NI CompactRIO User Manual

2026-08-19

Contents

NI CompactRIO User Manual 22
NI CompactRIO New Features and Changes 23
NI CompactRIO Device Drivers 26
CompactRIO Device Drivers Version Numbers 26
CompactRIO Device Drivers and LabVIEW Compatibility 26
CompactRIO Embedded System 32
Single-Board RIO System 34
Getting Started with a CompactRIO System in LabVIEW 35
Programming Interfaces 35
Using the Scan Interface with Individual Modules 38
Selecting an Appropriate Scan Period for Your Application 42
Programming Modes 43
Configuring a Project 44
Configuring a Project with Connected Hardware 46
Discovering your Hardware 46
Adding Additional Hardware 48
Configuring a Project with Offline Hardware 49
Adding a Controller Target 49
Adding a Chassis Item 50
Adding an FPGA Target 52
Adding I/O Items 53
Configuring a Project with DAQmx 55
Configuring Devices 56
Configuring Specialty Digital Functions 56
Configuring Counters 57
Count Edges 58
Measure the Period 60
Measure Pulse Width 61
Measure Frequency 61
Counter-Driven Outputs 64
Pulse-Width Modulation 64
Quadrature Input 65

Configuring the Chassis I/O (FPGA Interface) 67
Enabling Sleep Mode 67
Confirming the State of the Sleep Channel 68
Programmatically Resetting the System 68
Programming the FPGA LED 68
Reading the Temperature 70
Building and Troubleshooting an Application 70
CompactRIO Interface Methods (FPGA Interface) 71
CompactRIO Reconfigurable Embedded Chassis (FPGA Interface) 72
Converting the Temperature of a Chassis or Device 72
Synchronizing FPGA VIs with the NI Scan Engine (FPGA Interface) 73
Installing the FPGA Interface C API 76
Synchronizing Multiple C Series Modules 76
Configuring the Master Module 76
Configuring the Slave Modules 77
Acquiring Data from Synchronized C Series Modules 77
Synchronizing Multiple C Series Modules with NI 9469 Modules 79
How do I set a custom configuration for NI 9469 modules? 80
Using C Series Modules in a LabVIEW Project 80
Using the cRIO I/O Device Control 82
Using the RIO Device I/O Control 83
Configuring the C Series Module Data Rate (FPGA Interface) 84
Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface) 86
Reading from C Series Channels 87
Writing to C Series Channels 89
Configuring the cRIO I/O Node Name Control (FPGA Module) 90
Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface) 91
Supporting External Data Value References 93
Configuring the Host Memory Buffer 94
CompactRIO Controller and Chassis Reference 97
CompactRIO Reconfigurable Chassis 97
NI CompactRIO Controllers 97
CompactRIO DRAM Interface 97
Serial Ports 100

Configuring Controller Startup Settings in MAX 101
cRIO-903x Controllers 102
cRIO-904x Controllers 106
cRIO-905x Controllers 109
cRIO-906x Controllers 112
Ethernet RIO Chassis 115
MXIe-RIO Chassis 117
C Series Module Reference 120
C Series Module IDs 120
Analog Input Modules 131
Converting and Calibrating CompactRIO Analog Input Values (FPGA Interface) 134
Using a VI to Convert and Calibrate Values (NI-9204, NI-9205, and NI-9206) . . 135
Using an Equation to Convert and Calibrate Values 135
Detecting Out-of-Range Channels for CompactRIO Analog Input Channels (FPGA Interface) 138
Synchronizing Multiple Modules (FPGA Interface) 140
Configuring the Master Timebase Source for a Module (FPGA Interface) . 146
Understanding the Maximum Sample Rate when Synchronizing Multiple Modules (FPGA Interface) 148
NI 9202 148
NI 9218 150
NI 9225 151
NI 9227 153
NI 9229 154
NI 9230 155
NI 9231 157
NI 9232 158
NI 9234 160
NI 9235 161
NI 9236 162
NI 9237 164
NI 9238 165
NI 9239 167
NI 9242 168

NI 9244 169
NI 9246 171
NI 9247 172
NI 9250 174
NI 9251 175
NI 9252 176
NI 9253 178
NI 9260 179
NI 9470 181
Acquiring Data from a Module (FPGA Interface) 182
Configuring the Data Rate for a Module (FPGA Interface) 184
Using the C Series Module Properties Dalog Box 185
Using the FPGA I/O Property Node 185
Accessing TEDS Information from a Module (FPGA Interface) 185
Understanding Loop Timing (FPGA Interface) 186
NI 9201 188
NI 9201 (FPGA Interface) 188
Configuring the Minimum Time Between Conversions for the NI 9201/9221 (FPGA Interface) 190
Understanding NI 9201/9221 Scanning (FPGA Interface) 193
NI 9201 (Scan Interface) 194
NI 9202 194
NI 9202 (FPGA Interface) 195
Converting NI 9202 Data (FPGA Interface) 197
NI 9203 198
NI 9203 (FPGA Interface) 199
NI 9203 (Scan Interface) 201
NI 9204 201
NI 9204 (FPGA Interface) 202
AI Sample Method (FPGA Interface) 207
IO Sample Method (FPGA Interface) 209
Set Terminal Mode Method (FPGA Interface) 212
Set Triggers Method (FPGA Interface) 213
Set Voltage Range Method (FPGA Interface) 216
Conversion Timing for the NI 9204/9205/9206 (FPGA Interface) 217
Self-Calibration for the NI 9204/9205/9206 (FPGA Interface) 220

NI 9204 (Scan Interface) 221
NI 9205 222
NI 9205 (FPGA Interface) 223
AI Sample Method (FPGA Interface) 228
IO Sample Method (FPGA Interface) 231
Set Terminal Mode Method (FPGA Interface) 234
Set Triggers Method (FPGA Interface) 235
Set Voltage Range Method (FPGA Interface) 238
Conversion Timing for the NI 9204/9205/9206 (FPGA Interface) 239
Self-Calibration for the NI 9204/9205/9206 (FPGA Interface) 242
NI 9205 (Scan Interface) 242
NI 9206 243
NI 9206 (FPGA Interface) 244
AI Sample Method (FPGA Interface) 248
IO Sample Method (FPGA Interface) 251
Set Terminal Mode Method (FPGA Interface) 254
Set Triggers Method (FPGA Interface) 255
Set Voltage Range Method (FPGA Interface) 258
Conversion Timing for the NI 9204/9205/9206 (FPGA Interface) 259
Self-Calibration for the NI 9204/9205/9206 (FPGA Interface) 262
NI 9206 (Scan Interface) 262
NI 9207 263
NI 9207 (FPGA Interface) 264
Converting NI 9207 Data (FPGA Interface) 266
NI 9207 (Scan Interface) 266
NI 9208 267
NI 9208 (FPGA Interface) 268
Converting NI 9208 Data (FPGA Interface) 270
NI 9208 (Scan Interface) 270
NI 9209 271
NI 9209 (FPGA Interface) 272
Converting NI 9209 Data (FPGA Interface) 273
NI 9209 (Scan Interface) 274
NI 9210 275
NI 9210 (FPGA Interface) 275
Converting NI 9210 Data (FPGA Interface) 277

NI 9210 (Scan Interface) 279
NI 9211 280
NI 9211 (FPGA Interface) 280
Converting NI 9211 Data (FPGA Interface) 282
NI 9211 (Scan Interface) 284
NI 9212 285
NI 9212 (FPGA Interface) 286
Converting NI 9212 Data (FPGA Interface) 288
NI 9212 (Scan Interface) 290
NI 9213 291
NI 9213 (FPGA Interface) 292
Converting NI 9213 Data (FPGA Interface) 294
NI 9213 (Scan Interface) 296
NI 9214 297
NI 9214 (FPGA Interface) 298
Converting NI 9214 Data (FPGA Interface) 301
NI 9214 (Scan Interface) 302
NI 9215 304
NI 9215 (FPGA Interface) 305
NI 9215 (Scan Interface) 307
NI 9216 307
NI 9216 (FPGA Interface) 308
Converting Nominal Values to Temperature Values for the NI 9216 (FPGA Interface) 310
NI 9216 (Scan Interface) 310
NI 9217 311
NI 9217 (FPGA Interface) 312
Converting Nominal Values to Temperature Values for the NI 9217 (FPGA Interface) 314
NI 9217 (Scan Interface) 314
NI 9218 315
NI 9218 (FPGA Interface) 317
NI 9218 Data Rates 321
Loop Timing for the NI 9218 (FPGA Interface) 322
NI 9218 (Scan Interface) 323
NI 9219 324

NI 9219 (FPGA Interface) 326
Converting NI 9219 Data (FPGA Interface) 328
Configuring Modes and Ranges for the NI 9219 (FPGA Interface) . . . 331
Configuring the Digital In Threshold for the NI 9219 (FPGA Interface) . . . 332
NI 9219 (Scan Interface) 334
NI 9220 336
NI 9220 (FPGA Interface) 336
NI 9220 (Scan Interface) 338
NI 9221 339
NI 9221 (FPGA Interface) 340
Configuring the Minimum Time Between Conversions for the NI 9201/9221 (FPGA Interface) 342
Understanding NI 9201/9221 Scanning (FPGA Interface) 345
NI 9221 (Scan Interface) 345
NI 9222 346
NI 9222 (FPGA Interface) 347
NI 9222 (Scan Interface) 349
NI 9223 350
NI 9223 (FPGA Interface) 350
NI 9223 (Scan Interface) 352
NI 9224 353
NI 9224 (FPGA Interface) 354
Converting NI 9224 Data (FPGA Interface) 356
NI 9224 (Scan Interface) 356
NI 9225 357
NI 9225 (FPGA Interface) 358
NI 9225 (Scan Interface) 361
NI 9226 361
NI 9226 (FPGA Interface) 362
Converting Nominal Values to Temperature Values for the NI 9226 (FPGA Interface) 364
NI 9226 (Scan Interface) 364
NI 9227 365
NI 9227 (FPGA Interface) 366
NI 9227 (Scan Interface) 368

NI 9228 369
NI 9228 (FPGA Interface) 370
Converting NI 9228 Data (FPGA Interface) 372
NI 9228 (Scan Interface) 372
NI 9229 373
NI 9229 (FPGA Interface) 374
NI 9229 (Scan Interface) 376
NI 9230 377
NI 9230 (FPGA Interface) 377
Configuring the NI 9230/9232/9234 Channel Input Modes (FPGA Interface) 380
NI 9231 382
NI 9231 (FPGA Interface) 382
Configuring the NI 9231 Channel Input Modes (FPGA Interface) 385
Converting NI 9231 Data (FPGA Interface) 386
NI 9232 387
NI 9232 (FPGA Interface) 387
Configuring the NI 9230/9232/9234 Channel Input Modes (FPGA Interface) 390
NI 9234 392
NI 9234 (FPGA Interface) 392
Configuring the NI 9230/9232/9234 Channel Input Modes (FPGA Interface) 395
NI 9234 (Scan Interface) 397
NI 9235 397
NI 9235 (FPGA Interface) 398
NI 9235 (Scan Interface) 401
NI 9236 401
NI 9236 (FPGA Interface) 402
NI 9236 (Scan Interface) 405
NI 9237 405
NI 9237 (FPGA Interface) 406
NI 9237 (Scan Interface) 409
NI 9238 410
NI 9238 (FPGA Interface) 410
NI 9239 413

NI 9239 (FPGA Interface) 413
NI 9239 (Scan Interface) 416
NI 9242 416
NI 9242 (FPGA Interface) 417
Converting NI 9242 and NI 9244 Data (FPGA Interface) 419
NI 9242 (Scan Interface) 420
NI 9244 421
NI 9244 (FPGA Interface) 421
Converting NI 9242 and NI 9244 Data (FPGA Interface) 423
NI 9244 (Scan Interface) 424
NI 9246 425
NI 9246 (FPGA Interface) 426
NI 9247 428
NI 9247 (FPGA Interface) 429
NI 9250 434
NI 9250 (FPGA Interface) 434
Configuring the NI 9250 Channel Input Modes (FPGA Interface) 436
Converting NI 9250 Data (FPGA Interface) 438
NI 9251 438
NI 9251 (FPGA Interface) 439
Configuring the NI 9251 Channel Input Modes (FPGA Interface) 441
Converting NI 9251 Data (FPGA Interface) 442
NI 9252 443
NI 9252 (FPGA Interface) 444
Converting NI 9252 Data (FPGA Interface) 445
NI 9253 446
NI 9253 (FPGA Interface) 447
Converting NI 9253 Data (FPGA Interface) 449
NI 9253 (Scan Interface) 450
NI 9313 452
NI 9313 (FPGA Interface) 453
Converting NI 9313 Data (FPGA Interface) 454
NI 9313 (Scan Interface) 456
NI 9320 457
NI 9320 (FPGA Interface) 458
Converting NI 9320 Data (FPGA Interface) 460

NI 9381 461
NI 9381 (FPGA Interface) 462
Avoiding Timing Uncertainty with the NI 9381 (FPGA Interface) 467
Converting Voltage Values to Binary Values for the NI 9381 (FPGA Interface) 468
NI 9381 (Scan Interface) 469
Analog Output Modules 470
NI 9260 471
NI 9260 (FPGA Interface) 472
Converting Voltage Values to Binary Values for the NI 9260 (FPGA Interface) 474
Outputting Data from an NI 9260 (FPGA Interface) 475
NI 9262 477
NI 9262 (FPGA Interface) 477
Converting Voltage Values to Binary Values for the NI 9262/9263/9264/9269 (FPGA Interface) 480
NI 9263 481
NI 9263 (FPGA Interface) 482
Converting Voltage Values to Binary Values for the NI 9262/9263/9264/9269 (FPGA Interface) 485
NI 9263 (Scan Interface) 486
NI 9264 486
NI 9264 (FPGA Interface) 487
Converting Voltage Values to Binary Values for the NI 9262/9263/9264/9269 (FPGA Interface) 490
NI 9264 (Scan Interface) 491
NI 9265 492
NI 9265 (FPGA Interface) 493
Converting Current Values to Binary Values for the NI 9265 (FPGA Interface) 497
NI 9265 (Scan Interface) 499
NI 9266 500
NI 9266 (FPGA Interface) 500
Converting Current Values to Binary Values for the NI 9266 (FPGA Interface) 504
NI 9266 (Scan Interface) 506

NI 9269 507
NI 9269 (FPGA Interface) 507
Converting Voltage Values to Binary Values for the NI 9262/9263/9264/9269 (FPGA Interface) 511
NI 9269 (Scan Interface) 513
NI 9381 514
NI 9381 (FPGA Interface) 515
Avoiding Timing Uncertainty with the NI 9381 (FPGA Interface) 520
Converting Voltage Values to Binary Values for the NI 9381 (FPGA Interface) 521
NI 9381 (Scan Interface) 522
Counter Input Modules 523
NI 9326 523
NI 9326 (FPGA Interface) 524
NI 9326 I/O Method Parameters 532
Synchronizing Multiple NI 9326 Modules (FPGA Interface) 534
NI 9361 534
NI 9361 (FPGA Interface) 535
NI 9361 I/O Method Parameters 541
Synchronizing Multiple NI 9361 Modules (FPGA Interface) 549
Digital Input Modules 550
NI 9344 550
NI 9344 (FPGA Interface) 551
NI 9344 (Scan Interface) 555
NI 9375 556
NI 9375 (FPGA Interface) 556
Avoiding Timing Uncertainty with the NI 9375 (FPGA Interface) 558
NI 9375 (Scan Interface) 559
NI 9381 560
NI 9381 (FPGA Interface) 560
Avoiding Timing Uncertainty with the NI 9381 (FPGA Interface) 566
Converting Voltage Values to Binary Values for the NI 9381 (FPGA Interface) 567
NI 9381 (Scan Interface) 568
NI 9401 569
NI 9401 (FPGA Interface) 569

Configuring the Initial Line Direction for the NI 9401 (FPGA Interface) . . 573
NI 9401 (Scan Interface) 574
NI 9402 575
NI 9402 (FPGA Interface) 576
Configuring the Line Direction for the NI 9402 (FPGA Interface) 580
NI 9402 (Scan Interface) 582
NI 9403 583
NI 9403 (FPGA Interface) 584
Configuring the Line Direction for the NI 9403 (FPGA Interface) 587
NI 9403 (Scan Interface) 589
NI 9411 590
NI 9411 (FPGA Interface) 591
NI 9411 (Scan Interface) 594
NI 9421 595
NI 9421 (FPGA Interface) 596
NI 9421 (Scan Interface) 599
NI 9422 600
NI 9422 (FPGA Interface) 601
NI 9422 (Scan Interface) 604
NI 9423 605
NI 9423 (FPGA Interface) 606
NI 9423 (Scan Interface) 609
NI 9425 610
NI 9425 (FPGA Interface) 611
NI 9425 (Scan Interface) 613
NI 9426 614
NI 9426 (FPGA Interface) 614
NI 9426 (Scan Interface) 616
NI 9435 617
NI 9435 (FPGA Interface) 618
NI 9435 (Scan Interface) 621
NI 9436 622
NI 9436 (FPGA Interface) 622
NI 9436 (Scan Interface) 625
NI 9437 626

NI 9437 (FPGA Interface) 627
NI 9437 (Scan Interface) 630
Digital Output Modules 631
NI 9344 632
NI 9344 (FPGA Interface) 632
NI 9375 636
NI 9375 (FPGA Interface) 637
Avoiding Timing Uncertainty with the NI 9375 (FPGA Interface) 638
NI 9375 (Scan Interface) 639
NI 9381 640
NI 9381 (FPGA Interface) 640
Avoiding Timing Uncertainty with the NI 9381 (FPGA Interface) 646
Converting Voltage Values to Binary Values for the NI 9381 (FPGA Interface) 647
NI 9381 (Scan Interface) 648
NI 9401 649
NI 9401 (FPGA Interface) 649
Configuring the Initial Line Direction for the NI 9401 (FPGA Interface) . . 653
NI 9401 (Scan Interface) 654
NI 9402 655
NI 9402 (FPGA Interface) 655
Configuring the Line Direction for the NI 9402 (FPGA Interface) 659
NI 9402 (Scan Interface) 661
NI 9403 662
NI 9403 (FPGA Interface) 662
Configuring the Line Direction for the NI 9403 (FPGA Interface) 665
NI 9403 (Scan Interface) 667
NI 9470 668
NI 9470 (FPGA Interface) 669
Configuring Channels on the NI 9470 674
Synchronizing the NI 9470 with Other Modules (FPGA Interface) . . . 676
NI 9472 678
NI 9472 (FPGA Interface) 679
NI 9472 (Scan Interface) 682
NI 9474 683

NI 9474 (FPGA Interface) 683
NI 9474 (Scan Interface) 686
NI 9475 687
NI 9475 (FPGA Interface) 687
NI 9475 (Scan Interface) 690
NI 9476 691
NI 9476 (FPGA Interface) 691
NI 9476 (Scan Interface) 694
NI 9477 695
NI 9477 (FPGA Interface) 696
NI 9477 (Scan Interface) 697
NI 9478 698
NI 9478 (FPGA Interface) 699
Configuring Current Limits for the NI 9478 (FPGA Interface) 702
NI 9478 (Scan Interface) 704
NI 9481 706
NI 9481 (FPGA Interface) 706
NI 9481 (Scan Interface) 709
NI 9482 709
NI 9482 (FPGA Interface) 710
NI 9482 (Scan Interface) 712
NI 9485 713
NI 9485 (FPGA Interface) 714
NI 9485 (Scan Interface) 717
Functional Safety Modules. 717
NI 9350 718
NI 9350 (Scan Interface) 718
NI 9351 722
NI 9351 (Scan Interface) 723
Modular Instruments Modules 727
NI 9770 727
NI 9770 (FPGA Interface) 728
Synchronizing Multiple NI 9770 Modules (FPGA Interface) 733
NI 9775 735
NI 9775 (FPGA Interface) 736
NI 9775 Acquisition Modes (FPGA Interface). 740

Timing and Synchronization Module Reference 742
NI 9467 742
NI 9467 (FPGA Interface) 742
Glossary 747
NI 9469 747
NI 9469 (FPGA Interface) 748
CAN Interface Module Reference 751
C Series CAN Interface Modules 751
NI 9852 751
NI 9853 754
CAN Module Configuration 756
Add Targets and Devices 756
Module Configuration 757
Port Properties 757
Module Properties 759
Advanced CAN Port Properties 759
Timouts 760
Bit Timing 761
Input Filter 761
CAN Input 763
CAN Output 767
I/O Methods 771
Abort Transmit 772
Reset 773
Start 774
Stop 775
Wait on Comm State Change 775
Wait on Transceiver Wakeup 776
Wait on Transmit Complete 777
I/O Properties 778
Bit Timing 779
Comm State 780
Listen Only 781
Log Bus Errors 781
Log Transceiver Faults 782
LS/FT Termination 783

Receive Error Counter . . . 784
Self Reception . . . 785
Single Shot Transmit . . . 785
SJA1000 Filter Code . . . 786
SJA1000 Filter Mask . . . 786
SJA1000 Filter Mode . . . 787
Transceiver Mode . . . 787
Transmit Error Counter . . . 788
Module Properties . . . 788
Arbitration . . . 790
Error Terminals . . . 794
Converting Frames To and From Channels (Signals) . . . 797
Serial Interface Module Reference . . . 800
Programming Overview . . . 800
C Series Serial Interface Modules . . . 801
NI-987x Port Assignments . . . 802
NI 987x (FPGA Interface) . . . 802
NI 987x (Scan Interface) . . . 807
Connector Pinouts . . . 808
Serial Communication . . . 810
Communication Standards . . . 810
RS232 . . . 811
RS422 . . . 811
RS485 . . . 812
RS485 Transceiver Control . . . 812
Bias Resistors . . . 814
Duplex Architectures . . . 814
Full Duplex . . . 815
Half Duplex . . . 816
RS232 DTE versus DCE . . . 816
RS485 Termination . . . 817
Does the serial hardware have built-in support for protocols such as SDLC and ModBus? . . . 818
How do I use 9 data bit framing? . . . 818
What is the relationship between bits per second, throughput, and bus utilization? . . . 819

Motion Module Reference 820
C Series Drive Motion Modules 820
NI 950x LEDs and Connectors 821
NI 9501 (FPGA Interface) 824
NI 9501 Module Faults 830
NI 9502 (FPGA Interface) 830
NI 9502 Module Faults 837
NI 9502 Commutation Mode (FPGA Interface) 838
Connecting Hall Effect Sensors for Trapezoidal Commutation (FPGA Interface) 842
NI 9502 Timing (FPGA Interface) 844
NI 9503 (FPGA Interface) 846
NI 9503 Module Faults 849
NI 9503 Timing (FPGA Interface) 850
NI 9505 (FPGA Interface) 851
NI 9505 Motor Output Pulse Width Considerations (FPGA Interface) 856
NI 9505 Current Measurement Considerations (FPGA Interface) 858
NI 9505 Motor Output and Current Sense Example (FPGA Interface) 859
C Series Motor Drive Interface Modules 861
NI 951x LEDs and Connectors 861
NI 951x Module Faults 864
Downloading Firmware to an NI 951x Module 866
NI 951x (Scan Interface) 867
Single-Board RIO Device Reference 868
Single-Board RIO Devices 869
sbRIO-9603/9608/9609/9628/9638/9629 CAN API 870
CAN Input 872
CAN Output 876
I/O Methods 881
Reset 881
Start 882
Stop 882
Abort Transmit 883
Wait on Comm State Change 884
Wait on Transceiver Wakeup 884
Wait on Transmit Complete 885

I/O Properties 886
Baud Rate 887
Baud Rate Advanced 888
FD Baud Rate 889
FD Baud Rate Advanced 889
Comm State 891
Listen Only 891
I/O Mode 892
Input Timeout 893
Output Timeout (ms) 893
Log Bus Errors 893
Log Transceiver Faults 895
Self Reception 896
Single Shot Transmit 896
Receive Error Counter 897
Transmit Error Counter 897
Transceiver Mode 897
sbRIO-9603 898
sbRIO-9607 899
sbRIO-9608 900
sbRIO-9609 901
sbRIO-9627 902
sbRIO-9627 (FPGA Interface) 904
sbRIO-9628 906
sbRIO-9628 (FPGA Interface) 907
Converting Analog Input Values for the sbRIO-9628/9629/9638 (FPGA Interface) 911
Converting Analog Output Values for the sbRIO-9628/9629/9638 (FPGA Interface) 912
sbRIO-9629 913
sbRIO-9629 (FPGA Interface) 914
sbRIO-9637 918
sbRIO-9637 (FPGA Interface) 920
sbRIO-9638 922
sbRIO-9638 (FPGA Interface) 924
Converting Analog Input Values for the sbRIO-9628/9629/9638 (FPGA

Interface) 928
Converting Analog Output Values for the sbRIO-9628/9629/9638 (FPGA Interface) 929
sbRIO-9651 930
Getting Started with the sbRIO-9651 in LabVIEW 931
Creating Component-Level IP for the sbRIO-9651 932
Adding a CLIP for a Single-Board RIO Device to a LabVIEW Project 933
Implementing FPGA VIs for the sbRIO-9651 934
RIO Mezzanine Cards 935
sbRIO-9683 935
sbRIO-9683 (FPGA Interface) 936
Avoiding Timing Uncertainty with the NI 9683 (FPGA Interface) 941
Scaling NI 9683 Analog Input Values (FPGA Interface) 941
Scaling NI 9683 Analog Output Values (FPGA Interface) 942
sbRIO-9684 943
sbRIO-9684 (FPGA Interface) 944
Avoiding Timing Uncertainty with the sbRIO-9684 (FPGA Interface) 949
Scaling sbRIO-9684 Analog Input Values (FPGA Interface) 950
Scaling sbRIO-9684 Analog Output Values (FPGA Interface). 951
sbRIO-9694 952
sbRIO-9694 (FPGA Interface) 952
sbRIO-9697 954
sbRIO-9697 (FPGA Interface) 954
sbRIO-9698 956
sbRIO-9698 (FPGA Interface) 957
NI Digital I/O Mezzanine Card (FPGA Interface) 958
NI Single-Board RIO CLIP Generator 960
Creating Component-Level IP for an NI sbRIO Target 961
Creating a New CLIP 961
Adding Peripherals to a CLIP 962
Serial Port Naming Conventions. 964
Configuring I/O Pins on the Target 965
Creating a LabVIEW I/O Node Interface for a CLIP. 967
Configuring Clock Resources for a CLIP 968
Finishing the CLIP Design 970

myRIO Module Reference 972
myRIO-1900 972
myRIO-1900 (FPGA Interface) 974
myRIO-1950 976
myRIO-1950 (FPGA Interface) 978
NI ELVIS RIO Control Module 980
NI ELVIS RIO Control Module (FPGA Interface) 980
NI ELVIS III 983
NI ELVIS III (FPGA Interface) 985
IO Sample Method (FPGA Interface) 988
Conversion Timing for the NI ELVIS III (FPGA Interface) 990

NI CompactRIO User Manual

The NI CompactRIO User Manual provides detailed descriptions of the product functionality and the step by step processes for use.

Looking for Something Else?

For information not found in the User Manual for your product, such as specifications and API reference, browse *Related Information*.

Related information:

- CompactRIO Systems (cRIO)
- CompactRIO Chassis
- Download NI CompactRIO
- Interactive Activation Guide
- NI CompactRIO Release Notes
- NI CompactRIO Waveform Reference Library
- LabVIEW for CompactRIO Developer's Guide
- Developing Embedded Applications using CompactRIO and LabVIEW FPGA
- LabVIEW FPGA Module Programming Reference Manual
- Software Support for CompactRIO, CompactDAQ, Single-Board RIO, R Series, and EtherCAT

NI CompactRIO New Features and Changes

Learn about updates, including new features and behavior changes, introduced in each version of NI CompactRIO.

Discover what is new in the latest releases of NI CompactRIO.

> **Note** If you cannot find new features and changes for your version, it might not include user-facing updates. However, your version might include non-visible changes such as bug fixes and compatibility updates. For information about non-visible changes, refer to your product *Release Notes*.

Related information:

- Software and Driver Downloads

NI CompactRIO 26Q3 Changes

NI CompactRIO 26Q3 adds support for the NI 9313.

New Hardware Support

This version of the NI CompactRIO adds support for the following hardware:

- NI-9313 C Series Temperature Input Module

NI CompactRIO 2026 Q2 Changes

Learn about new features, behavior changes, and other updates in NI CompactRIO 2026 Q2.

New Features

This version of NI CompactRIO adds support for the following features:

- Added Scan Interface support for the NI 9204.

- Bug fixes and improvements.

NI CompactRIO 2025 Q4 Changes

Learn about new features, behavior changes, and other updates in NI CompactRIO 2025 Q4.

Hardware Support

- Added support for the following NI CompactRIO devices:
  - NI 9320

NI CompactRIO 2025 Q1 Changes

Learn about new features, behavior changes, and other updates in NI CompactRIO 2025 Q1.

Hardware Support

- Added connector variants for the following NI CompactRIO devices:
  - NI 9222
  - NI 9223
  - NI 9225
  - NI 9227
  - NI 9228
  - NI 9229
  - NI 9230
  - NI 9232
  - NI 9238
  - NI 9239
  - NI 9269

NI CompactRIO 2024 Q4 Changes

Learn about new features, behavior changes, and other updates in NI CompactRIO 2024 Q4.

Hardware Support

- Added support for the following NI CompactRIO devices:
  - NI 9204

NI CompactRIO Device Drivers

Use NI CompactRIO software in LabVIEW to add CompactRIO controllers and chassis, C Series modules, single-board RIO devices, and myRIO devices to a CompactRIO embedded system.

NI CompactRIO Device Driver Versions

Tip To optimize product performance, update to the most recent driver version.

The NI CompactRIO Device Drivers software might appear under different names depending on the version.

Table 1. NI CompactRIO Device Driver Versions

Driver Version	NI CompactRIO Device Driver Name
4.1 or earlier	NI-RIO
12.0 (August 2012) through 14.5 (February 2015)	NI-RIO Device Drivers
15.0 (August 2015) or later	NI CompactRIO Device Drivers NI R Series Multifunction RIO Device Drivers [1]

Related reference:

- NI CompactRIO Device Drivers and LabVIEW Compatibility

NI CompactRIO Device Drivers and LabVIEW Compatibility

Each version of NI CompactRIO Device Drivers supports the following LabVIEW

1. Prior to version 15.0, both the NI CompactRIO Device Drivers and NI R Series Multifunction RIO Device Drivers were included in the NI-RIO Device Drivers.

versions:

The following table lists LabVIEW compatibility support.

- The version of LabVIEW that released with the version of NI CompactRIO Device Drivers
- The three previous LabVIEW service pack versions

Note

For driver versions released after February 2017, refer to *NI Driver and Development Software Compatibility* in *Related information*.

Table 2. NI CompactRIO Device Drivers LabVIEW Compatibility

NI CompactRIO Device Drivers Version	LabVIEW Version Support
February 2017	2017 2016 SP1 2015 SP1 2014 SP1
August 2016	2016 2015 SP1 2014 SP1 2013 SP1
February 2016	2015 SP1 2014 SP1

NI CompactRIO Device Drivers Version	LabVIEW Version Support
	2013 SP1 2012 SP1
August 2015	2015 2014 SP1 2013 SP1 2012 SP1
14.5 14.0.5	2014 SP1 2013 SP1 2012 SP1 2011 SP1
14.0.1 14.0	2014 2013 SP1 2012 SP1 2011 SP1
13.1.1 13.1	2013 SP1 2012 SP1 2011 SP1 2010 SP1

NI CompactRIO Device Drivers Version	LabVIEW Version Support
13.0.1 13.0	2013 2012 SP1 2011 SP1 2010 SP1
12.1	2012 SP1 2011 SP1 2010 SP1 2009 SP1
12.0	2012 2011 SP1 2010 SP1 2009 SP1
4.1	2011 SP1 2010 SP1 2009 SP1 8.6.1
4.0	2011 2010 SP1

NI CompactRIO Device Drivers Version	LabVIEW Version Support
	2009 SP1 8.6.1
3.6.1 3.6	2010 SP1 2009 SP1 8.6.1 8.5.1
3.5.1	2010 2009 SP1 8.6.1 8.5.1
3.4	2009 SP1 8.6.1 8.5.1 8.2.1
3.3.1 3.3	2009 8.6.1 8.5.1 8.2.1

NI CompactRIO Device Drivers Version	LabVIEW Version Support
3.2.1	2009 8.6.1 8.5.1 8.2.1
3.1.1	8.6.1 8.5.1 8.2.1
3.0.1 3.0	8.6 8.5.1 8.2.1
2.4.1 2.4	8.5.1 8.2.1 8.0
2.3.1	8.5 8.2.1 8.0
2.1.3	8.2.1

NI CompactRIO Device Drivers Version	LabVIEW Version Support
2.1.2 2.1.1	8.0
2.1	8.2 8.0
2.0.2 2.0.1 2.0	8.0

Related reference:

- NI CompactRIO Device Driver Versions

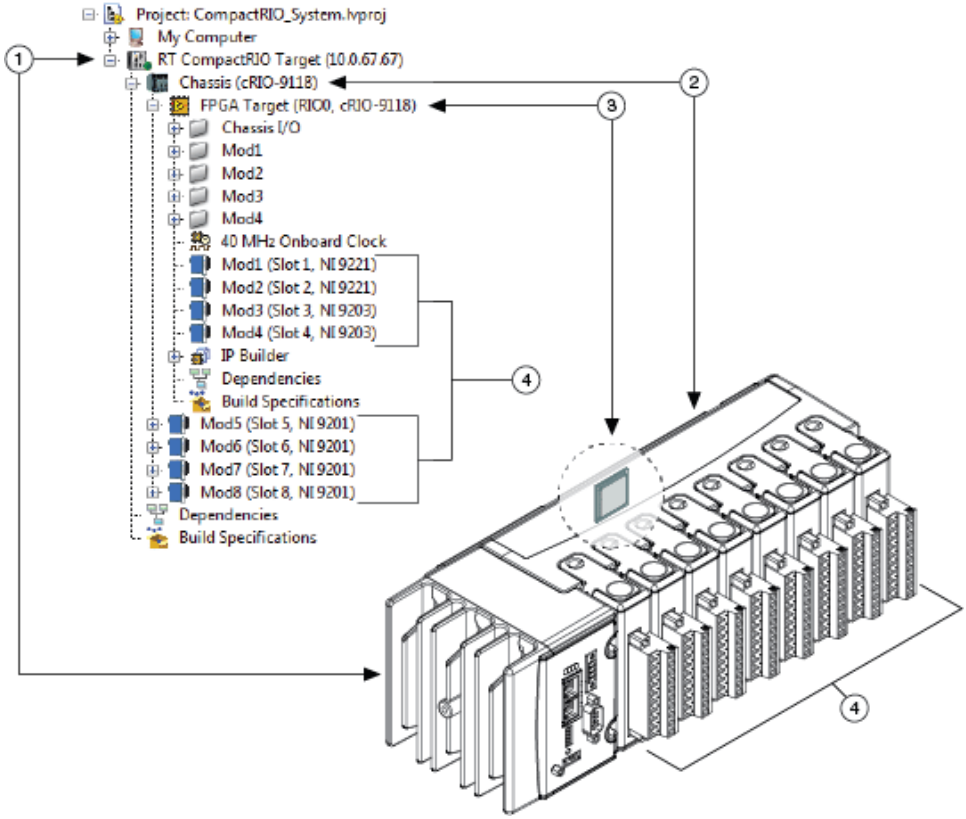
Related information:

- NI Driver and Development Software Compatibility

CompactRIO Embedded System

A CompactRIO embedded system includes a Real-Time processor, CompactRIO chassis, user-configurable FPGA, and C Series I/O modules.

Figure 1. Components in a CompactRIO Embedded System



1. Real-Time processor
2. CompactRIO chassis
3. User-configurable FPGA
4. C Series I/O modules

Real-Time Processor

CompactRIO controllers feature a real-time processor that deterministically executes LabVIEW applications.

CompactRIO Chassis

CompactRIO chassis feature a user-configurable FPGA that controls the digital and analog I/O lines on the reconfigurable embedded chassis.

User-configurable FPGA

The FPGA chip embedded within the chassis connects to the C Series I/O modules in a

star topology. The star topology provides direct access to each module for precise control and flexibility in timing, triggering, and synchronization.

To change the functionality of the chassis FPGA, use the LabVIEW FPGA Module and NI CompactRIO Device Drivers to create a custom VI. Download the custom VI to the FPGA.

C Series I/O Modules

C Series I/O modules provide integrated signal conditioning and connection terminals for different measurement types.

Related concepts:

- Configuring a Project

Single-Board RIO System

A single-board RIO (sbRIO) device consists of a real-time processor that is connected to a reconfigurable FPGA on a single PCB.

The user-reconfigurable FPGA controls the digital I/O lines and the analog I/O lines of the sbRIO device. sbRIO devices can include the onboard digital I/O, onboard C Series Modules, and/or expansion slots for C Series modules, and sbRIO mezzanine cards. Specific features vary by the device.

To configure sbRIO device timing and functionality, create a custom VI with the LabVIEW FPGA Module and NI CompactRIO Device Drivers, then download the custom VI to the FPGA. If you have only LabVIEW and NI CompactRIO Device Drivers but not the FPGA Module,Without the FPGA Module, you cannot create new FPGA VIs, but you can create VIs that run on Windows or LabVIEW Real-Time (RT) target to control an existing FPGA VI.

Related concepts:

- Configuring a Project
- Single-Board RIO (sbRIO) Device Reference

Getting Started with a CompactRIO System in LabVIEW

Use the information in this section to help you complete the following tasks as you get started with a CompactRIO embedded system in LabVIEW.

Learn Programming Techniques	Configure a LabVIEW Project	Configure Your Device
Use devices in Scan Interface mode, FPGA Interface mode, or Hybrid mode.	Set up a LabVIEW project to use hardware you have installed or hardware you have not installed.	Configure the I/O on your installed hardware.

Programming Interfaces

Communicate with devices using Scan Interface mode, FPGA Interface mode, or Hybrid mode.

Scan Interface Mode

Scan Interface mode enables you to use C Series modules directly from LabVIEW Real-Time. Modules that you use in Scan Interface mode appear under the Real-Time Scan Resources item in the **Project Explorer** window and I/O channels appear as I/O variables under the modules. To use I/O variables, you drag and drop them from the Project Explorer window to LabVIEW Real-Time VIs.

In Scan Interface mode, you do not need to do any LabVIEW FPGA development. LabVIEW programs the FPGA for you with a fixed FPGA bitfile that communicates with all the C Series modules that Scan Interface mode supports. LabVIEW also sends C Series data to the Real-Time host to be displayed in I/O variables. Scan Interface mode also enables you to dynamically detect which types of C Series modules are plugged into chassis slots.

Table 3. C Series Modules Supported in the Scan Interface

• NI 9201 • NI 9203	• NI 9220 • NI 9221	• NI 9264 • NI 9265	• NI 9426 • NI 9435

• NI 9204 • NI 9205 • NI 9206 • NI 9207 • NI 9208 • NI 9209 • NI 9210 • NI 9211 • NI 9212 • NI 9213 • NI 9214 • NI 9215 • NI 9216 • NI 9217 • NI 9218 • NI 9219	• NI 9222 • NI 9223 • NI 9224 • NI 9225 • NI 9226 • NI 9227 • NI 9228 • NI 9229 • NI 9234 • NI 9235 • NI 9236 • NI 9237 • NI 9239 • NI 9242 • NI 9244 • NI 9263	• NI 9266 • NI 9269 • NI 9313 • NI 9344 • NI 9350 • NI 9351 • NI 9375 • NI 9381 • NI 9401 • NI 9402 • NI 9403 • NI 9411 • NI 9421 • NI 9422 • NI 9423 • NI 9425	• NI 9436 • NI 9437 • NI 9472 • NI 9474 • NI 9475 • NI 9476 • NI 9477 • NI 9478 • NI 9481 • NI 9482 • NI 9485 • NI 9512 • NI 9514 • NI 9516

Table 4. CompactRIO Devices Supported in the Scan Interface

• cRIO-9022 • cRIO-9023 • cRIO-9024 • cRIO-9025 • cRIO-9030 • cRIO-9031 • cRIO-9032 • cRIO-9033 • cRIO-9034 • cRIO-9035 • cRIO-9035 (Sync) • cRIO-9036 • cRIO-9037 • cRIO-9038 • cRIO-9039 • cRIO-9039 (Sync) • cRIO-9040	• cRIO-9041 • cRIO-9042 • cRIO-9043 • cRIO-9045 • cRIO-9046 • cRIO-9047 • cRIO-9048 • cRIO-9049 • cRIO-9053 • cRIO-9054 • cRIO-9056 • cRIO-9057 • cRIO-9063 • cRIO-9064 • cRIO-9065 • cRIO-9066 • cRIO-9067	• cRIO-9068 • cRIO-9073 • cRIO-9074 • cRIO-9075 • cRIO-9076 • cRIO-9081 • cRIO-9082 • cRIO-9111 • cRIO-9112 • cRIO-9113 • cRIO-9114 • cRIO-9116 • cRIO-9118 • NI 9146 • NI 9147 • NI 9148 • NI 9149

Table 5. Single-Board RIO Devices Supported in the Scan Interface

• sbRIO-9601 • sbRIO-9602 • sbRIO-9602XT • sbRIO-9611 • sbRIO-9612 • sbRIO-9612XT	• sbRIO-9631 • sbRIO-9632 • sbRIO-9632XT • sbRIO-9641 • sbRIO-9642 • sbRIO-9642XT



LabVIEW FPGA Interface Mode

LabVIEW FPGA Interface mode enables you to use C Series modules from LabVIEW FPGA VIs. Modules that you use in LabVIEW FPGA Interface mode appear directly under the FPGA Target item in the **Project Explorer** window and I/O channels appear as FPGA I/O items under the FPGA Target. To access the I/O channels, you either configure FPGA I/O Nodes in a LabVIEW FPGA VI or drag and drop the I/O channels from the **Project Explorer** window to a LabVIEW FPGA VI block diagram.

In LabVIEW FPGA Interface mode, you can use LabVIEW FPGA programming to add more flexibility, customization, timing, and synchronization to your applications. To use the CompactRIO system in LabVIEW FPGA Interface mode, you must either have the LabVIEW FPGA Module installed on the host computer, or have access to a compiled bitfile that you can download to the FPGA. In either case, you use the Open FPGA VI Reference function in a LabVIEW Real-Time VI to access the FPGA VI or bitfile.

Hybrid Interface Mode

Hybrid Interface mode enables you to program custom FPGA functionality while simultaneously accessing some modules in Scan Interface mode. For example, you may want to use LabVIEW FPGA Interface mode to handle waveform buffered acquisition and inline processing, or you may be using some modules that do not support Scan Interface mode. To use Hybrid Interface mode, do the following:

1. Set up your project in LabVIEW FPGA Interface mode.
2. In the Project Explorer window, drag the modules you want to use in Scan Interface mode from the FPGA Target item to the Real-Time Scan Resources item. Alternatively, you can add modules under the Real-Time Scan Resources item by

right-clicking the Real-Time Scan Resources item and selecting **New»C Series Modules** and letting LabVIEW discover the modules.

Related concepts:

- Configuring a Project with Connected Hardware
- Using C Series Modules in a LabVIEW Project

Using the Scan Interface with Individual Modules

You may be able to use the Scan Interface for supported individual modules even if the chassis you are using is in **FPGA Interface** programming mode or does not support **Scan Interface** programming mode. First you must let LabVIEW discover the modules in the chassis and download a compiled bitfile to the chassis. Using modules that require the high-speed interface, such as the NI 951x, may also require a downloaded compiled bitfile. You must have the LabVIEW FPGA module installed to compile bitfiles for the FPGA.

After setting up your project, discovering modules, and downloading the bitfile to the chassis, use modules under the FPGA target in FPGA Interface programming mode and use modules under the Real-Time Scan Resources item in Scan Interface programming mode.

If you have both the LabVIEW Real-Time Module and the LabVIEW FPGA Module installed, you can create user-defined I/O variables to transfer custom I/O data between FPGA VIs and RT VIs. You can also synchronize the execution of code in FPGA VIs and RT VIs.

Note f the chassis is in FPGA Interface programming mode, you must always use the Open FPGA VI Reference function to download a compiled bitfile to the FPGA before using modules that are under the Real-Time Scan Resources item. You must ensure that the bitfile is running on the FPGA before accessing those modules.

Tip There is a short delay between the time when the Open FPGA VI Reference function finishes running and the time when user-defined variables and modules under the Real-Time Scan Resources item return valid

data. If you have user-defined variables in your VI, run the Open FPGA VI Reference function and then read the variables in a loop until they return no errors.

Note If the chassis is in FPGA Interface programming mode, using the Scan Interface reserves two of the available DMA FIFOs and occupies a portion of the FPGA space, resulting in a longer compile time than that for a bitfile with no modules in Scan Interface programming mode.

Note If any modules are under the Real-Time Scan Resources item, you cannot write to the Sleep channel of the FPGA target.

Related concepts:

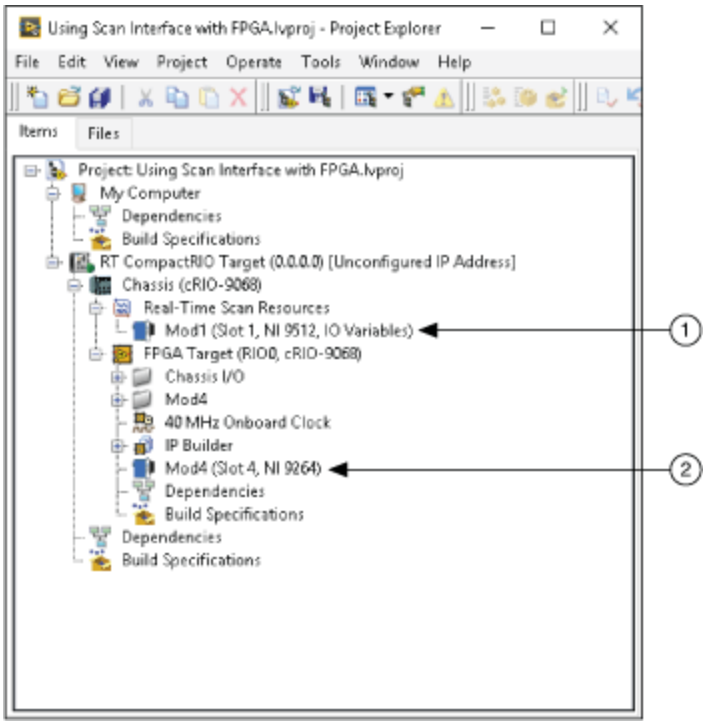
- Synchronizing FPGA VIs with the NI Scan Engine (FPGA Interface)
- Configuring a Project

Creating the LabVIEW Project

Complete the following steps to set up the LabVIEW project:

1. Configure the system.
2. Drag and drop the C Series module(s) that will run in Scan Interface mode under the Real-Time Scan Resources item. Leave any modules you plan to write FPGA code for under the FPGA target.

The LabVIEW Project Explorer window should look similar to the following figure:



1. Module in Scan Interface Programming Mode
2. Module in FPGA Interface Programming Mode

Downloading the Bitfile

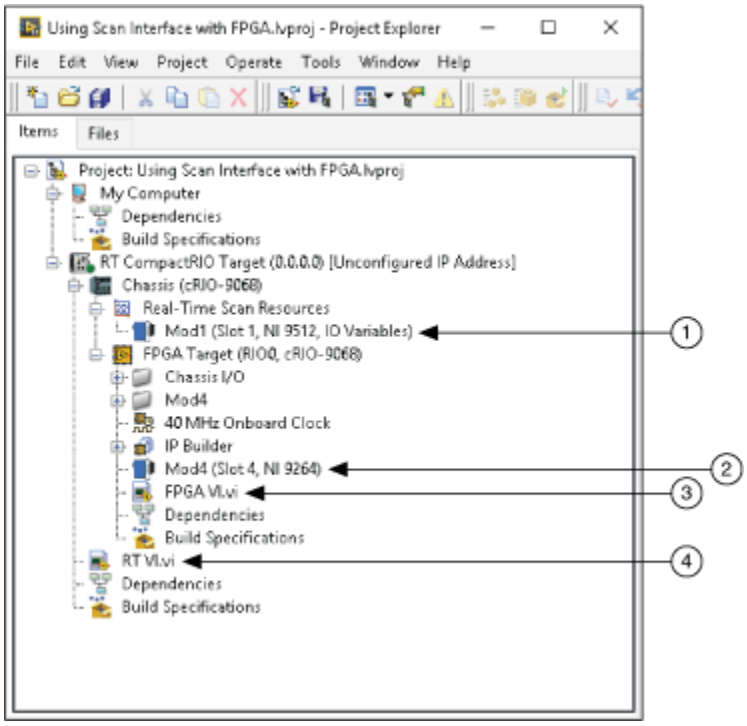
Complete the following steps to compile a bitfile and download it to the chassis:

1. Create a new VI under the FPGA target and save it with your project files. You will use this FPGA VI from an RT VI to download the module bitfile to the chassis.
2. Right-click the FPGA VI and select **Create Build Specification** from the shortcut menu. Under **Build Specifications**, right-click the new build specification for the example FPGA VI, select **Build**, and wait for the build to complete.
3. Create a new VI under the RT target and save it with your Project files.
4. Add an **Open FPGA VI Reference** function to the block diagram of the new RT VI.
5. Right-click the **Open FPGA VI Reference** function and select **Configure Open FPGA VI Reference** from the shortcut menu to open the **Configure Open FPGA VI Reference** dialog box.
6. Select the FPGA VI you created in step 1, make sure that the **Run the FPGA VI** checkbox is enabled, and click **OK** to close the dialog box.
7. Place the code you want to run in the RT VI on the block diagram to the right of the **Open FPGA VI Reference** function. Enclose the code in a structure such as a Timed

Loop or While Loop.

8. Add a **Close FPGA VI Reference** function to the block diagram to the right of the structure enclosing the code.
9. Connect the **Open FPGA VI Reference** function to the **Close FPGA VI Reference** function through the structure that encloses the code.
10. Run the RT VI.

The LabVIEW Project Explorer window should look similar to the following figure:



1. Module in Scan Interface Mode
2. Module in FPGA Interface Mode
3. FPGA VI to Download Bitfile
4. RT VI to Download Bitfile

Note The FPGA VI in the example procedure contains no FPGA code. The only purpose of this FPGA VI is to download to a module bitfile to the chassis. The module bitfile specifies the types and slot locations of all modules in the chassis. If you have LabVIEW FPGA code to execute in your application, you include it in the FPGA VI. When you compile and run the RT VI, it downloads the module bitfile and all FPGA code to the chassis.

Selecting an Appropriate Scan Period for Your Application

Take care when deciding how fast you want the I/O Scan to run because it runs in a high-priority thread.

The Scan Period must be long enough for the Scan Engine to scan all the I/O variables and for the CPU to execute any code running in timed loops synchronized with the Scan Engine. If you select a Scan Period below 1 ms, the system can become unstable. If the system can become unstable you can lose communication with the controller or device, or the Scan Engine may not have enough time to read and write all of the I/O.

There are a few things you can check to determine whether the system has enough resources to run your application at the speed you require.

Checking Base CPU Usage

Use the **NI Distributed System Manager** or the **RT Get CPU Loads VI** to determine what percentage of the CPU capacity the controller or device is using. Check CPU usage without an application running to see how much CPU capacity is left for your application.

Detecting Modules Not Being Scanned Quickly Enough

The **Percent Hardware Scan Utilization I/O Variable** indicates what percentage of the allotted time for data transfers is taken up by scanning I/O. If the percentage value reaches 100, a fault will occur. Use the NI Distributed System Manager to monitor the variable or read the variable programmatically.

You can monitor Scan Engine faults to determine whether modules are being scanned quickly enough for your application. You can configure a fault to stop execution if that is appropriate for your application. Look for the following fault.

Table 6. Scan Engine Fault Data Transfer Error

Error Code	Description
–65512	The data transfer for some I/O Variables on this target could not be completed in the allotted time and the updates for some values may have been delayed. Increase the Scan

Error Code	Description
	Period to avoid this problem.

Programming Modes

Configure the programming mode for each slot on a chassis.

There are three programming modes.

The default mode is only set after a fresh install and is determined by what is installed on the controller.

Table 7. Programming Modes

Programming Mode	Resource Item	Usage
Real-Time	Real-Time Resources	cRIO-904x and cRIO-905x controllers. Real Time Driver API support such as NI-DAQmx and NI-XNET support.
Real-Time Scan	Real-Time Scan Resources	IO Variables and Scan Engine support
LabVIEW FPGA	FPGA Target	IO Nodes on FPGA

Configuring a Program Mode

Complete the following steps to configure a programming mode. Programming modes can also be configured by using the NI System Configuration API.

1. Drag and drop the **C Series Module IO Item** into the resource item for the mode that you want. Refer to the *Programming Modes* table above for the resource item to use for each mode.
2. Right-click the **Controller Target Item** and select **Deploy All** or right-click the Module IO Item and select **Deploy Module Modes**.

Dynamic Mode Switching

> **Note** Dynamic Mode Switching is only available on cRIO-904x and cRIO-905x controllers.

You can change the programming mode of a cRIO-904x or cRIO-905x controller at any time regardless of deployed chassis slot operations. To execute a mode switch safely, the module will be put into a safe state. Modules have different behaviors when this occurs.

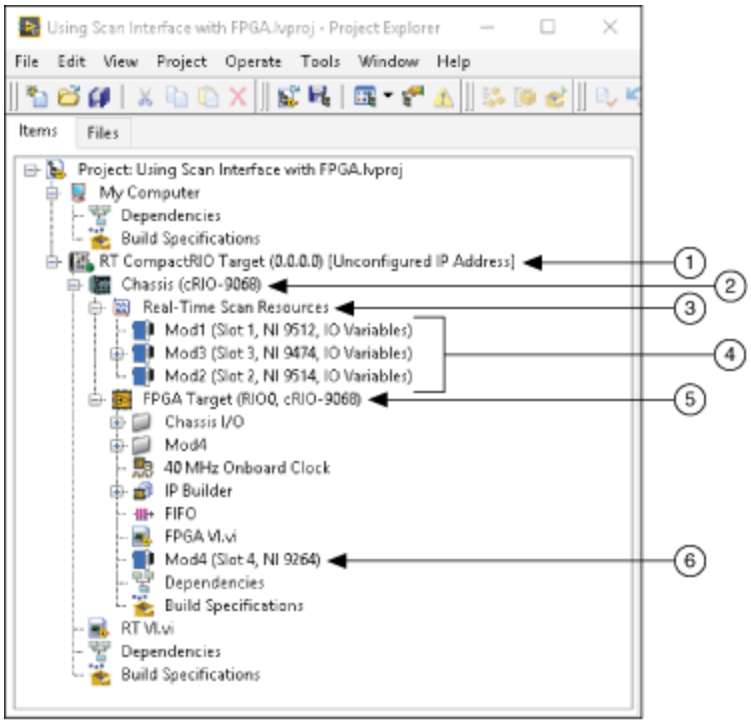
Configuring a Project

Configure a LabVIEW project with connected hardware or offline hardware.

Use a LabVIEW project to work with NI CompactRIO Device Drivers and C Series modules. Configure the project with targets and items for your hardware devices:

- **Target**—Any device on which you can run LabVIEW VIs, such as CompactRIO controllers, Single-Board RIO devices, myRIO devices, roboRIO devices, and NI ELVIS RIO Control Module (NI ELVIS RIO CM) devices.
- **Item**—A container for reconfigurable I/O or devices, such as C Series modules, CompactRIO chassis, Ethernet RIO chassis, MXIe-RIO chassis, and Single-Board RIO devices.

The following figure shows a LabVIEW project configured with targets and items.



1. Controller Target
2. Chassis Item
3. Real-Time Scan Resources Item
4. C Series Module Items (Scan Interface)
5. FPGA Target
6. C Series Module Item (FPGA Interface)

Note To discover targets and add them to a project, you must have LabVIEW RT and LabVIEW FPGA installed on the host system.

For an example of a LabVIEW project using C Series I/O, refer to the *Using the Scan Interface with FPGA Interface* LabVIEW project in the `labview\examples\CompactRIO\NI Scan Engine\Getting Started\Using Scan Interface with FPGA Interface\Using Scan Interface with FPGA Interface.lvproj`.

Related concepts:

- Single-Board RIO System
- Using C Series Modules in a LabVIEW Project

- Using the Scan Interface with Individual Modules

Configuring a Project with Connected Hardware

Configure your project to discover installed hardware.

Discovering your Hardware

Complete the following steps to discover your hardware in a project.

1. Create a new project in LabVIEW or open an existing project.
2. Right-click the project root in the **Project Explorer** window and select **New»Targets and Devices** from the shortcut menu to display the **Add Targets and Devices** dialog box.
3. Select the appropriate device from the **Targets and Devices** list.

   > **Note** If you cannot find the appropriate device, you may not have configured it for your network using Measurement & Automation Explorer (MAX). Launch MAX and refer to the MAX help for CompactRIO.

4. Click the **OK** button.
5. If the **Select Programming Mode** dialog box appears, select **LabVIEW FPGA Interface** or **Scan Interface** as appropriate.
6. Click the **Continue** button.
7. If the **Discover C Series Modules?** dialog box appears, click the **Discover** button.
8. Right-click a module in the **Project Explorer** window and select **Properties** from the shortcut menu to configure module-specific settings.
9. Repeat steps 2 through 8 to add an expansion chassis or additional expansion chassis to the project.

Related concepts:

- Programming Interfaces

Project Targets and Items

LabVIEW adds targets and items to your project for the hardware in your system.

The following table shows the target icons LabVIEW displays in the project.

Table 8. Target Icons in LabVIEW

Target Icon	Device	Description
	CompactRIO controller	LabVIEW adds this controller target to a project for a CompactRIO controller.
	Ethernet RIO chassis	LabVIEW adds this target to a project for a Ethernet RIO chassis. The Ethernet RIO chassis is not a real-time target, but it can add expansion I/O to a real-time target.
	Single-Board RIO device	LabVIEW adds this controller target to a project for a Single-Board RIO device.
	myRIO device	LabVIEW adds this controller target to a project for a myRIO-1900.
	myRIO device	LabVIEW adds this controller target to a project for a myRIO-1950.
	roboRIO device	LabVIEW adds this controller target to a project for a roboRIO.
	NI ELVIS RIO Control Module (NI ELVIS RIO CM) device	LabVIEW adds this controller target to a project for an NI ELVIS RIO CM.

The following table shows the item icons LabVIEW displays in the project.

Table 9. Item Icons in LabVIEW

Item Icon	Device	Description
	CompactRIO chassis, Ethernet RIO chassis	LabVIEW adds this chassis target to a project for a CompactRIO chassis or Ethernet RIO chassis.
	MXIe-RIO chassis	LabVIEW adds this chassis target to a project for a MXIe-RIO chassis. You can add a MXIe-RIO chassis to a real-time controller or the host PC.
	Single-Board RIO, myRIO, roboRIO, or NI ELVIS RIO CM device	LabVIEW adds this chassis target to a project for a Single-Board RIO, myRIO, roboRIO, or NI ELVIS RIO CM device.
	FPGA	LabVIEW adds this FPGA target to a project for the FPGA on a CompactRIO chassis, Ethernet RIO chassis, MXIe-RIO chassis, or Single-Board RIO device in the FGPA Interface.

Item Icon	Device	Description
	C Series modules	LabVIEW adds this module target to a project for C Series modules.
	Onboard C Series modules	LabVIEW adds this module target to a project for C Series modules onboard a Single-Board RIO device.

Adding Additional Hardware

LabVIEW cannot discover all compatible hardware. Complete the following steps to add an R Series expansion chassis or RIO mezzanine cards to your project.

Adding an R Series Expansion Chassis

1. Right-click **My Computer** in the **Project Explorer** window and select **New»Targets and Devices** from the shortcut menu to display the Add Targets and Devices dialog box.
2. Select the appropriate R Series device and click the **OK** button. LabVIEW adds an FPGA target item for the R Series device to the project.
3. Right-click the **FPGA Target** for the R Series device in the **Project Explorer** window and select **New»R Series Expansion Chassis** from the shortcut menu to display the **New R Series Expansion Chassis** dialog box.
4. Select the connector to which the R Series Expansion chassis is connected in the **Location** pull-down menu.
5. Check the **Discover C Series modules** checkbox and click the **OK** button to add the items to the project.
6. Right-click a module in the **Project Explorer** window and select **Properties** from the shortcut menu to configure module-specific settings.Enter your first step here. LabVIEW adds the following target to your project.

   Table 10. Project/Target Icon

Item Icon	Device	Description
	R Series Expansion Chassis or RIO Mezzanine Card	LabVIEW adds this target/item to a project for a R Series expansion chassis and RIO Mezzanine cards.

   Enter the result of your step here (optional).

Enter an example that illustrates the current task (optional).
Enter the tasks the user should do after finishing this task (optional).

Adding a RIO Mezzanine Card

1. Right-click the **FPGA Target** in the **Project Explorer** window and select **New»RIO Mezzanine Card** from the shortcut menu
   The **New RIO Mezzanine Card** dialog box opens.
2. Select the appropriate RIO mezzanine card from the **Type** pull-down menu and click the **OK** button.
   LabVIEW adds a RIO mezzanine card item and FPGA I/O items for I/O to the project.
3. To configure RIO mezzanine card specific settings, right-click the RIO mezzanine card item in the **Project Explorer** window and select **Properties** from the shortcut menu.

> **Note** LabVIEW discovers any C Series modules connected to RIO mezzanine cards with C Series slots after you add the card to the project.

LabVIEW adds the following target to your project.

Table 11. Project/Target Icon

Item Icon	Device	Description
	R Series Expansion Chassis or RIO Mezzanine Card	LabVIEW adds this target/item to a project for a R Series expansion chassis and RIO Mezzanine cards.

Configuring a Project with Offline Hardware

To configure a project with C Series modules when you do not have hardware installed, perform these procedures.

Adding a Controller Target

A controller target can be a CompactRIO controller, Ethernet RIO chassis, Single-Board RIO device, myRIO device, roboRIO device, or NI ELVIS RIO Control Module (NI ELVIS RIO CM) device.

Complete the following steps to add a controller target to your project.

1. Create a new project or open an existing project.
2. Right-click the project root in the **Project Explorer** window and select **New»Targets and Devices** from the shortcut menu to display the **Add Targets and Devices** dialog box.
3. Click the **New target or device** radio button.
4. Select the appropriate target or device.
5. Click the **OK** button.
   LabVIEW adds the controller target icon to the project.

Table 12. LabVIEW Controller Target Icons

Target Icon	Device	Description
	CompactRIO controller	LabVIEW adds this controller target when you configure a project with a CompactRIO real-time controller or CompactRIO integrated controller and chassis.
	Ethernet RIO chassis	LabVIEW adds this target when you configure a project with an Ethernet RIO chassis. The Ethernet RIO chassis is not a real-time target, but it can add expansion I/O to a real-time target.
	Single-Board RIO device	LabVIEW adds this controller target when you configure a project with a Single-Board RIO device.
	myRIO device	LabVIEW adds this controller target when you configure a project with a myRIO-1900.
	myRIO device	LabVIEW adds this controller target when you configure a project with a myRIO-1950.
	roboRIO device	LabVIEW adds this controller target when you configure a project with a roboRIO.
	NI ELVIS RIO Control Module (NI ELVIS RIO CM) device	LabVIEW adds this controller target when you configure a project with an NI ELVIS RIO CM.

Adding a Chassis Item

Add a CompactRIO, Ethernet RIO, MXIe-RIO, Single-Board RIO, myRIO, roboRIO, or NI ELVIS RIO Control Module (NI ELVIS RIO CM) chassis item to your project.

Complete the following steps to add a chassis item to your project.

1. Right-click the target in the **Project Explorer** window and select **New»Targets and Devices** from the shortcut menu to display the **Add Targets and Devices** dialog box.
2. Click the **New target or device** radio button.
3. Select the appropriate device.
4. Click the **OK** button.

> **Note** A chassis item is automatically added to your project for cRIO integrated controller and chassis and sbRIO devices.

LabVIEW adds the chassis item to the project.

Table 13. LabVIEW Chassis Item Icons

Item Icon	Device	Description
	CompactRIO chassis, Ethernet RIO chassis	LabVIEW adds this chassis item when you configure a project with a CompactRIO reconfigurable chassis, CompactRIO integrated chassis and controller, or Ethernet RIO chassis.
	MXIe-RIO chassis	LabVIEW adds this chassis item when you configure a project with a MXIe-RIO chassis. You can add a MXIe-RIO chassis to a real-time target or the host PC.
	Single-Board RIO, myRIO, roboRIO, or NI ELVIS RIO CM device	LabVIEW adds this chassis item when you configure a project with a Single-Board RIO, myRIO, roboRIO, or NI ELVIS RIO CM device.

Adding an R Series Expansion Chassis Item

You can add an R Series expansion chassis to the *FPGA target* for an R Series device in your project.

Enter the prerequisites here (optional).
Enter the context of your task here (optional).

1. Right-click the **FPGA Target** for the R Series device in the **Project Explorer** window and select **New»R Series Expansion Chassis** from the shortcut menu to display the

New R Series Expansion Chassis dialog box.
2. Click the **OK** button. LabVIEW adds a chassis item to the project.

1. Right-click the **FPGA Target** for the R Series device in the **Project Explorer** window and select **New»R Series Expansion Chassis** from the shortcut menu to display the **New R Series Expansion Chassis** dialog box.
Enter the result of your step here (optional).
2. Click the **OK** button.
LabVIEW adds a chassis item to the project.

Table 14. LabVIEW R Series Chassis Item Icon

Item Icon	Device	Description
	R Series Expansion chassis	LabVIEW adds this chassis item when you configure a project with an R Series expansion chassis.

Related tasks:

- Adding an FPGA Target

Adding an FPGA Target

You do not need to add an FPGA target to your project when programming in Scan Interface Mode, Real-Time Mode, or Real-Time Scan Mode.

Complete the following steps to add an FPGA target to your project.



Note *Real-Time Mode* and *Real-Time Scan Mode* are only available on cRIO-904x and cRIO-905x controllers.

1. Right-click the chassis item in the **Project Explorer** window and select **New»FPGA Target** from the shortcut menu.
2. If the **Deploy CompactRIO Chassis Settings?** dialog box appears, click the **Deploy Later** button to return to the project.
LabVIEW adds an FPGA target item for the chassis to the project and puts the system into **LabVIEW FPGA Interface mode**.

Table 15. LabVIEW FPGA Target Icon

Target Icon	Device	Description
	FPGA	LabVIEW adds this FPGA target when you configure a project with a CompactRIO chassis, Ethernet RIO chassis, MXIe-RIO chassis, or Single-Board RIO device in the FGPA Interface.

Related concepts:

- Programming Interfaces
- Programming Modes

Related tasks:

- Adding a Controller Target for the cRIO-904x/cRIO-905x

Adding I/O Items

An I/O item can be a RIO mezzanine card or a C Series module. RIO mezzanine cards have I/O or slots for C Series modules.

RIO Mezzanine Cards

Complete the following steps to add an RIO mezzanine card in **FPGA Interface Mode** to a project.

1. Right-click the **FPGA Target** in the **Project Explorer** window and select **New»RIO Mezzanine Card** from the shortcut menu to display the **New RIO Mezzanine Card** dialog box.
2. Select the appropriate RIO mezzanine card from the **Type** pull-down menu and click the **OK** button. LabVIEW adds an RIO mezzanine card item and FPGA I/O items for I/O to the project.
3. Right-click the RIO mezzanine card item in the **Project Explorer** window and select **Properties** from the shortcut menu to configure RIO mezzanine card specific settings or select **New»C Series Modules** to add C Series modules.
   LabVIEW adds the mezzanine card icon to the project.

Table 16. LabVIEW RIO Mezzanine Card Icon

Item Icon	Description
	LabVIEW adds this icon when you configure a project with a RIO mezzanine card.

Related concepts:

- Programming Interfaces

C Series Modules

Enter a short description of your task here (optional).

Complete the following steps to add modules in FPGA Interface Mode, Scan Interface Mode or Real-Time Mode to a project.



Note Real-Time Mode is only available on cRIO-904x and cRIO-905x controllers.

1. Right-click one of the resource items from the following table in the **Project Explorer** window and select **New»C Series Modules** from the shortcut menu to display the **Add Targets and Devices** dialog box.

For This Mode	Right-Click This Resource Item
FPGA Interface Mode	FPGA Target
Scan Interface Mode	Real-Time Scan Resources Item
Real-Time Mode	Real-Time Resources Item

2. Click the **New target or device** radio button, select **C Series Module**, and click the **OK** button to display the **New C Series Module** dialog box.
3. Select the appropriate C Series module from the **Module Type** pull-down menu and click the **OK** button. LabVIEW adds a module item and FPGA I/O items for the module I/O to the project.
4. Repeat steps 1 through 3 to add additional C Series modules to the project.
5. Right-click a module item in the **Project Explorer** window and select **Properties** from the shortcut menu to configure module-specific settings.

LabVIEW adds a C Series module icon to the project.

Table 17. LabVIEW C Series Module Icons

Item Icon	Device	Description
	C Series Modules	LabVIEW adds this module item when you configure a project with C Series modules.
	Onboard C Series Modules	LabVIEW adds this module item when you configure a project with a Single-Board RIO device that has onboard modules. LabVIEW adds onboard modules to the project in Scan Interface Mode. You can drag the onboard module item to FPGA target to place the module in the FPGA interface mode.

Related concepts:

- Programming Interfaces
- Programming Modes
- C Series Module Reference

Configuring a Project with DAQmx (cRIO-904x and cRIO-905x)

CompactRIO with DAQmx is only supported on cRIO-904x and cRIO-905x controllers.

Adding a Controller Target for the cRIO-904x/cRIO-905x

Complete the following steps to add a controller target for the cRIO-904x or cRIO-905x to a project.

1. Create a new project in LabVIEW or open an existing project.
2. Right-click the project root in the **Project Explorer** window and select **New»Targets and Devices** from the shortcut menu.
   The **Add Targets and Devices** dialog box opens.
3. Enable the applicable checkbox:
   - Click the **Existing target or device** radio button if your hardware is *connected*.
   - Click the **New target or device** radio button if your hardware is *offline*.
4. Select your cRIO-904x or cRIO-905x controller from the **Real-Time CompactRIO** folder.
5. Click the **OK** button.
   LabVIEW adds the target item to the project.

Related tasks:

- Adding an FPGA Target

Configuring Devices

Configure I/O functions on installed devices and chassis hardware.

Configuring Specialty Digital Functions

You can configure some C Series digital input and output modules for specialty digital functions, including:

- counter input
- counter-driven output
- pulse-width modulation output
- quadrature input

Two methods are available to configure specialty digital functions on C Series DIO modules:

- Configure specialty digital functions with the *Specialty Digital Configuration* page of the C Series Module Properties dialog box.
- Configure specialty digital functions by wiring a reference to a specialty digital module to a generic-class property node in a VI block diagram.

Witing a reference specifies the class of the property node as `SD Module` and provides programmatic access to all the specialty digital functions that the module supports.

To configure specialty digital functions for one or two modules, program the chassis for `Scan Interface` mode. To configure more than two modules for specialty digital functions, program the chassis to use `FPGA Interface` mode.

The maximum update rate for modules configured for specialty digital functions is approximately 4 MHz.

Related concepts:

- Using the Scan Interface with Individual Modules

Configuring Counters

You can configure channels of some C Series digital input modules as counters. Counter channels measure digital signals. Counters are used commonly to count edges. Counters are also used for time measurements, such as measuring the frequency or period of a signal.

Input Filter

You can add an input filter to all module input channels. A filter removes noise, glitches, and spikes on inputs by rejecting signals with periods shorter than the specified length. The following filter options are available:

- **Disabled**—The hardware synchronizes the digital inputs using a 4 MHz timebase. Pulses longer than 250 ns are always detected. Pulses shorter than 250 ns may be rejected.
- **1 µs**—Pulses shorter than 1 µs are rejected. Pulses between 1 µs and 2 µs are sometimes detected. Pulses longer than 2 µs are always detected.
- **16 µs**—Pulses shorter than 16 µs are rejected. Pulses between 16 µs and 32 µs are sometimes detected. Pulses longer than 32 µs are always detected.
- **256 µs**—Pulses shorter than 256 µs are rejected. Pulses between 256 µs and 512 µs are sometimes detected. Pulses longer than 512 µs are always detected.
- **4096 µs**—Pulses shorter than 4096 µs are rejected. Pulses between 4096 µs and 8192 µs are sometimes detected. Pulses longer than 8192 µs are always detected.

Counter Measurement Modes

You can select one of four counter measurement modes for each counter channel:

- Count Edges
- Period Measurement
- Pulse-Width Measurement
- Frequency Measurement

Configuring a Counter to Count Edges

You can configure a counter to count signal edges and perform an action on another channel when the count reaches a specified number called the *terminal count*. If you make any configuration changes, the counter resets.

You can configure the following options for edge counting.

Input Filter

You can add an input filter to all input channels of the module. A filter removes noise, glitches, and spikes on inputs by rejecting signals with periods shorter than the specified length. The following filter options are available:

- **Disabled**—The hardware synchronizes the digital inputs using a 4 MHz timebase. Pulses longer than 250 ns are always detected. Pulses shorter than 250 ns may be rejected.
- **1 µs**—Pulses shorter than 1 µsare rejected. Pulses between 1 µs and 2 µs are sometimes detected. Pulses longer than 2 µs are always detected.
- **16 µs**—Pulses shorter than 16 µs are rejected. Pulses between 16 µs and 32 µs are sometimes detected. Pulses longer than 32 µs are always detected.
- **256 µs**—Pulses shorter than 256 µs are rejected. Pulses between 256 µs and 512 µs are sometimes detected. Pulses longer than 512 µs are always detected.
- **4096 µs**—Pulses shorter than 4096 µs are rejected. Pulses between 4096 µs and 8192 µs are sometimes detected. Pulses longer than 8192 µs are always detected.

Counter Event

- **Rising Edge**—The counter counts rising edges.
- **Falling Edge**—The counter counts falling edges.
- **Any Edge**—The counter counts all edges.

Counter Source

- **This Channel**—The counter counts edges on an input signal connected to this channel.
- **Terminal Count of Previous Channel**—The counter increments when the previous channel reaches the terminal count. You can use this option to create counters

with terminal counts larger than 4,294,967,296.

Count Direction

- **Count Up**—The counter starts at zero, counts up to the specified terminal count, resets, and starts counting up again from zero.
- **Count Down**—The counter starts at the specified terminal count, counts down to zero, resets, and starts counting down again from the terminal count.

Terminal Count

You can specify a terminal count up to a maximum of 2^{32} = 4,294,967,296. When the counter reaches the terminal count, it resets and starts counting again from zero.

Terminal Count Output Mode

You can select one of the following four actions to perform on a digital output channel on another module when the counter reaches the specified terminal count.

- **Toggle, Reset Off**—The counter-driven channel starts in the off state and changes state each time the counter channel reaches the terminal count.
- **Toggle, Reset On**—The counter-driven channel starts in the on state and changes state each time the counter channel reaches the terminal count.
- **On Pulse**—The counter-driven channel turns on each time the counter channel reaches the terminal count and remains on until the next counter increment.
- **Off Pulse**—The counter-driven channel turns off each time the counter channel reaches the terminal count and remains off until the next counter increment.



Note In order for the counter to drive an output channel, you must configure the module that the output channel is on for *counter-driven output*. The counter-driven output channel has the same number as the counter channel that drives it. For example, CTR3 drives DO3 on an eight-channel digital output module in another slot in the same chassis.

Gate Mode

- **Always Enabled**—The counter increments on each edge of the type selected with the **Counter Event** control.

- **Next Channel**—The counter increments only on edges that occur when the next higher channel is in the on state.

Configuring a Counter to Measure Period

You can configure a digital input channel as a counter to measure the period of a signal connected to the channel. The counter returns data in µs.

You can configure the following options for a period measurement counter.

Input Filter

You can add an input filter to all input channels of the module. A filter removes noise, glitches, and spikes on inputs by rejecting signals with periods shorter than the specified length. You can select one of the following options.

- **Disabled**—The hardware synchronizes the digital inputs using a 4 MHz timebase. Pulses longer than 250 ns are always detected. Pulses shorter than 250 ns may be rejected.
- **1 µs**—Pulses shorter than 1 µs are rejected. Pulses between 1 µs and 2 µs are sometimes detected. Pulses longer than 2 µs are always detected.
- **16 µs**—Pulses shorter than 16 µs are rejected. Pulses between 16 µs and 32 µs are sometimes detected. Pulses longer than 32 µs are always detected.
- **256 µs**—Pulses shorter than 256 µs are rejected. Pulses between 256 µs and 512 µs are sometimes detected. Pulses longer than 512 µs are always detected.
- **4096 µs**—Pulses shorter than 4096 µs are rejected. Pulses between 4096 µs and 8192 µs are sometimes detected. Pulses longer than 8192 µs are always detected.

Measurement Edge

- **Rising Edge**—The counter measures the time from one rising edge to the next rising edge.
- **Falling Edge**—The counter measures the time from one falling edge to the next falling edge.

Configuring a Counter to Measure Pulse Width

You can configure a digital input channel as a counter to measure the pulse width of a signal connected to the channel. The counter returns data in µs. You can configure the following options for a pulse-width measurement counter.

Input Filter

You can add an input filter to all input channels of the module. A filter removes noise, glitches, and spikes on inputs by rejecting signals with periods shorter than the specified length. You can select one of the following options.

- **Disabled**—The hardware synchronizes the digital inputs using a 4 MHz timebase. Pulses longer than 250 ns are always detected. Pulses shorter than 250 ns may be rejected.
- **1 µs**—Pulses shorter than 1 µs are rejected. Pulses between 1 µs and 2 µs are sometimes detected. Pulses longer than 2 µs are always detected.
- **16 µs**—Pulses shorter than 16 µs are rejected. Pulses between 16 µs and 32 µs are sometimes detected. Pulses longer than 32 µs are always detected.
- **256 µs**—Pulses shorter than 256 µs are rejected. Pulses between 256 µs and 512 µs are sometimes detected. Pulses longer than 512 µs are always detected.
- **4096 µs**—Pulses shorter than 4096 µs are rejected. Pulses between 4096 µs and 8192 µs are sometimes detected. Pulses longer than 8192 µs are always detected.

Measurement Type

- **High Pulse**—The counter returns the time that the connected signal is high.
- **Low Pulse**—The counter returns the time that the connected signal is low.
- **Most Recent**—The counter returns the lengths of the most recent high and low pulses.

Configuring a Counter to Measure Frequency

You can configure a counter to measure the frequency of a signal connected to the channel. The counter counts the number of rising or falling signal edges during the

length of the Frequency Timebase, divides that number by the Frequency Timebase in µs, and returns the frequency in kHz.

You can configure the following options for a frequency measurement counter.

Input Filter

You can add an input filter to all input channels of the module. A filter removes noise, glitches, and spikes on inputs by rejecting signals with periods shorter than the specified length. You can select one of the following options.

- **Disabled**—The hardware synchronizes the digital inputs using a 4 MHz timebase. Pulses longer than 250 ns are always detected. Pulses shorter than 250 ns may be rejected.
- **1 µs**—Pulses shorter than 1 µs are rejected. Pulses between 1 µs and 2 µs are sometimes detected. Pulses longer than 2 µs are always detected.
- **16 µs**—Pulses shorter than 16 µs are rejected. Pulses between 16 µs and 32 µs are sometimes detected. Pulses longer than 32 µs are always detected.
- **256 µs**—Pulses shorter than 256 µs are rejected. Pulses between 256 µs and 512 µs are sometimes detected. Pulses longer than 512 µs are always detected.
- **4096 µs**—Pulses shorter than 4096 µs are rejected. Pulses between 4096 µs and 8192 µs are sometimes detected. Pulses longer than 8192 µs are always detected.

Measurement Edge

- **Rising**—The counter counts rising edges.
- **Falling**—The counter counts falling edges.

Frequency Timebase

Selecting a **Frequency Timebase** is a tradeoff between resolution and response speed. A shorter timebase provides lower resolution but the data it returns is more up to date. A longer timebase provides higher resolution but may lag behind or miss changes in signal frequency. The guidelines below for selecting a timebase for different frequency ranges will result in a maximum error of approximately 1%. If you require higher resolution and can tolerate less frequent updates, use a longer timebase than indicated.

- **256 µs**—The counter counts the edges of the selected type that occur in 256 µs and

returns the frequency of the signal. Use this setting for frequencies of 500 kHz or higher.

- **512 µs**—The counter counts the edges of the selected type that occur in 512 µs and returns the frequency of the signal. Use this setting for frequencies of 250 kHz or higher.
- **1024 µs**—The counter counts the edges of the selected type that occur in 1024 µs and returns the frequency of the signal. Use this setting for frequencies of 125 kHz or higher.
- **2048 µs**—The counter counts the edges of the selected type that occur in 2048 µs and returns the frequency of the signal. Use this setting for frequencies of 62.5 kHz or higher.
- **4096 µs**—The counter counts the edges of the selected type that occur in 4096 µs and returns the frequency of the signal. Use this setting for frequencies of 31.3 kHz or higher.
- **8192 µs**—The counter counts the edges of the selected type that occur in 8192 µs and returns the frequency of the signal. Use this setting for frequencies of 15.6 kHz or higher.
- **16384 µs**—The counter counts the edges of the selected type that occur in 16384 µs and returns the frequency of the signal. Use this setting for frequencies of 7.8 kHz or higher.
- **32768 µs**—The counter counts the edges of the selected type that occur in 32768 µs and returns the frequency of the signal. Use this setting for frequencies of 3.9 kHz or higher.

Note For frequencies lower than 3.9 kHz, configure a counter to *measure period* and calculate the frequency using the inverse of the measurement.

Frequency Measurement Example

Refer to the Frequency Measurement (Host) VI in the `labview\examples\CompactRIO\NI Scan Engine\Advanced\Specialty Digital Configuration - Counter Mode\Specialty Digital Configuration - Counter Mode.lvproj` for an example of frequency measurement. The example uses an *NI 9401*, but you can modify it to use a different digital input module.

Related concepts:

- NI 9401 (Scan Interface)

Configuring Counter-Driven Outputs

Counter-driven output channels respond in a predetermined way when a counter channel on a digital input module reaches the terminal count. You must configure a digital input channel on another module for *edge counting* and select an action for the Terminal Count Output Mode. Then you select the digital output channel with the same channel number and configure it as a counter-driven output. You can configure the following options for a counter-driven output.

Drive from Slot

Use this control to select a slot with a digital input module that has at least one channel configured to count edges. Refer to the **Project Explorer** window and select a slot that has a channel called CTRx, where x is the number of the selected output channel.

Counter Driven

Use this control to enable or disable counter-driven output for the selected output channel. If you select **Enabled**, the counter selected with the **Drive from Slot** control drives the selected output channel. If counter-driven output is enabled for the selected channel, LabVIEW removes the I/O variable for the selected channel from the **Project Explorer** window. If an I/O variable for the selected channel is used in a block diagram of a VI, the VI will not run.

Configuring a Channel for Pulse-Width Modulation

You can configure a digital output channel to output a signal with a set frequency and period. Use a control on the front panel of a VI to control the on time (also called the active time or duty cycle) of the pulse. Pulse-Width Modulation Example

Refer to the Pulse Width Modulation (Host) VI in the `labview\examples\CompactRIO\NI Scan Engine\Advanced\Specialty Digital Configuration - PWM\Specialty Digital Configuration - Pulse-Width Modulation Mode.lvproj` for an example of pulse-width modulation. The example uses an *NI 9474*, but you can modify it to use a different

digital output module.

Related concepts:

- NI 9474 (Scan Interface)

Configuring a Module for Quadrature Input

You can configure a digital input module for quadrature input.

A quadrature input returns position and velocity data based on the values of three channels. In quadrature input mode, channels 0, 1, and 2 are the A, B, and Index channels, respectively, of Quadrature0.

If the module has six or more channels, channels 3, 4, and 5 can form a second quadrature input, Quadrature1. A C Series module can have only two quadrature inputs. Quadrature inputs return position data in counts and velocity in counts per second.

You can configure the following options for quadrature input.

Input Filter

You can add an input filter to all input channels of the module. A filter removes noise, glitches, and spikes on inputs by rejecting signals with periods shorter than the specified length. You can select one of the following options.

- **Disabled**—The hardware synchronizes the digital inputs using a 4 MHz timebase. Pulses longer than 250 ns are always detected. Pulses shorter than 250 ns may be rejected.
- **1 µs**—Pulses shorter than 1 µs are rejected. Pulses between 1 µs and 2 µs are sometimes detected. Pulses longer than 2 µs are always detected.
- **16 µs**—Pulses shorter than 16 µs are rejected. Pulses between 16 µs and 32 µs are sometimes detected. Pulses longer than 32 µs are always detected.
- **256 µs**—Pulses shorter than 256 µs are rejected. Pulses between 256 µs and 512 µs are sometimes detected. Pulses longer than 512 µs are always detected.
- **4096 µs**—Pulses shorter than 4096 µs are rejected. Pulses between 4096 µs and 8192 µs are sometimes detected. Pulses longer than 8192 µs are always detected.

Velocity Timebase

Selecting a **Velocity Timebase** is a tradeoff between resolution and response speed. A shorter timebase provides lower resolution but the data it returns is more up to date. A longer timebase provides higher resolution but may lag behind or miss changes in velocity.

- **256 µs**—The counter counts the edges of the selected type that occur in 256 µs and returns the frequency of the signal. Use this setting for frequencies of 500 kHz or higher.
- **512 µs**—The counter counts the edges of the selected type that occur in 512 µs and returns the frequency of the signal. Use this setting for frequencies of 250 kHz or higher.
- **1024 µs**—The counter counts the edges of the selected type that occur in 1024 µs and returns the frequency of the signal. Use this setting for frequencies of 125 kHz or higher.
- **2048 µs**—The counter counts the edges of the selected type that occur in 2048 µs and returns the frequency of the signal. Use this setting for frequencies of 62.5 kHz or higher.
- **4096 µs**—The counter counts the edges of the selected type that occur in 4096 µs and returns the frequency of the signal. Use this setting for frequencies of 31.3 kHz or higher.
- **8192 µs**—The counter counts the edges of the selected type that occur in 8192 µs and returns the frequency of the signal. Use this setting for frequencies of 15.6 kHz or higher.
- **16384 µs**—The counter counts the edges of the selected type that occur in 16384 µs and returns the frequency of the signal. Use this setting for frequencies of 7.8 kHz or higher.
- **32768 µs**—The counter counts the edges of the selected type that occur in 32768 µs and returns the frequency of the signal. Use this setting for frequencies of 3.9 kHz or higher.

Index Mode

If you enable an Index Mode, the position is reset to zero when A and B have the specified values and the Index is in the on state.

- **Disabled**—The position is never reset to zero.

- **Phase B0 A0**—The position is reset to zero when B and A are both in the off state and the Index is in the on state.
- **Phase B0 A1**—The position is reset to zero when B is in the off state and A and the Index are both in the on state.
- **Phase B1 A0**—The position is reset to zero when B is in the on state, A is in the off state, and the Index is in the on state.
- **Phase B1 A1**—The position is reset to zero when A, B, and the Index are all in the on state.

Configuring the Chassis I/O (FPGA Interface)

Related concepts:

- Configuring a Project with Connected Hardware

Enabling Sleep Mode on a CompactRIO Chassis, Ethernet RIO Chassis, MXIe-RIO Chassis, or sbRIO Device

Installed and onboard C Series I/O modules in the chassis or device enter a low-power state when you enable sleep mode. Communicating with a module in sleep mode typically returns a module communication error from the FPGA I/O function. Sleep mode does not affect the chassis or controller.

Complete the following steps to enable sleep mode.

1. Configure a project for the system.
2. Make sure the **Sleep** I/O item is added to the system.
3. Right-click the FPGA target and select **New»VI** from the shortcut menu to add a new VI to the system.
4. Place an FPGA I/O Node on the block diagram of the VI.
5. Click the element section of the FPGA I/O Node and select **Chassis I/O»Sleep** or **Onboard I/O»Sleep**.
6. Right-click the **Sleep** input of the FPGA I/O Node and select **Create»Control**. When **Sleep** is TRUE, the system enters sleep mode. When **Sleep** is FALSE, the system enters active mode. The default state of the system is active mode.

Confirming the State of the Sleep Channel

Complete the following steps to read the state of the Sleep channel.

1. Place another FPGA I/O Node function on the block diagram. This FPGA I/O Node must execute after the FPGA I/O Node you added above.
2. Click the element section of the FPGA I/O Node and select **Chassis I/O»Sleep** or **Onboard I/O»Sleep**.
3. Right-click the FPGA I/O Node and select **Change to Read**.
4. Right-click the **Sleep** output of the FPGA I/O Node and select **Create»Indicator**.

Programmatically Resetting the System on a CompactRIO Chassis or sbRIO Device

Use a digital output line to programmatically reset the system.

1. Configure a project for the system.
2. Make sure the **System Reset** I/O item is added to the system.
3. Right-click the FPGA target in the **Project Explorer** window and select **New»VI** from the shortcut menu to add a new FPGA VI to the system.
4. Place an FPGA I/O Node on the block diagram of the FPGA VI.
5. Click the element section of the FPGA I/O Node and select **Chassis I/O (or Onboard I/O)»System Reset**.
6. Right-click the **System Reset** input of the FPGA I/O Node and select **Create»Control**. When **System Reset** is TRUE, the system resets.
   The system FPGA deasserts the **System Reset** line during the system reset. You do not need to wire a FALSE Boolean value to the **System Reset** input or ensure that the **System Reset** input is TRUE for a minimum amount of time.

> **Caution** If the FPGA VI uses the **System Reset** output and you configure the system to load the FPGA VI when there is a software or hardware reset, make sure that the system is not set up in a way in which the system continually resets before you run the FPGA VI. If the system is in a state in which it continually resets, contact NI support.

Programming the FPGA LED

Programming the FPGA LED on a CompactRIO Chassis or sbRIO Device

You can use the FPGA LED to help debug your application or easily retrieve application status. Complete the following steps to configure the FPGA LED.

1. Configure a project for the MXIe-RIO system.
2. Make sure the **FPGA LED** I/O item is added to the MXIe-RIO system.
3. Right-click the FPGA target in the **Project Explorer** window and select **New»VI**to add a new VI to the system.
4. Place an FPGA I/O Node on the block diagram of the VI.
5. Click the element section of the FPGA I/O Node and select **Chassis I/O (or Onboard I/O)»FPGA LED**.
6. Right-click the **FPGA LED** input of the FPGA I/O Node and select **Create»Control**. When **FPGA LED** is TRUE, the FPGA LED is lit. When **FPGA LED** is FALSE, the FPGA LED is off.

Refer to the 915x Switches and LEDs VI in the `labview\examples\CompactRIO\Chassis Specific\915x Switches and LEDs\915x Switches and LEDs.lvproj` for an example of programming the USER FPGA LEDs.

Programming the FPGA LED on a MXIe-RIO Chassis

The MXIe-RIO reconfigurable embedded chassis has bi-color FPGA LEDs. Use the FPGA LEDs to help debug your application or to retrieve the application status. Complete the following steps to configure the FPGA LEDs.

1. Configure a project for the MXIe-RIO system.
2. Make sure the **USER FPGA LED** I/O item is added to the MXIe-RIO system.
3. Right-click the FPGA target in the **Project Explorer** window and select **New»VI**to add a new VI to the MXIe-RIO system.
4. Place an FPGA I/O Node on the block diagram of the VI.
5. Click the element section of the FPGA I/O Node and select **Chassis I/O»USER FPGA1 LED**. USER FGPA2 LED and USER FGPA3 LED function the same as USER FPGA1 LED.
6. Right-click the **USER FPGA1 LED** input of the FPGA I/O Node and select **Create»Control**. When **USER FPGA1 LED** is 0, the USER FPGA1 LED is off. When **USER FPGA1 LED** is 1, the USER FPGA1 LED is lit green. When **USER FPGA1 LED** is 2, the USER FPGA1 LED is lit yellow.

> **Note** Values other than 0, 1, and 2 are ignored and return an error. You must wire the **Error Out** terminal of the FPGA I/O Node to receive the error notification.

Refer to the 915x Switches and LEDs VI in the `labview\examples\CompactRIO\Chassis Specific\915x Switches and LEDs\915x Switches and LEDs.lvproj` for an example of programming the USER FPGA LEDs.

Reading the Temperature of a CompactRIO Chassis, Ethernet RIO Chassis, MXIe-RIO Chassis, or sbRIO Device

Complete the following steps to read the temperature of the chassis or device using the onboard temperature sensor.

1. Configure a project for the system.
2. Make sure the **Chassis Temperature** I/O item is added to the system.
3. Right-click the FPGA target in the **Project Explorer** window and select **New»VI** from the shortcut menu to add a new VI to the system.
4. Place an FPGA I/O Node on the block diagram of the VI.
5. Click the element section of the FPGA I/O Node and select **Chassis I/O (or Onboard I/O)»Chassis Temperature**.
6. Right-click the **Chassis Temperature** output of the FPGA I/O Node and select **Create»Indicator**.
   The **Chassis Temperature** output returns 16-bit values for the temperature. You can convert this value into meaningful engineering units.

Related concepts:

- Converting the Temperature of a CompactRIO Chassis, Ethernet RIO Chassis, MXIe-RIO Chassis, or a Single-Board RIO Device (FPGA Interface)

Building and Troubleshooting a CompactRIO Embedded Application

CompactRIO Interface Methods (FPGA Interface)

In addition to the methods described in the LabVIEW API topic *Invoke Method*, you can use the Read TEDS method with the LabVIEW API Invoke Method function for CompactRIO devices.

Read TEDs Method

Read TEDS method reads TEDS[2] information from a C Series module that has TEDS support enabled.

- **Connector**—specifies the connector to which the NI 9151 R Series Expansion chassis is connected. This input is available only when the Open FPGA VI Reference function is configured for an R Series device. For the NI 781xR, valid values are 0 through 3.
- **Slot**—specifies the chassis slot of the C Series module from which you want to read TEDS information. If the slot does not have a C Series module that supports this feature, LabVIEW returns an error. Valid values are 1 through *N*, where *N* is the number of slots in the chassis.
- **Channel**—specifies the channel on the module from which you want to read TEDS information. If a sensor is not connected to the channel, the Invoke Method function returns an error. For the NI 9219, NI 9233, NI 9234, and NI 9237, valid values are 0 through 3 .
- **TEDS Binary**—represents the TEDS information of the sensor connected to the specified channel in an array of bits. Use the TEDS Toolkit to parse this data.
- **v0.9 (TEDS)**—determines whether the sensor stored in the TEDS is in the released IEEE 1451.4 standard format or in the legacy version (v0.9) format. v0.9 returns *TRUE* if the TEDS is in the legacy format. The TEDS Toolkit VIs require this output to properly parse information.

Related information:

- Invoke Method

2. Transducer Electronic Data Sheet. A TEDS contains the critical information needed by an instrument or measurement system to identify, characterize, interface, and properly use the signal from an analog sensor.

CompactRIO Reconfigurable Embedded Chassis (FPGA Interface)

A user-reconfigurable FPGA controls the digital and analog I/O lines on the CompactRIO reconfigurable embedded chassis. The FPGA chip is embedded within the CompactRIO chassis and is connected to the C Series I/O modules in a star topology, providing direct access to each module for precise control and flexibility in timing, triggering, and synchronization. You can *configure the CompactRIO system* and change the functionality of the FPGA on the CompactRIO chassis in LabVIEW using the LabVIEW FPGA Module and NI CompactRIO Device Drivers to create and download a custom VI to the FPGA.

Converting the Temperature of a CompactRIO Chassis, Ethernet RIO Chassis, MXIe-RIO Chassis, or a Single-Board RIO Device (FPGA Interface)

The FPGA I/O Node returns binary values for the chassis temperature. You can convert these values into meaningful engineering units. To ensure that the FPGA VI is as efficient as possible, convert the values in the host VI.

Using a VI to Convert Values

Refer to the RIO Chassis Temperature Monitor (FPGA) VI in the `labview\examples\CompactRIO\Chassis Specific\RIO Chassis Temperature Monitor\CRIO Chassis Temperature Monitor.lvproj` for an example of converting binary values. You can use the RIO Chassis Temperature Monitor (FPGA) VI as a subVI in the host VI to convert the values.

Using an Equation to Convert Values

Use the following equation in the host VI to convert binary values to temperature:

Temperature = Binary Value × 0.25 °C

where

- Binary Value is the value returned by the FPGA I/O Node.

Related concepts:

- Configuring the Chassis I/O (FPGA Interface)

Synchronizing FPGA VIs with the NI Scan Engine (FPGA Interface)

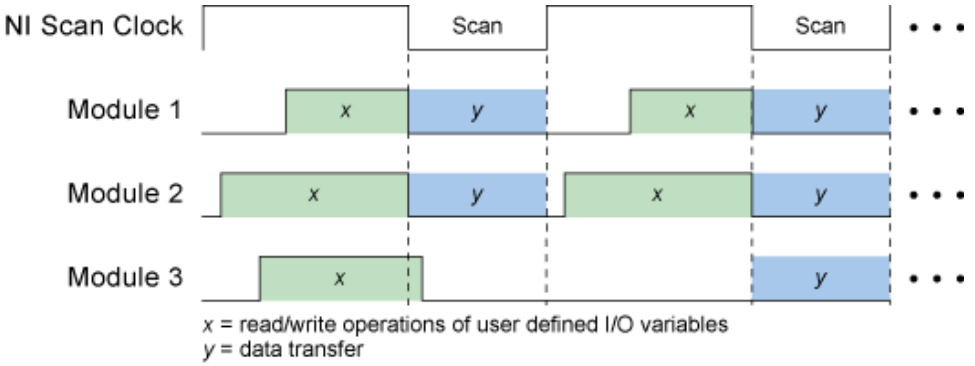
The CompactRIO reconfigurable embedded chassis, integrated controllers and chassis, Ethernet RIO chassis, and Single-Board RIO (sbRIO) devices have a Scan Clock I/O item. Use the Scan Clock item to monitor when the NI Scan Engine transfers data between the FPGA VI and the RT host VI.

Use the Scan Clock information with user-defined I/O variables to synchronize data transfers with the NI Scan Engine. Synchronizing user-defined I/O variable data transfers with the Scan Engine ensures the coherency of the data sets you transfer between the FPGA VI and the RT host VI.

NI recommends you design the application so that read/write operations occur when the Scan Engine is not transferring data between the FPGA VI and the RT host VI. If you read from or write to user-defined I/O variables while the Scan Engine is transferring data, the Scan Engine cannot guarantee data coherency, and the data transfer gets delayed to the next opportunity.

The following image illustrates how the Scan Engine reacts when you read from or write to user-defined I/O variables during the Scan Engine high time and when a user-defined I/O variable read or write operation misses the falling edge of the Scan Engine.

Figure 2. Scan Engine Reactions to User-Defined I/O Variable Read/Write Operations



You can also use the Scan Clock information to synchronize the execution of code in the FPGA VI with code in the RT host VI that is synchronized to the Scan Engine.

Use the following Scan Clock I/O item, method, and properties to monitor information from the Scan Engine.

Scan Clock I/O Item

Use the FPGA I/O Node to access the Scan Clock I/O item.

To access the Scan Clock I/O item, drag the Scan Clock I/O item from the **Chassis I/O** folder in the project to the block diagram of the FPGA VI. You can also place an FPGA I/O Node on the block diagram, click the element section of the FPGA I/O Node, and select **Chassis I/O»Scan Clock** from the shortcut menu.

- **Scan Clock**—returns TRUE when the Scan Clock signal is high and the Scan Engine is not transferring data between the FPGA VI and the RT host VI. **Scan Clock** returns FALSE when the Scan Clock signal is low and the Scan Engine is transferring data between the FPGA VI and the RT host VI.

Scan Clock Method: Wait on Rising Edge

Use the FPGA I/O Method Node to access the **Wait on Rising Edge** method for the Scan Clock I/O item.

The **Wait on Rising Edge** method pauses the execution of the FPGA I/O Method Node until the next rising edge of the Scan Clock signal or until the **Timeout** expires.

- **Timeout**—specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
- **Timed Out**—returns TRUE if a timeout occurred.
- **Missed Falling Edge**—returns TRUE if the method detects one or more falling edges of the Scan Clock signal after the last execution of the method and before the current execution. The first time the method executes, **Missed Falling Edge** returns FALSE. Use the **Missed Falling Edge** and **Missed Rising Edge** outputs to monitor if the FPGA VI is synchronized with the Scan Engine.

- **Missed Rising Edge**—returns TRUE if the method detects one or more rising edges of the Scan Clock signal after the last execution of the method and before the current execution. The first time the method executes, **Missed Rising Edge** returns FALSE. Use the **Missed Falling Edge** and **Missed Rising Edge** outputs to monitor if the FPGA VI is synchronized with the Scan Engine.
- **High Time**—displays the number of FPGA clock ticks that the Scan Clock signal was high during the last period of the signal. During the first period of the Scan Clock signal, **High Time** returns 0. The **High Time** output of the Wait on Rising Edge method performs the same as the High Time property accessible from the FPGA I/O Property Node.

Scan Clock Properties

Use the FPGA I/O Property Node to access the following properties for the Scan Clock I/O item.

- **Enabled**—returns TRUE when the Scan Engine is enabled. **Enabled** returns FALSE if the Scan Engine has not started, is reconfiguring, or has stopped
- **High Time**—displays the number of FPGA clock ticks that the Scan Clock signal was high during the last period of the signal. During the first period of the Scan Clock signal, **High Time** returns 0. The High Time property performs the same as the **High Time** output of the Wait on Rising Edge method.

Example VIs

Refer to the User-Defined IOV Synchronized VI in the `labview\examples\CompactRIO\NI Scan Engine\Advanced\Scan Engine Synchronization\Scan Engine Synchronization.lvproj` for an example of using the Scan Clock I/O item and the Wait for Rising Edge method with user-defined I/O variables.

Refer to the User-Defined IOV Advanced Sync VI in the `labview\examples\CompactRIO\NI Scan Engine\Advanced\Scan Engine Synchronization - Advanced\Scan Engine Synchronization - Advanced.lvproj` for an example of using the Scan Clock I/O item and the Wait for Rising Edge method, including the **High Time** output, with user-defined I/O variables.

Related concepts:

- Using the Scan Interface with Individual Modules

FPGA Interface C API

The FPGA Interface C API is a development tool used to create C/C++ applications that interact with LabVIEW FPGA VIs.

Refer to *Introduction to the FPGA Interface C API* for information about downloading and using the FPGA Interface C API.

> **Note** FPGA Interface C API software installed with NI CompactRIO Device Drivers versions earlier than 3.3 have been uninstalled.

Related information:

- Introduction to the FPGA Interface C API

Synchronizing Multiple C Series Modules

1. Configuring the Master Module
2. Configuring the Slave Modules
3. Acquiring Data from Synchronized C Series Modules
4. Synchronizing Multiple C Series Modules with NI 9469 Modules

Related concepts:

- Synchronizing Multiple Modules (FPGA Interface)
- Understanding the Maximum Sample Rate When Synchronizing Multiple Modules (FPGA Interface)

Configuring the Master Module

Complete the following steps to configure the master module.

1. Configure the system.
2. Right-click the module you want to configure as the master in the **Project Explorer**

window and select **Properties**. to display .
The **C Series Module Properties** dialog box opens.
3. Select **<Onboard Clock>** from the **Master Timebase Source** pull-down menu.
4. Enable the **Export Onboard Clock** checkbox.
5. Click the **OK** button.

Configuring the Slave Modules

Complete the following steps to configure the slave modules.

1. Right-click the module you want to configure as a slave in the **Project Explorer** window and select **Properties**.
The **C Series Module Properties** dialog box opens.
2. Select the name of the master module from the **Master Timebase Source** pull-down menu.
3. Repeat steps 1 and 2 for each slave module you want to configure.
4. Click the **OK** button.
5. Select **File»Save All** in the **Project Explorer** window.

Acquiring Data from Synchronized C Series Modules

Note the following when synchronizing multiple C Series modules:

- All modules must start acquisition mode at the same time.
- A single FPGA I/O Node function must read the synchronous data.

Block Diagram Objects used to Synchronize Multiple C Series Modules

Use the FPGA I/O Property Node, FPGA I/O Node, and FPGAMethod Node block diagram objects to synchronize multiple C Series modules.

Figure 3. FPGA I/O Property Node

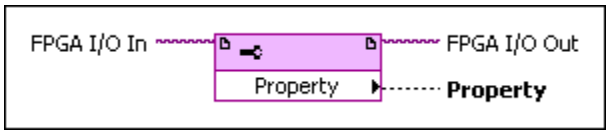


Figure 4. FPGA I/O Node

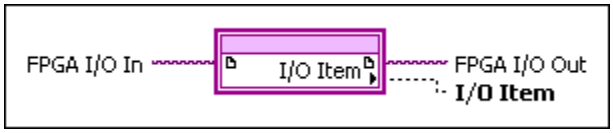
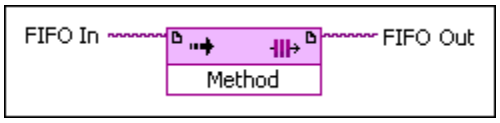


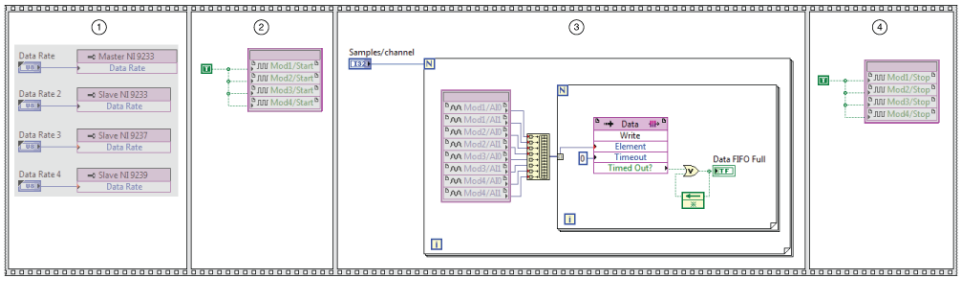
Figure 5. FIFO Method Node



Creating the Block Diagram

Reference the following image when creating a block diagram to synchronize multiple C Series modules. Customize the gray sections for your unique programming goals.

Figure 6. Block Diagram for Synchronizing Multiple C Series Modules



1. Configure FPGA I/O Property Nodes for the master module and slave modules and wire a control to each Node.
2. Configure an FPGA I/O Node with **Start** channels for the modules you want to synchronize and wire a Boolean constant set to TRUE to each **Start** channel.
3. Configure an FPGA I/O Node with all of the channels from which you want to synchronously sample.
4. Configure an FPGA I/O Node with **Stop** channels for the modules you want to synchronize and wire a Boolean constant set to TRUE to each **Stop** channel.

Tip You can acquire data from synchronized modules with different rates. If you synchronize modules that are configured for different data rates, create a separate loop for each data rate in the FPGA VI. In each loop, configure an FPGA I/O Node with all of the channels that are configured for the data rate of that loop. There is a delay before the FPGA I/O Node returns the first data point. The length of the delay depends on the data rate of the module.

Note Modules are internally timed. Do not use the **Loop Timer** or **Wait** functions in a loop with an FPGA I/O Node that acquires data from a module that supports synchronization.

Note The above block diagram example omits error handling to emphasize programming logic. Error handling enables debugging and controls data flow. **Always wire the error in terminal and the error out terminal**.

Examples

Refer to the Synchronizing NI 923x Modules (FPGA) VI in the `labview\examples\CompactRIO\Module Specific\NI 923x\Synchronizing NI 923x Modules\Synchronizing NI 923x Modules.lvproj` for an example of synchronizing multiple modules with the same data rate.

What happens if I change the master timebase source?

When you change the master timebase source, the **Data Rate** property controls or constants on the block diagram do not automatically update.

Note You might need to delete and recreate controls or constants for any **Data Rate** property on the block diagram after changing the master timebase source.

Synchronizing Multiple C Series Modules with NI 9469 Modules

To synchronize multiple C Series modules using NI 9469 modules, complete the following procedure.

Related concepts:

- Synchronizing Multiple Modules (FPGA Interface)
- Understanding the Maximum Sample Rate When Synchronizing Multiple Modules (FPGA Interface)

Configuring the Master NI 9469 Module

Complete the following steps to configure the master NI 9469 module.

1. Configure the system.
2. Right-click the NI 9469 module you want to configure as the master in the **Project Explorer** window and select **Properties**.
   The **C Series Module Properties** dialog box opens.
3. Select **Master** from the **Configuration** pull-down menu.
4. Enable the **Export Clock** checkbox.
5. Click the **OK** button.

Configuring the Slave NI 9469 Modules

Complete the following steps to configure the slave NI 9469 modules.

1. Right-click the NI 9469 module you want to configure as a slave in the **Project Explorer** window and select **Properties**.
   The **C Series Module Properties** dialog box opens.
2. Select **Slave** from the **Configuration** pull-down menu.
3. Place a checkmark in the **Import Clock** checkbox.
4. Click the **OK** button.
5. Repeat Steps 1 through 4 for each additional NI 9469 module you want to configure as a slave.
6. Select **File»Save All** in the **Project Explorer** window.

How do I set a custom configuration for NI 9469 modules

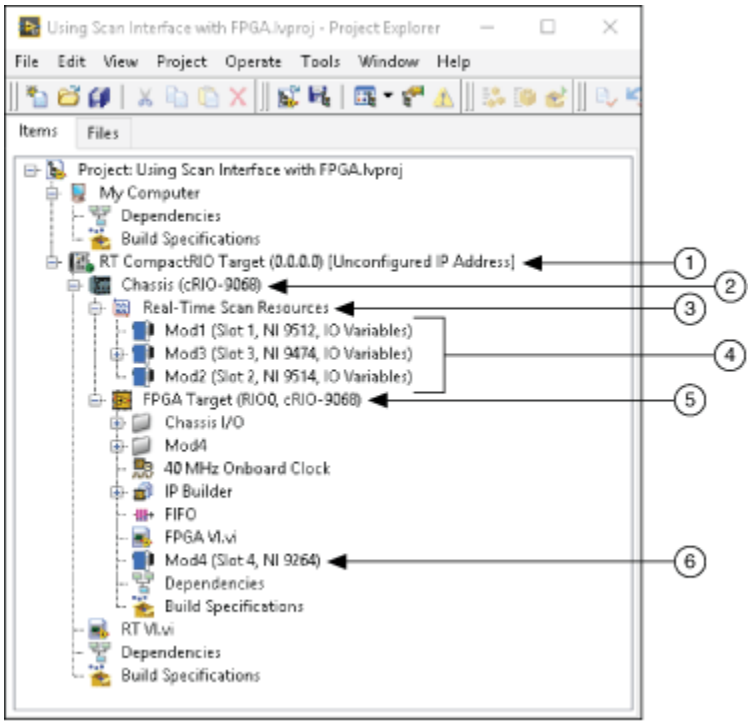
Perform the following procedure to set a custom configuration for NI 9469 modules.

1. Open the **Module Properties** dialog box
2. Select **Custom** from the **Configuration** pull-down menu.
3. Configure the lines and triggers for each port.

Using C Series Modules in a LabVIEW Project

To use C Series modules in a LabVIEW project, configure the project with your controller, chassis, and FPGA.

The following figure shows a project configured with a controller, chassis, FPGA target, and C Series modules in **Scan Interface** mode and **FPGA Interface** mode.



1. Target
2. Chassis Item
3. Real-Time Scan Resources Item
4. C Series Modules (Scan Interface)
5. FPGA Target
6. C Series Module (FPGA Interface)

For an example of a LabVIEW project using C Series I/O, refer to the *Using the Scan Interface with FPGA Interface LabVIEW* project in `labview\examples\CompactRIO\NI Scan Engine\Getting Started\Using Scan Interface with FPGA Interface\Using Scan Interface with FPGA Interface.lvproj`.

Target and Chassis Items

Table 18. Target Item Icons

Target Item Icon	Device
	CompactRIO Controller

Target Item Icon	Device
	Ethernet RIO Chassis
	CompactRIO Chassis Ethernet RIO Chassis

Table 19. Chassis Item Icons

Chassis Item Icon	Device
	CompactRIO Chassis, Ethernet RIO Chassis
	MXIe-RIO Chassis
	R Series Expansion Chassis Single-Board RIO Mezzanine Card
	Single-Board RIO Device

Related concepts:

- Configuring a Project
- Programming Interfaces

Using the cRIO I/O Device Control

Use the cRIO I/O Device to specify a cRIO I/O item on the block diagram.

To configure the cRIO I/O Device, right-click the Device and select **Configure I/O Type**. The Configure I/O Type dialog box opens to determine which methods and properties a cRIO I/O Device supports.

- To select a cRIO I/O item from the shortcut menu on the block diagram, click the arrow on the right side of the Device. A shortcut menu opens with items that appear in the Project Explorer window under the same target as the active VI.
- Specify an I/O item by clicking inside the cRIO I/O Device and typing the name of

the I/O item.
- To enable the cRIO I/O Device to reference a cRIO I/O resource, add the I/O resource under the cRIO target in the Project Explorer window.
- When creating a reusable subVI, use the cRIO I/O control to specify a cRIO I/O item on the block diagram.

Configure I/O Type Dialog Box

Use this dialog box to configure the specific data type of a cRIO I/O Device, such as the supported methods and properties. This dialog box contains the following components:

- **I/O Items**—Lists all the cRIO I/O devices that you previously added to the project. Select one of these I/O items as a model for the specific data type of the I/O control.
- **I/O Item Type**—Lists the methods and properties that the I/O item you select from the I/O Items list supports.
- **Replace All**—Copies all methods or properties from the **I/O Item Type** list to the **I/O Name Control Type** list.
- **I/O Item Type Details**—Provides information about the method or property you select in the **I/O Item Type** list.
- **I/O Name Control Type**—Lists the methods and properties that the I/O control supports.
- **Remove**—Removes unwanted methods or properties you select from the **I/O Name Control Type** list.
- **I/O Name Control Type Details**—Provides information about the method or property you select in the **I/O Name Control Type** list.

Using the RIO Device I/O Control

Place the RIO Device I/O control on the front panel of a VI.

Use the I/O control pull-down menu to select a RIO device.

Selecting a RIO Device in a Project VI Targeted to My Computer

If the VI is part of a project and is targeted to My Computer, the I/O control pull-down menu shows the following sections from top to bottom.

- RIO devices in the project

- Local RIO devices and local RIO aliases
- **Browse**, which opens the **Browse RIO Devices** dialog box
- Recently used RIO devices

Selecting a RIO Device in a Project VI Targeted to an RT Controller

If the VI is part of a project and is targeted to an RT controller, the I/O control pull-down menu shows **RIO devices local to the controller**.

Selecting a RIO Device in a Non-Project VI

If the VI is not part of a project, the I/O control pull-down menu shows the following sections from top to bottom.

- Local RIO devices and local RIO aliases
- Browse, which opens the **Browse RIO Devices** dialog box
- Recently used RIO devices

Browse RIO Devices Dialog Box

Use this dialog box to select from local and remote RIO devices. Click the **+** sign to the left of a device to show the FPGA target associated with that device. This dialog box includes the following components:

- **Add Remote System**—Adds a remote RIO device outside the local subnet. Enter the hostname or IP address of the device in the Add Remote System dialog that appears.
- **Refresh Device List**—Refreshes the Remote devices list.

Configuring the C Series Module Data Rate (FPGA Interface)

To configure the data rate at which C Series modules acquire and returns data at edit time, use the **C Series Module Properties** dialog box.

To programmatically change the data rate at run time, use the **FPGA I/O Property Node**. The execution of an I/O Property Node that is configured with a **Data Rate** property overwrites the value you configured in the **C Series Module Properties** dialog box.

Note Refer to the module documentation on ni.com/docs for information about the data rates of specific modules.

Related concepts:

- Configuring a Project
- Synchronizing Multiple Modules (FPGA Interface)

Configuring the Data Rate Using the C Series Module Properties Dialog Box

Complete the following steps to configure the data rate using the C Series Module Properties dialog box.

1. Configure the system and add a module.
2. Right-click the module in the **Project Explorer** window and select **Properties** to open the **C Series Module Properties** dialog box.
3. Select the rate from the **Data Rate** pull-down menu.
4. Click the **OK** button.
5. Select **File**»**Save All** in the **Project Explorer** window.

Configuring the Data Rate Using the FPGA I/O Property Node

Complete the following steps to configure the data rate using the FPGA I/O Property Node.

1. Place an FPGA I/O Property Node on the block diagram and configure it for the module.
2. Right-click the **Property** section and select **Data Rate** from the shortcut menu.
3. Right-click the **Data Rate** input and select **Create**»**Control** from the shortcut menu.
4. On the front panel of the VI, select a rate from the **Data Rate** pull-down menu.

How do I change the data rate from the host VI?

Change the data rate at run time by writing to the control from the host VI.

What happens to the data rate if you change the master timebase source?

When you change the master timebase source, the data rate in the C Series Module Properties dialog box automatically updates to reflect the actual data rate of the module. However, the **Data Rate** property controls or constants on the block diagram do not automatically update when you change the master timebase source.

> **Note** You might need to delete and recreate controls or constants for any **Data Rate** property on the block diagram after changing the master timebase source.

Does the Data Rate property return an error?

The **Data Rate** property returns `error 65538` when the module is acquiring data.

To stop acquiring data before you can write properties to the modules, use **Stop channel**.

Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)

Configure a single **FPGA I/O Node** with the channels you want to simultaneously sample or update.

If you configure a single FPGA I/O Node with channels from different modules, FPGA I/O execution begins at the same time for all the modules, but the actual sampling or update times may differ in the following instances:

- Between modules depending on the types of modules
- The number of channels used on each module
- The arbitration settings
- Idle state when the I/O Node begins execution

If you configure an FPGA I/O Node with multiple channels on a non-simultaneous module, the function accesses the channels sequentially.

The following table lists the C Series modules that support simultaneously reading

from or writing to multiple CompactRIO channels.

Table 20. Supported C Series Hardware

C Series Module	Description
NI 9215	±10 V, 100 kS/s/ch, 16-Bit, Simultaneous Input, 4-Channel C Series Voltage Input Module
NI 9220	16-Channel, ±10 V, 100 kS/s/ch, 16-Bit Simultaneous Analog Input Module
NI 9222	±10 V, 100 kS/s/ch, 16-Bit, Simultaneous Input, 4-Channel C Series Voltage Input Module
NI 9223	±10 V, 1 MS/s, 16-Bit, Simultaneous Input, 4-Channel C Series Voltage Input Module
NI 9225	300 Vrms, 50 kS/s/ch, 24-Bit, Simultaneous Input, 3-Channel C Series Voltage Input Module
NI 9227	50 kS/s/ch, 5 Arms, 24-Bit, 4-Channel C Series Current Input Module
NI 9229	±60 V, 50 kS/s/ch, 24-Bit, Simultaneous Input, 4-Channel C Series Voltage Module
NI 9239	±10 V, 50 kS/s/ch, 24-Bit, Simultaneous Input, 4-Channel C Series Voltage Input Module
NI 9263	100 kS/s/ch Simultaneous, ±10 V, 4-Channel C Series Voltage Output Module
NI 9264	25 kS/s/ch Simultaneous, ±10 V, 16-Channel C Series Voltage Output Module
NI 9265	4-Channel C Series Current Output Module
NI 9269	100 kS/s/ch Simultaneous, ±10 V, Isolated, 4-Channel C Series Voltage Output Module
NI 9320	16-Channel, ±10 V, 200 kS/s/ch, 16-Bit Simultaneous Analog Input Module

Reading from C Series Channels

You can read from C series channels using the FPGA interface or Scan interface.

Related concepts:

- Configuring a Project with Connected Hardware

Read Data using the FPGA Interface

Complete the following steps to read data from an input channel in a CompactRIO, Ethernet RIO, MXIe-RIO, or Single-Board RIO system.

1. Configure a project for the system.
2. Make sure the input channel is added to the system.
3. Right-click the FPGA target in the **Project Explorer** window and select **New»VI** to add a new FPGA VI to the system.
4. Place an FPGA I/O Node on the block diagram of the FPGA VI.
5. Click the element section of the FPGA I/O Node and select **Module»Module/Channel** or **Onboard I/O»Port»Port/DIO**, where Module represents a C Series module and Module/Channel is the channel from which you want to read.
   If you configure an FPGA I/O Node for multiple channels, the I/O Node accesses the channels simultaneously or sequentially depending on the type of module and other factors.



Note LabVIEW can read CompactRIO digital input lines as ports for some I/O modules. Refer to the reference topic for the module you are using for more information. You can find the reference topic for the module by navigating on the **Contents** tab to **NI CompactRIO Device Drivers»Devices»C Series Modules»Digital Input Modules»NI 9xxx**.

Related concepts:

- Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)

Read Data using the Scan Interface

Complete the following steps to read data from an input channel in a CompactRIO, Ethernet RIO, MXIe-RIO, or Single-Board RIO system.

1. Configure a project for the system.
2. Make sure the input channel is added to the system.
3. Right-click the RT target item for the controller or device in the **Project Explorer** window and select **New»VI** to add a new LabVIEW Real-Time VI to the target.

4. Place a Timed Loop on the block diagram of the VI.
5. Place the I/O variable for the channel in the Timed Structure.
6. Right-click the output terminal of the I/O variable and select **Create**»**Indicator**.
7. Right-click the **error out** terminal of the I/O variable and select **Create**»**Indicator**.
8. Run the VI.

Writing to C Series Channels

You can read from C series channels using the FPGA interface or Scan interface.

Related concepts:

- CompactRIO Embedded System
- Configuring a Project with Connected Hardware
- Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)

Write Data using the FPGA Interface

Complete the following steps to write data to an output channel in a CompactRIO, Ethernet RIO, MXIe-RIO, or Single-Board RIO system.

1. Configure a project for the system.
2. Make sure the output channel is added to the system.
3. Right-click the FPGA target in the **Project Explorer** window and select **New»VI** to add a new FPGA VI to the system.
4. Place an FPGA I/O Node on the block diagram of the FPGA VI.
5. Click the element section of the FPGA I/O Node and select **Module»Module/Channel** or **Onboard I/O»Port»Port/DIO** from the shortcut menu, where Module represents a C Series module and Module/Channel is the channel to which you want to write.
   If you configure an FPGA I/O Node for multiple channels, the I/O Node accesses the channels simultaneously or sequentially depending on the type of module and other factors.

   Note LabVIEW can write to CompactRIO digital output lines as ports for some I/O modules.

Confirming the State of Digital Output Channels

For digital channels, you also can read the digital output channel.

Complete the following steps to read from the digital output channels.

1. Place another FPGA I/O Node function on the block diagram.
   This FPGA I/O Node must execute after the FPGA I/O Node you added above.
2. Click the element section of the FPGA I/O Node and select **Module»Module/Channel** or **Onboard I/O»Port»Port/DIO** from the shortcut menu, where Module represents a C Series module and Module/Channel is the channel you want to read.
3. Right-click the new FPGA I/O Node and select **Change to Read** from the shortcut menu.

Write Data using the Scan Interface

Complete the following steps to write data to an output channel in a CompactRIO or Single-Board RIO system.

1. Configure a project for the system.
2. Make sure the I/O variable for the output channel appears under the module item in the **Project Explorer** window.
3. Right-click the RT target item for the controller or device in the **Project Explorer** window and select **New»VI** from the shortcut menu to add a new LabVIEW Real-Time VI to the target.
4. Place a Timed Loop on the block diagram of the VI.
5. Configure the Timed Loop to use the Synchronize to Scan Engine timing source.
6. Place the I/O variable for the channel in the Timed Structure.
7. Right-click the input terminal of the I/O variable and select **Create»Control**.
8. Right-click the **error out** terminal of the I/O variable and select **Create»Indicator**.
9. Run the VI.

Configuring the cRIO I/O Node Name Control (FPGA Module)

You can use cRIO I/O name controls to create reentrant subVIs with configurable I/O items. When you use a cRIO I/O control as an input on a subVI, the input accepts only I/O items that support a superset of the methods and properties you enable for the

cRIO I/O control. All enabled methods must match in name, and their parameters must match in name, order, and data type. All enabled properties must match in name and data type.

Complete the following steps to configure the cRIO I/O name control using the **Configure cRIO I/O Name Control Type** dialog box.

1. Right-click a cRIO I/O Device and select **Configure I/O Type** from the shortcut menu to display the **Configure cRIO I/O Name Control Type** dialog box.
2. Select a cRIO I/O item from the **I/O Items** list that supports all the methods and properties you plan to use with the cRIO I/O control. The methods and properties of the I/O item you select appear in the **I/O Item Type** list.
3. Click the **Replace All** button to copy all the methods and properties from the **I/O Item Type** list to the **I/O Name Control Type** list.
   You cannot move individual I/O item types to the **I/O Name Control Type** box.
4. Click the **Remove** button to remove methods and properties from the **I/O Name Control Type** list that you do not want to require of I/O items passed to the control. To make the control broadly reusable, remove any resource-specific methods and properties that you do not want to use with the cRIO I/O control. You can click an item in the **I/O Name Control Type** list and read about the item in the **I/O Name Control Type Details** section.
5. Click the **OK** button to close the dialog box.

Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

The power-on output state is the state that a C Series output module is in when power is applied to the module. You cannot configure the power-on output state.



Note Refer to the module documentation on ni.com/docs for information about the power-on output state for your module.

C Series analog output modules have a *Hot Swap Behavior* feature that allows you to configure the state of the output channels when the module is removed and reinserted.

The startup output state is the state that a CompactRIO channel is in after you load and

run an FPGA VI that communicates with the output module. You can configure startup output states.

Configuring Startup Output States

Use the FPGA I/O Node analog or digital output options to change startup output states. You can develop the FPGA VI so that the output function executes when the FPGA VI starts running.

Understanding Output States

Table 21. Output Channel States

Action	**State of the Output Channels**
The chassis with the module powers on.	Power-on output state
An FPGA VI loads.	Startup output state
An Invoke Method function configured with the Abort method executes on the host VI.	Power-on output state
A Close FPGA VI Reference function that is configured to **Close and Reset** executes. **Close and Reset** is the default action for the Close FPGA VI Reference function.	Power-on output state
A Close FPGA VI Reference function that is configured to **Close** executes.	Last output value
You click the **Abort** button on the host VI or select **Operate»Stop** while LabVIEW is targeted to an FPGA device and the FPGA VI is running with Interactive Front Panel Communication.	Last output value
You remove and reinsert one of the following modules: NI 9476, NI 9477, or NI 9478.	Power-on output state
You remove and reinsert one of the following modules and the FPGA VI is not loaded: NI 9263, NI 9264, NI 9265, NI 9269.	Power-on output state
You remove and reinsert one of the following modules and the FPGA VI is loaded, but not running: NI-9263, NI-9264, NI-9265, NI-9269.	Power-on output state

Action	State of the Output Channels
You remove and reinsert one of the following modules and the FPGA VI is loaded and running: NI-9263, NI-9264, NI-9265, NI-9269.	User-configurable
You remove and reinsert one of the following modules: NI-9375, NI-9401, NI-9402, NI-9403, NI-9472, NI-9474, NI-9475, NI-9481 or NI-9485. NI recommends that you do not replace a module that is a different type with one of these modules after the FPGA VI is loaded. Replacing a module that is a different type than the module with which you are replacing it after the FPGA VI is loaded can place the output channels in an undefined or unexpected state.	Last output value

Related concepts:

- NI 9263
- NI 9264
- NI 9265
- NI 9269
- NI 9375
- NI 9401
- NI 9402
- NI 9403
- NI 9472
- NI 9474
- NI 9475
- NI 9476
- NI 9477
- NI 9478
- NI 9481
- NI 9485

Supporting External Data Value References

The following NI CompactRIO Device Drivers FPGA targets support using an external data value reference to implement DMA FIFOs without a data copy and while making

better use of memory resources:

- cRIO-9030
- cRIO-9031
- cRIO-9033
- cRIO-9034
- cRIO-9038
- cRIO-9066
- cRIO-9067
- cRIO-9068
- NI 9149
- sbRIO-9651 SOM
- myRIO-1900
- myRIO-1950
- roboRIO

Configuring the Host Memory Buffer

Some FPGA targets contain a host memory buffer that you can access directly from the FPGA VI. Use the memory item interface in the project to use the host memory buffer in the same way you use block memory and LUT. Host memory buffer items appear in the Project Explorer window under the FPGA target.

Most FPGA applications using the host memory buffer can take advantage of the usability and VHDL optimization that the FPGA memory item interface provides. You cannot use VI-defined memory items to configure the host memory buffer.

Determining Whether Host Memory Buffer is Available

Complete the following steps to determine whether host memory buffer is available for your target.

1. Display the Project Explorer window.
2. Right-click the FPGA target and select Memory Properties.
3. Open the Implementation drop-down list and select DRAM.
4. Open the DRAM bank drop-down list. If host memory buffer is available, it will be included in the list.

Partitioning the Physical Host Memory Buffer into Multiple Memory Items

You can use the FPGA memory item interface to partition the physical host memory buffer banks available on a target into multiple memory items. Use the Memory Properties dialog box to create and configure memory partitions on a host memory buffer bank. For example, if a target has two physical host memory buffer banks, you could partition one bank into three different memories and the other bank into five memories, as shown below. LabVIEW treats each memory independently of each other.

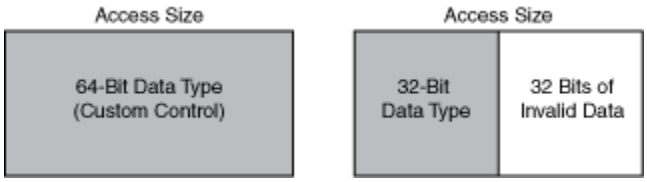
Understanding Latency and Host Memory Buffer

Access to the host memory buffer involves some non-deterministic latency. To compensate for this latency, use the Request Data and Retrieve Data methods to read data from the host memory buffer. You can queue multiple requests for data using the Request Data method and retrieve requested data using the Retrieve Data method. The host memory buffer returns requested data when you indicate you are ready to receive it using handshaking signals.

Data Recommendations

If you use a data type that is smaller than the access size, the remaining bits receive an unknown and invalid value, but still get written and take up both space and bandwidth. For example, if the access size is 64 bits wide and you choose a 32-bit data type when configuring the host memory buffer, the remaining 32 bits are of an unknown and invalid data type. The following figure shows an optimized memory element and a memory element in which the data type is smaller than the access size.

Figure 7.



NI recommends using data types that are exactly the same width as the access size of the host memory buffer to ensure that each access is optimized. Memory items accept clusters to be used as data types, and information can be packaged into clusters to achieve data types larger than those native to LabVIEW.

NI recommends that you push data into the memory item interface within the 40 MHz onboard clock domain. Right-click the FPGA target in a LabVIEW project, select **New»Memory**. Bandwidth is maximized when data is pushed into the memory item interface at the clock rate.

Request Pipelining

The host memory buffer architecture is highly pipelined, resulting in relatively long latency between data requests and the execution of the requests. NI recommends that prerequest samples, which helps maintain high throughput. To prerequest samples, request the samples you want to read without waiting for the data valid strobe of the retrieve method.

CompactRIO Controller and Chassis Reference

Use CompactRIO controllers and chassis with NI CompactRIO Device Drivers in LabVIEW.

CompactRIO Reconfigurable Chassis

Table 22. CompactRIO reconfigurable chassis

Chassis	Description
cRIO-9111	CompactRIO 4-Slot, Virtex-5 LX30 Reconfigurable Chassis
cRIO-9112	CompactRIO 8-Slot, Virtex-5 LX30 Reconfigurable Chassis
cRIO-9113	CompactRIO 4-Slot, Virtex-5 LX50 Reconfigurable Chassis
cRIO-9114	CompactRIO 8-Slot, Virtex-5 LX50 Reconfigurable Chassis
cRIO-9116	CompactRIO 8-Slot, Virtex-5 LX85 Reconfigurable Chassis
cRIO-9118	CompactRIO 8-Slot, Virtex-5 LX110 Reconfigurable Chassis

Related concepts:

- Configuring a Project with Connected Hardware
- Configuring a Project with Offline Hardware
- Configuring the Chassis I/O (FPGA Interface)
- Converting the Temperature of a CompactRIO Chassis, Ethernet RIO Chassis, MXIe-RIO Chassis, or a Single-Board RIO Device (FPGA Interface)

NI CompactRIO Controllers

CompactRIO DRAM Interface

Several CompactRIO targets contain onboard DRAM that is directly accessible from the LabVIEW FPGA VI. LabVIEW allows DRAM to be accessible as an FPGA Memory Item.

To determine the available amount of onboard DRAM, refer to the *Specifications* document of the CompactRIO target.

Using DRAM Effectively with CompactRIO

The following design considerations can affect the throughput and storage capacity that you can achieve with the dynamic random access memory (DRAM) interface of the FPGA on CompactRIO devices:

- Access size and frequency
- Request pipelining
- Sequential access

Access Size and Frequency

The access size is the amount of information stored in one memory address. You can set up memory to use a variety of data types. To achieve the best performance and to utilize the maximum amount of data, use a data type that matches the access size of the controller. The access size is the exact number of bits that are written and read in a given memory access.

The following table shows the specifications for CompactRIO targets that support FPGA-accessible DRAM, including the optimum access size.

Table 23. CompactRIO targets supporting FPGA-accessible DRAM

CompactRIO Target	Number of DRAM Banks	Size per Bank	Maximum Theoretical Bandwidth per Bank	Access Size
cRIO-9034	1	128 MB	1.6 GB/s	128 Bits
cRIO-9039	1	128 MB	1.6 GB/s	128 Bits

Data Recommendations

If you use a data type that is smaller than the access size, the remaining bits receive an unknown and invalid value, but still get written and take up both space and bandwidth. For example, if the access size is 128 bits wide and you choose a 32-bit data type when configuring the DRAM, the remaining 96 bits are of an unknown and invalid data type. The following figure shows an optimized memory element and a

memory element in which the data type is smaller than the access size.

Figure 8. Optimized memory elements



To ensure that each access is optimized, use data types with the exact same width as the access size of the DRAM. Memory items accept clusters to be used as data types, and information can be packaged into clusters to achieve data types larger than those native to LabVIEW.

Push data into the memory item interface within the DRAM clock domain, which is 100 MHz. Right-click the FPGA target in a LabVIEW project, select **New»FPGA Base Clock**, and choose the **DRAM Clock** resource. Bandwidth is maximized when data is pushed into the memory item interface at the clock rate.

Note You can use clock sources other than DRAM Clock, but performance will not be optimized.

Request Pipelining

The DRAM architecture is highly pipelined, resulting in relatively long latency between data requests and the execution of the requests. NI recommends that prerequest samples, which helps maintain high throughput.

To prerequest samples, request the samples you want to read without waiting for the data valid strobe of the retrieve method. Even though each individual request is still subject to latency and some non-determinism, you now get much higher transfer rates because DRAM can access several pieces of data sequentially instead of treating each request separately.

Sequential Access

DRAM is optimized for high storage density and high bandwidth. DRAM accesses data sequentially and in large blocks. For example, you have to read the data in address 0x1 after you have read the data in address 0x0 and the processor reads large blocks of memory into cache, even if the program being executed requests a single byte.

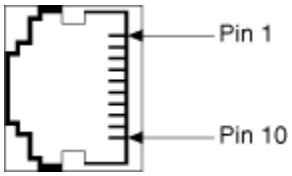
To maximize performance, avoid switching between reading and writing, accessing non-contiguous addresses, or writing to memory in decrementing-address fashion. The most efficient access strategy is to perform only one type of access, either reading or writing, on a large number of sequential addresses. Although this is optimal, it is not practical for most applications. A more practical approach is to maximize the amount of sequential data being accessed and minimizing changes in access modes.

Serial Ports

Note The cRIO-903x and 904x feature RS-232 and RS-485 Ethernet ports.

RS-232 Serial Port

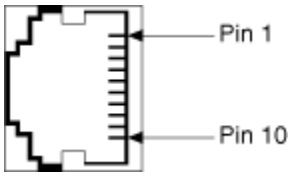
Figure 9. RS-232 Serial Port Pinout



Pin	Signal
1	No Connect
2	RI
3	CTS
4	RTS
5	DSR
6	GND
7	DTR
8	TXD
9	RXD
10	DCD

RS-485 Serial Port

Figure 10. RS-485 Serial Port Pinout



Pin	Signal Description
1	No Connect
2	TXD-
3	TXD+
4	No Connect
5	No Connect
6	RXD-
7	RXD+
8	No Connect
9	No Connect
10	Isolated GND

Configuring Controller Startup Settings in MAX

Note The following information applies to NI CompactRIO Controllers cRIO-903x, 904x, and 905x.

Perform the following steps to configure controller startup settings in Measurement & Automation Explorer (MAX):

1. Select the controller under **Remote Systems**.
2. Select the **System Settings** tab in the center pane.
3. In the **Startup Settings** section of the **System Settings** tab, place a check in the box for the startup setting you want to enable.
4. Click the **Restart** button in MAX to restart the system.

You can configure the following controller startup settings in MAX:

- **Force Safe Mode**—Launch only the services necessary for updating configuration and installing software. Do not launch LabVIEW RT or any startup applications.
- **Enable Console Out**—Redirect output to the serial port. You can use a serial port terminal to read the IP address and firmware version of the controller.
- **Disable RT Startup App**—Do not run any LabVIEW RT startup applications.
- **Disable FPGA Startup App**—Do not autoload any FPGA applications.

> **Note** If you restart the system at any time by disconnecting and reconnecting power, the system will disregard the **Disable FPGA Startup App** setting and load an FPGA startup application if one is present.

- **Enable Secure Shell Server (sshd)**—Enables login over SSH, an encrypted communication protocol. Visit ni.com/info and enter the Info Code `openssh` for more information about SSH.
- **LabVIEW Project Access**—Enables adding the target to a LabVIEW project.
- **Enable Embedded UI**—Enables the embedded UI the next time the target is restarted. When you enable the embedded UI, you can interact with the front panels of VIs running on the target using input and display devices connected directly to the target. You also can browse and edit files on the target within a graphical working environment.

> **Note** You can reboot the system into safe mode and clear the FPGA by pressing and holding the Reset button for 5 s.

cRIO-903x Controllers

The following table lists cRIO-903x Controllers.

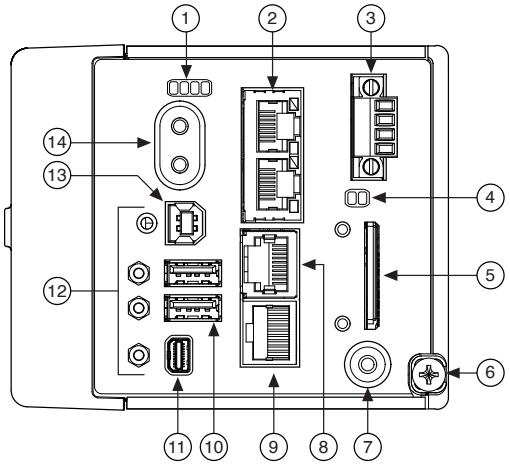
Table 24. cRIO-903x Controllers

cRIO-903x Controller Model	Description
cRIO-9030	1.33 GHz Dual-Core CPU, 1 GB DRAM, 4 GB Storage, Kintex-7 70T FPGA, 4-Slot CompactRIO Controller

cRIO-903x Controller Model	Description
cRIO-9031	1.33 GHz Dual-Core CPU, 1 GB DRAM, 4 GB Storage, Kintex-7 70T FPGA, Extended Temperature, 4-Slot CompactRIO Controller
cRIO-9032	1.33 GHz Dual-Core CPU, 2 GB DRAM, 8 GB Storage, Kintex-7 160T FPGA, WiFi, 4-Slot CompactRIO Controller
cRIO-9033	1.33 GHz Dual-Core CPU, 2 GB DRAM, 8 GB Storage, Kintex-7 160T FPGA, Extended Temperature, 4-Slot CompactRIO Controller
cRIO-9034	1.91 GHz Quad-Core CPU, 2 GB DRAM, 16 GB Storage, and Kintex-7 325T FPGA, 4-Slot CompactRIO Controller
cRIO-9035	1.33 GHz Dual-Core CPU, 1 GB DRAM, 4 GB Storage, and Kintex-7 70T FPGA, 8-Slot CompactRIO
cRIO-9036	1.33 GHz Dual-Core CPU, 1 GB DRAM, 4 GB Storage, Kintex-7 70T FPGA, Extended Temperature, 8-Slot CompactRIO Controller
cRIO-9037	1.33 GHz Dual-Core CPU, 2 GB DRAM, 8 GB Storage, Kintex-7 160T FPGA, WiFi, 8-Slot CompactRIO Controller
cRIO-9038	1.33 GHz Dual-Core CPU, 2 GB DRAM, 8 GB Storage, Kintex-7 160T FPGA, and Extended Temperature, 8-Slot CompactRIO Controller
cRIO-9039	1.91 GHz Quad-Core CPU, 2 GB DRAM, 16 GB Storage, and Kintex-7 325T FPGA, 8-Slot CompactRIO Controller

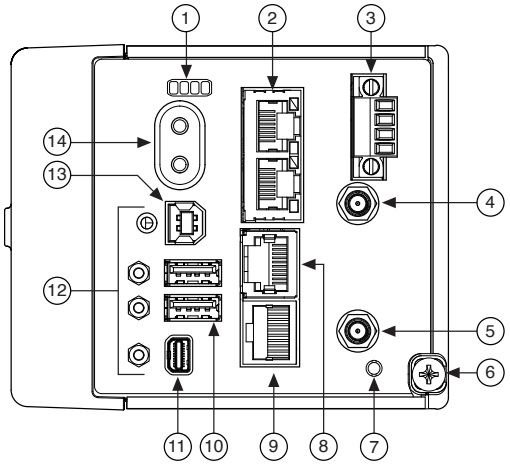
cRIO-903x Front Panel

Figure 11. cRIO-9030/9031/9033/9034/9035/9036/9038/9039 Front Panel



1. LEDs
2. Ethernet Ports
3. Power Connector
4. SD LEDs
5. SD Card Removable Storage
6. Ground Screw
7. USER1 Button
8. RS-232 Serial Port
9. RS-485/422 (DTE) Serial Port
10. USB Host Ports
11. Mini DisplayPort
12. Cable Retention Mounts
13. USB Device Port
14. Power and Reset Buttons

Figure 12. cRIO-9032/9037 Front Panel



1. LEDs
2. Ethernet Ports
3. Power Connector
4. Antenna Port 2
5. Antenna Port 1
6. Ground Screw
7. WLAN LED
8. RS-232 Serial Port
9. RS-485/422 (DTE) Serial Port
10. USB Host Ports
11. Mini DisplayPort

12. Cable Retention Mounts
13. USB Device Port
14. Power and Reset Buttons

Status LED Indicators

The following table describes the STATUS LED on NI CompactRIO cRIO-903x Controllers.

Table 25. cRIO-903x STATUS LED Indicator

LED Color/Pattern	Indication
Yellow, Blinks twice and pauses	The cRIO-903x is in safe mode. Software is not installed, which is the factory default state, or software has been improperly installed on the cRIO-903x. An error can occur when an attempt to upgrade the software is interrupted. Reinstall software on the cRIO-903x. Refer to *Installing Software on the Controller* for information about installing software on the cRIO-903x.
Yellow, Blinks three times and pauses	The cRIO-903x is in user-directed safe mode, or the cRIO-903x is in install mode to indicate that software is currently being installed. This pattern may also indicate that the user has forced the cRIO-903x to boot into safe mode by pressing the reset button for longer than five seconds or by enabling safe mode in MAX. Refer to *RESET Button* for information about safe mode.
Yellow, Blinks four times and pauses	The cRIO-903x is in safe mode. The software has crashed twice without rebooting or cycling power between crashes.
Yellow, On momentarily	The cRIO-903x is booting. No action required.
Red, Continuously blinks	Indicates a hardware error. An internal power supply has failed. Check front-panel I/O and C Series module connections for shorts. Remove any shorts and cycle power the cRIO-903x. If the problem persists, contact NI.
Off	The cRIO-903x is in run mode. Software is installed and the operating system is running.

cRIO-904x Controllers

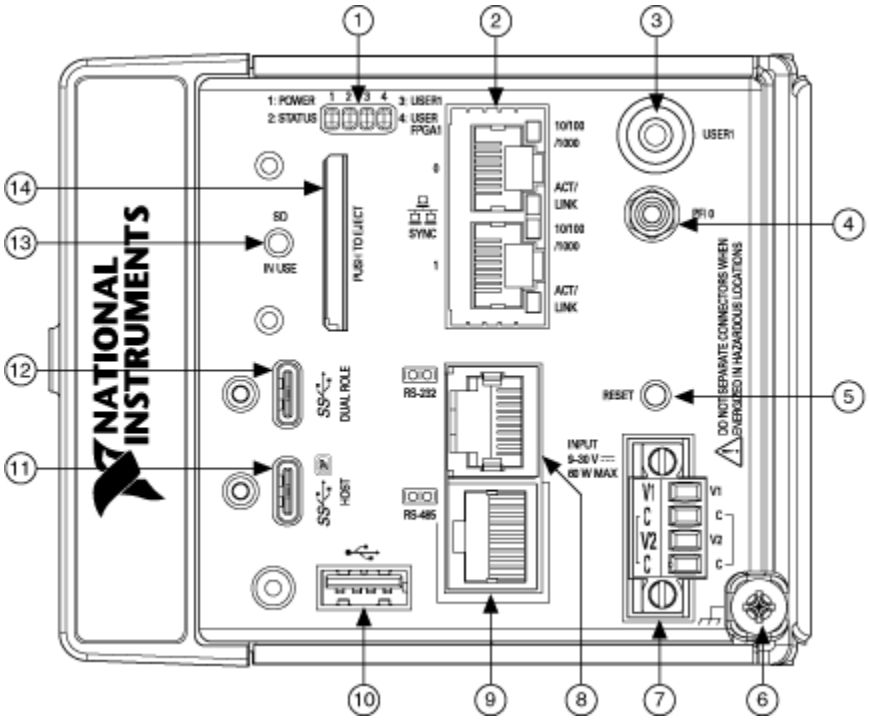
The following table lists cRIO-904x Controllers.

Table 26. cRIO-904x Controllers

cRIO-904x Controller Model	Description
cRIO-9040	1.30 GHz Dual-Core CPU, 2 GB DRAM, 4 GB Storage, -20 °C to 55 °C, Kintex-7 70T FPGA, 4-Slot CompactRIO Controller
cRIO-9041	1.30 GHz Dual-Core CPU, 2 GB DRAM, 4 GB Storage, -40 °C to 70 °C, Kintex-7 70T FPGA, 4-Slot CompactRIO Controller
cRIO-9042	1.60 GHz Quad-Core CPU, 4 GB DRAM, 4 GB Storage, -40 °C to 70 °C, Kintex-7 70T FPGA, 4-Slot CompactRIO Controller
cRIO-9043	1.30 GHz Dual-Core CPU, 2 GB DRAM, 8 GB Storage, -40 °C to 70 °C, Kintex-7 160T FPGA, 4-Slot CompactRIO Controller
cRIO-9045	1.30 GHz Dual-Core CPU, 2 GB DRAM , 4 GB Storage, -20 °C to 55 °C, Kintex-7 70T FPGA, 8-Slot CompactRIO Controller
cRIO-9046	1.30 GHz Dual-Core CPU, 2 GB DRAM, 4 GB Storage, -40 °C to 70 °C, Kintex-7 70T FPGA, 8-Slot CompactRIO Controller
cRIO-9047	1.60 GHz Quad-Core CPU, 4 GB Storage, -40 °C to 70 °C, Kintex-7 70T FPGA, 8-Slot CompactRIO Controller
cRIO-9048	1.30 GHz Dual-Core CPU, 2 GB DRAM, 8 GB Storage, -40 °C to 70 °C, Kintex-7 160T FPGA, Extended Temperature, 8-Slot CompactRIO Controller
cRIO-9049	1.60 GHz Quad-Core CPU, 4 GB DRAM, 16 GB Storage, -20 to 55 °C, Kintex-325T FPGA, 8-Slot CompactRIO Controller

cRIO-904x Front Panel

Figure 13. cRIO-904x Front Panel



1. LEDs
2. RJ-45 Ethernet Ports
3. User1 Button
4. PFI 0 SMB Connector
5. Resst Button
6. Grounding Screw
7. Power Connector
8. RS-232 Serial Port
9. RS-485 Serial Port
10. USB 2.0 Type-A (Port 1)
11. USB 3.1 Type-C with DisplayPort Alt Mode (Port 2)
12. USB 3.1 Type-C Dual Role (Port 3)
13. SD Indicator LED
14. SD Card Slot

Status LED Indicators

The following table describes the STATUS LED on NI CompactRIO cRIO-904x Controllers.

Table 27. cRIO-904x STATUS LED Indicator

LED Color/Pattern	Indication
Yellow, Blinks twice and pauses	The cRIO-904x is in safe mode. Software is not installed, which is the factory default state, or software has been improperly installed on the cRIO-904x. An error can occur when an attempt to upgrade the software is interrupted. Reinstall software on the cRIO-904x. Refer to *Installing Software on the Controller* for information about installing software on the cRIO-904x.
Yellow, Blinks three times and pauses	The cRIO-904x is in user-directed safe mode, or the cRIO-904x is in install mode to indicate that software is currently being installed. This pattern may also indicate that the user has forced the cRIO-904x to boot into safe mode by pressing the reset button for longer than five seconds or by enabling safe mode in MAX. Refer to *RESET Button* for information about safe mode.
Yellow, Blinks four times and pauses	The cRIO-904x is in safe mode. The software has crashed twice without rebooting or cycling power between crashes.
Yellow, Continuously blinks	The cRIO-904x has not booted into NI Linux Real-Time. The cRIO-904x either booted into an unsupported operating system, was interrupted during the boot process, or detected an unrecoverable software error.
Yellow, On momentarily	The cRIO-904x is booting. No action required.
Red, Continuously blinks	Indicates a hardware error. An internal power supply has failed. Check front-panel I/O and C Series module connections for shorts. Remove any shorts and cycle power the cRIO-904x. If the problem persists, contact NI.
Red, Solid	The cRIO-904x internal temperature has exceeded a critical threshold. Ensure the ambient operating temperature does not exceed the specified operating temperature. If the problem persists, contact NI.
Off	The cRIO-904x is in run mode. Software is installed and the operating system is running.

cRIO-905x Controllers

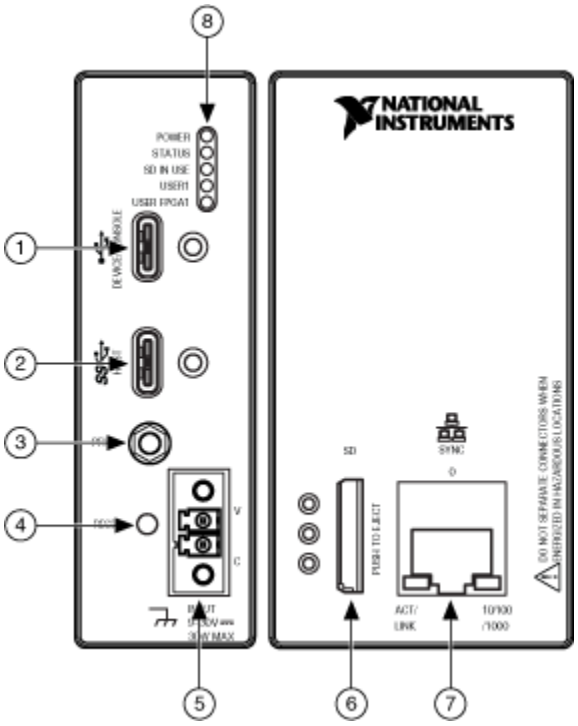
The following table lists cRIO-905x Controllers.

Table 28. cRIO-905x Controllers

cRIO-905x Controller Model	Description
cRIO-9053	1.33 GHz Dual-Core CPU, 1 GB DRAM, 4 GB Storage, -20 °C to 55 °C, Artix 7 50T FPGA, 4-Slot CompactRIO Controller
cRIO-9054	1.33 GHz Dual-Core CPU, 2 GB DRAM, 4 GB Storage, -20 °C to 55 °C, Artix 7 100T FPGA, 4-Slot CompactRIO Controller
cRIO-9055	1.33 GHz Dual-Core CPU, 2 GB DRAM, 4 GB Storage, -40 °C to 70 °C, Artix-7 A100T FPGA, Extended Temperature, 4-Slot CompactRIO Controller
cRIO-9056	1.33 GHz Dual-Core CPU, 1 GB DRAM 4 GB Storage, -20 °C to 55 °C, Artix 7 75T FPGA, 8-Slot CompactRIO Controller
cRIO-9057	1.33 GHz Dual-Core CPU, 2 GB DRAM, 4 GB Storage, -20 °C to 55 °C, Artix 7 100T FPGA, 8-Slot CompactRIO Controller
cRIO-9058	1.33 GHz Dual-Core CPU, 2 GB DRAM, 4 GB Storage, -40 °C to 70 °C, Artix-7 A100T FPGA, Extended Temperature, 8-Slot CompactRIO Controller

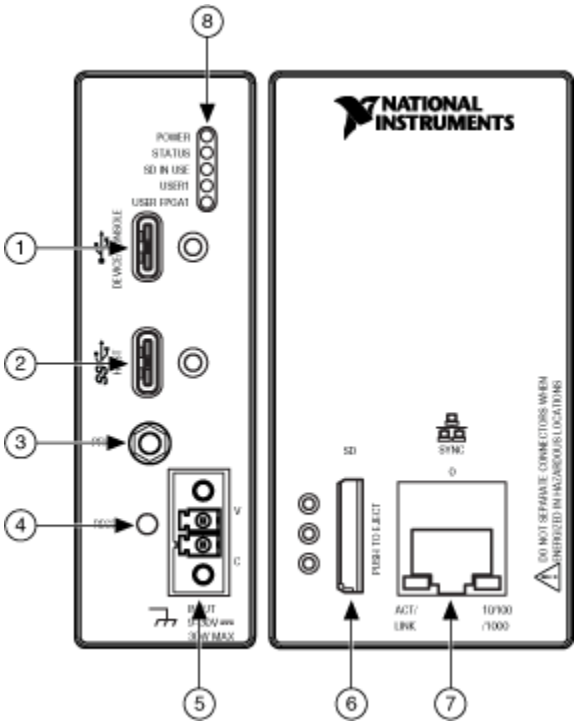
cRIO-905x Front Panel

Figure 14. cRIO-9053/9056 Front Panel



1. USB 2.0 Type-C Device/Console
2. USB 3.1 Type-C Host
3. PFI 0
4. Reset Button
5. Power Connector
6. µSD Card Slot
7. RJ-45 Gigabit Ethernet Port
8. LEDs

Figure 15. cRIO-9054/9055/9057/9058 Front Panel



1. USB 2.0 Type-C Device/Console
2. USB 3.1 Type-C Host
3. PFI 0
4. Reset Button
5. Power Connector
6. µSD Card Slot
7. RJ-45 Gigabit Ethernet Port
8. LEDs

Status LED Indicators

The following table describes the STATUS LED on NI CompactRIO cRIO-905x Controllers.

Table 29. cRIO-905x STATUS LED Indicator

LED Color/Pattern	Indication
Yellow, Blinks twice and pauses	The cRIO-905x is in safe mode. Software is not installed, which is the factory default state, or software has been improperly installed on the cRIO-905x.

LED Color/Pattern	Indication
	An error can occur when an attempt to upgrade the software is interrupted. Reinstall software on the cRIO-905x. Refer to *Installing Software on the Controller* for information about installing software on the cRIO-905x.
Yellow, Blinks three times and pauses	The cRIO-905x is in user-directed safe mode, or the cRIO-905x is in install mode to indicate that software is currently being installed. This pattern may also indicate that the user has forced the cRIO-905x to boot into safe mode by pressing the reset button for longer than five seconds or by enabling safe mode in MAX. Refer to *RESET Button* for information about safe mode.
Yellow, Blinks four times and pauses	The cRIO-905x is in safe mode. The software has crashed twice without rebooting or cycling power between crashes.
Yellow, Continuously blinks	The cRIO-905x has not booted into NI Linux Real-Time. The cRIO-905x either booted into an unsupported operating system, was interrupted during the boot process, or detected an unrecoverable software error.
Yellow, On momentarily	The cRIO-905x is booting. No action required.
Red, Continuously blinks	Indicates a hardware error. An internal power supply has failed. Check front-panel I/O and C Series module connections for shorts. Remove any shorts and cycle power the cRIO-905x. If the problem persists, contact NI.
Red, Solid	The cRIO-905x internal temperature has exceeded a critical threshold. Ensure the ambient operating temperature does not exceed the specified operating temperature. If the problem persists, contact NI.
Off	The cRIO-905x is in run mode. Software is installed and the operating system is running.

cRIO-906x Controllers

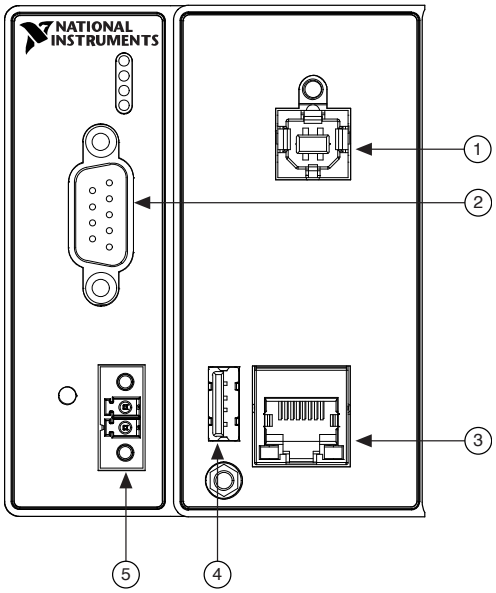
The following table lists cRIO-906x Controllers.

Table 30. cRIO-906x Controllers

cRIO-906x Controller Model	Description
cRIO-9063	667 MHz Dual-Core CPU, 256 MB DRAM, 512 MB Storage, Zynq-7020 FPGA, 4-Slot CompactRIO Controller
cRIO-9064	667 MHz Dual-Core CPU, 512 MB DRAM, 1 GB Storage, and Zynq-7020 FPGA, 4-Slot CompactRIO Controller
cRIO-9065	667 MHz Dual-Core CPU, 512 MB DRAM, 1 GB Storage, Zynq-7020 FPGA, Extended Temperature, 4-Slot CompactRIO Controller
cRIO-9066	667 MHz Dual-Core CPU, 256 MB DRAM, 512 MB Storage, Zynq-7020 FPGA, 8-Slot CompactRIO Controller
cRIO-9067	667 MHz Dual-Core CPU, 512 MB DRAM, 1 GB Storage, Zynq-7020 FPGA, 8-Slot CompactRIO Controller
cRIO-9068	667 MHz Dual-Core CPU, 512 MB DRAM, 1 GB Storage, Zynq-7020 FPGA, Extended Temperature, 8-Slot CompactRIO Controller

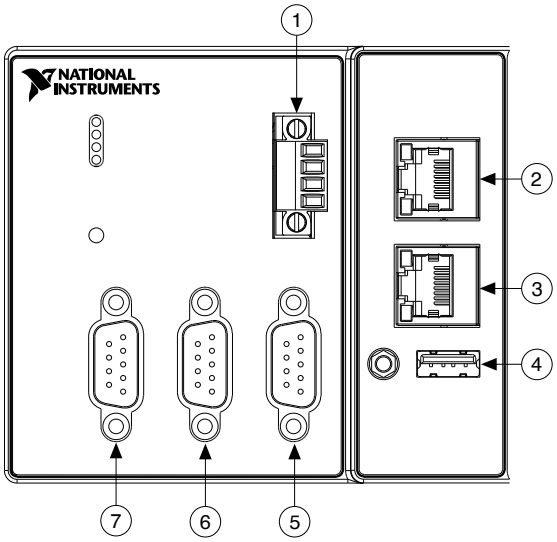
cRIO-906x Front Panel

Figure 16. cRIO-9063/9064/9065/9066/9067 Front Panel



1. USB Device Port
2. RS-232 Serial Port
3. RJ-45 Gigabit Ethernet Ports (one or two, depending on model)
4. USB Host Port
5. Power Connector

Figure 17. cRIO-9068 Front Panel



1. Power Connector
2. RJ-45 Gigabit Ethernet Port 2
3. RJ-45 Gigabit Ethernet Port 1
4. USB Host Port
5. RS-485 Serial Port
6. RS-232 Serial Port
7. RS-232 Serial Port

Configuring Controller Settings

Perform the following procedure to configure controller startup settings in Measurement & Automation Explorer (MAX).

1. Launch MAX.
2. Locate the controller under **Remote Systems**.
3. Select the **Controller Settings** tab.

To reboot the device into safe mode, press and hold the **Reset** button for 5 s.

Resetting the Controller also Resets the FPGA

Resetting the device real-time controller resets the FPGA, clears volatile memory and unloads any loaded FPGA bitfile. After reset, the device is put in Sleep Mode and module output channels revert to default output values. Refer to the module documentation for information about Sleep Mode behavior.

Autoloading the FPGA Bitfile on Startup

If you download a bitfile to the device, the bitfile is autoloaded by default at startup. To disable bitfile autoloading, enable the **Disable FPGA Startup Application** setting in MAX, or click the **Erase Firmware** button in the **RIO Device Setup Utility**.

Ethernet RIO Chassis

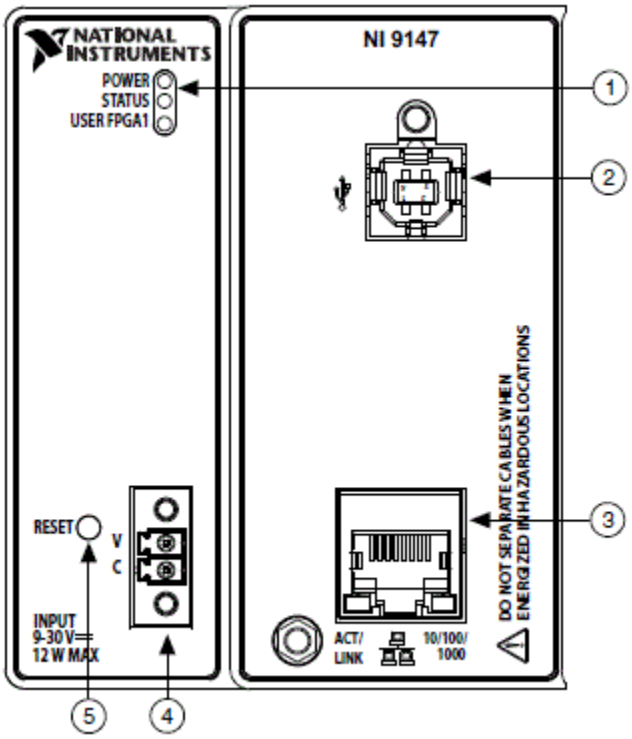
The following table lists Ethernet RIO Chassis.

Table 31. Ethernet RIO Chassis

Ethernet RIO Chassis Model	Description
NI-9147	4-Slot, Zynq-7020 FPGA, Ethernet CompactRIO Chassis
NI-9149	8-Slot, Zynq-7020 FPGA, Ethernet CompactRIO Chassis

Ethernet RIO Chassis Front Panel

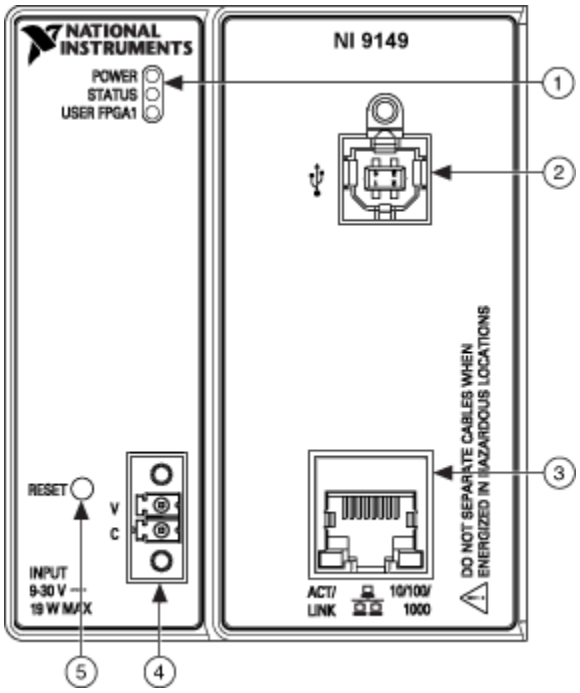
Figure 18. NI-9147 Front Panel



1. LEDs

2. USB Device Port
3. RJ-45 Ethernet Port
4. Power Connector
5. RESET Button

Figure 19. NI-9149 Front Panel



1. LEDs
2. USB Device Port
3. RJ-45 Ethernet Port
4. Power Connector
5. RESET Button

Configuring Controller Settings

Perform the following procedure to configure controller startup settings in Measurement & Automation Explorer (MAX).

1. Launch MAX.
2. Locate the controller under **Remote Systems**.
3. Select the **Controller Settings** tab.

To reboot the device into safe mode, press and hold the **Reset** button for 5 s.

Resetting the Controller also Resets the FPGA

Resetting the device real-time controller resets the FPGA, clears volatile memory and unloads any loaded FPGA bitfile. After reset, the device is put in Sleep Mode and module output channels revert to default output values. Refer to the module documentation for information about Sleep Mode behavior.

MXIe-RIO Chassis

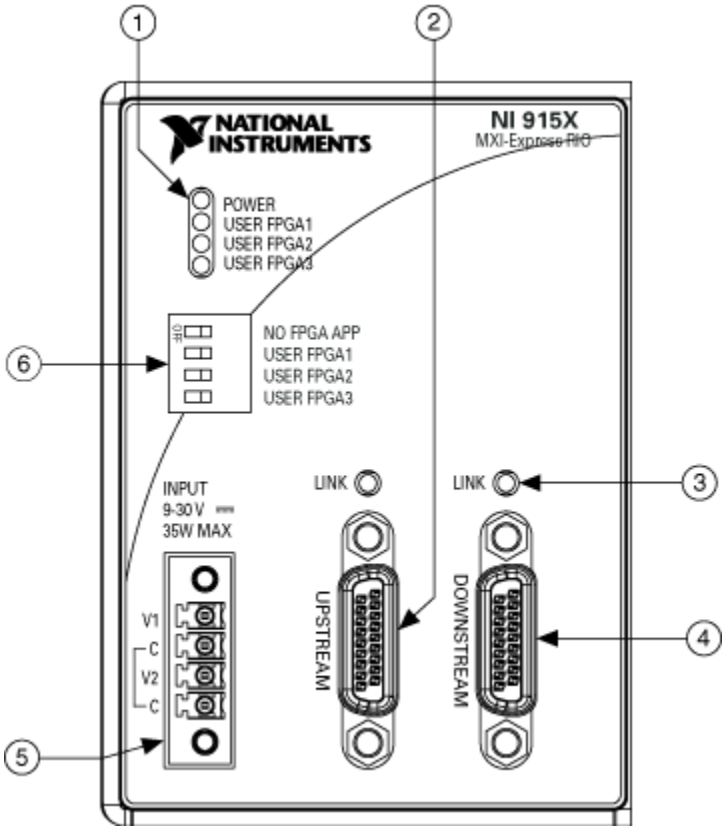
The following table lists MXIe-RIO Chassis.

Table 32. MXIe-RIO Chassis

MXIe-RIO Chassis Model	Description
NI-9154	8-Slot, Virtex-5 LX50 FPGA, MXI-Express CompactRIO Chassis
NI-9155	8-Slot, Virtex-5 LX85 FPGA, MXI-Express CompactRIO Chassis
NI-9157	14-Slot, Virtex-5 LX85 FPGA, MXI-Express CompactRIO Chassis
NI-9159	14-Slot, Virtex-5 LX110 FPGA, MXI-Express CompactRIO Chassis

MXI-RIO Chassis Front Panel

Figure 20. MXI-RIO Chassis Front Panel



1. LEDs
2. Upstream Port
3. MXI-Express LINK LEDs
4. Downstream Port
5. Power Connector
6. DIP Switches

Reading the MXIe-RIO Chassis Switch State (FPGA Interface)

The MXIe-RIO reconfigurable embedded chassis has USER FPGA DIP switches. Complete the following steps to read the switch state of DIP switches on the FPGA.

1. Configure the MXIe-RIO system.
2. Verify that the **USER FPGA Switch** I/O item is added to the MXIe-RIO system.
3. To add a new VI to the MXIe-RIO system, right-click the FPGA target in the **Project Explorer** window and select **New»VI** from the shortcut menu.
4. Place an FPGA I/O Node on the block diagram of the VI.

5. Click the element section of the FPGA I/O Node and select **Chassis I/O»USER FPGA1 Switch** from the shortcut menu. USER FPGA2 Switch and USER FPGA3 Switch function the same as USER FPGA1 Switch.
6. Right-click the **USER FPGA1 Switch** output of the FPGA I/O Node and select **Create»Indicator** from the shortcut menu.

Refer to the 915x Switches and LEDs VI in the `labview\examples\CompactRIO\Chassis Specific\915x Switches and LEDs\915x Switches and LEDs.lvproj` for an example of reading USER FPGA switches.

Related concepts:

- Configuring a Project with Connected Hardware

C Series Module Reference

You can use C Series modules with NI CompactRIO Device Drivers in LabVIEW.

- When programming in **FPGA Programming** mode, C Series modules support FPGA I/O functions, I/O resources, arbitration options, methods, and properties.
- When programming in **Scan Interface** mode, C Series modules support I/O variables and properties.

Refer to the C Series module topics for details on specific support.

C Series Module IDs

You can use the Module ID property with the **FPGA I/O Property Node** to read the IDs associated with C Series modules. You also can wire a reference to a module in Scan Interface mode to a generic-class property node in a VI block diagram, and select the **ProductID** property to read the ID programmatically. Note that the **ProductID** output will be displayed in decimal by default, and the following IDs are in hexadecimal. By ID

Table 33. C Series Modules List by Product ID

ID	Module
0x709C	*NI 9435*
0x709D	*NI 9411*
0x709E	*NI 9423*
0x709F	*NI 9421 with screw terminal*
0x70A0	*NI 9474*
0x70A1	*NI 9472 with screw terminal*
0x70A2	*NI 9481*
0x70A3	*NI 9211*
0x70A4	*NI 9201*
0x70A5	*NI 9221*
0x70A6	*NI 9215 with screw terminal*

ID	Module
0x70A7	*NI 9263*
0x7129	*NI 9203*
0x712A	*NI 9205 with DSUB*
0x712B	*NI 9217*
0x712C	*NI 9265*
0x712E	*NI 9421 with DSUB*
0x712F	*NI 9425*
0x7130	*NI 9401*
0x7131	*NI 9403*
0x7132	*NI 9472 with DSUB*
0x7133	*NI 9476*
0x7135	*NI 9215 with BNC*
0x714F	*NI 9853*
0x71A1	*NI 9201 with DSUB*
0x71A2	*NI 9221 with DSUB*
0x71C2	*NI 9239 with screw terminal*
0x71C3	*NI 9237 with RJ-50*
0x71C4	*NI 9204 with spring terminal*
0x71CA	*NI 9422*
0x71CB	*NI 9477*
0x71EA	*NI 9205 with spring terminal*
0x71EB	*NI 9206*
0x71ED	*NI 9505*
0x71F3	*NI 9852*
0x71F6	*NI 9485*
0x72B5	*NI 9234*
0x72F6	*NI 9264*

ID	Module
0x72FD	*NI 9229 with screw terminal*
0x7304	*NI 9375*
0x7305	*NI 9870*
0x7306	*NI 9871*
0x730C	*NI 9219*
0x731A	*NI 9478*
0x7328	*NI 9402*
0x732A	*NI 9235*
0x732B	*NI 9236*
0x736A	*NI 9426*
0x7367	*NI 9239 with BNC*
0x7368	*NI 9229 with BNC*
0x7377	*NI 9475*
0x7383	*NI 9225 with screw terminal*
0x73A4	*NI 9237 with DSUB*
0x73E0	*NI 9220 with spring terminal*
0x7408	*NI 9225E*
0x7409	*NI 9481E*
0x740A	*NI 9211E*
0x740B	*NI 9203E*
0x740C	*NI 9403E*
0x740D	*NI 9215E*
0x7415	*NI 9269 with screw terminal*
0x7416	*NI 9227 with screw terminal*
0x741C	*NI 9239E*
0x741D	*NI 9229E*
0x741E	*NI 9263E*

ID	Module
0x741F	*NI 9237E*
0x743A	*NI 9213E*
0x743B	*NI 9478E*
0x743F	*NI 9870E*
0x7440	*NI 9871E*
0x7441	*NI 9219E*
0x7442	*NI 9265E*
0x7445	*NI 9222 with screw terminal*
0x7446	*NI 9222E*
0x7447	*NI 9264E*
0x7449	*NI 9213*
0x744B	*NI 9205E*
0x744C	*NI 9227E*
0x744D	*NI 9269E*
0x745D	*NI 9215E with BNC*
0x747A	*NI 9375 with DSUB*
0x749B	*NI 9201E with screw terminal*
0x749C	*NI 9221E with screw terminal*
0x749D	*NI 9201E with DSUB*
0x749E	*NI 9221E with DSUB*
0x74A1	*NI 9505E*
0x74AA	*NI 9207E with DSUB*
0x74AB	*NI 9208E with DSUB*
0x74AC	*NI 9208 with DSUB*
0x74AD	*NI 9207 with DSUB*
0x74EC	*NI 9223*
0x74ED	*NI 9223E*

ID	Module
0x74EE	*NI 9214*
0x753F	*NI 9232 with screw terminal*
0x7615	*NI 9220 with DSUB*
0x7642	*NI 9381*
0x76E3	*NI 9482*
0x76E8	*NI 9244*
0x7733	*NI 9222 with BNC*
0x7734	*NI 9223 with BNC*
0x7750	*NI 9238 with screw terminal*
0x77EA	*NI 9250*
0x77B8	*NI 9247*
0x77C5	*NI 9344*
0x77CC	*NI 9230 with screw terminal*
0x77E3	*NI 9246*
0x77E9	*NI 9251*
0x777E	*NI 9361*
0x7786	*NI 9260 with BNC*
0x778C	*NI 9260 with mini XLR*
0x780D	*NI 9224*
0x780E	*NI 9228*
0x7827	*NI 9770*
0x7866	*NI 9232 with BNC*
0x7867	*NI 9230 with BNC*
0x7889	*NI 9775*
0x788A	*NI 9202 with DSUB*
0x788C	*NI 9202 with spring terminal*
0x788F	*NI 9266*

ID	Module
0x78B1	*NI 9210*
0x78B6	*NI 9262 with DSUB*
0x7900	*NI 9476 with spring terminal*
0x7930	*NI 9231*
0x79C4	*NI 9253*
0x79C8	*NI 9252 with screw terminal*
0x79CA	*NI 9252 with DSUB*
0x7A59	*NI 9470*
0x7A5E	*NI 9326*
0x7B3B	*NI 9204 with DSUB*
0x7B45	*NI 9222 with snap in terminal*
0x7B47	*NI 9223 with snap in terminal*
0x7B49	*NI 9225 with snap in terminal*
0x7B4B	*NI 9227 with snap in terminal*
0x7B4D	*NI 9230 with snap in terminal*
0x7B4F	*NI 9232 with snap in terminal*
0x7B51	*NI 9238 with snap in terminal*
0x7B53	*NI 9269 with snap in terminal*
0x7B55	*NI 9229 with snap in terminal*
0x7B57	*NI 9239 with snap in terminal*
0x7B5C	*NI 9320 with spring terminal*
0x7B5D	*NI 9320 with DSUB*
0x7BA8	*NI 9313*

Table 34. C Series Modules Listed By Module Name

Module	ID
NI 9201	0x70A4

Module	ID
NI 9201 with DSUB	0x71A1
NI 9201E with DSUB	0x749D
NI 9201E with screw terminal	0x749B
NI 9202 with DSUB	0x788A
NI 9202 with spring terminal	0x788C
NI 9203	0x7129
NI 9203E	0x740B
NI 9204 with DSUB	0x7B3B
NI 9204 with spring terminal	0x71C4
NI 9205 with DSUB	0x712A
NI 9205 with spring terminal	0x71EA
NI 9205E	0x744B
NI 9206	0x71EB
NI 9207 with DSUB	0x74AD
NI 9207E with DSUB	0x74AA
NI 9208 with DSUB	0x74AC
NI 9208E with DSUB	0x74AB
NI 9210	0x78B1
NI 9211	0x70A3
NI 9211E	0x740A
NI 9213	0x7449
NI 9213E	0x743A
NI 9214	0x74EE
NI 9215 with BNC	0x7135
NI 9215 with screw terminal	0x70A6
NI 9215E with BNC	0x745D
NI 9215E with screw terminal	0x740D

Module	ID
NI 9217	0x712B
NI 9219	0x730C
NI 9219E	0x7441
NI 9220 with spring terminal	0x73E0
NI 9220 with DSUB	0x7615
NI 9221	0x70A5
NI 9221 with DSUB	0x71A2
NI 9221E with DSUB	0x749E
NI 9221E with screw terminal	0x749C
NI 9221E with screw terminal	0x749C
NI 9222 with BNC	0x7733
NI 9222 with screw terminal	0x7445
NI 9222 with snap in terminal	0x7B45
NI 9222E	0x7446
NI 9223 with BNC	0x7734
NI 9223 with screw terminal	0x74EC
NI 9223 with snap in terminal	0x7B47
NI 9223E	0x74ED
NI 9224	0x780D
NI 9225 with screw terminal	0x7383
NI 9225E	0x7408
NI 9225 with snap in terminal	0x7B49
NI 9227 with screw terminal	0x7416
NI 9227 with snap in terminal	0x7B4B
NI 9227E	0x744C
NI 9228	0x780E
NI 9229 with screw terminal	0x72FD

Module	ID
NI 9229E	0x741D
NI 9229 with BNC	0x7368
NI 9229 with snap in terminal	0x7B55
NI 9230 with BNC	0x7867
NI 9230 with screw terminal	0x77CC
NI 9230 with snap in terminal	0x7B4D
NI 9231	0x7930
NI 9232 with BNC	0x7866
NI 9232 with screw terminal	0x753F
NI 9232 with snap in terminal	0x7B4F
NI 9234	0x72B5
NI 9235	0x732A
NI 9236	0x732B
NI 9237 with DSUB	0x73A4
NI 9237 with RJ-50	0x71C3
NI 9237E	0x741F
NI 9238 with screw terminal	0x7750
NI 9238 with snap in terminal	0x7B51
NI 9239 with screw terminal	0x71C2
NI 9239E	0x741C
NI 9239 with BNC	0x7367
NI 9239 with snap in terminal	0x7B57
NI 9244	0x76E8
NI 9246	0x77E3
NI 9247	0x77B8
NI 9250	0x77EA
NI 9251	0x77E9

Module	ID
NI 9252 with screw terminal	0x79C8
NI 9252 with DSUB	0x79CA
NI 9253	0x79C4
NI 9260 with BNC	0x7786
NI 9260 with mini XLR	0x778C
NI 9262 with DSUB	0x78B6
NI 9263	0x70A7
NI 9263E	0x741E
NI 9264	0x72F6
NI 9264E	0x7447
NI 9265	0x712C
NI 9265E	0x7442
NI 9266	0x788F
NI 9269 with screw terminal	0x7415
NI 9269 with snap in terminal	0x7B53
NI 9269E	0x744D
NI 9313	0x7BA8
NI 9320 with spring terminal	0x7B5C
NI 9320 with DSUB	0x7B5D
NI 9326	0x7A5E
NI 9344	0x77C5
NI 9361	0x777E
NI 9375	0x7304
NI 9375 with DSUB	0x747A
NI 9381	0x7642
NI 9401	0x7130
NI 9402	0x7328

Module	ID
NI 9403	0x7131
NI 9403E	0x740C
NI 9411	0x709D
NI 9421 with DSUB	0x712E
NI 9421 with screw terminal	0x709F
NI 9422	0x71CA
NI 9423	0x709E
NI 9425	0x712F
NI 9426	0x736A
NI 9435	0x709C
NI 9470	0x7A59
NI 9472 with DSUB	0x7132
NI 9472 with screw terminal	0x70A1
NI 9474	0x70A0
NI 9475	0x7377
NI 9476	0x7133
NI 9476 with spring terminal	0x7900
NI 9477	0x71CB
NI 9478	0x731A
NI 9478E	0x743B
NI 9481	0x70A2
NI 9481E	0x7409
NI 9482	0x76E3
NI 9485	0x71F6
NI 9505	0x71ED
NI 9505E	0x74A1
NI 9770	0x7827

Module	ID
NI 9775	0x7889
NI 9852	0x71F3
NI 9853	0x714F
NI 9870	0x7305
NI 9870E	0x743F
NI 9871	0x7306
NI 9871E	0x7440

Analog Input Modules

Table 35. C Series Analog Input Modules

C Series Analog Input Module	Description
NI-9201	±10 V, 500 kS/s, 12-Bit, 8-Channel C Series Voltage Input Module
NI-9202	±10 V, 10 kS/s, 24-Bit, 16-Channel C Series Voltage Input Module
NI-9203	200 kS/s, ±20 mA, 8-Channel C Series Current Input Module
NI-9204	±10 V, 250 kS/s, 16-Bit, 16-Channel C Series Voltage Input Module
NI-9205	±10 V, 250 kS/s, 16-Bit, 32-Channel C Series Voltage Input Module
NI-9206	±10 V, 250 kS/s, 16-Bit, 32-Channel, 600 VDC Isolation C Series Voltage Input Module
NI-9207	500 S/s, 16-Channel C Series Voltage and Current Input Module
NI-9208	500 S/s, ±20 mA, 16-Channel C Series Current Input Module
NI-9209	±10 V, 500 S/s, 16-Channel C Series Voltage Input Module
NI-9210	4-Channel, 14 S/s Aggregate, ±80 mV C Series Temperature Input Module
NI-9211	4-Channel, 14 S/s, ±80 mV C Series Temperature Input Module
NI-9212	8-Channel, 95 S/s/ch Simultaneous, ±78 mV C Series Temperature Input Module
NI-9213	16 channel, up to 75 S/s per channel, ±78 mV C Series Temperature Input Module
NI-9214	16-Channel, 68 S/s Aggregate, ±78 mV, Isothermal C Series Temperature Input Module

C Series Analog Input Module	Description
NI-9215	±10 V, 100 kS/s/ch, 16-Bit, Simultaneous Input, 4-Channel C Series Voltage Input Module
NI-9216	8-Channel, 400 S/s Aggregate, 0 Ω to 400 Ω, PT100 RTD C Series Temperature Input Module
NI-9217	4-Channel, 400 S/s Aggregate, 0 Ω to 400 Ω, PT100 RTD C Series Temperature Input Module
NI-9218	51.2 kS/s/ch, 2-Channel C Series Universal Analog Input Module
NI-9219	100 S/s/ch, 4-Channel C Series Universal Analog Input Module
NI-9220	±10 V, 100 kS/s/ch, 16-Bit, Simultaneous Input, 16-Channel C Series Voltage Input Module
NI-9221	±10 V, 1 MS/s, 16-Bit, Simultaneous Input, 4-Channel C Series Voltage Input Module
NI-9222	±10 V, 500 kS/s/ch, 16-Bit, Simultaneous Input, 4-Channel C Series Voltage Input Module
NI-9223	±10 V, 1 MS/s, 16-Bit, Simultaneous Input, 4-Channel C Series Voltage Input Module
NI-9224	±10 V, 1 kS/s/ch, 24-Bit, Simultaneous Input, 8-Channel C Series Voltage Input Module
NI-9225	±10 V, 1 kS/s/ch, 24-Bit, Simultaneous Input, 8-Channel C Series Voltage Input Module
NI-9226	8-Channel, 400 S/s Aggregate, 0 Ω to 4000 Ω, PT1000 RTD C Series Temperature Input Module
NI-9227	50 kS/s/ch, 5 Arms, 24-Bit, 4-Channel C Series Current Input Module
NI-9228	±60 V, 1 kS/s/ch, 24-Bit, Simultaneous Input, 8-Channel C Series Voltage Input Module
NI-9229	±60 V, 50 kS/s/ch, 24-Bit, Simultaneous Input, 4-Channel C Series Voltage Module
NI-9230	3-Channel, 12.8 kS/s/channel, ±30 V, C Series Sound and Vibration Input Module
NI-9231	8-Channel, 51.2 kS/s/channel, -5 V to 5 V, C Series Sound and Vibration Input Module
NI-9232	3-Channel, 102.4 kS/s/channel, ±30 V, C Series Sound and Vibration Input Module
NI-9233	4-Channel, 50kS/s/channel, ±5 V, C Series Sound and Vibration Input Module

C Series Analog Input Module	Description
NI-9234	4-Channel, 51.2 kS/s/channel, ±5 V, C Series Sound and Vibration Input Module
NI-9235	10 kS/s/channel, 120 Ω Quarter-Bridge Strain Gage, 8-Channel C Series Strain/Bridge Input Module
NI-9236	10 kS/s/channel, 350 Ω Quarter-Bridge Strain Gage, 8-Channel C Series Strain/Bridge Input Module
NI-9237	50 kS/s/channel, Bridge Analog Input, 4-Channel C Series Strain/Bridge Input Module
NI-9238	±500 mV, 50 kS/s/ch, 24-Bit, Simultaneous Input, 4-Channel C Series Voltage Input Module
NI-9239	±10 V, 50 kS/s/ch, 24-Bit, Simultaneous Input, 4-Channel C Series Voltage Input Module
NI-9242	250 Vrms L-N, 400 Vrms L-L, 50 kS/s/ch, 24-Bit, 3-Phase C Series Voltage Input Module
NI-9244	400 Vrms L-N, 690 Vrms L-L, 50 kS/s/ch, 24-Bit, 3-Phase C Series Voltage Input Module
NI-9246	50 kS/s/ch, 20 Arms, 30 Apk, 24-Bit, 3-Channel C Series Current Input Module
NI-9247	50 kS/s/ch, 50 Arms, 147 Apk, 24-Bit, 3-Channel C Series Current Input Module
NI-9250	2-Channel, 102.4 kS/s/ch Simultaneous, ±5 V, C Series Sound and Vibration Input Module
NI-9251	3 Vrms, 102.4 kS/s/ch Simultaneous, 2-Channel C Series Voltage Input Module
NI-9252	±10 V, 50 kS/s/ch, 24-Bit, 8-Channel C Series Voltage Input Module
NI-9253	50 kS/s/ch, ±20 mA, 24-Bit, 8-Channel C Series Current Input Module
NI-9313	16 channel, 1 S/s per channel, ±78 mV C Series Temperature Input Module
NI-9320	±10 V, 200 kS/s/ch, 16-Bit, Simultaneous Input, 16-Channel C Series Voltage Input Module
NI-9381	8 AI, 8 AO, 4 DIO, 0 V to 5 V, C Series Multifunction I/O Module

Converting and Calibrating CompactRIO Analog Input Values (FPGA Interface)

Configure the calibration mode for an analog input module using the C Series Module Properties dialog box for the module. Available options are Calibrated and Raw.

- The Calibrated option configures the FPGA I/O Node to return calibrated, fixed-point data for the analog input module.
- The Raw option configures the FPGA I/O Node to return uncalibrated, binary values for the following analog input modules.

Table 36. Modules that Return Uncalibrated, Binary Data (Calibrated Mode = Raw)

• NI 9201 • NI 9203 • NI 9204 • NI 9205 • NI 9206 • NI 9215 • NI 9216 • NI 9217[3] • NI 9218	• NI 9220 • NI 9221 • NI 9222 • NI 9223 • NI 9225 • NI 9226 • NI 9227 • NI 9229 • NI 9230	• NI 9232 • NI 9234 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9246 • NI 9247 • NI 9381

When Calibrated is set to Raw, you must convert the binary values into meaningful engineering units and apply calibration constants. for higher accuracy. Convert and calibrate the values in the host VI.

The binary data returned by some modules is already calibrated. The following table lists the modules that return calibrated, binary data.

Table 37. Modules that Return Calibrated, Binary Data

• NI 9202 • NI 9207	• NI 9219 • NI 9224	• NI 9251 • NI 9252

3. After converting binary values to engineering units for the NI-9217, you can convert the nominal resistance values to temperature.

• NI 9208 • NI 9211 • NI 9212 • NI 9213 • NI 9214	• NI 9228 • NI 9231 • NI 9242 • NI 9244 • NI 9250	• NI 9253 • NI 9313 • NI 9320

To convert the calibrated, binary values to engineering units, refer to the *Converting <module> Data (FPGA Interface)* topic for each hardware module.

Using a VI to Convert and Calibrate Values (NI-9204, NI-9205, and NI-9206)

When using the NI-9204, NI-9205, and NI-9206, refer to the Binary to Nominal Polynomial (Host) VI in the LabVIEW example program below. This example uses the straight-line conversion algorithm with the NI-MCal linearization correction to convert binary analog input values to calibrated engineering units.

Use the Binary to Nominal Polynomial (Host) VI as a subVI in the host VI.

LabVIEW Example Program

```
labview\examples\CompactRIO\Module Specific\NI
9204\<module> Linearization Coefficients\NI 9204
Linearization Coefficients.lvproj
```

where

- <module> is `NI 9204`, `NI 9205`, or `NI 9206`.

Using an Equation to Convert and Calibrate Values

Use the following equation in the host VI to convert binary analog input values to calibrated engineering units:

- $\text{Input Engineering Units} = (\text{Binary Value} \times \text{LSB Weight} - \text{Offset})^4$

Where

- `Binary Value` is the signed or unsigned value returned by the FPGA I/O Node
- `LSB Weight` is the value returned by the LSB Weight property
- `Offset` is the value returned by the Offset property.

The `LSB Weight` and `Offset` units differ per module.

To convert to calibrated engineering units, use the **FPGA I/O Property Node** to read the *LSB Weight* and *Offset* properties. If you do not want to read the LSB Weight and Offset values from the module, convert to uncalibrated engineering units. Use the following values for Offset and LSB Weight:

$$\texttt{Offset} = 0^{5}$$

$$\texttt{LSB Weight} = \texttt{Typical Input Span} \div 2^{\texttt{ADC Resolution}}$$

Where

- `Typical Input Span` value is listed in the following table for each module.
- `ADC Resolution` value is the ADC resolution of the module. Refer to module documentation to obtain this value.

Table 38. C Series Module Typical Input Span

Module	Typical Input Span
NI-9201	21.06 V
NI-9203	21.56 mA (unipolar), 43.12 mA (bipolar)
NI-9205	20.8 V
NI-9206	21.5 V
NI-9215	20.8 V
NI-9216	100 Ω RTD
NI-9217	100 Ω RTD

4. When converting and calibrating data acquired from the NI-9203 in ±20 mA range, the equation is Input Engineering Units = ((Binary Value – 32768) × LSB Weight – Offset).
5. When calculating engineering units for the NI-9203 in ±20 mA range, Offset = 20 mA.

Module	Typical Input Span
NI-9218	Varies by mode; refer to the following table *NI-9218 Typical Input Span*
NI-9220	20.8 V
NI-9221	125 V
NI-9222	21.2 V
NI-9223	21.2 V
NI-9225	850 V
NI-9226	1,000 Ω RTD
NI-9227	29.954 A
NI-9229	125.28 V
NI-9230	63 V
NI-9232	63 V
NI-9234	10.2 V
NI-9235	52.6 mV/V
NI-9236	52.6 mV/V
NI-9237	50 mV/V
NI-9238	1.25 V
NI-9239	21.04 V
NI-9246	62.5 A
NI-9247	294 A
NI-9381	5 V

Table 39. NI-9218 Typical Input Span

Channel Mode	Typical Measurement Range
±16 V	±16.3 V
±20 mA	24.4 mA
±22 mV/V Bridge	22.1 mV/V
±5 V IEPE	5.33 V

Channel Mode	Typical Measurement Range
±60 V	±62.1 V
±65 mV	73.5 mV

Detecting Out-of-Range Channels for CompactRIO Analog Input Channels (FPGA Interface)

You can detect an out-of-range CompactRIO analog input channel when you *read* the channel.

If you set **Calibration Mode** to **Calibrated** in the **C Series Module Properties** dialog box, and the **FPGA I/O Node** returns a value greater than the minimum operating range value, the channel might be out of range.

Note Refer to the *Specifications* document for your module for more information about operating ranges.

The FPGA I/O Node returns the full-scale binary value if a channel is out of range and **Calibration Mode** is set to **Raw**.

Table 40. C Series AI Module Full-Scale Binary Value

Module	Full-Scale Binary Value
NI 9201	–2,048 (0xF800) or 2,047 (0x7FF)
NI 9202[*]	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9203	0 (0x0000) or 65,535 (0xFFFF)
NI 9204[*]	–32,768 (0x8000) or 32,767 (0x7FFF)
NI 9205[*]	–32,768 (0x8000) or 32,767 (0x7FFF)
NI 9206[*]	–32,768 (0x8000) or 32,767 (0x7FFF)
NI 9207	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9208	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9209	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)

Module	Full-Scale Binary Value
NI 9210	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9211	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9212	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9213	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9214	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9215	–32,768 (0x8000) or 32,767 (0x7FFF)
NI 9217	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9218	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9219	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9220	–32,768 (0x8000) or 32,767 (0x7FFF)
NI 9221	–2,048 (0xF800) or 2,047 (0x7FF)
NI 9222	–32,768 (0x8000) or 32,767 (0x7FFF)
NI 9223	–32,768 (0x8000) or 32,767 (0x7FFF)
NI 9224[*]	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9225	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9227	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9228[*]	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9229	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9230	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9231[*]	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9232	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9234	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9235	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9236	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9237	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9238	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9239	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)

Module	Full-Scale Binary Value
NI 9242[*]	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9244[*]	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9246	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9247	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9250[*]	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9251[*]	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9252[*]	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9253[*]	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9313	–8,388,608 (0xFF800000) or 8,388,607 (0x7FFFFF)
NI 9320	–32,768 (0x8000) or 32,767 (0x7FFF)
NI 9381	0 (0x0000) or 4,095 (0x0FFF)

[*]For the NI 9202/9204/9205/9206/9224/9228/9231/9242/9244/9250/9251/9252/9253, a channel might be out of range before the FPGA I/O Node reaches the minimum or maximum full-scale binary value.

Synchronizing Multiple Modules (FPGA Interface)

Learn how to synchronize multiple C Series modules connected to the same FPGA. Find equations for calculating the time-to-first-sample and sampling guidance for supported delta-sigma modules with different data rates.

You can synchronize multiple modules that are connected to the same FPGA device when your application meets the following requirements:

- The modules use the same master timebase source
- The modules start acquisition mode at the same time
- A single FPGA I/O Node function reads the synchronous data

Create FPGA I/O items for the module before you can configure the items using the FPGA I/O Node. Develop the FPGA VI to meet the guidelines described in the following

table.

For delta-sigma modules, you must synchronize multiple sample rates.

Table 41. Synchronizing Multiple Sample Rates

Guideline	Details
Share a master timebase source	Configure the modules to share master timebase source.
Start the synchronized acquisition	• Configure an FPGA I/O Node with **Start** channels for the module you want to synchronize. • Wire a Boolean constant set to TRUE to each **Start** channel. • Ensure that all I/O channels are in the same FPGA I/O Node. Otherwise, the FPGA I/O Node will not return synchronized data.
Acquire data from synchronized modules with the same data rate	• Configure an FPGA I/O Node with all of the channels from which you want to synchronously sample. • Ensure that all I/O channels are in the same FPGA I/O Node. Otherwise, the FPGA I/O Node will not return synchronized data. • Refer to the Synchronizing NI 923x Modules (FPGA) VI in `labview\examples\CompactRIO\Module Specific\NI 923x\Synchronizing NI 923x Modules\Synchronizing NI 923x Modules.lvproj` for an example of synchronizing multiple modules with the same data rate.
Acquire data from synchronized modules with different data rates	• If you synchronize modules that are configured for different data rates, create a *separate loop* for each data rate in the FPGA VI. • In each loop, configure an FPGA I/O Node with all of the channels that are configured for the data rate of that loop. If you place channels that are configured for different rates in the same loop, LabVIEW returns an overrun warning, *error 65539*, from the FPGA I/O Node when you run the VI. • A delay occurs before the FPGA I/O Node returns the first data point. The length of the delay depends on the data rate and model number of the module. The equation for the delay is provided in the following table.

Synchronizing Multiple NI 9231, NI 9250, and NI 9251 Modules

Use the following table to find the equation for synchronizing multiple sample rates for these modules:

- NI 9231
- NI 9250
- NI 9251

Table 42. Equation for Synchronizing Multiple Sample Rates for the NI 9231, NI 9250, and NI 9251

Decimation Rate, *m*	Clock Divider, *n*	Time to First Sample (s)
32	*n* = 1	(281.625 * *m* * *n* + 5.5) * *Master_Timebase_Period* ± 1 *Master_Timebase_Period*
64	*n* = 1	(281.625 * *m* * *n* + 8.5) * *Master_Timebase_Period* ± 1 *Master_Timebase_Period*
128, ..., 1,024	*n* = 1	(281.625 * *m* * *n* + 6.5) * *Master_Timebase_Period* ± 1 *Master_Timebase_Period*
32, ..., 1,024	*n* = 2, ..., 12	(281.625 * *m* * *n* + 5.5) * *Master_Timebase_Period* ± 1 *Master_Timebase_Period*

Synchronizing Multiple NI 9218, NI 9225, NI 9227, NI 9229, NI 9234, NI 9237, NI 9238, NI 9239, NI 9242, NI 9244, NI 9246, NI 9247 Modules

Use the following table to find the equation for synchronizing multiple sample rates for these modules:

- NI 9218
- NI 9225
- NI 9227
- NI 9229
- NI 9234
- NI 9237
- NI 9238

- NI 9239
- NI 9242
- NI 9244
- NI 9246
- NI 9247

Table 43. Equation for Synchronizing Multiple Sample Rates for the NI 9218, NI 9225, NI 9227, NI 9229, NI 9234, NI 9237, NI 9238, NI 9239, NI 9242, NI 9244, NI 9246, NI 9247

Decimation Rate, *m*	**Clock Divider, *n***	**Time to First Sample (s)**
256	*n* = 1	(8.5 + *n* * 34,152) * *Master_Timebase_Period*
256	*n* = 2, ..., 31	(5.5 + *n* * 34,152) * *Master_Timebase_Period*
64	*n* = 1, ..., 31	77/*Fs* + 4-5 *Master_Timebase_Period*

Synchronizing Multiple NI 9230 and NI 9232 Modules

Use the following table to find the equation for synchronizing multiple sample rates for these modules:

- NI 9230
- NI 9232

Table 44. Equation for Synchronizing Multiple Sample Rates for the NI 9230 and NI 9232

Decimation Rate, *m*	**Clock Divider, *n***	**Time to First Sample (s)**
128	n = 1, ..., 31	72/*Fs* + 4-5 *Master_Timebase_Period*
256	n = 1, ..., 31	68/*Fs* + 4-5 *Master_Timebase_Period*

Synchronizing Multiple NI 9235 and NI 9236 Modules

Use the following table to find the equation for synchronizing multiple sample rates for these modules:

- NI 9235
- NI 9236

Table 45. Equation for Synchronizing Multiple Sample Rates for the NI 9235 and NI 9236

Decimation Rate, *m*	Clock Divider, *n*	Time to First Sample (s)
512	n = 1, ..., 31	39/*Fs* + 110.5 MClk + 1 OClk + *Analog Delay*

- The *Master_Timebase_Period* = period of the internal or external clock that the module uses (1/13.1072 MHz, 1/12.8 MHz, or 1/10 MHz)
- *Fs* = sample rate. The divider and decimation rate depend on which sample rate you select

The module *Specifications* document provides an equation to help you determine the clock divider and decimation rate and/or lists the clock divider and decimation rate for each sample rate.

- For example, the *NI-9232 Specifications* lists the decimation rate as 64 and clock divider as 1 when sampling at 102.4 kHz.
- Using the equation in the *NI-9218 Specifications*, you can determine that the clock divider is 1 when the sample rate is 51.2 kHz. *n* = Master_Timebase_Frequency/256/*fs* [6]

Synchronizing Multiple NI 9202 Modules

Use the following table to find the equation for synchronizing multiple sample rates for NI 9202 modules.

Table 46. Equation for Synchronizing Multiple Sample Rates for the NI 9202

Clock Divider, *b*	Time to First Sample (s)
1	((5.40625 + *e*) * *a* * *b* * *c* * *d* + 4.5) * *Master_Timebase_Period* ± 1 *Master_Timebase_Period*
2, ..., 11	((5.40625 + *e*) * *a* * *b* * *c* * *d* + 5.5) * *Master_Timebase_Period* ± 1 *Master_Timebase_Period*

- The *Master_Timebase_Period* = period of the internal or external clock that the module uses (1/13.1072 MHz, 1/12.8 MHz, or 1/10 MHz)
- *a* = ADC Decimation Rate
- *b* = Timebase Clock Divider

6. This is the equation provided in the *NI-9218 Specifications* reorganized to solve for *n*.

- *c* = ADC Clock Divider
- *d* = Filter Decimation Rate
- *e* = Filter Frequency
  - 0 for notch at *Fs*
  - 1 for notch at *Fs*/2
  - 3 for notch at *Fs*/4
  - 7 for notch at *Fs*/8
  - 15 for notch at *Fs*/16

Refer to the *NI-9202 Specifications* for the values of *a*, *b*, *c*, and *d* for each sample rate.

Synchronizing Multiple NI 9252 and NI 9253 Modules

Use the following table to find the equation for synchronizing multiple sample rates for these modules:

- NI 9252
- NI 9253

Table 47. Equation for Synchronizing Multiple Sample Rates for the NI 9252 and NI 9253

Filter	Clock Divider, *n*	Time to First Sample (s)
Comb	2	((5.40625 + *b*) * *128* * *a* + 8.5) * *Master_Timebase_Period* ± 1 *Master_Timebase_Period*
Comb	Others	((5.40625 + *b*) * *128* * *a* + 8.5) * *Master_Timebase_Period* ± 1 *Master_Timebase_Period*
Butterworth	2	(5.40625 * *128* * *a* + 5.5) * *Master_Timebase_Period* ± 1 *Master_Timebase_Period*
Butterworth	Others	(5.40625 * *128* * *a* + 5.5) * *Master_Timebase_Period* ± 1 *Master_Timebase_Period*

- The *Master_Timebase_Period* = period of the internal or external clock that the module uses (1/13.1072 MHz, 1/12.8 MHz, or 1/10 MHz)
- *a* = Decimation Rate

- *b* = Filter Frequency
  - 0 for Filter Frequency—1 (notch at Fs)
  - 1 for Filter Frequency—2 (notch at Fs/2)
  - 3 for Filter Frequency—3 (notch at Fs/4)
  - 7 for Filter Frequency—4 (notch at Fs/8)
  - 15 for Filter Frequency—5 and 6 (notch at Fs/16)

Refer to the *NI 9252 Specifications* and *NI 9253 Specifications* for the values of (Decimation Rate) *a* for each sample rate.

Related concepts:

- Configuring the Master Timebase Source for a C Series Module (FPGA Interface)
- Understanding Loop Timing (FPGA Interface)
- Understanding the Maximum Sample Rate When Synchronizing Multiple Modules (FPGA Interface)

Configuring the Master Timebase Source for a C Series Module (FPGA Interface)

C Series modules with a selectable timebase source are listed in the following table.

Table 48. C Series Modules with a Selectable Timebase Source

• NI 9202 • NI 9218 • NI 9225 • NI 9227 • NI 9229 • NI 9230 • NI 9231	• NI 9232 • NI 9234 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239	• NI 9242 • NI 9244 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9252	• NI 9253 • NI 9260 • NI 9469 • NI 9770 • NI 9775 • NI 9470

In a system that contains more than one of these modules, you can configure one module as the master timebase source (*master*) and configure the other modules to use that master timebase source (*slaves*).

Sharing the same master timebase source allows you to synchronize multiple

modules. The master timebase source is divided to acquire data at the data rate you configure.

Note The NI 9151 R Series Expansion chassis does not support synchronizing multiple modules.

Note The NI 9770 does not support synchronization with other non-NI 9770 modules.

Note You can only use another NI 9775 or the NI 9469 as the master timebase source for the NI 9775. If you select the NI 9469, you can only use the 12.8 MHz clock as the exported clock frequency.

Configuring the Onboard Clock of a Module as the Master Timebase Source

Complete the following steps to configure the internal Onboard Clock of one module as the master timebase source.

1. Configure the CompactRIO system and add a module.
2. Right-click the module in the **Project Explorer** window and select **Properties** to launch the **C Series Module Properties** dialog box.
3. Select **<Onboard Clock>** from the **Master Timebase Source** pull-down menu.
4. Click the **OK** button.
5. Select **File»Save All** in the **Project Explorer** window.

Configuring a Master Module to Share the Master Timebase Source with Slaves

Complete the following steps to configure the master timebase source for a master module and share that timebase with slave modules.

1. Configure the CompactRIO system and add the module to use as the master.
2. Right-click the module in the **Project Explorer** window and select **Properties** to launch the **C Series Module Properties** dialog box.
3. Select **<Onboard Clock>** from the **Master Timebase Source** pull-down menu.

4. Enable the **Export Onboard Clock** checkbox.
5. Add a module you want to use as a slave.
6. Right-click the slave module in the **Project Explorer** window and select **Properties** to display the **C Series Module Properties** dialog box.
7. Select the name of the master module from the **Master Timebase Source** pull-down menu.
8. Repeat steps 5 through 7 for each slave module you want to configure.
9. Click the **OK** button.
10. Select **File»Save All** in the **Project Explorer** window.

Related concepts:

- Configuring the Data Rate for a Module (FPGA Interface)

Understanding the Maximum Sample Rate When Synchronizing Multiple Modules (FPGA Interface)

The following C Series modules derive their sampling rate from the internal master timebase.

To show information about the internal master timebase and maximum sample rate for a specific module, select the module from the following list.

Related concepts:

- Configuring the Master Timebase Source for a C Series Module (FPGA Interface)
- Synchronizing Multiple Modules (FPGA Interface)

NI 9202

The NI 9202 derives its sample rate from the internal master timebase.

Table 49. NI 9202 internal master timebase and sample rate

Internal master timebase	12.8 MHz
Maximum sample rate when using the internal master timebase	10 kS/s (12.8 MHz/1280)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9202 maximum sample rate when it is configured to use the master timebase source.

Table 50. NI 9202 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9202 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	10 kS/s
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250	10.24 kS/s

Master module	Slave NI 9202 maximum sample rate
• NI 9251 • NI 9260	

NI 9218

The NI 9218 derives its sample rate from the internal master timebase.

Table 51. NI 9218 internal master timebase and sample rate

Internal master timebase	13.1072 MHz
Maximum sample rate when using the internal master timebase	51.2 kS/s (13.1072 MHz/256)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9218 maximum sample rate when it is configured to use the master timebase source.

Table 52. NI 9218 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9218 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236	50 kS/s

Master module	Slave NI 9218 maximum sample rate
• NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	51.2 kS/s

NI 9225

The NI 9225 derives its sample rate from the internal master timebase.

Table 53. NI 9225 internal master timebase and sample rate

Internal master timebase	12.8 MHz
Maximum sample rate when using the internal master timebase	50 kS/s (12.8 MHz/256)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The

maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9225 maximum sample rate when it is configured to use the master timebase source.

Table 54. NI 9225 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9225 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	50 kS/s
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	51.2 kS/s

NI 9227

The NI 9227 derives its sample rate from the internal master timebase.

Table 55. NI 9227 internal master timebase and sample rate

Internal master timebase	12.8 MHz
Maximum sample rate when using the internal master timebase	50 kS/s (12.8 MHz/256)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9227 maximum sample rate when it is configured to use the master timebase source.

Table 56. NI 9227 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9227 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470	50 kS/s

Master module	Slave NI 9227 maximum sample rate
• NI 9775	
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	51.2 kS/s

NI 9229

The NI 9229 derives its sample rate from the internal master timebase.

Table 57. NI 9229 internal master timebase and sample rate

Internal master timebase	12.8 MHz
Maximum sample rate when using the internal master timebase	50 kS/s (12.8 MHz/256)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9229 maximum sample rate when it is configured to use the master timebase source.

Table 58. NI 9229 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9229 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	50 kS/s
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	51.2 kS/s

NI 9230

The NI 9230 derives its sample rate from the internal master timebase.

Table 59. NI 9230 internal master timebase and sample rate

Internal master timebase	13.1072 MHz
Maximum sample rate when using the internal	12.8 kS/s (13.1072 MHz/(128×8))

master timebase	

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9230 maximum sample rate when it is configured to use the master timebase source.

Table 60. NI 9230 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9230 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	12.5 kS/s
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234	12.8 kS/s

Master module	Slave NI 9230 maximum sample rate
• NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	

NI 9231

The NI 9231 derives its sample rate from the internal master timebase.

Table 61. NI 9231 internal master timebase and sample rate

Internal master timebase	13.1072 MHz
Maximum sample rate when using the internal master timebase	51.2 kS/s (13.1072 MHz/256)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9231 maximum sample rate when it is configured to use the master timebase source.

Table 62. NI 9231 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9231 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229	50 kS/s

Master module	Slave NI 9231 maximum sample rate
• NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	51.2 kS/s

NI 9232

The NI 9232 derives its sample rate from the internal master timebase.

Table 63. NI 9232 internal master timebase and sample rate

Internal master timebase	13.1072 MHz
Maximum sample rate when using the internal master timebase	102.4 kS/s (13.1072 MHz/128)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9232 maximum sample rate when it is configured to use the master timebase source.

Table 64. NI 9232 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9232 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	100 kS/s
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	102.4 kS/s

NI 9234

The NI 9234 derives its sample rate from the internal master timebase.

Table 65. NI 9234 internal master timebase and sample rate

Internal master timebase	13.1072 MHz
Maximum sample rate when using the internal master timebase	51.2 kS/s (13.1072 MHz/256)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9234 maximum sample rate when it is configured to use the master timebase source.

Table 66. NI 9234 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9234 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470	50 kS/s

Master module	Slave NI 9234 maximum sample rate
• NI 9775	
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	51.2 kS/s

NI 9235

The NI 9235 derives its sample rate from the internal master timebase.

Table 67. NI 9235 internal master timebase and sample rate

Internal master timebase	12.8 MHz
Maximum sample rate when using the internal master timebase	10 kS/s (12.8 MHz/(256×5))

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9235 maximum sample rate when it is configured to use the master timebase source.

Table 68. NI 9235 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9235 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	10 kS/s
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	10.24 kS/s

NI 9236

The NI 9236 derives its sample rate from the internal master timebase.

Table 69. NI 9236 internal master timebase and sample rate

Internal master timebase	12.8 MHz
Maximum sample rate when using the internal	10 kS/s (12.8 MHz/(256×5))

master timebase	

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9236 maximum sample rate when it is configured to use the master timebase source.

Table 70. NI 9236 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9236 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	10 kS/s
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234	10.24 kS/s

Master module	Slave NI 9236 maximum sample rate
• NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	

NI 9237

The NI 9237 derives its sample rate from the internal master timebase.

Table 71. NI 9237 internal master timebase and sample rate

Internal master timebase	12.8 MHz
Maximum sample rate when using the internal master timebase	50 kS/s (12.8 MHz/256)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9237 maximum sample rate when it is configured to use the master timebase source.

Table 72. NI 9237 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9237 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229	50 kS/s

Master module	Slave NI 9237 maximum sample rate
• NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	51.2 kS/s

NI 9238

The NI 9238 derives its sample rate from the internal master timebase.

Table 73. NI 9238 internal master timebase and sample rate

Internal master timebase	12.8 MHz
Maximum sample rate when using the internal master timebase	50 kS/s (12.8 MHz/256)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9238 maximum sample rate when it is configured to use the master timebase source.

Table 74. NI 9238 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9238 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	50 kS/s
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	51.2 kS/s

NI 9239

The NI 9239 derives its sample rate from the internal master timebase.

Table 75. NI 9239 internal master timebase and sample rate

Internal master timebase	12.8 MHz
Maximum sample rate when using the internal master timebase	50 kS/s (12.8 MHz/256)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9239 maximum sample rate when it is configured to use the master timebase source.

Table 76. NI 9239 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9239 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470	50 kS/s

Master module	Slave NI 9239 maximum sample rate
• NI 9775	
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	51.2 kS/s

NI 9242

The NI 9242 derives its sample rate from the internal master timebase.

Table 77. NI 9242 internal master timebase and sample rate

Internal master timebase	12.8 MHz
Maximum sample rate when using the internal master timebase	50 kS/s (12.8 MHz/256)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9242 maximum sample rate when it is configured to use the master timebase source.

Table 78. NI 9242 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9242 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	50 kS/s
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	51.2 kS/s

NI 9244

The NI 9244 derives its sample rate from the internal master timebase.

Table 79. NI 9244 internal master timebase and sample rate

Internal master timebase	12.8 MHz
Maximum sample rate when using the internal	50 kS/s (12.8 MHz/256)

master timebase	

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9244 maximum sample rate when it is configured to use the master timebase source.

Table 80. NI 9244 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9244 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	50 kS/s
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234	51.2 kS/s

Master module	Slave NI 9244 maximum sample rate
• NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	

NI 9246

The NI 9246 derives its sample rate from the internal master timebase.

Table 81. NI 9246 internal master timebase and sample rate

Internal master timebase	13.1072 MHz
Maximum sample rate when using the internal master timebase	51.2 kS/s (13.1072 MHz/256)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9246 maximum sample rate when it is configured to use the master timebase source.

Table 82. NI 9246 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9246 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229	50 kS/s

Master module	Slave NI 9246 maximum sample rate
• NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	51.2 kS/s

NI 9247

The NI 9247 derives its sample rate from the internal master timebase.

Table 83. NI 9247 internal master timebase and sample rate

Internal master timebase	13.1072 MHz
Maximum sample rate when using the internal master timebase	51.2 kS/s (13.1072 MHz/256)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9247 maximum sample rate when it is configured to use the master timebase source.

Table 84. NI 9247 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9247 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	50 kS/s
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	51.2 kS/s

NI 9250

The NI 9250 derives its sample rate from the internal master timebase.

Table 85. NI 9250 internal master timebase and sample rate

Internal master timebase	13.1072 MHz
Maximum sample rate when using the internal master timebase	102.4 kS/s (13.1072 MHz/128)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9250 maximum sample rate when it is configured to use the master timebase source.

Table 86. NI 9250 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9250 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470	100 kS/s

Master module	Slave NI 9250 maximum sample rate
• NI 9775	
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	102.4 kS/s

NI 9251

The NI 9251 derives its sample rate from the internal master timebase.

Table 87. NI 9251 internal master timebase and sample rate

Internal master timebase	13.1072 MHz
Maximum sample rate when using the internal master timebase	102.4 kS/s (13.1072 MHz/128)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9251 maximum sample rate when it is configured to use the master timebase source.

Table 88. NI 9251 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9251 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	100 kS/s
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	102.4 kS/s

NI 9252

The NI 9252 derives its sample rate from the internal master timebase.

Table 89. NI 9252 internal master timebase and sample rate

Internal master timebase	12.8 MHz
Maximum sample rate when using the internal	50 kS/s (12.8 MHz/256)

master timebase	

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9252 maximum sample rate when it is configured to use the master timebase source.

Table 90. NI 9252 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9252 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	50 kS/s
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234	51.2 kS/s

Master module	Slave NI 9252 maximum sample rate
• NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	

NI 9253

The NI 9253 derives its sample rate from the internal master timebase.

Table 91. NI 9253 internal master timebase and sample rate

Internal master timebase	12.8 MHz
Maximum sample rate when using the internal master timebase	50 kS/s (12.8 MHz/256)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9253 maximum sample rate when it is configured to use the master timebase source.

Table 92. NI 9253 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9253 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229	50 kS/s

Master module	Slave NI 9253 maximum sample rate
• NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	51.2 kS/s

NI 9260

The NI 9260 derives its sample rate from the internal master timebase.

Table 93. NI 9260 internal master timebase and sample rate

Internal master timebase	13.1072 MHz
Maximum sample rate when using the internal master timebase	51.2 kS/s (13.1072 MHz/256)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9260 maximum sample rate when it is configured to use the master timebase source.

Table 94. NI 9260 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9260 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470 • NI 9775	50 kS/s
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	51.2 kS/s

NI 9470

The NI 9470 derives its sample rate from the internal master timebase.

Table 95. NI 9470 internal master timebase and sample rate

Internal master timebase	12.8 MHz
Maximum sample rate when using the internal master timebase	3.200 kS/s (12.8 MHz/4000)

When the module uses the internal master timebase, it can not be sourced to different model types for synchronization.

When synchronizing multiple modules, the maximum sample rate of the slave module depends on the frequency of the master module's internal master timebase. The maximum sample rate of the system is the lowest sample rate of all the modules being synchronized.

The following table lists the NI 9470 maximum sample rate when it is configured to use the master timebase source.

Table 96. NI 9470 maximum sample rate when synchronizing multiple modules

Master module	Slave NI 9470 maximum sample rate
• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9233 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9252 • NI 9253 • NI 9470	3.200 kS/s

Master module	Slave NI 9470 maximum sample rate
• NI 9775	
• NI 9218 • NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9260	3.277 kS/s

Acquiring Data from a Module (FPGA Interface)

The information in this topic applies to analog input modules with a master timebase.

Table 97. Analog Input Modules with a Master Timebase

• NI 9202 • NI 9218 • NI 9225 • NI 9227 • NI 9229 • NI 9230 • NI 9231 • NI 9232	• NI 9234 • NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244	• NI 9246 • NI 9247 • NI 9250 • NI 9251 • NI 9252 • NI 9253 • NI 9770 • NI 9775

The module input channels are sampled simultaneously at the data rate configured for the module. Use the **Start** and **Stop** channels to put the module in and out of acquisition mode.

A module that is in acquisition mode can only acquire data. When a module is in acquisition mode, you cannot perform other operations, such as:

- You can not access module properties when a module is in acquisition mode.
- You can not access TEDS information, if supported, when a module is in acquisition mode.

Use the FPGA I/O Node to read the data from the module.



Note Create FPGA I/O items for the module before you can configure the items using the FPGA I/O Node.

Example

The module Getting Started (FPGA) VI at `labview\examples\CompactRIO\Module Specific\<module>\<module> Getting Started` shows how to read from a module.

Putting the Module in Acquisition Mode

Complete the following steps to put the module in acquisition mode.

1. Configure an FPGA I/O Node with the **Start** channel of the module.
2. Wire a Boolean constant set to *TRUE* to the **Start** input of the FPGA I/O Node.
   A synchronization pulse is sent to the module. The module starts acquiring data at the data rate you configure.

Reading Data from the Module

When the module starts acquiring data, you can use an FPGA I/O Node to read data from the module.

Connect the AI output of the FPGA I/O Node [7] to various types of functions, including an FPGA Memory function or an FPGA FIFO function.

To ensure that the VI reads the data synchronously when reading multiple channels on the module, place the channels in the same FPGA I/O Node.

The module internally acquires data at a specified rate. The FPGA I/O Node does not return data until new data has been acquired by the module.

7. For the NI 9770, connect the RF In/I or RF In/Q items.

Note If the module did not start acquiring data or stops acquiring data while an FPGA I/O Node is waiting for data from the module, the FPGA I/O Node returns *I/O Not Start error 65582*.

Exiting Acquisition Mode

Complete the following steps to exit acquisition mode.

1. Configure an FPGA I/O Node with the **Stop** channel of the module.
2. Write a *TRUE* to the **Stop** input.
   The module is no longer in acquisition mode. You can now access its properties or, for some modules, TEDS information.

Configuring the Data Rate for a Module (FPGA Interface)

Table 98. C Series Modules with a Configurable Data Rate

• NI 9202 • NI 9218 • NI 9225 • NI 9227 • NI 9229	• NI 9230 • NI 9231 • NI 9232 • NI 9234 • NI 9235	• NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242	• NI 9244 • NI 9246 • NI 9247 • NI 9250 • NI 9251	• NI 9252 • NI 9253 • NI 9260 • NI 9770 • NI 9775

You can configure the rate at which the module acquires and returns data at edit time using the C Series Module Properties dialog box.

To programmatically change the data rate at run time, use the FPGA I/O Property Node. The execution of an I/O Property Node that is configured with a **Data Rate** property overwrites the value you configured in the C Series Module Properties dialog box.

Related concepts:

- Loop Timing for the NI 9218 (FPGA Interface)

Configuring the Data Rate Using the C Series Module Properties Dalog Box

Complete the following steps to configure the data rate for the module using the **C Series Module Properties** dialog box.

1. Configure the CompactRIO system and add the module.
2. Right-click the module in the **Project Explorer** window and select **Properties** to launch the **C Series Module Properties** dialog box.
3. Select the rate from the **Data Rate** pull-down menu.
4. Click the **OK** button.
5. Select **File»Save All** in the **Project Explorer** window.

Configuring the Data Rate Using the FPGA I/O Property Node

Complete the following steps to configure the data rate for the module using the FPGA I/O Property Node.

1. Place an FPGA I/O Property Node on the block diagram and configure it for the module.
2. Click the **Property** section and select **Data Rate** from the shortcut menu.
3. Right-click the **Data Rate** input and select **Create»Control** from the shortcut menu.
4. On the front panel of the VI, select a rate from the **Data Rate** pull-down menu.

You can change the data rate at run time by writing to the control from the host VI. Refer to the module Getting Started (FPGA) VI at `labview\examples\CompactRIO\Module Specific\<module>\<module> Getting Started` for an example of configuring the data rate using the FPGA I/O Property Node.

The **Data Rate** property might return *error 65538* if the module is acquiring data. Use the **Stop** channel to stop acquiring data before you can write properties to the module.

Accessing TEDS Information from a Module (FPGA Interface)

The following table lists the modules that support TEDS-compatible transducers.

Table 99. C Series Modules that Support TEDS-Compatible Transducers

• NI 9218 • NI 9219 • NI 9230	• NI 9231 • NI 9232 • NI 9234	• NI 9237 • NI 9250

Access Transducer Electronic Data Sheet (TEDS) information from TEDS-compatible transducers connected to the channels of the module.

Before you can read TEDS information, enable TEDS support for the module. Access TEDS information only from the host VI.

Example

For an example about how to access TEDS information, refer to the module Getting Started (FPGA) VI in the `labview\examples\CompactRIO\Module Specific\<module>\<module> Getting Started` directory.

Understanding Loop Timing (FPGA Interface)

Table 100. C Series Modules that have an Internal Master Timebase and are Internally Timed

• NI 9202 • NI 9225 • NI 9227 • NI 9229 • NI 9230 • NI 9231 • NI 9232 • NI 9234	• NI 9235 • NI 9236 • NI 9237 • NI 9238 • NI 9239 • NI 9242 • NI 9244 • NI 9246	• NI 9247 • NI 9250 • NI 9251 • NI 9252 • NI 9253 • NI 9260 • NI 9770 • NI 9775

Do not use the Loop Timer function or the Wait function in a loop with an FPGA I/O Node that acquires data from one of these modules.

When you create a loop that reads data from one of these modules, make sure the loop does not execute slower than the data rate of the module. If the loop execution time is slower than the data rate, the FPGA I/O Node returns an overrun warning and

continues to read data. The overrun warning means that the data the FPGA I/O Node returns is valid, but the function missed one or more data points since the last time it read data from the module.

The function returns the overrun warning when all of the following conditions are true:

- The module is in acquisition mode.
- An FPGA I/O Node that is acquiring data from the module executes at least once after you put the module in acquisition mode.
- The FPGA I/O Node did not read one or more data points since the previous time the function executed.

If the application acquires multiple buffers of data from the module and the timing relationship between them is not important, you can ignore the overrun warning returned with the first point of each buffer. Avoiding Overrun Warnings

To avoid overrun warnings, develop the FPGA VI to meet the following guidelines:

Table 101. Guidelines to Avoid Overrun Warnings

Conditions	Guidelines
Your application acquires multiple buffers of data from a module with an internal master timebase.	If the timing relationship between the buffers is not important, you can ignore the overrun warning returned with the first point of each buffer.
You are reading from multiple modules with an internal master timebase in the same loop.	• Use one FPGA I/O Node to read the channels. • Configure the modules to share a master timebase source and have the same data rate.
You are reading from a module with an internal master timebase and another analog input module in the same loop.	• If the rate at which you can acquire data from the other module is as fast or faster than the data rate configured for the module with an internal master timebase, you can read from both modules in the same loop. • If you use the same FPGA I/O Node to read data from all modules, the FPGA I/O Node does not return data for the other module until the

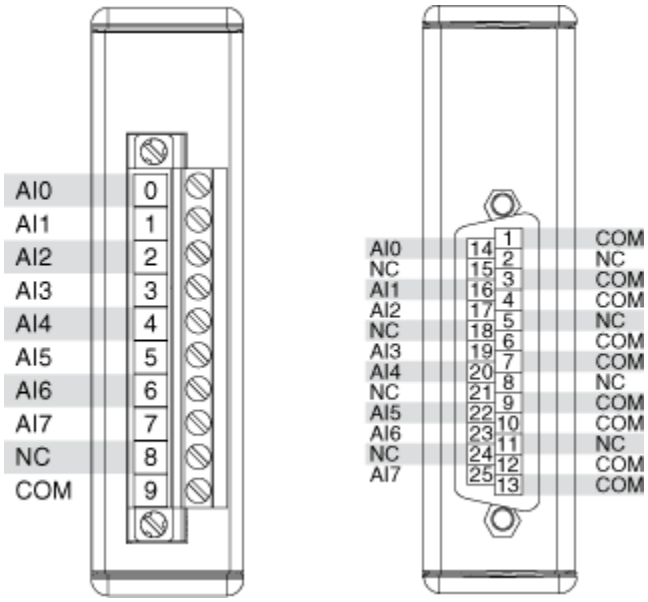
Conditions	Guidelines
	module with an internal master timebase acquires data. • If the other module has a slower data rate than the module with an internal master timebase and you read from both modules in the same loop, the FPGA I/O Node for the module with an internal master timebase returns an overrun warning and continues reading data. To avoid missing data, you can either change the data rate of the module with an internal master timebase or read from each module in a different loop.

NI 9201

CompactRIO 8-Channel, ±10 V, 12-Bit Analog Input Module

NI 9201 Pinout

Figure 21. NI 9201 Pinout



NI 9201 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 102. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9201 has AI channels 0 to 7.

You can configure the minimum time between conversions and the channels to scan.

Arbitration

This device only supports the Arbitrate if Multiple Requestors Only option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

To access the following properties for this device, use the **FPGA I/O Property Node**.

Table 103. I/O Properties

Property	I/O Type	Description
LSB Weight	AO	Returns the LSB weight in nV/LSB for the channel. Use this value to convert data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
Offset	AO	Returns the calibration offset in nV for the channel. Use this value to convert data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.

Module Properties

To access the following module properties for this device, use the FPGA I/O Property Node.

Table 104. Module Properties

Property	Description
Module ID	Returns the module ID, 0x70A4.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Configuring the Minimum Time Between Conversions for the NI 9201/9221 (FPGA Interface)

You can configure the minimum time between conversions for the NI 9201/9221 in the C Series Module Properties dialog box.

Complete the following steps to configure the minimum time between conversions for the NI 9201/9221.

1. Configure the CompactRIO system, and add an NI 9201/9221.
2. Right-click the NI 9201/9221 in the **Project Explorer** window and select **Properties** from the shortcut menu to display the C Series Module Properties dialog box.
3. Enter a value between 0 and 10 µs in increments of 25 ns in the **Minimum Time Between Conversions** text box.
4. Click the **OK** button.
5. Select **File»Save All** in the **Project Explorer** window.

Note The default minimum time between conversions for the NI 9201 is 2 µs, and the default minimum time between conversions for the NI 9221 is 1.25 µs. The accuracy specifications in the NI 9201 and NI 9221 hardware documentation on ni.com/manuals are based on these default values. Refer to the following tables for examples of how the value you enter in the **Minimum Time Between Conversions** text box affects the actual time between conversions and the accuracy of the module.

The following table lists the minimum time between conversions when you are using a chassis *other than* the NI 9151 R Series Expansion chassis.

Table 105. Minimum Time Between Conversions When Using a Chassis Other Than the NI 9151 R Series Expansion Chassis

Module	Sampling Data from a Singleor Channel or Multiple Channels?	Minimum Time Between Conversions	Able to Achieve the Specified Time Between Conversions? (Refer to the following Note)	Module Accuracy
NI 9201	Single	≥ 1.25 µs	Yes	Module accuracy not affected
< 1.25 µs	No	Module accuracy not affected		
Multiple	≥ 2 µs	Yes	Module accuracy not affected	
< 2 µs	No	Module accuracy degrades		
NI 9221	Single	≥ 1.25 µs	Yes	Module accuracy not affected
< 1.25 µs	No	Module accuracy not affected		
Multiple	≥ 1.25 µs	Yes	Module accuracy not affected	
< 1.25 µs	No	Module accuracy degrades		

Set the minimum time between conversions in the C Series Module Properties dialog box.



Note
The actual time between conversions depends on how you develop the FPGA VI:

- *Yes* indicates that it is possible to write an FPGA VI that can sustain the minimum time between conversions specified in the C Series Module Properties dialog box.
- *No* indicates that the minimum time between conversions specified in the C Series Module Properties dialog box is too low and you may not be able to write an FPGA VI that can sustain the specified time.

The following table lists the minimum time between conversions when you are using the NI 9151 R Series Expansion chassis.

Table 106. Minimum Time Between Conversions when using the NI 9151 R Series Expansion Chassis

Module	Sampling Data from a Single Channel or Multiple Channels?	Minimum Time Between Conversions	Able to Achieve the Specified Time Between Conversions? (Refer to the following Note)	Module Accuracy
NI 9201	Single	≥ 2.1 µs	Yes	Module accuracy not affected
< 2.1 µs	No	Module accuracy not affected		
Multiple	≥ 2.1 µs	Yes	Module accuracy not affected	
< 2.1 µs and ≥ 2 µs	No	Module accuracy not affected		
< 2 µs	No	Module accuracy degrades		
NI 9221	Single	≥ 2.1 µs	Yes	Module accuracy not affected
< 2.1 µs	No	Module accuracy not affected		
Multiple	≥ 2.1 µs	Yes	Module accuracy not affected	

Module	Sampling Data from a Single Channel or Multiple Channels?	Minimum Time Between Conversions	Able to Achieve the Specified Time Between Conversions? (Refer to the following Note)	Module Accuracy
< 2.1 µs and ≥ 1.25 µs	No	Module accuracy not affected		
< 1.25 µs	No	Module accuracy degrades		

Set the minimum time between conversions in the C Series Module Properties dialog box.



Note

The actual time between conversions depends on how you develop the FPGA VI:

- *Yes* indicates that it is possible to write an FPGA VI that can sustain the minimum time between conversions specified in the C Series Module Properties dialog box.
- *No* indicates that the minimum time between conversions specified in the C Series Module Properties dialog box is too low and you may not be able to write an FPGA VI that can sustain the specified time.

Understanding NI 9201/9221 Scanning (FPGA Interface)

To scan the channels of the NI 9201/9221, configure an FPGA I/O Node with the channels you want to acquire from the module.

The module implements a pipeline that is automatically managed by the FPGA I/O Node. Channels within the FPGA I/O Node are sampled in numerical order regardless of the order they appear in the node.

The first time an FPGA I/O Node configured with channels on an NI 9201/9221 module executes, the module performs two setup conversions before converting the first

channel. The two setup conversions prime the pipeline for subsequent FPGA I/O Node reads. The module does not repeat the setup conversions unless the FPGA I/O Node channel configuration changes and the pipeline needs to be primed again.

NI 9201 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9201 has the following channels.

Table 107. Module Channels

Channel	Description
AIx	Analog input channel x, where x is the number of the channel. For the NI 9201, x is 0 to 7.

C Series Module Properties Dialog Box

Use the C Series Module Properties dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

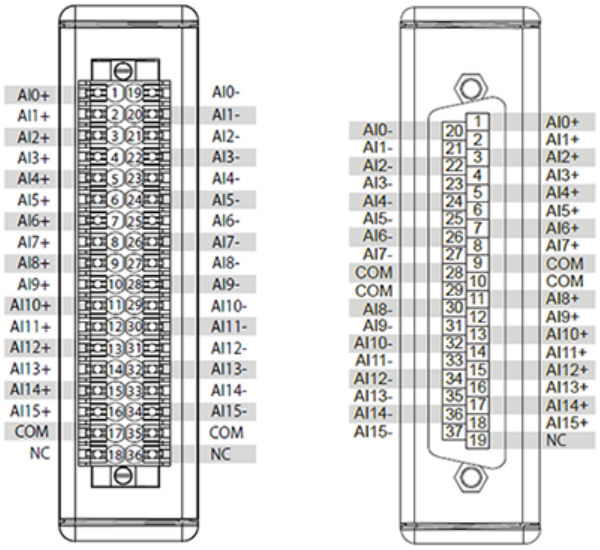
- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.

NI 9202

CompactRIO 16-Channel, 24-Bit Differential Analog Input Module

NI 9202 Pinout

Figure 22. NI 9202 Pinout



NI 9202 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 108. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9202 has channels 0 to 15.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The onboard clock frequency is 12.8 MHz. Use the FPGA I/O Node in a single-cycle timed loop to access this channel. Export the onboard clock of the NI 9202 to access this channel.
Start	Channel that controls when the NI 9202 starts acquiring data. If TRUE is written to the **Start** channel, the NI 9202 starts acquiring data. When the NI 9202 is acquiring data, you must write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9202 stops acquiring data. If TRUE is written to the

Terminal	Description
	Stop channel, the NI 9202 stops acquiring data. When the NI 9202 is acquiring data, you must write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Stop** channel, no operation is performed.

Arbitration

This device only supports the Arbitrate if Multiple Requestors Only option for arbitration. You cannot configure arbitration settings for this device.

Module Properties

To access the following module properties for this device, use the FPGA I/O Property Node.

Table 109. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9202 acquires data.
Module ID	Returns the module ID, 0x788C for the spring terminal and 0x788A for the DSUB variant.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.
Input Delay	Returns the filter delay based on the data rate and filter setting.
Filter Frequency Config	Sets the frequency of the first notch in the filter response.

Note Changes to the filter settings apply to all channels.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Converting NI 9202 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9202 if you want the FPGA I/O Node to return calibrated, fixed-point data from the module.

Set the **Calibration Mode** to **Raw** if you want the FPGA I/O Node to return calibrated, binary data from the module. If you set the **Calibration Mode** to **Raw**, convert the binary values to engineering units. Convert these values in the host VI. For an example, refer to `labview\examples\CompactRIO\Module Specific\NI 9202\NI 9202 Scaling\NI 9202 Raw To Scaled.lvproj`.

The binary values are not uniform across data rates. To get the **Corrected Raw**, multiply the binary value with the **Gain Correction** with respect to the data rate used. For an example, refer to `labview\examples\CompactRIO\Module Specific\NI 9202\NI 9202 Raw Correction\NI 9202 Raw Correction.lvproj`.

Data Rates[1]	**Scaling**	**Gain Correction**
10 kS/s, 5 kS/s	Volts = Binary Value × 2018176 pV/LSB	Corrected Binary Value = Binary Value × 1.6
60 S/s[2]	Volts = Binary Value × 1356704 pV/LSB	Corrected Binary Value = Binary Value × 1.07563
400 S/s, 200 S/s, 100 S/s, 10 S/s[2]	Volts = Binary Value × 1291512 pV/LSB	Corrected Binary Value = Binary Value × 1.024
2 kS/s, 1 kS/s, 500 S/s, 250 S/s, 125 S/s, 50 S/s[2]	Volts = Binary Value × 1614448 pV/LSB	Corrected Binary Value = Binary Value × 1.28
60 S/s[3]	Volts = Binary Value × 2274057 pV/LSB	Corrected Binary Value = Binary Value × 1.802817
10 S/s[3], 50 S/s[3], Others	Volts = Binary Value × 1261244 pV/LSB	Corrected Binary Value = Binary Value × 1

where

- *Binary Value* is the value returned by the FPGA I/O Node.

[1] The actual data rate changes with the master timebase when using an external master timebase. In order to provide power line frequency rejection when using master timebase frequencies of both 12.8 MHz and 13.1072 MHz, there are two sets of settings for the data rates of 10 S/s, 50 S/s and 60 S/s. Refer to the following footnotes for the correct scaling constants and gain corrections, depending on your master timebase frequency.

[2] Data rates setting when using internal master timebase or an external master timebase of 12.8 MHz.

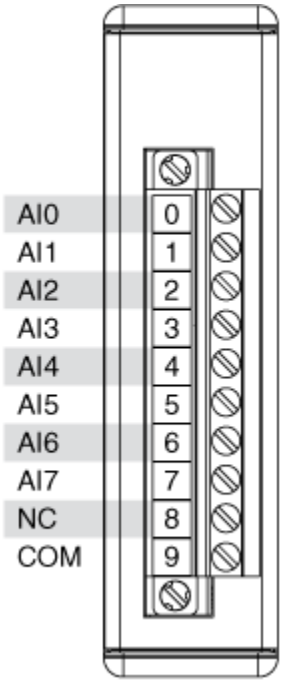
[3] Data rates setting when using an external master timebase of 13.1072 MHz.

NI 9203

CompactRIO 8-Channel, ±20 mA, 16-Bit Analog Input Module

NI 9203 Pinout

Figure 23. NI 9203 Pinout



NI 9203 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

Use the **FPGA I/O Node** to access the following terminals for this device.

Table 110. Terminals in Software

Terminal	Description
AIx	Analog input channel x, where x is the number of the channel. The NI 9203 has AI channels 0 to 7.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

To access the following method for this device, use the **FPGA I/O Method Node**.

Method	Description
Set Input Range	Sets the input range for a channel as either 0–20 mA or ±20 mA. This method overwrites the value you configure in the C Series Module Properties dialog box.

I/O Properties

To access the following properties for this device, use the **FPGA I/O Property Node**.

Property	Description
LSB Weight (±20 mA range)	Returns the LSB weight in pA/LSB for the channel. Use this value to convert and calibrate NI 9203 data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box, and you set the input range to ±20 mA.
LSB Weight (4–20 mA range)	Returns the LSB weight in pA/LSB for the channel. Use this value to convert and calibrate NI 9203 data if you set the **Calibration Mode** to **Raw** in the **C Series Module Properties** dialog box, and you set the input range to 0–20 mA.

Module Properties

Use the **FPGA I/O Property Node** to access the following properties for this device.

Table 111. Module Properties

Property	Description
Module ID	Returns the module ID, 0x7129.
Offset (±20 mA range)	Returns the calibration offset in pA for the channel. Use this value to *convert and calibrate* NI 9203 data if you set the **Calibration Mode** to **Raw** in the *C Series Module Properties* dialog box, and you set the input range to ±20 mA.
Offset (4–20 mA range)	Returns the calibration offset in pA for the channel. Use this value to convert and calibrate NI 9203 data if you set the **Calibration Mode** to **Raw** in the **C Series Module Properties** dialog box, and you set the input range to 0–20 mA.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

NI 9203 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in amps.

Module Channels

Table 112. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9203, *x* is 0 to 7.

C Series Module Properties Dialog Box

Use the C Series Module Properties dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

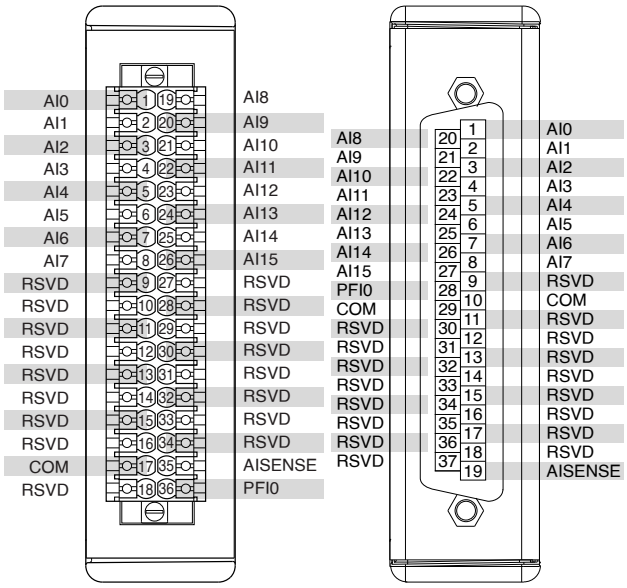
- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.
- **Channel Configuration**—You can specify the input range for each channel.
  - **Channels**—Specifies the channel(s) for which you want to select the input range.
  - **Input Range**—Specifies the input range for the selected channel(s) as either 0–20 mA or ±20 mA.

NI 9204

CompactRIO 16-Channel Single-Ended/8-Channel Differential, ±200 mV to ±10 V, 16-Bit Analog Input Module

NI 9204 Pinout

Figure 24. NI 9204 Pinout



NI 9204 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Terminal	Description
AIx	Analog input channel x, where x is the number of the channel. The NI 9204 has AI channels 0 to 15. The channel is sampled using the currently configured terminal mode and voltage range. Configure these settings at edit time using the C Series Module Properties dialog box or at run time using the Set Terminal Mode or Set Voltage Range methods.
DI0	Digital input channel used to access the module digital line, which is labeled PFI0.
Trig	Returns the output of the module trigger circuitry. Use the Set Triggers method to configure the trigger circuit.

Refer to *Conversion Timing for the NI 9204/9205/9206 (FPGA Interface)* for information

about the order and timing of conversions for these I/O channels.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

To access the following methods for this device, use the FPGA I/O Method Node.

Table 113. Module Methods

Method	Description
AI Sample	Acquires a single AI sample from the module. You can configure the channel number, terminal mode, and voltage range at run time. **Note** Avoid configuring the VI to run an AI Sample method and another FPGA I/O Method Node simultaneously when both target the same C Series module. Do not configure the VI to run multiple AI Sample methods simultaneously when the AI Sample methods target the same C Series module.
IO Sample	Acquires a single sample from the module. You can configure the channel number, terminal mode, and voltage range at run time. **Note** Avoid configuring the VI to run an IO Sample method and another FPGA I/O Method Node simultaneously when both target the same C Series module. Do not configure the VI to run multiple IO Sample methods simultaneously when the IO Sample methods target the same C Series module. Refer to the NI 9204 Basic IO VI and the NI 9205 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Basic IO\NI 9205 Basic IO.lvproj` and `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Advanced IO\NI 9204`

Method	Description
	`Advanced IO.lvproj` for examples related to this method.
Read Module Temperature	Reads the module temperature, represented as -40 °C to 125 °C. **Note** The Read Module Temperature method is only available when Calibration Mode is set to Calibrated in the C Series Module Properties dialog box.
Restore Last External Calibration Constants	Restores the calibration constants of the module to the factory values. Enable self-calibration on the C Series Module Properties dialog box to use this method. **Notice** The Restore Last External Calibration Constants method node must be used with the example VI to prevent damaging the module. Refer to the NI 9204 Self-Calibration VI in the `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Self Calibration\NI 9204 Self Calibration.lvproj` for an example of using this method.
Set Terminal Mode	Sets the terminal mode for a channel as RSE (referenced single-ended), NRSE (non-referenced single-ended), or DIFF (differential). This method overwrites the value you configure in the C Series Module Properties dialog box.
Set Triggers	Sets the trigger output to one of the following values: • None • AI Below Low • AI Above High • AI Inside Region • AI Below Low with Hysteresis • AI Above High with Hysteresis • DI Trigger Refer to the NI 9204 Basic Triggering VI and the NI 9204 Advanced Triggering VI in the `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Basic Triggering\NI 9204 Basic Triggering.lvproj` and `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Advanced Triggering\NI 9204 Advanced Triggering.lvproj` for examples related to this method.
Set Voltage	Sets the input range for a channel as ±10 V, ±5 V, ±1 V, or ±200 mV. This method

Method	Description
Range	overwrites the value you configure in the C Series Module Properties dialog box.
Write Self-Calibration Constants	Restores the calibration constants of the module to the factory values. Enable self-calibration on the C Series Module Properties dialog box to use this method. • **Self-Calibration Constants**—Contains offset and gain constants for the analog input ranges of the module. **Notice** The Write Self-Calibration Constants method node must be used with the example VI to prevent damaging the module. Refer to the NI 9204 Self-Calibration VI in the `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Self Calibration\NI 9204 Self Calibration.lvproj` for an example of using this method.

I/O Properties

This device does not support any I/O properties.

Module Properties

To access the following module properties for this device, use the FPGA I/O Property Node.

Table 114. Module Properties

Property	Description
Linearization Coefficient x	(Optional) NI-MCal linearization coefficient x, where *x* is the number of the linearization coefficient. The NI 9204 has linearization coefficients 3 to 0. You can use these coefficients in calibration to compensate for some types of measurement nonlinearity inherent in this device. When returning calibrated data from the FPGA I/O Node, the Linearization Coefficients are not used to calculate the calibrated data. This means that calibrated data returned in FPGA does not compensate for the measurement nonlinearities in

Property	Description
	the device. Refer to the NI 9204 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Advanced IO\NI 9204 Advanced IO.lvproj` for an example of this concept.
LSB Weight (±10 V range)	Returns the LSB weight in pV/LSB for the ±10 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
LSB Weight (±1 V range)	Returns the LSB weight in pV/LSB for the ±1 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
LSB Weight (±200 mV range)	Returns the LSB weight in pV/LSB for the ±200 mV range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
LSB Weight (±5 V range)	Returns the LSB weight in pV/LSB for the ±5 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Module ID	Returns the module ID, 0x7B3B (NI 9204 with DSUB) or 0x71C4 (NI 9204 with spring terminal).
Offset (±10 V range)	Returns the calibration offset in nV for the ±10 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Offset (±1 V range)	Returns the calibration offset in nV for the ±1 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Offset (±200 mV range)	Returns the calibration offset in nV for the ±200 mV range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Offset (±5 V range)	Returns the calibration offset in nV for the ±5 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

AI Sample Method (FPGA Interface)

Use AI Sample method to acquire a single AI sample from the module.

You can configure the channel number, terminal mode, and voltage range at run time. To use this module method, select it in an **FPGA I/O Method Node** that is configured for the appropriate device and/or channel.

> **Note** Avoid configuring the VI to run an AI Sample method and another **FPGA I/O Method Node** simultaneously when both target the same C Series module. Do not configure the VI to run multiple AI Sample methods simultaneously when the AI Sample methods target the same C Series module.

	**AI Channel** [**i + 2**] is the channel you need to load into the module conversion pipeline. The channel number is part of the configuration information that controls what data the module will sample two iterations later. Refer to *Conversion Timing* for more information.

	**Terminal Mode** [**i + 2**] is an enumeration of the following terminal modes: • RSE is a single-ended measurement between any of the 16 individual channels and the module ground. • NRSE is a single-ended measurement between any of the 16 individual channels and the AISENSE line into the module. • DIFF is a differential measurement between two of the 16 individual channels, with the name of the differential channel matching the channel tied to the positive side of the differential measurement. You can configure channels AI0 to AI7 to DIFF mode. The terminal mode is part of the configuration that controls the data to be sampled two iterations into the future. Refer to *Conversion Timing* for more information.

	**Voltage Range** [**i + 2**] is an enumeration of the following voltage ranges: • ±10 V • ±5 V • ±1 V • ±200 mV The voltage range is part of the configuration information that controls the data to be sampled two iterations into the future. Refer to *Conversion Timing* for more information.

	**Data** [**i**] returns the data from the current sample.

	**error in** describes error conditions that occur before this VI or function runs. The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node. •  **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE. •  **code** is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code. •  **source** always contains an empty string because strings are not supported in LabVIEW FPGA.

	**error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error.

	• **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred. • **code** is the error code number identifying an error. The default is 0. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is 0 or a warning code. • **source** always contains an empty string because strings are not supported in LabVIEW FPGA.

Using This Method

The AI Sample method provides an efficient and flexible interface to the module. You can use this method to acquire a single sample from any of the AI channels on the module, at any range, and with any available input mode.

When you execute this method, the module performs a single conversion on the next channel in the conversion pipeline. The **Data [i]** method output returns the data from this conversion. As the conversion data is read from the module, the new configuration specified by the **AI Channel [i + 2]**,**Terminal Mode [i + 2]**, and **Voltage Range [i + 2]** inputs is loaded into the module's configuration pipeline. The pipeline is two samples deep. Therefore, the configuration you set during one execution of the AI Sample method determines which channel the AI Sample method will sample two iterations later. Refer to *Conversion Timing* for more details.

Related concepts:

- Conversion Timing for the NI 9204/9205/9206 (FPGA Interface)

IO Sample Method (FPGA Interface)

This module method acquires a single sample from the module. The channel number, terminal mode, and voltage range are all configurable at run time. To use this module method, select it in an **FPGA I/O Method Node** that is configured for the appropriate device and/or channel.

Note NI suggests that you do not configure the VI to run an IO Sample method and another FPGA I/O Method Node in parallel when both are

targeted to the same C Series module. Do not configure the VI to run multiple IIO Sample methods in parallel when the methods are targeted to the same C Series module.

	**Configuration** [i+2] contains encoded configuration information that gets loaded into the module conversion pipeline. This configuration information controls the data to be sampled two iterations into the future. Refer to the *Conversion Timing* topic for more information. Refer to the Configuration Encoding table below for an example of how the configuration information is encoded.

	**Data** [i] Returns the data from the current sample.

	**error in** describes error conditions that occur before this VI or function runs. The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node. •  **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE. •  **code** is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code. •  **source** always contains an empty string because strings are not supported in LabVIEW FPGA.

	**error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error.

	• **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred. • **code** is the error code number identifying an error. The default is 0. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is 0 or a warning code. • **source** always contains an empty string because strings are not supported in LabVIEW FPGA.

Configuration Encoding Table

The following table describes how to construct the Configuration [i+2] value to write to the IO Sample method to perform a particular operation. The **Operation** column lists the types of operations that can be performed using the IO Sample method. The **Configuration Bit Fields** column lists the bit positions within the **Configuration [i+2]** number that need to be set. The **Values** column specifies the binary value to apply to the specified **Configuration Bit Fields**.

Operation	Configuration Bit Fields	Values
AI Read	15:11	001b
	10:8	000b for channels 0 or 8, 001b for channels 1 or 9, … 110b for channels 6 or 14, 111b for channels 7 or 15
	7:6	00b
	5:4	00b for NRSE mode11b for RSE or DIFF modes
	3:2	11b for channels 8-15, 01b for DIFF mode on channels 0-7, 10b for RSE or NRSE mode on channels 0-7
	1:0	00b for ±10 V, 01b for ±5 V, 10b for ±1 V, 11b for ±200 mV
DI0 Read	15:0	0000000000000000b

Using this Method

The IO Sample method provides an efficient and flexible interface to the module. You can use this method to acquire a single sample from any of the channels on the module, at any range, and with any available input mode.

When this method is executed, the module performs a single conversion on the next channel present in the conversion pipeline on the module. The data from this conversion is returned via the **Data [i]** method output. At the same time that the conversion data is read from the module, the new configuration information specified by the **Configuration [i+2]** input is loaded into the configuration pipeline on the module. The pipeline is two samples deep. So, the configuration information specified on one execution of the IO Sample method determines which channel will be sampled by the IO Sample method two iterations into the future. Refer to the *Conversion Timing* topic for more details.

Examples

Refer to the NI 9204 Basic IO VI and the NI 9204 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Basic IO\NI 9204 Basic IO.lvproj` and `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Advanced IO\NI 9204 Advanced IO.lvproj` for examples of the IO concepts discussed above.

Set Terminal Mode Method (FPGA Interface)

The Set Terminal Mode method overrides the default setting provided in the dialog box. To use this module method, select it in an **FPGA I/O Method Node** that is configured for the appropriate device and/or channel.

	**AI Channel** is the channel you want to set the terminal mode for.
	**Terminal Mode** is an enumeration of the following trigger modes: • **RSE** is a single-ended measurement between any of the 16 (NI 9204) or 32 (NI 9205/9206) individual channels and the module ground. • **NRSE** is a single-ended measurement between any of the 16 (NI 9204) or 32 (NI 9205/9206) individual channels and the AISENSE line into the module. • **DIFF** is a differential measurement between two of the 16 (NI 9204) or 32 (NI 9205/9206) individual channels, with the name of the differential channel matching the channel tied to the positive side of the differential measurement. Channels AI0 to AI7 and AI16 to AI23 (NI 9205/9206 only) may be set to DIFF configuration.

	error in describes error conditions that occur before this VI or function runs. The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

	status is *TRUE* (X) if an error occurred before this VI or function ran or *FALSE* (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
	code is the error or warning code. The default is `0`. If **status** is TRUE, **code** is a nonzero *error code*. If **status** is FALSE, **code** is `0` or a warning code.
	source always contains an empty string because strings are not supported in LabVIEW FPGA.

	error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error.

	status is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred.
	code is the error or warning code. If **status** is TRUE, **code** is a nonzero *error code*. If **status** is FALSE, **code** is `0` or a warning code.
	source always contains an empty string because strings are not supported in LabVIEW FPGA.

Set Triggers Method (FPGA Interface)

This module method controls the data returned by the trigger digital input line. Use this module method by selecting it in an **FPGA I/O Method Node** that is configured for the appropriate device and/or channel.

	Trigger Mode is an enumeration of the following trigger mode options: • **None** disables the trigger function. • **AI Below Low** detects when the analog signal is below the low threshold you specify. • **AI Above High** detects when the analog signal is above the high threshold you specify. • **AI Inside Region** detects when the analog signal is between the high and low thresholds you specify. • **AI Below Low with Hysteresis** asserts the trigger when the signal starts above the high threshold you specify and then crosses below the low threshold you specify. The trigger deasserts when the signal crosses above the high threshold you specify. • **AI Above High with Hysteresis** asserts the trigger when the signal starts below the low threshold you specify and then crosses above the high threshold you specify. The trigger deasserts when the signal crosses below the low threshold you specify. • **DI Trigger** detects when the digital signal on PFI0 is logic high. **Note** The onboard NI 9205 on *Single-Board RIO* (sbRIO) devices does not support DI triggering.
	AI High Threshold defines the upper threshold for the analog trigger circuit.
	AI Low Threshold defines the lower threshold for the analog trigger circuit.

	error in describes error conditions that occur before this VI or function runs. The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

	status is TRUE if an error occurred. If **status** is TRUE, the VI does not perform any operations.
	code is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code.
	source always contains an empty string because strings are not supported in LabVIEW FPGA.

	error out contains error information. If **error in** indicates that an error occurred before this VI

	or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error.

	status is TRUE if an error occurred. If **status** is TRUE, the VI does not perform any operations.
	code is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code.
	source always contains an empty string because strings are not supported in LabVIEW FPGA.

Using This Method

Use the Set Triggers method to configure the trigger circuit on the module. Once the trigger circuit is configured, use an FPGA I/O Node configured to access the Trig channel to monitor the output of the trigger circuit. The trigger circuit does not automatically initiate any conversions on the module. It is up to the FPGA VI to monitor the Trig line, and then initiate a conversion using an FPGA I/O Node or the IO Sample method when the appropriate trigger event occurs.

Perform the following steps to trigger an acquisition using the Set Triggers method:

1. Ensure that the module pipeline is primed with the next two channels that you want to sample after the trigger event. This step will minimize the delay between the trigger event and the AI channels getting sampled. For analog triggering modes, this step also configures the analog trigger circuitry to monitor the desired channel. The AI trigger circuit monitors the channel waiting in the module pipeline for the next conversion. If you are using FPGA I/O Nodes to perform the acquisition, this step can be accomplished by executing an FPGA I/O Node configured to match the FPGA I/O Node that you will be executing after the trigger event. If you are using one of the analog triggering modes along with the FPGA I/O Nodes, the AI trigger circuit will operate on the lowest numbered channel present in the FPGA I/O Node. Refer to the *IO Sample* method and *Conversion Timing* topics for more information on managing the module pipeline using the IO Sample method. If you are using one of the analog trigger modes along with the IO Sample method, the AI trigger circuit will operate on the channel waiting in the pipeline to be converted

by the next IO Sample method.
2. Execute the *Set Triggers* method with the appropriate trigger mode selected. The trigger mode circuitry is implemented in the hardware as you specify in this method. Refer to the NI 9204, NI 9205, and NI 9206 hardware documentation for more information about the trigger modes.
3. Monitor the Trig channel in a loop until you see the appropriate trigger condition occur.
4. Execute an FPGA I/O Node or IO Sample method function to acquire data in response to the trigger event.

Examples

Refer to the NI 9204 Basic Triggering VI and the NI 9204 Advanced Triggering VI in the `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Basic Triggering\NI 9204 Basic Triggering.lvproj` and `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Advanced Triggering\NI 9204 Advanced Triggering.lvproj` for examples of the triggering concepts discussed above or to the NI 9205 Basic Triggering VI and the NI 9205 Advanced Triggering VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Basic Triggering\NI 9205 Basic Triggering.lvproj` and `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Advanced Triggering\NI 9205 Advanced Triggering.lvproj` for examples of the triggering concepts discussed above.

Set Voltage Range Method (FPGA Interface)

This module method overrides the default setting provided in the *C Series Module Properties* dialog box. To use this module method, select it in an **FPGA I/O Method Node** that is configured for the appropriate device and/or channel.

	**AI Channel** is the channel you want to set the voltage range for.
	**Voltage Range** is an enumeration of the following voltage ranges: • ±10 V • ±5 V

	• ±1 V • ±200 mV

	error in describes error conditions that occur before this VI or function runs. The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node. **status** is TRUE if an error occurred. If **status** is TRUE, the VI does not perform any operations. **code** is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code. **source** always contains an empty string because strings are not supported in LabVIEW FPGA.

	error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error. **status** is TRUE if an error occurred. If **status** is TRUE, the VI does not perform any operations. **code** is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code. **source** always contains an empty string because strings are not supported in LabVIEW FPGA.

Conversion Timing for the NI 9204/9205/9206 (FPGA Interface)

The *NI 9204*, *NI 9205*, and *NI 9206* modules implement a two-element deep pipeline for

access to the AI, DI0, and DO0 (NI 9205/9206 only) channels. This pipeline results in maximum sample rate and maximum sample quality. The Trig channel has a parallel data path that bypasses the pipeline.

When using the FPGA I/O Node to sample channels, the pipeline is automatically managed by the FPGA I/O Node, and the channels within the FPGA I/O Node are sampled in numerical order regardless of the order they appear in the node.

If the first two channel requests in the FPGA I/O Node do not match the two channel requests stored in the module pipeline, there will be a delay before the first channel sample occurs. This delay is caused by the FPGA I/O Node automatically updating the module channel sample pipeline, which takes two channel sample cycles.

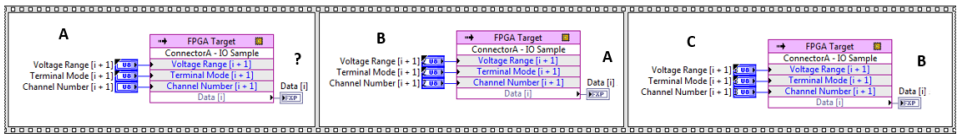
To minimize the need for the pipeline updates, each FPGA I/O Node leaves the module pipeline primed to repeat itself. If you use only one FPGA I/O Node in a looping structure, only the first iteration of the loop will incur the two-cycle time delay. All subsequent iterations operate with no delay.

When using the IO Sample method, take steps to manage this pipeline in the VIs.

- To read one channel, that channel configuration must be requested two cycles before the time it is to be sampled. The U16 configuration value for an AI channel sample contains the channel number, channel input range, and terminal mode. The U16 configuration value also can represent an access to the DIO or DOO (NI 9205/9206 only) channels, which is treated the same as an AI operation.

The following diagram illustrates how the pipeline works within a sequence structure. A typical application uses a loop structure to continuously iterate through a scan list. The scan list is a predefined list of configurations. The sequence structure and art elements show how data moves into the NI 9204/9205/9206 pipeline and then is converted. The IO Sample method first converts the configuration that is in the ADC, and then shifts the configuration values through the pipeline to prepare for the next time LabVIEW calls the IO Sample method.

Figure 25. Pipeline Behavior within a Sequence Structure



The simplest form of pipeline management is to force the pipeline on every set of samples. This assumes that the pipeline is always wrong at the first sample, and sends all IO Sample requests in order. The last two data points need some configuration to be requested, but it does not matter, in this use case, what configuration that is.

A better form of pipeline management emulates the behavior of the FPGA I/O Node. Instead of always assuming that the pipeline is wrong, you choose the last two configuration values based on the first two samples you want to include in the next iteration. This pipeline management system can be accomplished in two parts: initialization and sample iteration.



Tip NI recommends you use the IO Sample method for the following reasons:

- Smaller on FPGA—With large channel count, the IO Sample method is more efficient with FPGA resources, and thus requires less space on the FPGA for the same application.
- Sample flexibility—The IO Sample method gives you more flexibility to change channel range, terminal mode, and channel order at run time and at full sample rate.
- Application timing flexibility—The IO Sample method returns all sample data as soon as it is available. The FPGA I/O Node will gather data for all requested samples and then return all at once.

The **Minimum Time Between Conversions** control you can set in the *C Series Module Properties* dialog box determines the shortest possible time between any two conversions. For channels sampled within the same FPGA I/O Node, the time you set determines the exact time between conversions. For channels sampled within separate FPGA I/O Nodes or for conversions caused by looping on an FPGA I/O Node, the time you set may be less than the actual time between conversions. However, the minimum time you set is never greater than the time between conversions. If the application tries to execute an FPGA I/O Node or IO Sample method faster than the specified minimum time between conversions, the conversion is delayed until the minimum time you set is satisfied.

The default minimum time between conversions for the NI 9204/9205/9206 is 8 µs. The

accuracy specifications in the NI 9204, NI 9205, and NI 9206 hardware documentation on ni.com/docs are based on this default value. If you set the minimum time between conversions to at least 8 µs, the accuracy of the module is not affected. If you set the minimum time between conversions to less than 8 µs, the accuracy of the module degrades if you sample data from multiple channels.

Examples

Refer to the NI 9204 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Advanced IO\NI 9204 Advanced IO.lvproj` for an example of conversion timing on the NI 9204 C Series Module.

Refer to the NI 9205 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Advanced IO\NI 9205 Advanced IO.lvproj` for an example of conversion timing on the NI 9205 and NI 9206 C Series Modules.

Related concepts:

- NI 9204 (FPGA Interface)
- NI 9205 (FPGA Interface)
- NI 9206 (FPGA Interface)
- IO Sample Method (FPGA Interface)

Self-Calibration for the NI 9204/9205/9206 (FPGA Interface)

Self-Calibration adjusts the calibration constants stored onboard the module. Perform a self-calibration at any time to account for changes in environmental conditions. Enable self-calibration on the *C Series Module Properties* dialog box.

Enabling Self-Calibration on the *NI 9204*, *NI 9205*, or *NI 9206* allows access to the Write Self-Calibration Constants and Restore Last External Calibration Constants module methods.

Caution The Write Self-Calibration Constants and Restore Last External Calibration Constants method nodes must be used with the example VI to

> prevent damaging the module.

Refer to the NI 9204 Self-Calibration VI in the `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Self Calibration\NI 9204 Self Calibration.lvproj` or to the NI 9205 Self-Calibration VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Self Calibration\NI 9205 Self Calibration.lvproj` for examples of self-calibration.

NI 9204 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9204 has the following channels.

Table 115. Module Channels

Channel	Description
AIx	Analog input channel x, where x is the number of the channel. For the NI 9204, x is 0 to 15.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9204. Right-click the NI 9204 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.

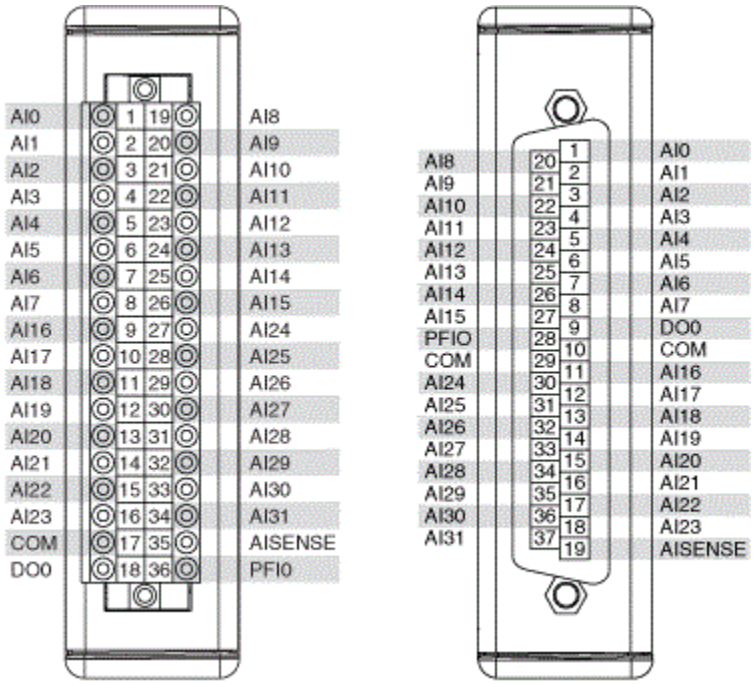
- **Channel Configuration**—You can specify the input range for each channel.
  - **Channels**—Specifies the channel(s) for which you want to select the input range.
  - **Input Range**—Specifies the input range for the selected channel(s) as ±10 V, ±5 V, ±1 V, or ±200 mV.
  - **Terminal Mode**—Specifies the terminal mode for the selected channel(s) as RSE (referenced single-ended), NRSE (non-referenced single-ended), or DIFF (differential).
  - **Minimum Time between Conversions**—Specifies the time (in uSec) between channel samples. The Scan Engine samples all channels of the module sequentially. The default time between sampling one channel and the next is 8 µs. If you connect all unused channels to ground, you should not have to increase this setting. If for some reason you cannot connect unused channels to ground, you can reduce noise by increasing the minimum time between conversions.

NI 9205

CompactRIO 32-Channel Single-Ended/16-Channel Differential, ±200 mV to ±10 V, 16-Bit Analog Input Module

NI 9205 Pinout

Figure 26. NI 9205 Pinout



NI 9205 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 116. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9205 has AI channels 0 to 31. The channel is sampled using the currently configured terminal mode and voltage range. Configure these settings at edit time using the **C Series Module Properties** dialog box or at run time using the Set Terminal Mode or Set Voltage Range methods.
DI0	Digital input channel used to access the module digital line, which is labeled PFI0. This terminal is not available for an onboard NI 9205 on a Single-Board RIO (sbRIO) device.

Terminal	Description
Trig	Returns the output of the module trigger circuitry. Use the Set Triggers method to configure the trigger circuit.
DO0	Digital output channel 0. This terminal is not available for an onboard NI 9205 on an sbRIO device.

Refer to *Conversion Timing for the NI 9204/9205/9206 (FPGA Interface)* for information about the order and timing of conversions for these I/O channels.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

To access the following methods for this device, use the FPGA I/O Method Node.

Method	Description
AI Sample	Acquires a single AI sample from the module. You can configure the channel number, terminal mode, and voltage range at run time. **Note** Avoid configuring the VI to run an AI Sample method and another FPGA I/O Method Node simultaneously when both target the same C Series module. Do not configure the VI to run multiple AI Sample methods simultaneously when the AI Sample methods target the same C Series module.
IO Sample	Acquires a single sample from the module. The channel number, terminal mode, and voltage range are all configurable at run time. **Note** Avoid configuring the VI to run an IO Sample method and another

Method	Description
	FPGA I/O Method Node simultaneously when both target the same C Series module. Do not configure the VI to run multiple IO Sample methods simultaneously when the IO Sample methods target the same C Series module. Refer to the NI 9205 Basic IO VI and the NI 9205 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Basic IO\NI 9205 Basic IO.lvproj` and `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Advanced IO\NI 9205 Advanced IO.lvproj` for examples related to this method.
Read Module Temperature	Reads the module temperature, represented as -40 °C to 125 °C. **Note** The Read Module Temperature method is only available when Calibration Mode is set to Calibrated in the C Series Module Properties dialog box.
Restore Last External Calibration Constants	Restores the calibration constants of the module to the factory values. You must enable *self-calibration* on the *C Series Module Properties* dialog box to use this method. **Notice** The Restore Last External Calibration Constants method node must be used with the example VI to prevent damaging the module. Refer to the NI 9205 Self-Calibration VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Self Calibration\NI 9205 Self Calibration.lvproj` for an example of using this method.
Set Terminal Mode	Sets the terminal mode for a channel as RSE (referenced single-ended), NRSE (non-referenced single-ended), or DIFF (differential). This method overwrites the value you configure in the **C Series Module Properties** dialog box.
Set Triggers	Sets the trigger output to one of the following values: • None • AI Below Low • AI Above High • AI Inside Region

Method	Description
	• AI Below Low with Hysteresis • AI Above High with Hysteresis • DI Trigger **Note** The onboard NI 9205 on sbRIO devices does not support DI triggering. Refer to the NI 9205 Basic Triggering VI and the NI 9205 Advanced Triggering VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Basic Triggering\NI 9205 Basic Triggering.lvproj` and `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Advanced Triggering\NI 9205 Advanced Triggering.lvproj` for examples related to this method.
Set Voltage Range	Sets the input range for a channel as ±10 V, ±5 V, ±1 V, or ±200 mV. This method overwrites the value you configure in the **C Series Module Properties** dialog box.
Write Self-Calibration Constants	Restores the calibration constants of the module to the factory values. Enable self-calibration on the C Series Module Properties dialog box to use this method. • **Self-Calibration Constants**—Contains offset and gain constants for the analog input ranges of the module. **Notice** The Write Self-Calibration Constants method node must be used with the example VI to prevent damaging the module. Refer to the NI 9205 Self-Calibration VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Self Calibration\NI 9205 Self Calibration.lvproj` for an example of using this method.

I/O Properties

This device does not support any I/O properties.

Module Properties

To access the following module properties for this device, use the FPGA I/O Property

Node.

Property	Description
Linearization Coefficient x	(Optional) NI-MCal linearization coefficient x, where *x* is the number of the linearization coefficient. The NI 9205 has linearization coefficients 3 to 0. You can use these coefficients in calibration to compensate for some types of measurement nonlinearity inherent in this device. When returning calibrated data from the FPGA I/O Node, the Linearization Coefficients are not used to calculate the calibrated data. This means that calibrated data returned in FPGA does not compensate for the measurement nonlinearities in the device. Refer to the NI 9205 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Advanced IO\NI 9205 Advanced IO.lvproj` for an example of this concept.
LSB Weight (±10 V range)	Returns the LSB weight in pV/LSB for the ±10 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
LSB Weight (±1 V range)	Returns the LSB weight in pV/LSB for the ±1 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
LSB Weight (±200 mV range)	Returns the LSB weight in pV/LSB for the ±200 mV range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
LSB Weight (±5 V range)	Returns the LSB weight in pV/LSB for the ±5 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Module ID	Returns the module ID, 0x712A (NI 9205 with DSUB) or 0x71EA (NI 9205 with spring terminal).
Offset (±10 V range)	Returns the calibration offset in nV for the ±10 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Offset (±1 V range)	Returns the calibration offset in nV for the ±1 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.

Property	Description
Offset (±200 mV range)	Returns the calibration offset in nV for the ±200 mV range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Offset (±5 V range)	Returns the calibration offset in nV for the ±5 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Conversion Timing for the NI 9204/9205/9206 (FPGA Interface)

AI Sample Method (FPGA Interface)

Use AI Sample method to acquire a single AI sample from the module.

You can configure the channel number, terminal mode, and voltage range at run time. To use this module method, select it in an **FPGA I/O Method Node** that is configured for the appropriate device and/or channel.

Note Avoid configuring the VI to run an AI Sample method and another **FPGA I/O Method Node** simultaneously when both target the same C Series module. Do not configure the VI to run multiple AI Sample methods simultaneously when the AI Sample methods target the same C Series module.

U8	**AI Channel** [**i + 2**] is the channel you need to load into the module conversion pipeline. The channel number is part of the configuration information that controls what data the module will sample two iterations later. Refer to

	Conversion Timing for more information.

	**Terminal Mode** [**i + 2**] is an enumeration of the following terminal modes: • RSE is a single-ended measurement between any of the 32 individual channels and the module ground. • NRSE is a single-ended measurement between any of the 32 individual channels and the AISENSE line into the module. • DIFF is a differential measurement between two of the 32 individual channels, with the name of the differential channel matching the channel tied to the positive side of the differential measurement. You can configure channels AI0 to AI7 and AI16 to AI23 to DIFF mode. The terminal mode is part of the configuration that controls the data to be sampled two iterations into the future. Refer to *Conversion Timing* for more information.

	**Voltage Range** [**i + 2**] is an enumeration of the following voltage ranges: • ±10 V • ±5 V • ±1 V • ±200 mV The voltage range is part of the configuration information that controls the data to be sampled two iterations into the future. Refer to *Conversion Timing* for more information.

	**Data** [**i**] returns the data from the current sample.

	**error in** describes error conditions that occur before this VI or function runs. The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

	• **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE. • **code** is the error code number identifying an error. The default is 0. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is 0 or a warning code. • **source** always contains an empty string because strings are not supported in LabVIEW FPGA.

	**error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error. • **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred. • **code** is the error code number identifying an error. The default is 0. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is 0 or a warning code. • **source** always contains an empty string because strings are not supported in LabVIEW FPGA.

Using This Method

The AI Sample method provides an efficient and flexible interface to the module. You can use this method to acquire a single sample from any of the AI channels on the module, at any range, and with any available input mode.

When you execute this method, the module performs a single conversion on the next channel in the conversion pipeline. The **Data [i]** method output returns the data from this conversion. As the conversion data is read from the module, the new configuration specified by the **AI Channel [i + 2]**,**Terminal Mode [i + 2]**, and **Voltage Range [i + 2]** inputs is loaded into the module's configuration pipeline. The pipeline is two samples deep. Therefore, the configuration you set during one execution of the AI Sample method determines which channel the AI Sample method will sample two iterations later. Refer to *Conversion Timing* for more details.

IO Sample Method (FPGA Interface)

This module method acquires a single sample from the module. The channel number, terminal mode, and voltage range are all configurable at run time. To use this module method, select it in an FPGA I/O Method Node that is configured for the appropriate device and/or channel.



Note NI suggests that you do not configure the VI to run an IO Sample method and another FPGA I/O Method Node in parallel when both are targeted to the same C Series module. Similarly, NI suggests that you do not configure the VI to run multiple IO Sample methods in parallel when the IO Sample methods are targeted to the same C Series module.

	**Configuration [i+2]** contains encoded configuration information that gets loaded into the module conversion pipeline. This configuration information controls the data to be sampled two iterations into the future. Refer to the *Conversion Timing* topic for more information. Refer to the Configuration Encoding table below for an example of how the configuration information is encoded.

	**Data [i]** Returns the data from the current sample.

	**error in** describes error conditions that occur before this VI or function runs. The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node. •  **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE. •  **code** is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code. •  **source** always contains an empty string because strings are not supported in LabVIEW FPGA.

	**error out** contains error information. If **error in** indicates that an error occurred before this VI

	or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error. • **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred. • **code** is the error code number identifying an error. The default is 0. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is 0 or a warning code. • **source** always contains an empty string because strings are not supported in LabVIEW FPGA.

Configuration Encoding

The following table describes how to construct the Configuration [i+2] value to write to the IO Sample method to perform a particular operation.

- The **Operation** column lists the types of operations that can be performed using the IO Sample method.
- The **Configuration Bit Fields** column lists the bit positions within the **Configuration [i+2]** number that need to be set.
- The **Values** column specifies the binary value to apply to the specified **Configuration Bit Fields**.

Table 117. Configuration Encoding

Operation	Configuration Bit Fields	Values
AI Read	15:13	001b
	12:11	01b for channels 0-1510b for channels 16-31
	10:8	000b for channels 0, 8, 16, or 24 001b for channels 1, 9, 17, or 25…110b for channels 6, 14, 22, or 30111b for channels 7, 15, 23, or 31
	7:6	00b
	5:4	00b for NRSE mode11b for RSE or DIFF modes
	3:2	11b for channels 8-15 or 24-31 01b for DIFF mode on channels 0-7 or 16-2310b for RSE or NRSE mode on channels 0-7 or 16-23

Operation	Configuration Bit Fields	Values
	1:0	00b for ±10 V01b for ±5 V10b for ±1 V11b for ±200 mV
DI0 Read	15:0	0000000000000000b
DO0 Write	15:1	010000000000000b
	0	Binary value to write to DO0

Using this Method

The IO Sample method provides an efficient and flexible interface to the module. You can use this method to acquire a single sample from any of the channels on the module, at any range, and with any available input mode.

When this method is executed, the module performs a single conversion on the next channel present in the conversion pipeline on the module. The data from this conversion is returned via the **Data [i]** method output. At the same time that the conversion data is read from the module, the new configuration information specified by the **Configuration [i+2]** input is loaded into the configuration pipeline on the module. The pipeline is two samples deep. So, the configuration information specified on one execution of the IO Sample method determines which channel will be sampled by the IO Sample method two iterations into the future. Refer to the *Conversion Timing* topic for more details.

Examples

Refer to the NI 9205 Basic IO VI and the NI 9205 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Basic IO\NI 9205 Basic IO.lvproj` and `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Advanced IO\NI 9205 Advanced IO.lvproj` for examples of the IO concepts discussed above.

Related concepts:

- Conversion Timing for the NI 9204/9205/9206 (FPGA Interface)

Set Terminal Mode Method (FPGA Interface)

The Set Terminal Mode method overrides the default setting provided in the dialog box. To use this module method, select it in an **FPGA I/O Method Node** that is configured for the appropriate device and/or channel.

	**AI Channel** is the channel you want to set the terminal mode for.
	**Terminal Mode** is an enumeration of the following trigger modes: • **RSE** is a single-ended measurement between any of the 16 (NI 9204) or 32 (NI 9205/9206) individual channels and the module ground. • **NRSE** is a single-ended measurement between any of the 16 (NI 9204) or 32 (NI 9205/9206) individual channels and the AISENSE line into the module. • **DIFF** is a differential measurement between two of the 16 (NI 9204) or 32 (NI 9205/9206) individual channels, with the name of the differential channel matching the channel tied to the positive side of the differential measurement. Channels AI0 to AI7 and AI16 to AI23 (NI 9205/9206 only) may be set to DIFF configuration.

	**error in** describes error conditions that occur before this VI or function runs. The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

	**status** is *TRUE* (X) if an error occurred before this VI or function ran or *FALSE* (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE.
	**code** is the error or warning code. The default is `0`. If **status** is TRUE, **code** is a nonzero *error code*. If **status** is FALSE, **code** is `0` or a warning code.
	**source** always contains an empty string because strings are not supported in LabVIEW FPGA.

	**error out** contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the

	error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error. **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred. **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero *error code*. If **status** is FALSE, **code** is 0 or a warning code. **source** always contains an empty string because strings are not supported in LabVIEW FPGA.

Set Triggers Method (FPGA Interface)

This module method controls the data returned by the trigger digital input line. Use this module method by selecting it in an **FPGA I/O Method Node** that is configured for the appropriate device and/or channel.

	Trigger Mode is an enumeration of the following trigger mode options: • **None** disables the trigger function. • **AI Below Low** detects when the analog signal is below the low threshold you specify. • **AI Above High** detects when the analog signal is above the high threshold you specify. • **AI Inside Region** detects when the analog signal is between the high and low thresholds you specify. • **AI Below Low with Hysteresis** asserts the trigger when the signal starts above the high threshold you specify and then crosses below the low threshold you specify. The trigger deasserts when the signal crosses above the high threshold you specify. • **AI Above High with Hysteresis** asserts the trigger when the signal starts below the low threshold you specify and then crosses above the high threshold you specify. The trigger deasserts when the signal crosses below the low threshold you specify. • **DI Trigger** detects when the digital signal on PFI0 is logic high. **Note** The onboard NI 9205 on *Single-Board RIO* (sbRIO) devices does not support DI triggering.
	AI High Threshold defines the upper threshold for the analog trigger circuit.
	AI Low Threshold defines the lower threshold for the analog trigger circuit.

	error in describes error conditions that occur before this VI or function runs. The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

	status is TRUE if an error occurred. If **status** is TRUE, the VI does not perform any operations.
	code is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code.
	source always contains an empty string because strings are not supported in LabVIEW FPGA.

	error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error.

	status is TRUE if an error occurred. If **status** is TRUE, the VI does not perform any operations.
	code is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code.
	source always contains an empty string because strings are not supported in LabVIEW FPGA.

Using This Method

Use the Set Triggers method to configure the trigger circuit on the module. Once the trigger circuit is configured, use an FPGA I/O Node configured to access the Trig channel to monitor the output of the trigger circuit. The trigger circuit does not automatically initiate any conversions on the module. It is up to the FPGA VI to monitor the Trig line, and then initiate a conversion using an FPGA I/O Node or the IO Sample method when the appropriate trigger event occurs.

Perform the following steps to trigger an acquisition using the Set Triggers method:

1. Ensure that the module pipeline is primed with the next two channels that you want to sample after the trigger event. This step will minimize the delay between the trigger event and the AI channels getting sampled. For analog triggering modes, this step also configures the analog trigger circuitry to monitor the desired channel. The AI trigger circuit monitors the channel waiting in the module pipeline for the next conversion. If you are using FPGA I/O Nodes to perform the acquisition, this step can be accomplished by executing an FPGA I/O Node configured to match the FPGA I/O Node that you will be executing after the trigger event. If you are using one of the analog triggering modes along with the FPGA I/O Nodes, the AI trigger circuit will operate on the lowest numbered channel present in the FPGA I/O Node. Refer to the *IO Sample* method and *Conversion Timing* topics for more information on managing the module pipeline using the IO Sample method. If you are using one of the analog trigger modes along with the IO Sample method, the AI trigger circuit will operate on the channel waiting in the pipeline to be converted by the next IO Sample method.
2. Execute the *Set Triggers* method with the appropriate trigger mode selected. The trigger mode circuitry is implemented in the hardware as you specify in this method. Refer to the NI 9204, NI 9205, and NI 9206 hardware documentation for more information about the trigger modes.
3. Monitor the Trig channel in a loop until you see the appropriate trigger condition occur.
4. Execute an FPGA I/O Node or IO Sample method function to acquire data in response to the trigger event.

Examples

Refer to the NI 9204 Basic Triggering VI and the NI 9204 Advanced Triggering VI in the `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Basic Triggering\NI 9204 Basic Triggering.lvproj` and `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Advanced Triggering\NI 9204 Advanced Triggering.lvproj` for examples of the triggering concepts discussed above or to the NI 9205 Basic Triggering VI and the NI 9205 Advanced Triggering VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Basic Triggering\NI 9205 Basic Triggering.lvproj` and `labview\`

`examples\CompactRIO\Module Specific\NI 9205\NI 9205 Advanced Triggering\NI 9205 Advanced Triggering.lvproj` for examples of the triggering concepts discussed above.

Set Voltage Range Method (FPGA Interface)

This module method overrides the default setting provided in the *C Series Module Properties* dialog box. To use this module method, select it in an **FPGA I/O Method Node** that is configured for the appropriate device and/or channel.

	**AI Channel** is the channel you want to set the voltage range for.
	**Voltage Range** is an enumeration of the following voltage ranges: • ±10 V • ±5 V • ±1 V • ±200 mV

	error in describes error conditions that occur before this VI or function runs. The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

	status is TRUE if an error occurred. If **status** is TRUE, the VI does not perform any operations.
	code is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code.
	source always contains an empty string because strings are not supported in LabVIEW FPGA.

	error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the

	error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error.

	status is TRUE if an error occurred. If **status** is TRUE, the VI does not perform any operations.
	code is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code.
	source always contains an empty string because strings are not supported in LabVIEW FPGA.

Conversion Timing for the NI 9204/9205/9206 (FPGA Interface)

The *NI 9204*, *NI 9205*, and *NI 9206* modules implement a two-element deep pipeline for access to the AI, DI0, and DO0 (NI 9205/9206 only) channels. This pipeline results in maximum sample rate and maximum sample quality. The Trig channel has a parallel data path that bypasses the pipeline.

When using the FPGA I/O Node to sample channels, the pipeline is automatically managed by the FPGA I/O Node, and the channels within the FPGA I/O Node are sampled in numerical order regardless of the order they appear in the node.

If the first two channel requests in the FPGA I/O Node do not match the two channel requests stored in the module pipeline, there will be a delay before the first channel sample occurs. This delay is caused by the FPGA I/O Node automatically updating the module channel sample pipeline, which takes two channel sample cycles.

To minimize the need for the pipeline updates, each FPGA I/O Node leaves the module pipeline primed to repeat itself. If you use only one FPGA I/O Node in a looping structure, only the first iteration of the loop will incur the two-cycle time delay. All subsequent iterations operate with no delay.

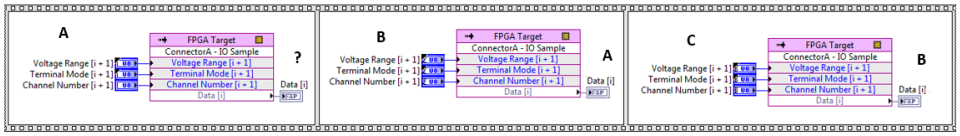
When using the IO Sample method, take steps to manage this pipeline in the VIs.

- To read one channel, that channel configuration must be requested two cycles before the time it is to be sampled. The U16 configuration value for an AI channel sample contains the channel number, channel input range, and terminal mode.

The U16 configuration value also can represent an access to the DIO or DOO (NI 9205/9206 only) channels, which is treated the same as an AI operation.

The following diagram illustrates how the pipeline works within a sequence structure. A typical application uses a loop structure to continuously iterate through a scan list. The scan list is a predefined list of configurations. The sequence structure and art elements show how data moves into the NI 9204/9205/9206 pipeline and then is converted. The IO Sample method first converts the configuration that is in the ADC, and then shifts the configuration values through the pipeline to prepare for the next time LabVIEW calls the IO Sample method.

Figure 27. Pipeline Behavior within a Sequence Structure



The simplest form of pipeline management is to force the pipeline on every set of samples. This assumes that the pipeline is always wrong at the first sample, and sends all IO Sample requests in order. The last two data points need some configuration to be requested, but it does not matter, in this use case, what configuration that is.

A better form of pipeline management emulates the behavior of the FPGA I/O Node. Instead of always assuming that the pipeline is wrong, you choose the last two configuration values based on the first two samples you want to include in the next iteration. This pipeline management system can be accomplished in two parts: initialization and sample iteration.

Tip NI recommends you use the IO Sample method for the following reasons:

- Smaller on FPGA—With large channel count, the IO Sample method is more efficient with FPGA resources, and thus requires less space on the FPGA for the same application.
- Sample flexibility—The IO Sample method gives you more flexibility to change channel range, terminal mode, and channel order at run time and at full sample rate.

- Application timing flexibility—The IO Sample method returns all sample data as soon as it is available. The FPGA I/O Node will gather data for all requested samples and then return all at once.

The **Minimum Time Between Conversions** control you can set in the *C Series Module Properties* dialog box determines the shortest possible time between any two conversions. For channels sampled within the same FPGA I/O Node, the time you set determines the exact time between conversions. For channels sampled within separate FPGA I/O Nodes or for conversions caused by looping on an FPGA I/O Node, the time you set may be less than the actual time between conversions. However, the minimum time you set is never greater than the time between conversions. If the application tries to execute an FPGA I/O Node or IO Sample method faster than the specified minimum time between conversions, the conversion is delayed until the minimum time you set is satisfied.

The default minimum time between conversions for the NI 9204/9205/9206 is 8 µs. The accuracy specifications in the NI 9204, NI 9205, and NI 9206 hardware documentation on ni.com/docs are based on this default value. If you set the minimum time between conversions to at least 8 µs, the accuracy of the module is not affected. If you set the minimum time between conversions to less than 8 µs, the accuracy of the module degrades if you sample data from multiple channels.

Examples

Refer to the NI 9204 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Advanced IO\NI 9204 Advanced IO.lvproj` for an example of conversion timing on the NI 9204 C Series Module.

Refer to the NI 9205 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Advanced IO\NI 9205 Advanced IO.lvproj` for an example of conversion timing on the NI 9205 and NI 9206 C Series Modules.

Self-Calibration for the NI 9204/9205/9206 (FPGA Interface)

Self-Calibration adjusts the calibration constants stored onboard the module. Perform a self-calibration at any time to account for changes in environmental conditions. Enable self-calibration on the *C Series Module Properties* dialog box.

Enabling Self-Calibration on the *NI 9204*, *NI 9205*, or *NI 9206* allows access to the Write Self-Calibration Constants and Restore Last External Calibration Constants module methods.

> **Caution** The Write Self-Calibration Constants and Restore Last External Calibration Constants method nodes must be used with the example VI to prevent damaging the module.

Refer to the NI 9204 Self-Calibration VI in the `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Self Calibration\NI 9204 Self Calibration.lvproj` or to the NI 9205 Self-Calibration VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Self Calibration\NI 9205 Self Calibration.lvproj` for examples of self-calibration.

NI 9205 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9205 has the following channels.

Table 118. Module Channels

Channel	Description
AIx	Analog input channel x, where x is the number of the channel. For the NI 9205, x is 0 to 31.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9205. Right-click the NI 9205 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

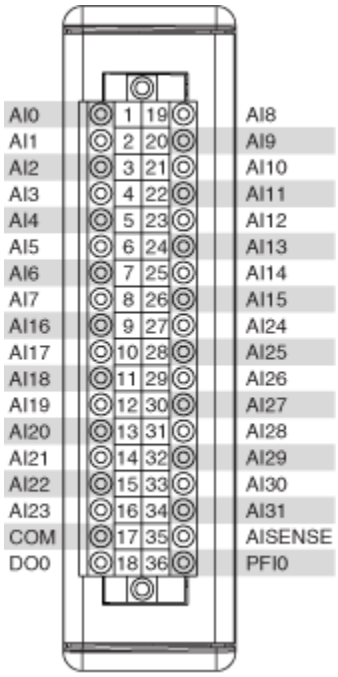
- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.
- **Channel Configuration**—You can specify the input range for each channel.
  - **Channels**—Specifies the channel(s) for which you want to select the input range.
  - **Input Range**—Specifies the input range for the selected channel(s) as ±10 V, ±5 V, ±1 V, or ±200 mV.
  - **Terminal Mode**—Specifies the terminal mode for the selected channel(s) as RSE (referenced single-ended), NRSE (non-referenced single-ended), or DIFF (differential).
  - **Minimum Time between Conversions**—Specifies the time (in uSec) between channel samples. The Scan Engine samples all channels of the module sequentially. The default time between sampling one channel and the next is 8 µs. If you connect all unused channels to ground, you should not have to increase this setting. If for some reason you cannot connect unused channels to ground, you can reduce noise by increasing the minimum time between conversions.

NI 9206

CompactRIO 32-Channel Single-Ended/16-Channel Differential, ±200 mV to ±10 V, 16-Bit Analog Input Module

NI 9206 Pinout

Figure 28. NI 9206 Pinout



NI 9206 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 119. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9206 has AI channels 0 to 31. The channel is sampled using the currently configured terminal mode and voltage range. Configure these settings at edit time using the **C Series Module Properties** dialog box or at run time using the Set Terminal Mode or Set Voltage Range methods.
DI0	Digital input channel used to access the module digital line, which is labeled PFI0. This terminal is not available for an onboard NI 9206 on a Single-Board RIO (sbRIO) device.

Terminal	Description
Trig	Returns the output of the module trigger circuitry. Use the Set Triggers method to configure the trigger circuit.
DO0	Digital output channel 0. This terminal is not available for an onboard NI 9206 on an sbRIO device.

Refer to *Conversion Timing for the NI 9204/9205/9206 (FPGA Interface)* for information about the order and timing of conversions for these I/O channels.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

To access the following methods for this device, use the FPGA I/O Method Node.

Method	Description
AI Sample	Acquires a single AI sample from the module. You can configure the channel number, terminal mode, and voltage range at run time. **Note** Avoid configuring the VI to run an AI Sample method and another FPGA I/O Method Node simultaneously when both target the same C Series module. Do not configure the VI to run multiple AI Sample methods simultaneously when the AI Sample methods target the same C Series module.
IO Sample	Acquires a single sample from the module. The channel number, terminal mode, and voltage range are all configurable at run time. **Note** Avoid configuring the VI to run an IO Sample method and another

Method	Description
	FPGA I/O Method Node simultaneously when both target the same C Series module. Do not configure the VI to run multiple IO Sample methods simultaneously when the IO Sample methods target the same C Series module. Refer to the NI 9206 Basic IO VI and the NI 9206 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9206\NI 9206 Basic IO\NI 9206 Basic IO.lvproj` and `labview\examples\CompactRIO\Module Specific\NI 9206\NI 9206 Advanced IO\NI 9206 Advanced IO.lvproj` for examples related to this method.
Read Module Temperature	Reads the module temperature, represented as -40 °C to 125 °C. **Note** The Read Module Temperature method is only available when Calibration Mode is set to Calibrated in the C Series Module Properties dialog box.
Set Terminal Mode	Sets the terminal mode for a channel as RSE (referenced single-ended), NRSE (non-referenced single-ended), or DIFF (differential). This method overwrites the value you configure in the **C Series Module Properties** dialog box.
Set Triggers	Sets the trigger output to one of the following values: • None • AI Below Low • AI Above High • AI Inside Region • AI Below Low with Hysteresis • AI Above High with Hysteresis • DI Trigger Refer to the NI 9206 Basic Triggering VI and the NI 9206 Advanced Triggering VI in the `labview\examples\CompactRIO\Module Specific\NI 9206\NI 9206 Basic Triggering\NI 9206 Basic Triggering.lvproj` and `labview\examples\CompactRIO\Module Specific\NI 9206\NI 9206 Advanced Triggering\NI 9206 Advanced Triggering.lvproj` for examples related to this method.
Set Voltage Range	Sets the input range for a channel as ±10 V, ±5 V, ±1 V, or ±200 mV. This method overwrites the value you configure in the **C Series Module Properties** dialog box.

I/O Properties

This device does not support any I/O properties.

Module Properties

To access the following module properties for this device, use the FPGA I/O Property Node.

Property	Description
Linearization Coefficient x	(Optional) NI-MCal linearization coefficient x, where *x* is the number of the linearization coefficient. The NI 9206 has linearization coefficients 3 to 0 You can use these coefficients in calibration to compensate for some types of measurement nonlinearity inherent in this device. When returning calibrated data from the FPGA I/O Node, the Linearization Coefficients are not used to calculate the calibrated data. This means that calibrated data returned in FPGA does not compensate for the measurement nonlinearities in the device. Refer to the NI 9206 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9206\NI 9206 Advanced IO\NI 9206 Advanced IO.lvproj` for an example of this concept.
LSB Weight (±10 V range)	Returns the LSB weight in pV/LSB for the ±10 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
LSB Weight (±1 V range)	Returns the LSB weight in pV/LSB for the ±1 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
LSB Weight (±200 mV range)	Returns the LSB weight in pV/LSB for the ±200 mV range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
LSB Weight (±5 V range)	Returns the LSB weight in pV/LSB for the ±5 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Module ID	Returns the module ID, 0x71EB.

Property	Description
Offset (±10 V range)	Returns the calibration offset in nV for the ±10 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Offset (±1 V range)	Returns the calibration offset in nV for the ±1 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Offset (±200 mV range)	Returns the calibration offset in nV for the ±200 mV range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Offset (±5 V range)	Returns the calibration offset in nV for the ±5 V range. Use this value to convert and calibrate data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Conversion Timing for the NI 9204/9205/9206 (FPGA Interface)

AI Sample Method (FPGA Interface)

Use AI Sample method to acquire a single AI sample from the module.

You can configure the channel number, terminal mode, and voltage range at run time. To use this module method, select it in an **FPGA I/O Method Node** that is configured for the appropriate device and/or channel.



Note Avoid configuring the VI to run an AI Sample method and another **FPGA I/O Method Node** simultaneously when both target the same C Series module. Do not configure the VI to run multiple AI Sample methods

simultaneously when the AI Sample methods target the same C Series module.

	**AI Channel** [**i + 2**] is the channel you need to load into the module conversion pipeline. The channel number is part of the configuration information that controls what data the module will sample two iterations later. Refer to *Conversion Timing* for more information.
	**Terminal Mode** [**i + 2**] is an enumeration of the following terminal modes: • RSE is a single-ended measurement between any of the 32 individual channels and the module ground. • NRSE is a single-ended measurement between any of the 32 individual channels and the AISENSE line into the module. • DIFF is a differential measurement between two of the 32 individual channels, with the name of the differential channel matching the channel tied to the positive side of the differential measurement. You can configure channels AI0 to AI7 and AI16 to AI23 to DIFF mode. The terminal mode is part of the configuration that controls the data to be sampled two iterations into the future. Refer to *Conversion Timing* for more information.
	**Voltage Range** [**i + 2**] is an enumeration of the following voltage ranges: • ±10 V • ±5 V • ±1 V • ±200 mV The voltage range is part of the configuration information that controls the data to be sampled two iterations into the future. Refer to *Conversion Timing* for more information.
	**Data** [**i**] returns the data from the current sample.
	**error in** describes error conditions that occur before this VI or function runs.

	The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node. • **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE. • **code** is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code. • **source** always contains an empty string because strings are not supported in LabVIEW FPGA.
	error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error. • **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred. • **code** is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code. • **source** always contains an empty string because strings are not supported in LabVIEW FPGA.

Using This Method

The AI Sample method provides an efficient and flexible interface to the module. You can use this method to acquire a single sample from any of the AI channels on the module, at any range, and with any available input mode.

When you execute this method, the module performs a single conversion on the next channel in the conversion pipeline. The **Data [i]** method output returns the data from

this conversion. As the conversion data is read from the module, the new configuration specified by the **AI Channel [i + 2]**,**Terminal Mode [i + 2]**, and **Voltage Range [i + 2]** inputs is loaded into the module's configuration pipeline. The pipeline is two samples deep. Therefore, the configuration you set during one execution of the AI Sample method determines which channel the AI Sample method will sample two iterations later. Refer to *Conversion Timing* for more details.

IO Sample Method (FPGA Interface)

This module method acquires a single sample from the module. The channel number, terminal mode, and voltage range are all configurable at run time. To use this module method, select it in an FPGA I/O Method Node that is configured for the appropriate device and/or channel.



Note NI suggests that you do not configure the VI to run an IO Sample method and another FPGA I/O Method Node in parallel when both are targeted to the same C Series module. Similarly, NI suggests that you do not configure the VI to run multiple IO Sample methods in parallel when the IO Sample methods are targeted to the same C Series module.

	**Configuration [i+2]** contains encoded configuration information that gets loaded into the module conversion pipeline. This configuration information controls the data to be sampled two iterations into the future. Refer to the *Conversion Timing* topic for more information. Refer to the Configuration Encoding table below for an example of how the configuration information is encoded.

	**Data [i]** Returns the data from the current sample.

	**error in** describes error conditions that occur before this VI or function runs. The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node. •  **status** is TRUE (X) if an error occurred before this VI or function ran or FALSE (checkmark) to indicate a warning or that no error occurred before this VI or function

	ran. The default is FALSE. • **code** is the error code number identifying an error. The default is 0. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is 0 or a warning code. • **source** always contains an empty string because strings are not supported in LabVIEW FPGA.
	error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error. • **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred. • **code** is the error code number identifying an error. The default is 0. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is 0 or a warning code. • **source** always contains an empty string because strings are not supported in LabVIEW FPGA.

Configuration Encoding

The following table describes how to construct the Configuration [i+2] value to write to the IO Sample method to perform a particular operation.

- The **Operation** column lists the types of operations that can be performed using the IO Sample method.
- The **Configuration Bit Fields** column lists the bit positions within the **Configuration [i+2]** number that need to be set.
- The **Values** column specifies the binary value to apply to the specified **Configuration Bit Fields**.

Table 120. Configuration Encoding

Operation	Configuration Bit Fields	Values
AI Read	15:13	001b
	12:11	01b for channels 0-1510b for channels 16-31
	10:8	000b for channels 0, 8, 16, or 24 001b for

Operation	Configuration Bit Fields	Values
		channels 1, 9, 17, or 25…110b for channels 6, 14, 22, or 30111b for channels 7, 15, 23, or 31
	7:6	00b
	5:4	00b for NRSE mode11b for RSE or DIFF modes
	3:2	11b for channels 8-15 or 24-31 01b for DIFF mode on channels 0-7 or 16-2310b for RSE or NRSE mode on channels 0-7 or 16-23
	1:0	00b for ±10 V01b for ±5 V10b for ±1 V11b for ±200 mV
DI0 Read	15:0	0000000000000000b
DO0 Write	15:1	010000000000000b
	0	Binary value to write to DO0

Using this Method

The IO Sample method provides an efficient and flexible interface to the module. You can use this method to acquire a single sample from any of the channels on the module, at any range, and with any available input mode.

When this method is executed, the module performs a single conversion on the next channel present in the conversion pipeline on the module. The data from this conversion is returned via the **Data [i]** method output. At the same time that the conversion data is read from the module, the new configuration information specified by the **Configuration [i+2]** input is loaded into the configuration pipeline on the module. The pipeline is two samples deep. So, the configuration information specified on one execution of the IO Sample method determines which channel will be sampled by the IO Sample method two iterations into the future. Refer to the *Conversion Timing* topic for more details.

Examples

Refer to the NI 9205 Basic IO VI and the NI 9205 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Basic IO\NI 9205 Basic IO.lvproj` and `labview\examples\CompactRIO\`

`Module Specific\NI 9205\NI 9205 Advanced IO\NI 9205 Advanced IO.lvproj` for examples of the IO concepts discussed above.

Set Terminal Mode Method (FPGA Interface)

The Set Terminal Mode method overrides the default setting provided in the dialog box. To use this module method, select it in an **FPGA I/O Method Node** that is configured for the appropriate device and/or channel.

	AI Channel is the channel you want to set the terminal mode for.
	**Terminal Mode** is an enumeration of the following trigger modes: • **RSE** is a single-ended measurement between any of the 16 (NI 9204) or 32 (NI 9205/9206) individual channels and the module ground. • **NRSE** is a single-ended measurement between any of the 16 (NI 9204) or 32 (NI 9205/9206) individual channels and the AISENSE line into the module. • **DIFF** is a differential measurement between two of the 16 (NI 9204) or 32 (NI 9205/9206) individual channels, with the name of the differential channel matching the channel tied to the positive side of the differential measurement. Channels AI0 to AI7 and AI16 to AI23 (NI 9205/9206 only) may be set to DIFF configuration.

	**error in** describes error conditions that occur before this VI or function runs. The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node. **status** is *TRUE* (X) if an error occurred before this VI or function ran or *FALSE* (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is FALSE. **code** is the error or warning code. The default is `0`. If **status** is TRUE, **code** is a nonzero *error code*. If **status** is FALSE, **code** is `0` or a warning code. **source** always contains an empty string because strings are not supported in LabVIEW FPGA.

	error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error. **status** is TRUE (X) if an error occurred or FALSE (checkmark) to indicate a warning or that no error occurred. **code** is the error or warning code. If **status** is TRUE, **code** is a nonzero *error code*. If **status** is FALSE, **code** is 0 or a warning code. **source** always contains an empty string because strings are not supported in LabVIEW FPGA.

Set Triggers Method (FPGA Interface)

This module method controls the data returned by the trigger digital input line. Use this module method by selecting it in an **FPGA I/O Method Node** that is configured for the appropriate device and/or channel.

	Trigger Mode is an enumeration of the following trigger mode options: • **None** disables the trigger function. • **AI Below Low** detects when the analog signal is below the low threshold you specify. • **AI Above High** detects when the analog signal is above the high threshold you specify. • **AI Inside Region** detects when the analog signal is between the high and low thresholds you specify. • **AI Below Low with Hysteresis** asserts the trigger when the signal starts above the high threshold you specify and then crosses below the low threshold you specify. The trigger deasserts when the signal crosses above the high threshold you specify. • **AI Above High with Hysteresis** asserts the trigger when the signal starts below the low threshold you specify and then crosses above the high threshold you specify. The trigger deasserts when the signal crosses below the low threshold you specify. • **DI Trigger** detects when the digital signal on PFI0 is logic high. **Note** The onboard NI 9205 on *Single-Board RIO* (sbRIO) devices does not support DI triggering.
	AI High Threshold defines the upper threshold for the analog trigger circuit.

	AI Low Threshold defines the lower threshold for the analog trigger circuit.

	error in describes error conditions that occur before this VI or function runs. The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

	status is TRUE if an error occurred. If **status** is TRUE, the VI does not perform any operations.
	code is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code.
	source always contains an empty string because strings are not supported in LabVIEW FPGA.

	error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error.

	status is TRUE if an error occurred. If **status** is TRUE, the VI does not perform any operations.
	code is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code.
	source always contains an empty string because strings are not supported in LabVIEW FPGA.

Using This Method

Use the Set Triggers method to configure the trigger circuit on the module. Once the trigger circuit is configured, use an FPGA I/O Node configured to access the Trig channel to monitor the output of the trigger circuit. The trigger circuit does not

automatically initiate any conversions on the module. It is up to the FPGA VI to monitor the Trig line, and then initiate a conversion using an FPGA I/O Node or the IO Sample method when the appropriate trigger event occurs.

Perform the following steps to trigger an acquisition using the Set Triggers method:

1. Ensure that the module pipeline is primed with the next two channels that you want to sample after the trigger event. This step will minimize the delay between the trigger event and the AI channels getting sampled. For analog triggering modes, this step also configures the analog trigger circuitry to monitor the desired channel. The AI trigger circuit monitors the channel waiting in the module pipeline for the next conversion. If you are using FPGA I/O Nodes to perform the acquisition, this step can be accomplished by executing an FPGA I/O Node configured to match the FPGA I/O Node that you will be executing after the trigger event. If you are using one of the analog triggering modes along with the FPGA I/O Nodes, the AI trigger circuit will operate on the lowest numbered channel present in the FPGA I/O Node. Refer to the *IO Sample* method and *Conversion Timing* topics for more information on managing the module pipeline using the IO Sample method. If you are using one of the analog trigger modes along with the IO Sample method, the AI trigger circuit will operate on the channel waiting in the pipeline to be converted by the next IO Sample method.
2. Execute the *Set Triggers* method with the appropriate trigger mode selected. The trigger mode circuitry is implemented in the hardware as you specify in this method. Refer to the NI 9204, NI 9205, and NI 9206 hardware documentation for more information about the trigger modes.
3. Monitor the Trig channel in a loop until you see the appropriate trigger condition occur.
4. Execute an FPGA I/O Node or IO Sample method function to acquire data in response to the trigger event.

Examples

Refer to the NI 9204 Basic Triggering VI and the NI 9204 Advanced Triggering VI in the `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Basic Triggering\NI 9204 Basic Triggering.lvproj` and `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Advanced Triggering\NI 9204 Advanced Triggering.lvproj` for examples of the triggering concepts discussed above or to the NI 9205 Basic Triggering

VI and the NI 9205 Advanced Triggering VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Basic Triggering\NI 9205 Basic Triggering.lvproj` and `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Advanced Triggering\NI 9205 Advanced Triggering.lvproj` for examples of the triggering concepts discussed above.

Set Voltage Range Method (FPGA Interface)

This module method overrides the default setting provided in the *C Series Module Properties* dialog box. To use this module method, select it in an **FPGA I/O Method Node** that is configured for the appropriate device and/or channel.

	**AI Channel** is the channel you want to set the voltage range for.
	**Voltage Range** is an enumeration of the following voltage ranges: • ±10 V • ±5 V • ±1 V • ±200 mV

	**error in** describes error conditions that occur before this VI or function runs. The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

	**status** is TRUE if an error occurred. If **status** is TRUE, the VI does not perform any operations.
	**code** is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code.
	**source** always contains an empty string because strings are not supported in LabVIEW FPGA.

	error out contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error.

	status is TRUE if an error occurred. If **status** is TRUE, the VI does not perform any operations.
	code is the error code number identifying an error. The default is `0`. If status is TRUE, code is a nonzero *error code*. If status is FALSE, code is `0` or a warning code.
	source always contains an empty string because strings are not supported in LabVIEW FPGA.

Conversion Timing for the NI 9204/9205/9206 (FPGA Interface)

The *NI 9204*, *NI 9205*, and *NI 9206* modules implement a two-element deep pipeline for access to the AI, DI0, and DO0 (NI 9205/9206 only) channels. This pipeline results in maximum sample rate and maximum sample quality. The Trig channel has a parallel data path that bypasses the pipeline.

When using the FPGA I/O Node to sample channels, the pipeline is automatically managed by the FPGA I/O Node, and the channels within the FPGA I/O Node are sampled in numerical order regardless of the order they appear in the node.

If the first two channel requests in the FPGA I/O Node do not match the two channel requests stored in the module pipeline, there will be a delay before the first channel sample occurs. This delay is caused by the FPGA I/O Node automatically updating the module channel sample pipeline, which takes two channel sample cycles.

To minimize the need for the pipeline updates, each FPGA I/O Node leaves the module pipeline primed to repeat itself. If you use only one FPGA I/O Node in a looping structure, only the first iteration of the loop will incur the two-cycle time delay. All subsequent iterations operate with no delay.

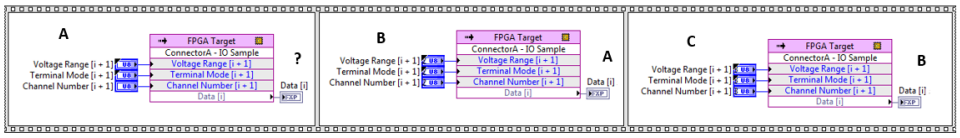
When using the IO Sample method, take steps to manage this pipeline in the VIs.

- To read one channel, that channel configuration must be requested two cycles

before the time it is to be sampled. The U16 configuration value for an AI channel sample contains the channel number, channel input range, and terminal mode. The U16 configuration value also can represent an access to the DIO or DOO (NI 9205/9206 only) channels, which is treated the same as an AI operation.

The following diagram illustrates how the pipeline works within a sequence structure. A typical application uses a loop structure to continuously iterate through a scan list. The scan list is a predefined list of configurations. The sequence structure and art elements show how data moves into the NI 9204/9205/9206 pipeline and then is converted. The IO Sample method first converts the configuration that is in the ADC, and then shifts the configuration values through the pipeline to prepare for the next time LabVIEW calls the IO Sample method.

Figure 29. Pipeline Behavior within a Sequence Structure



The simplest form of pipeline management is to force the pipeline on every set of samples. This assumes that the pipeline is always wrong at the first sample, and sends all IO Sample requests in order. The last two data points need some configuration to be requested, but it does not matter, in this use case, what configuration that is.

A better form of pipeline management emulates the behavior of the FPGA I/O Node. Instead of always assuming that the pipeline is wrong, you choose the last two configuration values based on the first two samples you want to include in the next iteration. This pipeline management system can be accomplished in two parts: initialization and sample iteration.

Tip NI recommends you use the IO Sample method for the following reasons:

- Smaller on FPGA—With large channel count, the IO Sample method is more efficient with FPGA resources, and thus requires less space on the FPGA for the same application.
- Sample flexibility—The IO Sample method gives you more flexibility to

change channel range, terminal mode, and channel order at run time and at full sample rate.
- Application timing flexibility—The IO Sample method returns all sample data as soon as it is available. The FPGA I/O Node will gather data for all requested samples and then return all at once.

The **Minimum Time Between Conversions** control you can set in the *C Series Module Properties* dialog box determines the shortest possible time between any two conversions. For channels sampled within the same FPGA I/O Node, the time you set determines the exact time between conversions. For channels sampled within separate FPGA I/O Nodes or for conversions caused by looping on an FPGA I/O Node, the time you set may be less than the actual time between conversions. However, the minimum time you set is never greater than the time between conversions. If the application tries to execute an FPGA I/O Node or IO Sample method faster than the specified minimum time between conversions, the conversion is delayed until the minimum time you set is satisfied.

The default minimum time between conversions for the NI 9204/9205/9206 is 8 µs. The accuracy specifications in the NI 9204, NI 9205, and NI 9206 hardware documentation on ni.com/docs are based on this default value. If you set the minimum time between conversions to at least 8 µs, the accuracy of the module is not affected. If you set the minimum time between conversions to less than 8 µs, the accuracy of the module degrades if you sample data from multiple channels.

Examples

Refer to the NI 9204 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Advanced IO\NI 9204 Advanced IO.lvproj` for an example of conversion timing on the NI 9204 C Series Module.

Refer to the NI 9205 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Advanced IO\NI 9205 Advanced IO.lvproj` for an example of conversion timing on the NI 9205 and NI 9206 C Series Modules.

Self-Calibration for the NI 9204/9205/9206 (FPGA Interface)

Self-Calibration adjusts the calibration constants stored onboard the module. Perform a self-calibration at any time to account for changes in environmental conditions. Enable self-calibration on the *C Series Module Properties* dialog box.

Enabling Self-Calibration on the *NI 9204*, *NI 9205*, or *NI 9206* allows access to the Write Self-Calibration Constants and Restore Last External Calibration Constants module methods.

> **Caution** The Write Self-Calibration Constants and Restore Last External Calibration Constants method nodes must be used with the example VI to prevent damaging the module.

Refer to the NI 9204 Self-Calibration VI in the `labview\examples\CompactRIO\Module Specific\NI 9204\NI 9204 Self Calibration\NI 9204 Self Calibration.lvproj` or to the NI 9205 Self-Calibration VI in the `labview\examples\CompactRIO\Module Specific\NI 9205\NI 9205 Self Calibration\NI 9205 Self Calibration.lvproj` for examples of self-calibration.

NI 9206 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9206 has the following channels.

Table 121. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9206, *x* is 0 to 31.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9206. Right-click the NI 9206 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

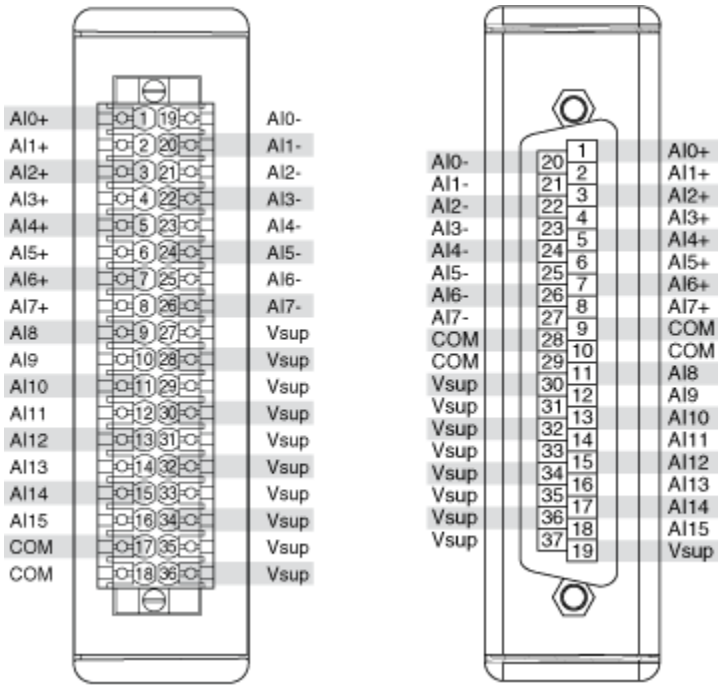
- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.
- **Channel Configuration**—You can specify the input range for each channel.
  - **Channels**—Specifies the channel(s) for which you want to select the input range.
  - **Input Range**—Specifies the input range for the selected channel(s) as ±10 V, ±5 V, ±1 V, or ±200 mV.
  - **Terminal Mode**—Specifies the terminal mode for the selected channel(s) as RSE (referenced single-ended), NRSE (non-referenced single-ended), or DIFF (differential).
  - **Minimum Time between Conversions**—Specifies the time between channel samples. The Scan Engine samples all channels of the module sequentially. The default time between sampling one channel and the next is 8 µs. If you connect all unused channels to ground, you should not have to increase this setting. If for some reason you cannot connect unused channels to ground, you can reduce noise by increasing the minimum time between conversions.

NI 9207

CompactRIO 16-Channel, ±20 mA/±10 V, 24-Bit Isolated Analog Input Module

NI 9207 Pinout

Figure 30. NI 9207 Pinout



NI 9207 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 122. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9207 has AI channels 0 to 15.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

This device does not support any module methods.

I/O Properties

This device does not support any I/O properties.

Module Properties

To access the following module properties for this device, use the FPGA I/O Property Node.

Table 123. Module Properties

Property	Description
Conversion Time	Sets the time it takes to acquire one point of data from one or more channels in a single FPGA I/O Node. Options are **High Speed** or **High Resolution**. This property overwrites the value configured in the C Series Module Properties dialog box. Returns the conversion time setting when configured to read. Refer to the NI 9207 hardware documentation for more information about conversion times.
Module ID	Returns the module ID, 0x74AD.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Converting NI 9207 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the *C Series Module Properties* dialog box for the *NI 9207* if you want the FPGA I/O Node to return calibrated, fixed-point data from the module. Set the **Calibration Mode** to **Raw** if you want the FPGA I/O Node to return calibrated, binary data from the module. If you set the **Calibration Mode** to **Raw**, you must convert the binary values to engineering units. You must convert these values in the host VI.

Using an Equation to Convert Binary Current Values to Amperes

Use the following equation in the host VI to convert binary current values to amperes:

Amps = *Binary Value* × 0.022 A ÷ 8,388,607

Where

- *Binary Value* is the value returned by the FPGA I/O Node.

Using an Equation to Convert Binary Voltage Values to Volts

Use the following equation in the host VI to convert binary voltage values to volts:

Volts = *Binary Value* × 10.4 V ÷ 8,388,607

Where

- *Binary Value* is the value returned by the FPGA I/O Node.

NI 9207 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for channels AI0–AI7 return calibrated floating-point data in volts. The I/O variables for channels AI8–AI15 return calibrated floating-point data in amps.

Module Channels

The NI 9207 has the following channels.

Table 124. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9207, *x* is 0 to 15.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. To display this dialog box, right-click the module in the **Project Explorer** window and select **Properties**. You can configure the following options.

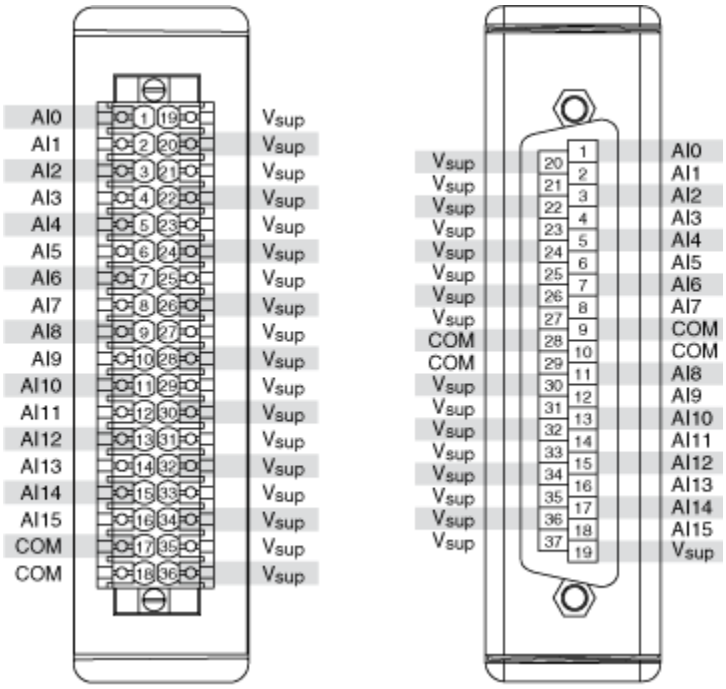
- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.
- **Conversion Time**—Determines the time it takes to acquire one point of data from one or more channels. Select **High Speed** or **High Resolution**. For more information about the conversion times, refer to the NI 9207 hardware documentation.

NI 9208

CompactRIO 16-Channel, ±20 mA, 24-Bit Isolated Analog Input Module

NI 9208 Pinout

Figure 31. NI 9208 Pinout



NI 9208 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 125. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9208 has AI channels 0 to 15.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

This device does not support any module methods.

I/O Properties

This device does not support any I/O properties.

Module Properties

To access the following module properties for this device, use the FPGA I/O Property Node.

Table 126. Module Properties

Property	Description
Conversion Time	Sets the time it takes to acquire one point of data from one or more channels in a single FPGA I/O Node. Options are **High Speed** or **High Resolution**. Returns the conversion time setting when configured to read. Refer to the NI 9207 hardware documentation for more information about conversion times.
Module ID	Returns the module ID, 0x74AC.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Converting NI 9208 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the *C Series Module Properties* dialog box for the *NI 9208* if you want the FPGA I/O Node to return calibrated, fixed-point data from the module in units of amps. Set the **Calibration Mode** to **Raw** if you want the FPGA I/O Node to return calibrated, binary data from the module. If you set the **Calibration Mode** to **Raw**, you must convert the binary current values to amps. You must convert these values in the host VI.

Use the following equation in the host VI to convert binary current values to amperes:

$$Amps = Binary\ Value \times 0.022\ A \div 8{,}388{,}607$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.

NI 9208 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9208 has the following channels.

Table 127. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9208, *x* is 0 to 15.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. To display this dialog box, right-click the module in the **Project Explorer** window and select **Properties**. You can configure the

following options.

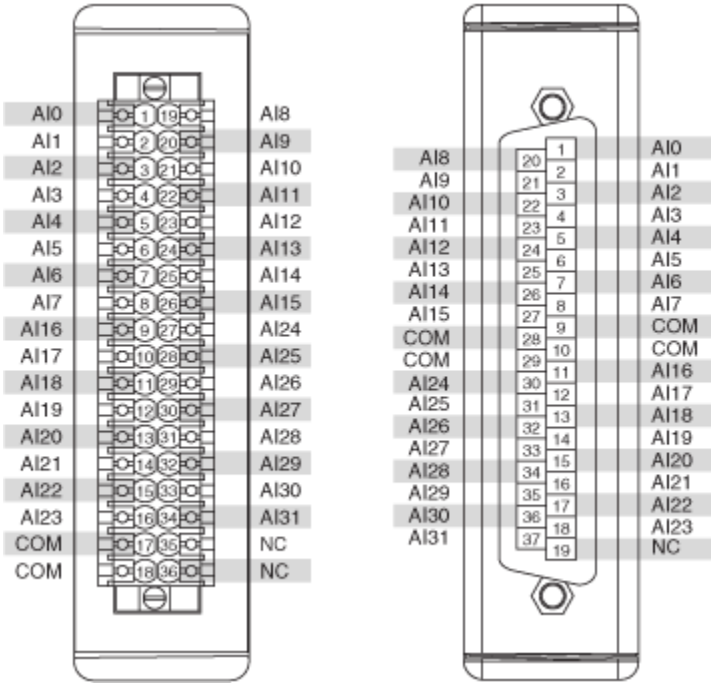
- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.
- **Conversion Time**—Determines the time it takes to acquire one point of data from one or more channels. Select **High Speed** or **High Resolution**. For more information about the conversion times, refer to the NI 9207 hardware documentation.

NI 9209

CompactRIO 16-Channel Differential, 32-Channel Single-Ended, ±10 V, 24-Bit Analog Input Module

NI 9209 Pinout

Figure 32. NI 9209 Pinout



NI 9209 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 128. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9209 has AI channels 0 to 31.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

To access the following method for this device, use the FPGA I/O Method Node.

Table 129. Module Methods

Method	Description
Set Terminal Mode	Sets the terminal mode for a channel as RSE (referenced single-ended) or DIFF (differential). This method overwrites the value configured in the C Series Module Properties dialog box.

I/O Properties

This device does not support any I/O properties.

Module Properties

To access the following module properties for this device, use the FPGA I/O Property Node.

Table 130. Module Properties

Property	Description
Conversion Time	Sets the time it takes to acquire one point of data from one or more channels in a single FPGA I/O Node. Options are **High Speed** or **High Resolution**. This property overwrites the value configured in the C Series Module Properties dialog box. Returns the conversion time setting when configured to read. Refer to the NI 9209 hardware documentation for more information about conversion times.
Module ID	Returns the module ID, 0x77F3.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Converting NI 9209 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the *C Series Module Properties* dialog box for the *NI 9209* if you want the FPGA I/O Node to return calibrated, fixed-point data from the module. Set the **Calibration Mode** to **Raw** if you want the FPGA I/O Node to return calibrated, binary data from the module. If you set the **Calibration Mode** to **Raw**, you must convert the binary values to engineering units. You must convert these

values in the host VI.

Using an Equation to Convert Binary Voltage Values to Volts

Use the following equation in the host VI to convert binary voltage values to volts:

$$Volts = Binary\ Value \times 10.4\ V \div 8{,}388{,}607$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.

NI 9209 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9209 has the following channels.

Table 131. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9209, *x* is 0 to 31.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. To display this dialog box, right-click the module in the **Project Explorer** window and select **Properties**. You can configure the following options.

- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.

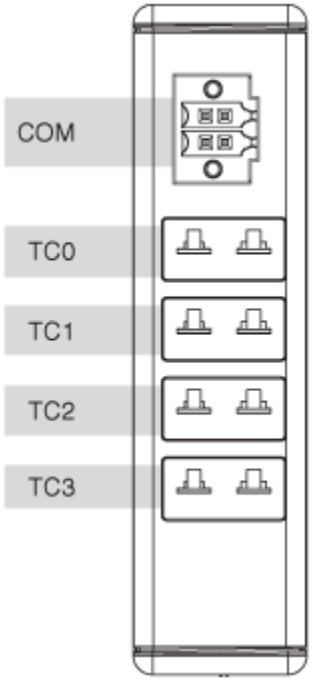
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.
- **Conversion Time**—Determines the time it takes to acquire one point of data from one or more channels. Select **High Speed** or **High Resolution**. For more information about the conversion times, refer to the NI 9209 hardware documentation.

NI 9210

CompactRIO 4-Channel, ±80 mV, 24-Bit Thermocouple Input Module

NI 9210 Pinout

Figure 33. NI 9210 Pinout



NI 9210 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 132. Terminals in Software

Terminal	Description
TC*x*	Thermocouple input channel *x*, where *x* is the number of the channel. The NI 9210 has TC channels 0 to 3.
CJC	Cold-junction compensation channel. For the best accuracy, read the CJC channel in the same FPGA I/O Node as the thermocouple input channels. Convert the CJC data to temperature.
Autozero	Autozero channel. For the best accuracy, read the Autozero channel in the same FPGA I/O Node as the thermocouple input channels.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

This device does not support any I/O properties.

Module Properties

To access the following module properties for this device, use the FPGA I/O Property Node.

Table 133. Module Properties

Property	Description
Module ID	Returns the module ID, 0x77F3.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Converting NI 9210 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the *C Series Module Properties* dialog box for the *NI 9210* if you want the FPGA I/O Node to return calibrated, fixed-point data from the module in units of volts. If you set the **Calibration Mode** to **Calibrated**, convert the fixed-point CJC data to binary CJC data and then convert the binary CJC data to temperature.

Set the **Calibration Mode** to **Raw** if you want the FPGA I/O Node to return calibrated, binary data from the module. If you set the **Calibration Mode** to **Raw**, you must convert the binary thermocouple values to voltage and convert the binary CJC data to temperature. Convert these values in the host VI.

Using a VI to Convert Data to Temperature

Refer to the Convert to Temperature (NI 9210) polymorphic VI in the `labview\examples\CompactRIO\Module Specific\NI 9210\NI 9210 Support Files.llb` for an example of converting calibrated or raw data to temperature. You can use the Convert to Temperature (NI 9210) VI as a subVI in the host VI.

Use the following equation in the host VI to convert binary thermocouple values to voltage:

$$Voltage = Binary\ Value \times 80\ mV \div 8{,}388{,}607$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.

Using an Equation to Convert Fixed-Point CJC Data to Binary CJC Data

Use the following equation in the host VI to convert fixed-point CJC data to binary CJC data:

$$Binary\ CJC\ Data = (Fixed\text{-}Point\ CJC\ Data) \div (0.160 \div (2^{24} - 1))$$

Where

- *Fixed-Point CJC Data* is the value returned by the FPGA I/O Node.

Using Equations to Convert Binary CJC Data to Temperature

Use the following equations in the host VI to convert binary CJC data to temperature:

Calculate the resistance of the thermistor:

$$R_T = (10000 \times Binary\ CJC\ Data) \div (2^{23} - Binary\ CJC\ Data)$$

Calculate the CJC temperature:

$$T = [\ 1 \div [A + B(\ln(R_T)) + C(\ln(R_T))^3]] - (273.15 + OffsetConstant)$$

Where

- T = temperature in °C
- $A = 1.2873851 \times 10^{-3}$
- $B = 2.3575235 \times 10^{-4}$
- $C = 9.4978060 \times 10^{-8}$
- R_T = thermistor resistance reading
- *OffsetConstant*[8] = 0.1

Refer to the National Institute of Standards and Technology (NIST) Monograph 175 thermocouple reference tables for more information about converting and adjusting thermocouple values.

8. The *OffsetConstant* is the typical temperature gradient between the CJC sensor and the thermocouple cold junction.

NI 9210 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in units as specified on the C Series Module Properties dialog box.

Note The Scan Interface applies Autozero and cold-junction compensation to NI 9210 data returned in temperature units.

Module Channels

The NI 9210 has the following channels.

Table 134. Module Channels

Channel	Description
AI*x*	Analog input channel x, where x is the number of the channel. For the NI 9210, x is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. To display this dialog box, right-click the module in the **Project Explorer** window and select **Properties**. You can configure the following options.

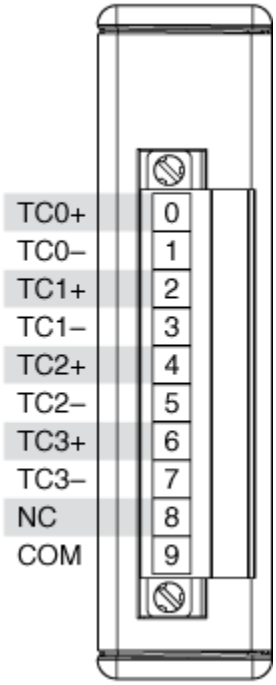
- **Name**—Specifies the name of the C Series module. The name is shown in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. Use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.
- **Thermocouple Type**—Specifies the type of thermocouple connected to the channel.
- **Measurement Units**—Specifies the units you want data to be returned in for the channel. You can select Raw Volts, Degrees Kelvin, Degrees Celsius, Degrees Fahrenheit, or Degrees Rankine.

NI 9211

CompactRIO 4-Channel, ±80 mV, 24-Bit Thermocouple Input Module

NI 9211 Pinout

Figure 34. NI 9211 Pinout



NI 9211 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 135. Terminals in Software

Terminal	Description
TC*x*	Thermocouple input channel *x*, where *x* is the number of the channel. The NI 9211 has TC channels 0 to 3.
CJC	Cold-junction compensation channel. For the best accuracy, read the CJC channel in the

Terminal	Description
	same FPGA I/O Node as the thermocouple input channels. Convert the CJC data to temperature.
Autozero	Autozero channel. For the best accuracy, read the Autozero channel in the same FPGA I/O Node as the thermocouple input channels.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

This device does not support any I/O properties.

Module Properties

To access the following module properties for this device, use the FPGA I/O Property Node.

Table 136. Module Properties

Property	Description
Module ID	Returns the module ID, 0x70A3.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Converting NI 9211 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the *C Series Module Properties* dialog box for the *NI 9211* if you want the FPGA I/O Node to return calibrated, fixed-point data from the module in units of volts. If you set the **Calibration Mode** to **Calibrated**, you must convert the fixed-point CJC data to binary CJC data and then convert the binary CJC data to temperature.

Set the **Calibration Mode** to **Raw** if you want the FPGA I/O Node to return calibrated, binary data from the module. If you set the **Calibration Mode** to **Raw**, you must convert the binary thermocouple values to voltage and convert the binary CJC data to temperature. You must convert these values in the host VI.



Note The equations below apply only to the NI 9211. If you are using the NI 9211E board-only module, refer to the KnowledgeBase for the correct equations.

Using a VI to Convert Data to Temperature

Refer to the Convert to Temperature (NI 9211) polymorphic VI in the `labview\examples\CompactRIO\Module Specific\NI 9211\NI 9211 Support Files.llb` for an example of converting calibrated or raw data to temperature. You can use the Convert to Temperature (NI 9211) VI as a subVI in the host VI.

Use the following equation in the host VI to convert binary thermocouple values to voltage:

$$Voltage = Binary\ Value \times 80\ mV \div 8{,}388{,}607$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.

Using an Equation to Convert Fixed-Point CJC Data to Binary CJC Data

Use the following equation in the host VI to convert fixed-point CJC data to binary CJC data:

Binary CJC Data = (*Fixed-Point CJC Data*) ÷ (0.160 ÷ (2^{24} – 1))

Where

- *Fixed-Point CJC Data* is the value returned by the FPGA I/O Node.

Using Equations to Convert Binary CJC Data to Temperature

Use the following equations in the host VI to convert binary CJC data to temperature:

Calculate the resistance of the thermistor:

$$R_T = (10000 \times \textit{Binary CJC Data}) \div (2^{23} - \textit{Binary CJC Data})$$

Calculate the CJC temperature:

$$T = [\ 1 \div [A + B(\ln(R_T)) + C(\ln(R_T))^3]] - (273.15 + \textit{OffsetConstant})$$

Where

- T = temperature in °C
- $A = 1.2873851 \times 10^{-3}$
- $B = 2.3575235 \times 10^{-4}$
- $C = 9.4978060 \times 10^{-8}$
- R_T = thermistor resistance reading
- *OffsetConstant*[9] = 0.7

Refer to the National Institute of Standards and Technology (NIST) Monograph 175 thermocouple reference tables for more information about converting and adjusting thermocouple values.

9. The *OffsetConstant* is the typical temperature gradient between the CJC sensor and the thermocouple cold junction.

NI 9211 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in units as specified on the C Series Module Properties dialog box.

Note The Scan Interface applies Autozero and cold-junction compensation to NI 9211 data returned in temperature units.

Module Channels

The NI 9211 has the following channels.

Table 137. Module Channels

Channel	Description
AI*x*	Analog input channel x, where x is the number of the channel. For the NI 9211, x is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. To display this dialog box, right-click the module in the **Project Explorer** window and select **Properties**. You can configure the following options.

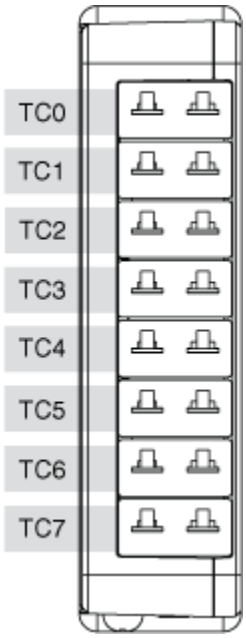
- **Name**—Specifies the name of the C Series module. The name is shown in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. Use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.
- **Thermocouple Type**—Specifies the type of thermocouple connected to the channel.
- **Measurement Units**—Specifies the units you want data to be returned in for the channel. You can select Raw Volts, Degrees Kelvin, Degrees Celsius, Degrees Fahrenheit, or Degrees Rankine.

NI 9212

CompactRIO 8-Channel, ±78 mV, 24-Bit Channel-to-Channel Isolated Thermocouple Input Module

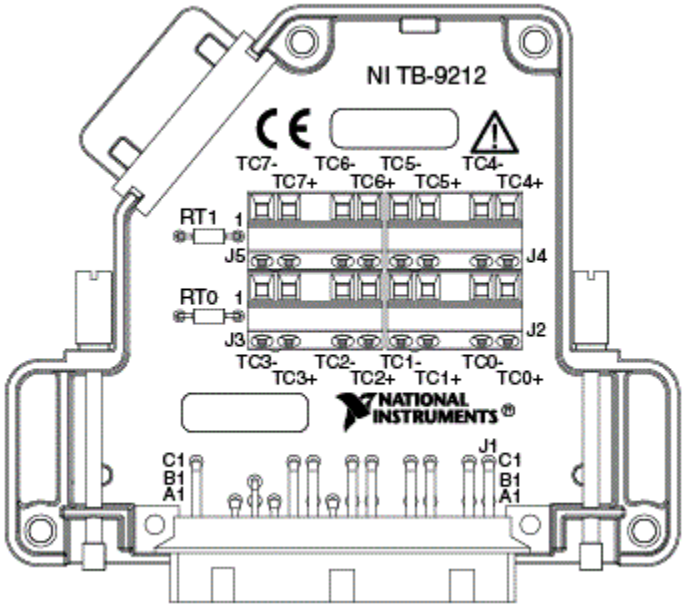
NI 9212 Pinout

Figure 35. NI 9212 Pinout



The NI TB-9212 provides connections for eight thermocouple channels on the NI 9212.

Figure 36. NI TB-9212 Pinout



NI 9212 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 138. Terminals in Software

Terminal	Description
TC*x*	Thermocouple input channel *x*, where *x* is the number of the channel. The NI 9212 has TC channels 0 to 7.
CJC	Cold-junction compensation channel *x*, where *x* is the number of the channel. The NI 9212 has CJC channels 0 to 1. For the best accuracy, read the appropriate CJC channel in the same FPGA I/O Node as the corresponding thermocouple input channels. Convert the CJC data to temperature. • **CJC 0**—corresponds to thermocouple input channels TC0, TC1, TC2, and TC3. • **CJC 1**—corresponds to thermocouple input channels TC4, TC5, TC6, and TC7.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

To access the following method for this device, use the FPGA I/O Method Node.

Table 139. Module Methods

Method	Description
Check Cached Status	Returns Booleans for each channel that indicate whether the channel was out of range or had an open thermocouple since the last execution of the Check Cached Status method. When the FPGA I/O Node reads the channels, the FPGA VI determines the state of the channels and caches any TRUE value until the Check Cached Status method executes. • **Force Status Read**—When the value of this input is FALSE, the method returns the cached status information since the last time that the Check Cached Status method executed. When the value is TRUE, the method forces the FPGA I/O Node to read all channels and update the status information. Forcing a status read can introduce jitter into an analog input loop. • **Open**—Returns an array of Boolean values. A value of TRUE in any index indicates that the channel sharing a number with that index detected an open thermocouple on the channel at some point after the last time that the Check Cached Status method executed.

I/O Properties

This device does not support any I/O properties.

Module Properties

To access the following module properties for this device, use the FPGA I/O Property Node.

Table 140. Module Properties

Property	Description
Conversion Time	Sets the time it takes to acquire one point of data from one or more channels in a single FPGA I/O Node. Select High Speed, Best 60 Hz Rejection, Best 50 Hz Rejection, or High Resolution. Refer to the *NI 9212 with NI TB-9212 Specifications* for more information about the conversion times. This property overwrites the value configured in the C Series Module Properties dialog box. Returns the conversion time setting when configured to read.
Module ID	Returns the module ID, 0x7705.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related information:

- NI-9212 with TB-9212 Specifications

Converting NI 9212 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the *C Series Module Properties* dialog box for the *NI 9212* if you want the FPGA I/O Node to return calibrated, fixed-point data from the module in units of volts. If you set the **Calibration Mode** to **Calibrated**, you must convert the CJC data from voltage to temperature.

Set the **Calibration Mode** to **Raw** if you want the FPGA I/O Node to return calibrated, binary data from the module. If you set the **Calibration Mode** to **Raw**, you must convert the binary thermocouple and CJC values to voltage and then convert the CJC data from voltage to temperature. You must convert these values in the host VI.

Using a VI to Convert Data to Temperature

Refer to the NI 9212 Convert to Temperature polymorphic VI in the `labview\examples\CompactRIO\Module Specific\NI 9212\NI 9212 Getting Started\NI 9212 Getting Started.lvproj` for an example of converting calibrated or raw data to temperature. Use the NI 9212 Convert to Temperature VI as a subVI in the host VI.

Use the following equation in the host VI to convert the binary thermocouple and CJC values to voltage:

$$Voltage = Binary\ Value \times 78.125\ mV \div 8{,}388{,}607$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.

Converting CJC Data from Voltage to Temperature

The NI 9212Convert Thermistor Reading VI is a subVI in the NI 9212 Convert to Temperature polymorphic VI that converts CJC data from voltage to temperature.

The VI uses the following equation to calculate the resistance of the thermistor:

$$R_T = [(CJC\ Data \div 0.078125) \div (1 - (CJC\ Data \div 0.078125))] \times 28{,}000$$

Using the resistance of the thermistor, the VI references a look-up table to interpolate the CJC temperature.

Refer to the National Institute of Standards and Technology (NIST) Monograph 175 thermocouple reference tables for more information about converting and adjusting thermocouple values.

NI 9212 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in units as specified on the C Series Module Properties dialog box.

Note The Scan Interface applies cold-junction compensation to NI 9212 data returned in temperature units.

Module Channels

The NI 9212 has the following channels.

Table 141. Module Channels

Channel	Description
TC*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9212, *x* is 0 to 7.

Module-Specific Errors

The NI 9212 can return the following module-specific error.

Table 142. Module-Specific Errors

Error Code	Description
–65582	An open thermocouple was detected.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. To display this dialog box, right-click the module in the **Project Explorer** window and select **Properties**. You can configure the following options.

- **Name**—Specifies the name of the C Series module, which appears in the **Project**

Explorer window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.

- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.
- **Conversion Time**—Determines the time it takes to acquire one point of data from one or more channels. Select High Speed, Best 60 Hz Rejection, Best 50 Hz Rejection, or High Resolution. Refer to the *NI 9212 with NI TB-9212 Specifications* for more information about the conversion times.
- **Channels**—Specifies the channel(s) for which you want to configure settings.
- **Thermocouple Type**—Specifies the type of thermocouple connected to the channel.
- **Measurement Units**—Specifies the units you want data to be returned in for the channel. You can select Raw Volts, Degrees Kelvin, Degrees Celsius, Degrees Fahrenheit, or Degrees Rankine.
- **Enable Open Thermocouple Detection**—If this box is checked, the NI 9212 returns a zero value and reports an error if it detects an open thermocouple on any channel. This box is checked by default. If you do not check this box, each channel of the NI 9212 independently returns full-scale data and the NI 9212 does not report an error if an open thermocouple is detected.

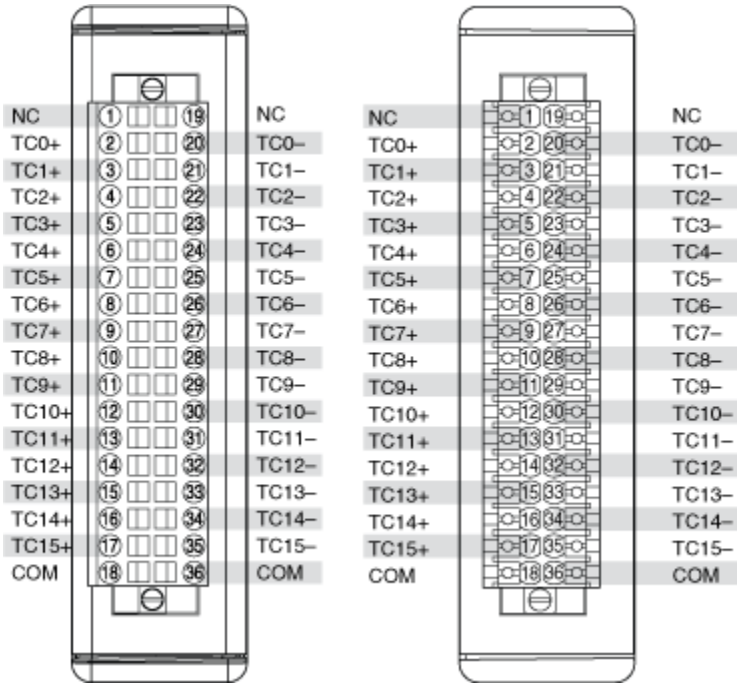
> **Note** Enabling or disabling open thermocouple detection on the NI 9212 has no effect on the accuracy of the measurement because this option on the module is implemented entirely from software.

NI 9213

CompactRIO 16-Channel, ±78 mV, 24-Bit Thermocouple Input Module

NI 9213 Pinout

Figure 37. NI 9213 Pinout



NI 9213 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 143. Terminals in Software

Terminal	Description
TC*x*	Thermocouple input channel *x*, where *x* is the number of the channel. The NI 9213 has TC channels 0 to 15.
Autozero	Autozero channel. For the best accuracy, read the Autozero channel in the same FPGA I/O Node as the thermocouple input channels.
CJC	Cold-junction compensation channel. For the best accuracy, read the CJC channel in the same FPGA I/O Node as the thermocouple input channels. Convert the CJC data to temperature.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Module Methods

To access the following method for this device, use the FPGA I/O Method Node.

Table 144. Module Methods

Method	Description
Check Cached Status	Returns a Boolean value for each channel that indicates whether the channel was out of range or had an open thermocouple since the last execution of the Check Cached Status method. When the FPGA I/O Node reads the channels, the FPGA VI determines the state of the channels and caches any *TRUE* value until the Check Cached Status method executes. • **Force Status Read**—When the value of this input is FALSE, the method returns the cached status information since the last time that the Check Cached Status method executed. When the value is *TRUE*, the method forces the FPGA I/O Node to read all channels and update the status information. Forcing a status read can introduce jitter into an analog input loop. • **Out of Range**—Returns an array of Boolean values. A value of *TRUE* in any index indicates that the channel sharing a number with that index exceeded the common-mode voltage range at some point after the last time that the Check Cached Status method executed. • **Open**—Returns an array of Boolean values. A value of *TRUE* in any index indicates that the channel sharing a number with that index detected an open thermocouple on the channel at some point after the last time that the Check Cached Status method executed.

I/O Properties

This device does not support any I/O properties.

Module Properties

To access the following module properties for this device, use the FPGA I/O Property Node.

Table 145. Module Properties

Property	Description
Conversion Time	Sets the time it takes to acquire one point of data from one or more channels in a single FPGA I/O Node. Select High Speed or High Resolution. Refer to the *NI 9212 with NI TB-9212 Specifications* for more information about the conversion times. This property overwrites the value configured in the C Series Module Properties dialog box. Returns the conversion time setting when configured to read.
Module ID	Returns the module ID, 0x7449.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Converting NI 9213 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9213 if you want the FPGA I/O Node to return calibrated, fixed-point data from the module in units of volts. If you set the **Calibration Mode** to **Calibrated**, you must convert the CJC data from voltage to temperature.

Set the **Calibration Mode** to **Raw** if you want the FPGA I/O Node to return calibrated, binary data from the module. If you set the **Calibration Mode** to **Raw**, you must convert the binary thermocouple and CJC values to voltage and then convert the CJC data from voltage to temperature. You must convert these values in the host VI.

Note The equations below apply only to the NI 9213. If you are using the NI 9213E board-only module, refer to the KnowledgeBase for the correct equations.

Using a VI to Convert Data to Temperature

Refer to the NI 9213 Convert to Temperature polymorphic VI in the `labview\examples\CompactRIO\Module Specific\NI 9213\NI 9213 Getting Started\NI 9213 Getting Started.lvproj` for an example of converting calibrated or raw data to temperature. Use the `NI 9213 Convert to Temperature` VI as a subVI in the host VI.

Using an Equation to Convert Binary Values to Voltage

Use the following equation in the host VI to convert the binary thermocouple and CJC values to voltage:

$$Voltage = Binary\ Value \times 78.125\ mV \div 8{,}388{,}607$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.

Using Equations to Convert CJC Data from Voltage to Temperature

Use the following equations in the host VI to convert CJC data from volts to temperature:

Calculate the resistance of the thermistor:

$$R_T = (10000 \times CJC\ Data \times 32) \div (2.5 - CJC\ Data \times 32)$$

Calculate the CJC temperature:

$$T = [\ 1 \div [A + B(\ln(R_T)) + C(\ln(R_T))^3]] - (273.15 + OffsetConstant)$$

Where

- T = temperature in °C
- $A = 1.2873851 \times 10^{-3}$

- $B = 2.3575235 \times 10^{-4}$
- $C = 9.4978060 \times 10^{-8}$
- R_T = thermistor resistance reading
- $OffsetConstant$[10] = 1

Refer to the National Institute of Standards and Technology (NIST) Monograph 175 thermocouple reference tables for more information about converting and adjusting thermocouple values.

NI 9213 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in units as specified on the C Series Module Properties dialog box.

Note The Scan Interface applies Autozero and cold-junction compensation to NI 9213 data returned in temperature units.

Module Channels

The NI 9213 has the following channels.

Table 146. Module Channels

Channel	Description
TC*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9213, *x* is 0 to 15.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. To display this dialog box, right-click the module in the **Project Explorer** window and select **Properties**. You can configure the following options.

10. The *OffsetConstant* is the typical temperature gradient between the CJC sensor and the thermocouple cold junction.

- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.
- **Conversion Time**—Determines the time it takes to acquire one point of data from one or more channels. Select High Speed or High Resolution. Refer to the NI 9213 hardware documentation for more information about the conversion times.
- **Channels**—Specifies the channel(s) for which you want to configure settings.
- **Thermocouple Type**—Specifies the type of thermocouple connected to the channel.
- **Measurement Units**—Specifies the units you want data to be returned in for the channel. You can select Raw Volts, Degrees Kelvin, Degrees Celsius, Degrees Fahrenheit, or Degrees Rankine.
- **Enable Open Thermocouple Detection**—If this box is checked, the NI 9213 returns a zero value and reports an error if it detects an open thermocouple on any channel. This box is checked by default. If you do not check this box, each channel of the NI 9213 independently returns full-scale data and the NI 9213 does not report an error if an open thermocouple is detected.

> **Note** Enabling or disabling open thermocouple detection on the NI 9213 has no effect on the accuracy of the measurement because this option on the module is implemented entirely from software.

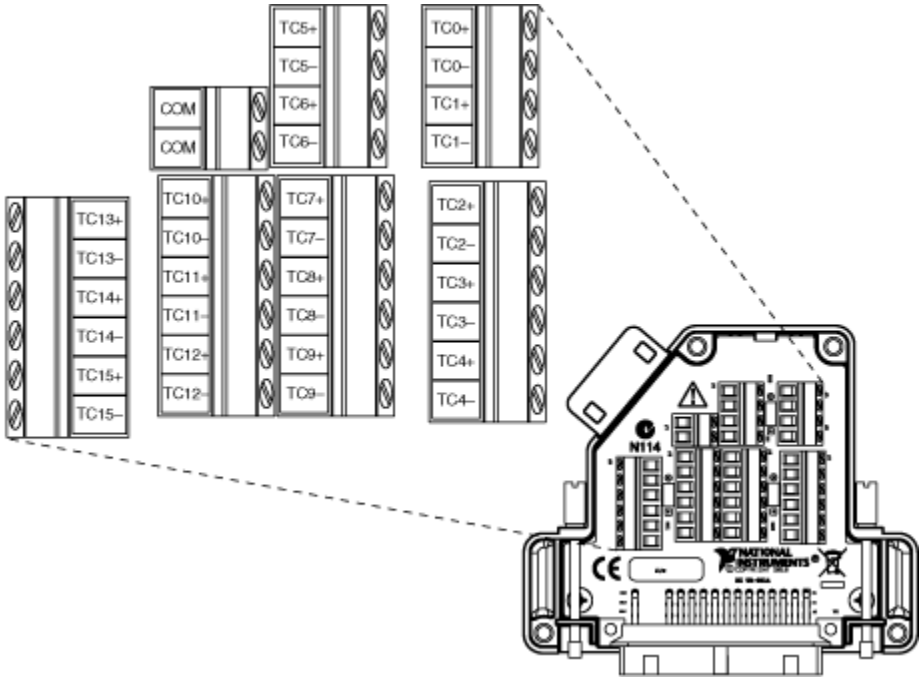
NI 9214

CompactRIO 16-Channel, ±78 mV, 24-Bit Thermocouple Input Module

NI TB-9214 Pinout

The NI TB-9214 provides connections for the 16 thermocouple channels of the NI 9214.

Figure 38. NI TB-9214 Pinout



NI 9214 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 147. Terminals in Software

Terminal	Description
TC*x*	Thermocouple input channel *x*, where *x* is the number of the channel. The NI 9214 has TC channels 0 to 15.
Autozero	Autozero channel. For the best accuracy, read the Autozero channel in the same FPGA I/O Node as the thermocouple input channels.
CJC	Cold-junction compensation channel *x*, where *x* is the number of the channel. The NI 9214 has CJC channels 0 to 2. For the best accuracy, read the appropriate CJC channel in the same FPGA I/O Node as the corresponding thermocouple input channels. Convert the CJC data to temperature. • **CJC 0**—corresponds to thermocouple input channels TC2, TC3, TC4, TC7, TC8, and

Terminal	Description
	TC9. • **CJC 1**—corresponds to thermocouple input channels TC10, TC11, TC12, TC13, TC14, and TC15. • **CJC 2**—corresponds to thermocouple input channels TC0, TC1, TC5, and TC6.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Module Methods

To access the following method for this device, use the FPGA I/O Method Node.

Table 148. Module Methods

Method	Description
Check Cached Status	Returns a boolean value for each channel that indicates whether the channel was out of range or had an open thermocouple since the last execution of the Check Cached Status method. When the FPGA I/O Node reads the channels, the FPGA VI determines the state of the channels and caches any *TRUE* value until the Check Cached Status method executes. • **Force Status Read**—When the value of this input is FALSE, the method returns the cached status information since the last time that the Check Cached Status method executed. When the value is *TRUE*, the method forces the FPGA I/O Node to read all channels and update the status information. Forcing a status read can introduce jitter into an analog input loop. • **Out of Range**—Returns an array of Boolean values. A value of *TRUE* in any index indicates that the channel sharing a number with that index exceeded the common-mode voltage range at some point after the last time that the Check Cached Status method executed. • **Open**—Returns an array of Boolean values. A value of *TRUE* in any index indicates that the channel sharing a number with that index detected an open thermocouple on the channel at some point after the last time that the Check Cached Status method executed.

I/O Properties

This device does not support any I/O properties.

Module Properties

To access the following module properties for this device, use the FPGA I/O Property Node.

Table 149. Module Properties

Property	Description
Conversion Time	Sets the time it takes to acquire one point of data from one or more channels in a single FPGA I/O Node. Select High Speed or High Resolution. Refer to the *NI 9214 Specifications* for more information about the conversion times. This property overwrites the value configured in the C Series Module Properties dialog box. Returns the conversion time setting when configured to read.
Open TC Detection Enable	Enables open thermocouple detection on the module. When this property is enabled, the NI 9214 returns an error if it detects an open thermocouple on any TC*x* channel. This property overwrites the value you configure in the *C Series Module Properties* dialog box. **Note** Enabling open thermocouple detection on the NI 9214 results in a small current leakage in the thermocouple measurement. Refer to the *Overview of Open Thermocouple Detection* support document for more information about compensating for this error. Returns open thermocouple detection setting when configured to read.
Module ID	Returns the module ID, 0x74EE.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related information:

- NI-9214 Specifications
- Overview of Open Thermocouple Detection

Converting NI 9214 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the *C Series Module Properties* dialog box for the *NI 9214* if you want the FPGA I/O Node to return calibrated, fixed-point data from the module in units of volts. If you set the **Calibration Mode** to **Calibrated**, you must convert the CJC data from voltage to temperature.

Set the **Calibration Mode** to **Raw** if you want the FPGA I/O Node to return calibrated, binary data from the module. If you set the **Calibration Mode** to **Raw**, you must convert the binary thermocouple and CJC values to voltage and then convert the CJC data from voltage to temperature. You must convert these values in the host VI.

Using a VI to Convert Data to Temperature

Refer to the NI 9214 Convert to Temperature polymorphic VI in the `labview\examples\CompactRIO\Module Specific\NI 9214\NI 9214 Getting Started\NI 9214 Getting Started.lvproj` for an example of converting calibrated or raw data to temperature. Use the NI 9214 Convert to Temperature VI as a subVI in the host VI.

Use the following equation in the host VI to convert the binary thermocouple and CJC values to voltage:

$$Voltage = Binary\ Value \times 78.125\ mV \div 8{,}388{,}607$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.

Converting CJC Data from Voltage to Temperature

The NI 9214 Convert Thermistor Reading VI is a subVI in the NI 9214 Convert to Temperature polymorphic VI that converts CJC data from voltage to temperature.

The VI uses the following equation to calculate the resistance of the thermistor:

$R_T = [(CJC\ Data \div 0.078125) \div (1 - (CJC\ Data \div 0.078125))] \times 20{,}000$

Using the resistance of the thermistor, the VI references a look-up table to interpolate the CJC temperature.

Refer to the National Institute of Standards and Technology (NIST) Monograph 175 thermocouple reference tables for more information about converting and adjusting thermocouple values.

NI 9214 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in units as specified on the C Series Module Properties dialog box.

Note The Scan Interface applies Autozero and cold-junction compensation to NI 9214 data returned in temperature units.

Module Channels

The NI 9214 has the following channels.

Table 150. Module Channels

Channel	Description
TC*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9214, *x* is 0 to 15.

Module-Specific Errors

The NI 9214 can return the following module-specific error.

Table 151. Module-Specific Errors

Error Code	Description
–65582	An open thermocouple was detected.
–65583	A common-mode range error was detected.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. To display this dialog box, right-click the module in the **Project Explorer** window and select **Properties**. You can configure the following options.

- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.
- **Conversion Time**—Determines the time it takes to acquire one point of data from one or more channels. Select High Speed or High Resolution. Refer to the NI 9214 hardware documentation for more information about the conversion times.
- **Channels**—Specifies the channel(s) for which you want to configure settings.
- **Thermocouple Type**—Specifies the type of thermocouple connected to the channel.
- **Measurement Units**—Specifies the units you want data to be returned in for the channel. You can select Raw Volts, Degrees Kelvin, Degrees Celsius, Degrees Fahrenheit, or Degrees Rankine.
- **Enable Open Thermocouple Detection**—If this box is checked, the NI 9214 returns a zero value and reports an error if it detects an open thermocouple on any channel. This box is checked by default. If you do not check this box, the Open Thermocouple Detection Circuit is disabled and the NI 9214 does not report an error or go to full-scale when an open thermocouple is detected.



Note Enabling open thermocouple detection on the NI 9214 results in a

> small current leakage in the thermocouple measurement. Refer to the *Overview of Open Thermocouple Detection* support document for more information about compensating for this error.

NI 9214 OTD Compensation Example VI

Refer to the NI 9214 OTD Compensation VI in the `labview\examples\CompactRIO\NI Scan Engine\Module Specific\Analog Input\NI 9214 - Compensation - Scan Mode\NI 9214 OTD Compensation - Scan Mode.lvproj` for an example that compensates for the error introduced by enabling open thermocouple detection on the NI 9214.

Related information:

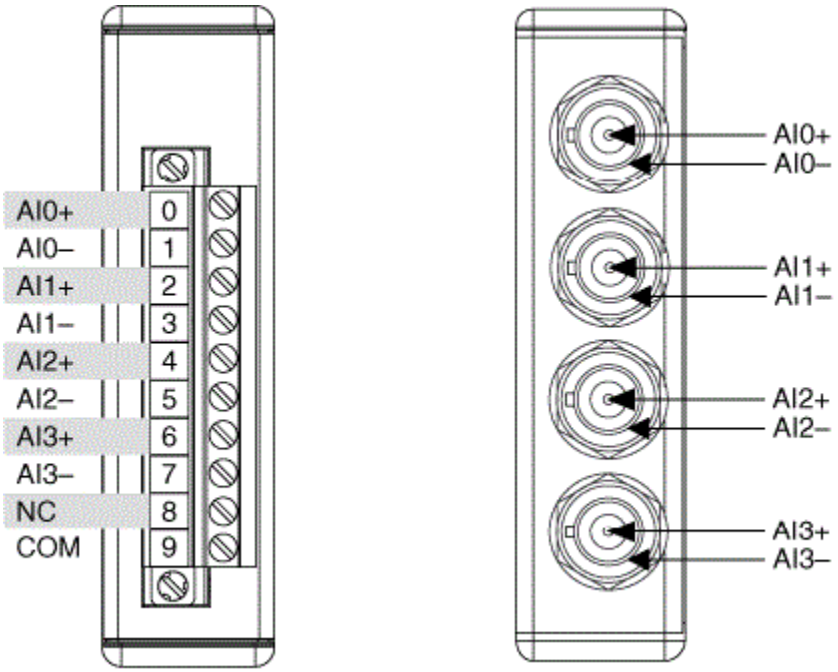
- Overview of Open Thermocouple Detection

NI 9215

CompactRIO 4-Channel, ±10 V, 16-Bit Simultaneous Analog Input Module

NI 9215 Pinout

Figure 39. NI 9215 Pinout



NI 9215 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Note You can simultaneously read from multiple channels on the NI 9215. Refer to *Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)*.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 152. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9215 has AI channels 0 to 3.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 153. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in nV/LSB for the channel. Use this value to convert and calibrate NI 9215 data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert and calibrate NI 9215 data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 154. Module Properties

Property	Description
Module ID	Returns the module ID, 0x7135 (NI 9215 with BNC) or 0x70A6 (NI 9215 with screw terminal).
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)

NI 9215 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9215 has the following channels.

Table 155. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9215, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9215. Right-click the NI 9215 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

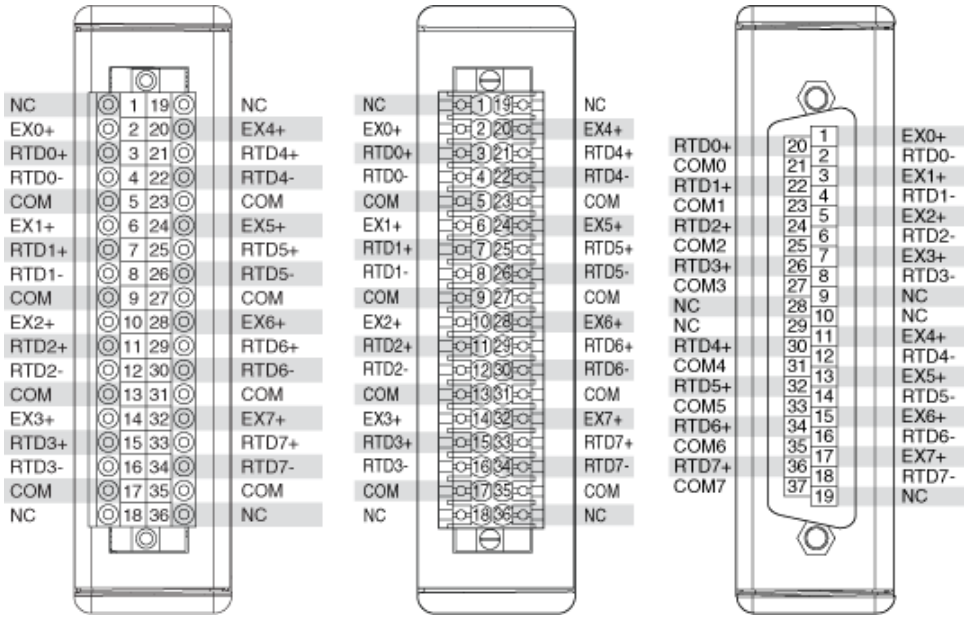
- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.

NI 9216

CompactRIO 8-Channel, 24-Bit, 100 Ω RTD Analog Input Module

NI 9216 Pinout

Figure 40. NI 9216 Pinout



NI 9216 (FPGA Interface)

CompactRIO 8-Channel, 24-Bit, 100 Ω RTD Analog Input Module

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 156. Terminals in Software

Terminal	Description
RTD*x*	RTD input channel *x*, where *x* is the number of the channel. The NI 9216 has RTD channels 0 to 7.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for

arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 157. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in pΩ/LSB for the channel. Use this value to convert and calibrate NI 9216 data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in µΩ for the channel. Use this value to convert and calibrate NI 9216 data if you set the **Calibration Mode** to **Raw** in the **C Series Module Properties** dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 158. Module Properties

Property	Description
Conversion Time	Sets the time in milliseconds it takes to acquire conversion data. This property overwrites the value you configure in the C Series Module Properties dialog box.
Module ID	Returns the module ID, 0x77C4 (NI 9216 with spring terminal) or 0x77DF (NI 9216 with DSUB).
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Converting Nominal Values to Temperature Values for the NI 9216 (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the *C Series Module Properties* dialog box for the *NI 9216* if you want the FPGA I/O Node to return calibrated, fixed-point data for the module in units of ohms. If you set the **Calibration Mode** to **Raw**, the FPGA I/O Node returns uncalibrated, binary values for the module. After you convert these binary values to nominal values, you can convert the nominal resistance values into temperature values. Convert these values in the host VI.

Using a VI to Convert Values

Refer to the `RTD to Temp` VI in the `labview\examples\CompactRIO\Module Specific\NI 9216\NI 9216 Getting Started\NI 9216 Getting Started.lvproj` for an example of converting nominal resistance values to temperature. Use the `RTD to Temp` VI as a subVI in the host VI to convert nominal resistance values to temperature.

Use a linearization curve known as the Callendar-Van Dusen equation in the host VI to measure the temperature of RTDs.

NI 9216 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in units as specified on the C Series Module Properties dialog box.

Module Channels

The NI 9216 has the following channels.

Table 159. Module Channels

Channel	Description
RTD*x*	RTD input channel *x*, where *x* is the number of the channel. The NI 9216 has RTD channels 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

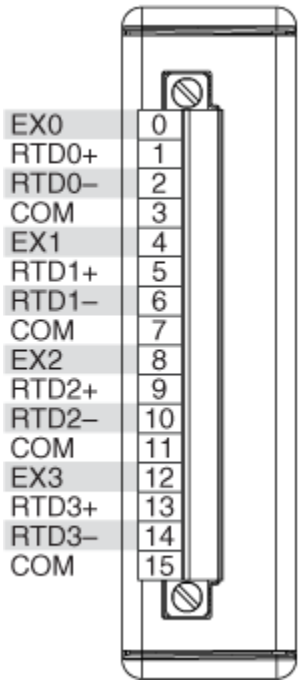
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- Conversion Time—Specifies the time it takes to acquire one point of data from all channels. Select 200 ms or 2.5 ms.
- Channels—Specifies the channel(s) for which you want to configure settings.
- Measurement Units—Specifies the units you want data to be returned in for the channel. You can select Raw Ohms, Degrees Kelvin, Degrees Celsius, Degrees Fahrenheit, or Degrees Rankine.
- RTD Type—Specifies the type of RTD connected to the channel by temperature coefficient of resistance (TCR). If you select Custom, enter the **Callendar-Van Dusen Coefficients** below.
  - **Callendar-Van Dusen Coefficients**—Specifies the coefficients to use in the Callendar-Van Dusen equation. the Callendar-Van Dusen equation is a linearization curve for measuring the temperature of an RTD.

NI 9217

CompactRIO 4-Channel, 24-Bit, 100 Ω RTD Analog Input Module

NI 9217 Pinout

Figure 41. NI 9217 Pinout



NI 9217 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 160. Terminals in Software

Terminal	Description
RTD*x*	RTD input channel *x*, where *x* is the number of the channel. The NI 9217 has RTD channels 0 to 3.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 161. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in pΩ/LSB for the channel. Use this value to convert and calibrate module data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in µΩ for the channel. Use this value to convert and calibrate module data if you set the **Calibration Mode** to **Raw** in the **C Series Module Properties** dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 162. Module Properties

Property	Description
Conversion Time	Sets the time in milliseconds it takes to acquire conversion data. This property overwrites the value you configure in the C Series Module Properties dialog box.
Module ID	Returns the module ID, 0x712B.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Converting Nominal Values to Temperature Values for the NI 9217 (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the *C Series Module Properties* dialog box for the *NI 9217* if you want the FPGA I/O Node to return calibrated, fixed-point data for the module in units of ohms. If you set the **Calibration Mode** to **Raw**, the FPGA I/O Node returns uncalibrated, binary values for the module. After you convert these binary values to nominal values, you can convert the nominal resistance values into temperature values. Convert these values in the host VI.

Using a VI to Convert Values

Refer to the RTD to Temp VI in the `labview\examples\CompactRIO\Module Specific\NI 9217\NI 9217 Getting Started\NI 9217 Getting Started.lvproj` for an example of converting nominal resistance values to temperature. Use the RTD to Temp VI as a subVI in the host VI to convert nominal resistance values to temperature.

Use a linearization curve known as the Callendar-Van Dusen equation in the host VI to measure the temperature of RTDs.

NI 9217 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in units as specified on the C Series Module Properties dialog box.

Module Channels

The NI 9217 has the following channels.

Table 163. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9217, *x* 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- Conversion Time—Specifies the time it takes to acquire one point of data from all channels. Select 200 ms or 2.5 ms.
- Channels—Specifies the channel(s) for which you want to configure settings.
- Measurement Units—Specifies the units you want data to be returned in for the channel. You can select Raw Ohms, Degrees Kelvin, Degrees Celsius, Degrees Fahrenheit, or Degrees Rankine.
- RTD Type—Specifies the type of RTD connected to the channel by temperature coefficient of resistance (TCR). If you select Custom, enter the **Callendar-Van Dusen Coefficients** below.
  - **Callendar-Van Dusen Coefficients**—Specifies the coefficients to use in the Callendar-Van Dusen equation. the Callendar-Van Dusen equation is a linearization curve for measuring the temperature of an RTD.

NI 9218

CompactRIO 2-Channel, 24-Bit Dynamic Universal Analog Input Module

NI 9218 Pinout

Figure 42. NI 9218 Pinout

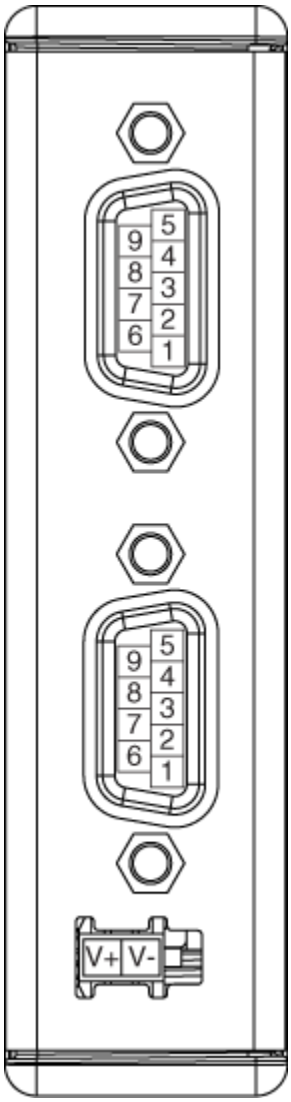


Table 164. NI 9218 Pin Assignments by Mode

Pin	Mode				
	Bridge	Voltage		IEPE	TEDS
	±22 mV/V	±65 mV	±16 V	±5 V	Class 1
1	EX+	EX+*	EX+*	—	—
2	—	—	—	AI+	TEDS+
3	EX-	EX-*	AI-, EX-*	AI-	TEDS-
4	RS+	—	—	—	—

Pin	Mode				
	Bridge	Voltage		IEPE	TEDS
	±22 mV/V	±65 mV	±16 V	±5 V	Class 1
5	RS-	—	—	—	—
6	AI+	AI+	AI+	—	—
7	AI-	AI-†	—	—	—
8	SC	—	—	—	—
9	SC	—	—	—	—

*Optional sensor excitation.

†In ±65 mV mode, you must tie pin 7 (AI-) to pin 3 (EX-).

NI 9218 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Note You can synchronize the NI 9218 with other modules that have a selectable timebase source.

Terminals in Software

Table 165. Terminals in Software

Terminal	Description
AI*x*	AI input channel *x*, where *x* is the number of the channel. The NI 9218 has RTD channels 0 to 1.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is 13.1072 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9218 to access this channel.
Start	Channel that controls when the NI 9218 starts acquiring data.

Terminal	Description
	To acquire data, write *TRUE* to the Start channel. When the NI 9218 is acquiring data, you must write *TRUE* to the Stop channel before you can access properties for the module. No operation is performed when writing *FALSE* to the Start channel.
Stop	Channel that controls when the NI 9218 stops acquiring data. To stop acquiring data, write *TRUE* to the Stop channel. When the NI 9218 is acquiring data, write *TRUE* to the Stop channel before you can access properties for the module. No operation is performed when writing *FALSE* to the Stop channel.

You can read TEDS information from the NI 9218.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Module Methods

To access the following method for this device, use the FPGA I/O Method Node.

Table 166. Module Methods

Method	Description
Check Cached Status	Returns Booleans for each channel that indicate whether there was an excitation fault or open loop on the channel since the last execution of the Check Cached Status method. When the FPGA I/O Node reads the channels, the FPGA VI determines the state of the channels and caches any *TRUE* value until the Check Cached Status method executes. • **Excitation Fault**—Returns an array of Boolean values. A value of *TRUE* in any index indicates that the channel sharing a number with that index

Method	Description
	detected an excitation fault on the channel at some point after the last time that the Check Cached Status method executed. • **Open Loop**—Returns an array of Boolean values. A value of *TRUE* in any index indicates that the channel sharing a number with that index detected an open loop on the channel at some point after the last time that the Check Cached Status method executed.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 167. I/O Properties

Property	Description
Input Configuration	Sets the input configuration of the corresponding channel to one of ten modes: ±60 V; ±16 V; ±16 V, 12 V Ex.; ±65 mV; ±65 mV, 12 V Ex.; ±20 mA; ±20 mA, 12 V Ex.; ±22 mV/V Bridge, 2 V Ex.; ±22 mV/V Bridge, 3.3 V Ex.; ±5 V IEPE AC Coupled.
LSB Weight (±16 V range)	Returns the LSB weight in pV/LSB for the ±16 V range.
LSB Weight (±20 mA range)	Returns the LSB weight in fV/LSB for the ±20 mA range.
LSB Weight (±22 mV/V range)	Returns the LSB weight in fV/LSB for the ±22 mV/V range.
LSB Weight (±5 V IEPE range)	Returns the LSB weight in pV/LSB for the ±5 V IEPE range.
LSB Weight (±65 mV range)	Returns the LSB weight in fV/LSB for the ±65 mV range.
LSB Weight (±60 V range)	Returns the LSB weight in pV/LSB for the ±60 V range.
Offset (±16 V range)	Returns the calibration offset in nV for the ±16 V range.
Offset (±20	Returns the calibration offset in nV for the ±20 mA range.

Property	Description
mA range)	
Offset (±22 mV/V range)	Returns the calibration offset in nV for the ±22 mV/V range.
Offset (±5 V IEPE range)	Returns the calibration offset in nV for the ±5 V IEPE range.
Offset (±65 mV range)	Returns the calibration offset in nV for the ±65 mV range.
Offset (±60 V range)	Returns the calibration offset in nV for the ±60 V range.
Offset Cal Enable	Enables offset calibration. This disconnects both signal input pins and internally connects a short to the ADC driver circuitry.
Shunt Cal Enable	Controls the shunt calibration switch for each channel.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 168. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI-9218 acquires data.
Module ID	Returns the module ID, 0x7757.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Excitation Protection

The NI 9218 excitation circuit is protected from overcurrent and overvoltage fault conditions. A fault condition also occurs when:

- The voltage supplied to the external power connector on the NI 9218 is below the minimum required voltage.

- The excitation pins on the NI 9218 are shorted together.
- The connected load exceeds the powered sensor output current limit.

The circuit is automatically disabled in the event of a fault condition. Whenever possible, a channel automatically recovers after the fault is removed.
Wire error terminals on the FPGA I/O Property Nodes to receive notification of overcurrent and overvoltage faults. LabVIEW returns *error 65654* if there is an overcurrent or overvoltage fault on at least one channel.

- If a warning occurs, only the channel(s) with the fault are affected and all other channels on the module continue to function properly without interruption.
- If an error occurs, the module is unable to recover from the fault condition and you must restart the module after the fault is removed.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

NI 9218 Data Rates

The frequency of a master timebase (f_M) controls the data rate (f_S) of the NI 9218. The NI 9218 includes an internal master timebase with a frequency of 13.1072 MHz, but the module can also accept an external master timebase or export its own master timebase.

Internal Master Timebase

The following equation provides the available data rates for the NI 9218 with the internal master timebase.

$$f_s = \frac{f_M \div 256}{n}$$

Where

- *n* is any integer from 1 to 31.

Refer to the following table for the available data rates with the internal master timebase.

Table 169. NI 9218 Data Rates (kS/s)

51.200	5.689	3.012	2.048
25.600	5.120	2.845	1.969
17.067	4.654	2.695	1.896
12.800	4.267	2.560	1.829
10.240	3.938	2.438	1.765
8.533	3.657	2.328	1.707
7.314	3.413	2.226	1.652
6.400	3.200	2.133	—

External Master Timebase

The NI 9218 has a different set of data rates when using an external master timebase with a frequency other than 13.1072 MHz. To synchronize the data rate of an NI 9218 with other modules that use master timebases to control sampling, all of the modules must share a single master timebase source.

Loop Timing for the NI 9218 (FPGA Interface)

The NI 9218 uses an internal master timebase. When creating a loop with an I/O Node that acquires data from the NI 9218, do not use the Loop Timer or Wait functions. If the loop execution time is slower than the data rate of the NI 9218, the FPGA I/O Node returns an overrun warning and continues to read from the module.

The overrun warning means that the data the FPGA I/O Node returns is valid, but the function missed one or more data points since the last time it read data from the NI 9218.

The function returns the overrun warning when all of the following conditions are true:

- The NI 9218 is in acquisition mode.
- An FPGA I/O Node that is acquiring data from the module executes at least once after you put the module in acquisition mode.
- The FPGA I/O Node did not read one or more data points since the previous time the function executed.

Avoiding Overrun Warnings with the NI 9218

Follow these guidelines when developing an FPGA VI to avoid overrun warnings.

- Ensure that the loop does not execute slower than the data rate of the NI 9218.
- When reading from the NI 9218 and an additional internally timed module in the same loop, use one FPGA I/O Node to read all module channels. You also must synchronize the NI 9218 and the additional internally timed module.
- You can read from an NI 9218 and a non-internally timed analog input module in the same loop if the other module can acquire data as fast or faster than the data rate of the NI 9218. If you use the same FPGA I/O Node to read data from the modules, the FPGA I/O Node does not return data for the other module until the NI 9218 acquires data.
- Use different loops for the NI 9218 and another analog input module that acquires data slower than the NI 9218.



Note If your application acquires multiple buffers of data from an internally timed module and the timing relationship between them is not important, you can ignore the overrun warning returned with the first point of each buffer.

Related concepts:

- Configuring the Data Rate for a Module (FPGA Interface)

NI 9218 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in units as specified on the C Series Module Properties dialog box.

Module Channels

The NI 9218 has the following channels.

Table 170. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9218, *x* 0 to 1.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- Channels—Specifies the channel(s) for which you want to configure settings.
- Input Configuration—Sets the input configuration of the corresponding channel to one of ten modes: ±60 V; ±16 V; ±16 V, 12 V Ex.; ±65 mV; ±65 mV, 12 V Ex.; ±20 mA; ±20 mA, 12 V Ex.; ±22 mV/V Bridge, 2 V Ex.; ±22 mV/V Bridge, 3.3 V Ex.; ±5 V IEPE AC Coupled.

NI 9219

CompactRIO 4-Channel, 24-Bit Universal Analog Input Module

NI 9219 Pinout

Figure 43. NI 9219 Pinout

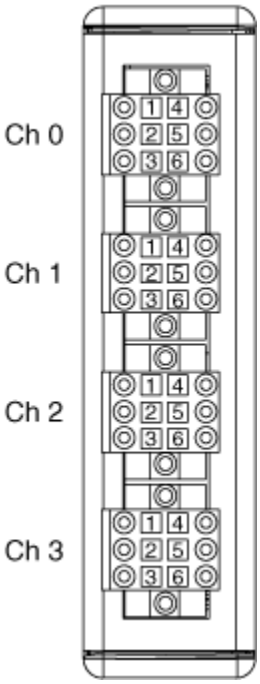


Table 171. Signal Descriptions

Terminal	Signal Name	Signal Description
1	T+	TEDS Data
2	T-	TEDS COM
3	EX+/HI[1]	Positive excitation or input signal
4	HI	Positive input signal
5	EX-/LO[1]	Negative excitation or input signal
6	LO	Negative input signal

[1] Depending on the mode, terminals 3 and 5 are either the excitation or input signals.

Table 172. Pin Assignments by Mode

Mode	Terminal					
	1	2	3	4	5	6
Voltage	T+	T-	—	HI	LO	—
Current	T+	T-	HI	—	LO	—

Mode	Terminal					
	1	2	3	4	5	6
4-Wire Resistance	T+	T-	EX+	HI	EX-	LO
2-Wire Resistance	T+	T-	HI	—	LO	—
Thermocouple	T+	T-	—	HI	LO	—
4-Wire RTD	T+	T-	EX+	HI	EX-	LO
3-Wire RTD	T+	T-	EX+	—	EX-	LO
Quarter-Bridge	T+	T-	HI	—	LO	—
Half-Bridge	T+	T-	EX+	HI	EX-	—
Full-Bridge	T+	T-	EX+	HI	EX-	LO
Digital In	T+	T-	—	HI	LO	—
Contact	T+	T-	HI	—	LO	—

NI 9219 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

Table 173. Terminals in Software

Terminal	Description
CH*x*	Channel *x*, where *x* is the number of the channel. The NI 9219 has channels 0 to 3.
CJC*x*	Cold-junction compensation channel *x*, where *x* is the number of the channel. The NI 9219 has CJC channels 0 to 3. If a channel is in Thermocouple mode, you can read CJC data for the channel. Convert the CJC data to temperature.

You can read TEDS information from the NI 9219.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for

arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 174. I/O Properties

Property	Description
Voltage Range	Sets the range of a channel in Voltage mode.
Resistance Range	Sets the range of a channel in Resistance mode.
RTD Range	Sets the range of a channel in RTD mode.
Quarter Bridge Range	Sets the range of a channel in Quarter Bridge mode.
Full Bridge Range	Sets the range of a channel in Full Bridge mode.
Digital Threshold	Sets the threshold of a channel in Digital In mode.

Module Properties

To access the following module properties for this device, use the FPGA I/O Property Node.

Table 175. Module Properties

Property	Description
Conversion Time	Sets the time it takes to acquire one point of data from one or more channels in a single FPGA I/O Node. Select High Speed, Best 60 Hz Rejection, Best 50 Hz Rejection, or High Resolution. Refer to the *NI 9219 Specifications* for more information about the conversion times. This property overwrites the value configured in the C Series Module Properties dialog box.

Property	Description
Module ID	Returns the module ID, 0x730C.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Excitation Protection

The NI 9219 excitation circuit is protected from overcurrent and overvoltage fault conditions. The circuit is automatically disabled in the event of a fault condition. Whenever possible, a channel automatically recovers after the fault is removed.

Wire error terminals on the FPGA I/O Property Nodes to receive notification of overcurrent and overvoltage faults. LabVIEW returns *error 65654* or *error 65548* if there is an overcurrent or overvoltage fault on at least one channel.

- If a warning occurs, only the channel(s) with the fault are affected and all other channels on the module continue to function properly without interruption.
- If an error occurs, the module is unable to recover from the fault condition and you must restart the module after the fault is removed.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related information:

- NI-9219 Specifications

Converting NI 9219 Data (FPGA Interface)

Set the **Calibration Mode** to Calibrated in the C Series Module Properties dialog box for the NI 9219 if you want the FPGA I/O Node to return calibrated, fixed-point data from the module. If you set the **Calibration Mode** to Calibrated and an NI 9219 channel is in Thermocouple mode, do the following:

- Convert the fixed-point CJC data to binary CJC data.

- Convert the binary CJC data to temperature.

Set the **Calibration Mode** to Raw if you want the FPGA I/O Node to return binary values from the module that are calibrated based on the *range* of the selected mode. If you set the **Calibration Mode** to Raw, you must convert the binary values to meaningful engineering units. If you set the **Calibration Mode** to Raw and a NI 9219 channel is in Thermocouple mode, convert the binary CJC data to temperature. Convert these values in the host VI.



Note The equations below apply only to the NI 9219. If you are using the NI 9219E board-only module, refer to the KnowledgeBase for the correct equations.

Using a VI to Convert and Adjust Binary Values

Refer to the NI 9219 Binary to Nominal VI in the `labview\examples\CompactRIO\Module Specific\NI 9219\NI 9219 Scaling Utility` directory for an example of converting and adjusting binary input values. Use the polymorphic NI 9219 Binary to Nominal VI as a subVI in the host VI to convert and adjust binary input values.

Using an Equation to Convert Binary Values

Use the following equation in the host VI to convert the binary input values to engineering units for each channel:

$$Engineering\ Units^{11} = Binary\ Value \times (Range\ High - Range\ Low) \div 2^{24}$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.
- *Range High* is the upper value of the input range [12]
- *Range Low* is the lower value of the input range.

11. Engineering units are equivalent to the units of the mode input range listed in the NI 9219 hardware documentation on ni.com/manuals.
12. Refer to the NI 9219 hardware documentation on ni.com/manuals for the input ranges for each mode. For modes that have only one input range value, use 0 as the lower value of the input range.

Digital In Mode returns a Boolean value.

- *FALSE* is equivalent to any voltage below the specified threshold.
- *TRUE* is equivalent to any voltage above the specified threshold.

Open Contact Mode returns a Boolean value.

- *FALSE* indicates a closed circuit.
- *TRUE* indicates a open circuit.

Using an Equation to Convert Fixed-Point CJC Data

You can use the following equation in the host VI to convert fixed-point CJC data to binary CJC data:

$$Binary\ CJC\ Data = (Fixed\text{-}Point\ CJC\ Data) \div (0.250 \div (2^{24} - 1))$$

Using Equations to Convert Binary CJC Data

Use the following equations in the host VI to convert binary CJC data to temperature:

Calculate the resistance of the thermistor:

$$R_T = 10000 \div [(2^{16} \div Binary\ CJC\ Data) - 1]$$

Calculate the CJC temperature:

$$T = [\ 1 \div [A + B(\ln(R_T)) + C(\ln(R_T))^3]] - (273.15 + OffsetConstant)$$

Where

- T = temperature in °C
- $A = 1.2873851 \times 10^{-3}$
- $B = 2.3575235 \times 10^{-4}$
- $C = 9.4978060 \times 10^{-8}$

- R_T = thermistor resistance reading
- *OffsetConstant*[13] = 0.1.5

Refer to the National Institute of Standards and Technology (NIST) Monograph 175 thermocouple reference tables for more information about converting and adjusting thermocouple values.

Configuring Modes and Ranges for the NI 9219 (FPGA Interface)

Configure the mode and range for each channel on the NI 9219 at edit time using the *C Series Module Properties* dialog box. Programmatically change the range for each channel at run time using the FPGA I/O Property Node. The execution of an I/O Property Node that is configured with a **Range** property overwrites the value you configured in the **C Series Module Properties** dialog box.

Configuring Modes and Ranges Using the C Series Module Properties Dialog Box

Complete the following steps to configure the channel(s) mode and range using the **C Series Module Properties** dialog box.

1. Configure the CompactRIO system, and add an NI 9219.
2. Right-click the NI 9219 in the **Project Explorer** window and select **Properties** from the shortcut menu to display the **C Series Module Properties** dialog box.
3. Select the channel(s) for which you want to configure the mode and range from the **Channels** table. You can select more than one channel by holding the <Ctrl> or <Shift> key when selecting channels.
4. Select the mode for the channel(s) from the **Channel Mode** pull-down menu. If you select **Digital In** mode, skip the following steps and *set the threshold*. If you select **Open Contact** mode, skip the following step.
5. Select the range from the **Range** pull-down menu for the selected channel(s).
6. Click the **OK** button.
7. Select **File»Save All** in the **Project Explorer** window.

13. The *OffsetConstant* is the typical temperature gradient between the CJC sensor and the thermocouple cold junction.

Configuring Ranges Using the FPGA I/O Property Node

Complete the following steps to configure the channel range using the FPGA I/O Property Node.

1. Create FPGA I/O items for the channel of the NI 9219 for which you want to configure a **Range** property. The channel must be set to Voltage mode, Resistance mode, RTD mode, Quarter Bridge mode, or Full Bridge mode.
2. Place an FPGA I/O Property Node on the block diagram and configure it for the NI 9219 channel for which you want to configure the range.
3. Click the **Property** section and select the **Range** property from the shortcut menu.
4. Right-click the **Range** input and select **Create»Control** from the shortcut menu.
5. On the front panel of the VI, select a rate from the **Range** pull-down menu.

You can change the channel range at run time by writing to the control from the host VI. Refer to the NI 9219 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9219\NI 9219 Getting Started\NI 9219 Getting Started.lvproj` for an example of changing a channel range using the FPGA I/O Property Node.

Related concepts:

- Configuring a Project
- Configuring the Digital In Threshold for the NI 9219 (FPGA Interface)

Configuring the Digital In Threshold for the NI 9219 (FPGA Interface)

Channels in Digital In mode on the NI 9219 have a 0–60 V unipolar threshold. The default value of the Digital In mode threshold is 1.5 V. Configure the threshold at edit time using the *C Series Module Properties* dialog box. Programmatically change the threshold at run time using the FPGA I/O Property Node. The execution of an I/O Property Node that is configured with a **Digital Threshold** property overwrites the value you configured in the **C Series Module Properties** dialog box.

Configuring the Digital In Threshold Using the C Series Module Properties Dialog Box

Complete the following steps to configure the Digital In threshold for the NI 9219 using

the **C Series Module Properties** dialog box.

1. *Configure* the CompactRIO system, and add an NI 9219.
2. Right-click the NI 9219 in the **Project Explorer** window and select **Properties** from the shortcut menu to display the **C Series Module Properties** dialog box.
3. Select the channel(s) for which you want to configure the Digital In threshold from the **Channels** table. You can select more than one channel by holding the <Ctrl> or <Shift> key when selecting channels.
4. Select **Digital In** from the **Channel Mode** pull-down menu.
5. Enter a value between 0 and 60 in the **Threshold** text box for the selected channel(s).
6. Click the **OK** button.
7. Select **File»Save All** in the **Project Explorer** window.

Configuring the Digital In Threshold Using the FPGA I/O Property Node

Complete the following steps to configure the Digital In threshold using the FPGA I/O Property Node.

1. Create FPGA I/O items for the channel of the NI 9219 for which you want to configure the **Digital Threshold** property. The channel must be set to Digital In mode.
2. Place an FPGA I/O Property Node on the block diagram and configure it for the NI 9219 channel for which you want to configure the Digital In threshold.
3. Click the **Property** section and select **Digital Threshold** from the shortcut menu.
4. Right-click the **Digital Threshold** input and select **Create»Control** from the shortcut menu.
5. On the front panel of the VI, enter a binary value in the **Digital Threshold** control. You can use the following equation in the host VI to calculate the threshold value:

   Threshold (binary) = Threshold (volts) x 2^{24} ÷ 60

You can change the Digital In threshold at run time by writing to the control from the host VI. Refer to the NI 9219 Digital Threshold VI in the `labview\examples\CompactRIO\Module Specific\NI 9219\NI 9219 Digital Threshold\NI 9219 Digital Threshold.lvproj` for an example of configuring the NI 9219 Digital In threshold using the FPGA I/O Property Node.

Related concepts:

- Configuring a Project

NI 9219 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the Project Explorer window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data or Boolean data in units determined by the measurement mode you select on the C Series Module Properties dialog box. The following table shows the units for the different modes.

Table 176. Units per Mode

Mode	Data Units
Voltage	Volts
Current	Amps
Resistance	Ohms
Thermocouple	Volts or Temperature Units
RTD	Ohms or Temperature Units
Bridge	Volts/Volt
Digital In	Boolean
Open Contact	Boolean

Note The Scan Interface applies Autozero and cold-junction compensation to NI 9219 data on channels configured for thermocouple mode.

Module Channels

The NI 9219 has the following channels.

Table 177. Module Channels

Channel	Description
CHx	Analog input channel x, where x is the number of the channel. For the NI 9219, x is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9219. Right-click the NI 9219 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

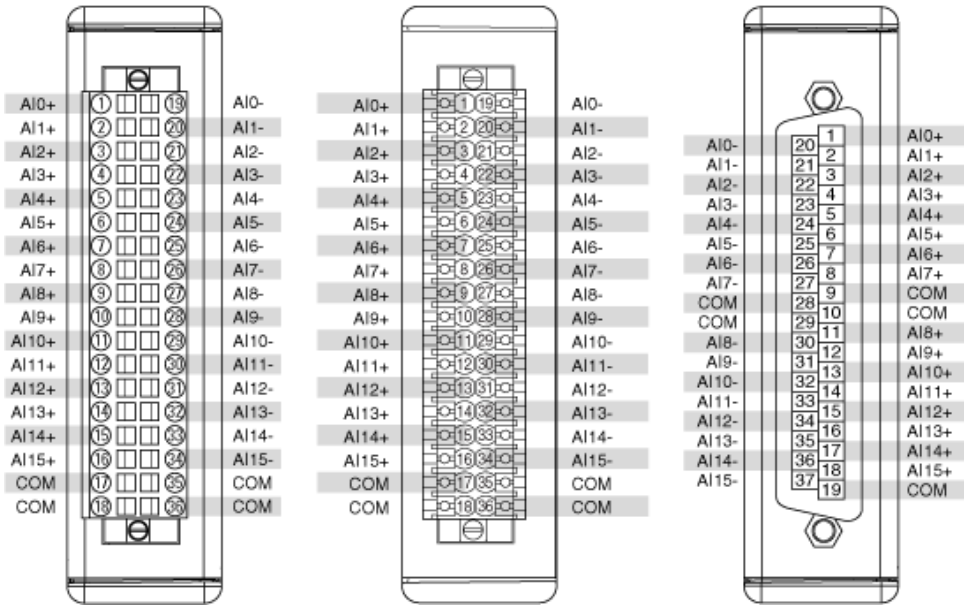
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.
- Channels—Specifies the channel(s) for which you want to configure settings.
- Mode—Specifies the measurement mode for the selected channel(s).
- Range—Specifies the range for the selected channel(s).
- Thermocouple Type—Specifies the type of thermocouple connected to a channel in thermocouple mode.
- Measurement Units—Specifies the units for data to be returned from the selected channel(s) configured for thermocouple or RTD mode.
- RTD Type—Specifies the type of RTD connected to the channel by temperature coefficient of resistance (TCR). If you select Custom, enter the **Callendar-Van Dusen Coefficients** below.
  - **Callendar-Van Dusen Coefficients**—Specifies the coefficients to use in the Callendar-Van Dusen equation, a linearization curve for measuring the temperature of an RTD.
- Threshold—Specifies the minimum high level in volts for the selected channel(s) configured for Digital In mode. The range of valid threshold values is 0 to 60.
- Conversion Time—Determines the time it takes to read all channels of the module. Select High Speed, Best 60 Hz Rejection, Best 50 Hz Rejection, or High Resolution. Refer to the NI 9219 hardware documentation on ni.com/manuals for more information about these conversion times.

NI 9220

CompactRIO 16-Channel, ±10 V, 16-Bit Simultaneous Analog Input Module

NI 9220 Pinout

Figure 44. NI 9220 Pinout



NI 9220 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

> **Note** You can simultaneously read from multiple channels on the NI 9220. Refer to *Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)*.

User-Controlled I/O Sampling

You can use the following User-Controlled I/O Sampling functions to perform I/O with more specific control over the I/O hardware on the FPGA.

Table 178. User-Controlled I/O Sampling Functions

Function	Type
Generate I/O Sample Pulse Method	Module function
Get I/O Read Status Method	I/O function
Read I/O Method	I/O function
Reset I/O Method	Module function

Note Call the Reset I/O Method function first to prepare the NI 9220 to use the other User-Controlled I/O Sampling functions.

Terminals in Software

Use the FPGA I/O Node or User-Controlled I/O Sampling functions to access the following terminals for this device.

Table 179. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9220 has AI channels 0 to 15.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 180. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in nV/LSB for the channel. Use this value to convert and calibrate NI 9220 data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert and calibrate NI 9220 data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 181. Module Properties

Property	Description
Module ID	Returns the module ID, 0x7757.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)

NI 9220 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9220 has the following channels.

Table 182. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9220 has AI channels 0 to 15.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

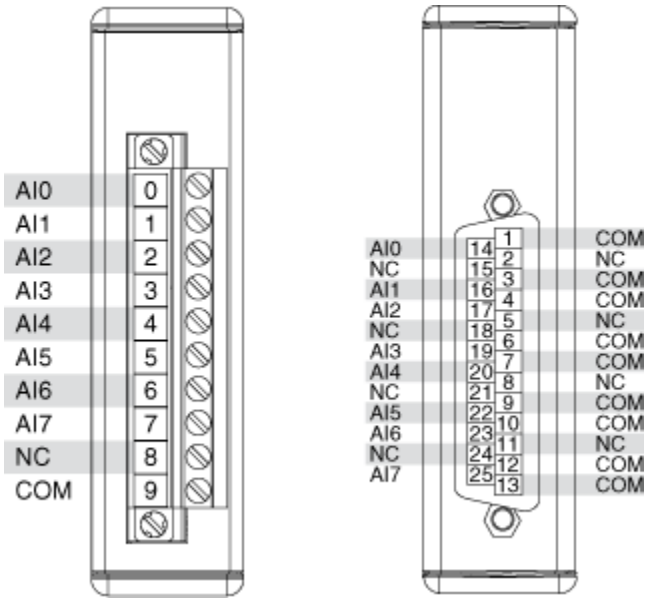
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9221

CompactRIO 8-Channel, ±60 V, 12-Bit Analog Input Module

NI 9221 Pinout

Figure 45. NI 9221 Pinout



NI 9221 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 183. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9221 has AI channels 0 to 7.

You can configure the minimum time between conversions.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 184. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in nV/LSB for the channel. Use this value to convert and calibrate module data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert and calibrate module data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 185. Module Properties

Property	Description
Module ID	Returns the module ID, 0x70A5.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Configuring the Minimum Time Between Conversions for the NI 9201/9221 (FPGA Interface)

You can configure the minimum time between conversions for the NI 9201/9221 in the C Series Module Properties dialog box.

Complete the following steps to configure the minimum time between conversions for the NI 9201/9221.

1. Configure the CompactRIO system, and add an NI 9201/9221.
2. Right-click the NI 9201/9221 in the **Project Explorer** window and select **Properties** from the shortcut menu to display the C Series Module Properties dialog box.
3. Enter a value between 0 and 10 µs in increments of 25 ns in the **Minimum Time Between Conversions** text box.
4. Click the **OK** button.
5. Select **File»Save All** in the **Project Explorer** window.

Note The default minimum time between conversions for the NI 9201 is 2 µs, and the default minimum time between conversions for the NI 9221 is 1.25 µs. The accuracy specifications in the NI 9201 and NI 9221 hardware documentation on ni.com/manuals are based on these default values. Refer to the following tables for examples of how the value you enter in the **Minimum Time Between Conversions** text box affects the actual time between conversions and the accuracy of the module.

The following table lists the minimum time between conversions when you are using a chassis *other than* the NI 9151 R Series Expansion chassis.

Table 186. Minimum Time Between Conversions When Using a Chassis Other Than the NI 9151 R Series Expansion Chassis

Module	Sampling Data from a Singleor Channel or Multiple Channels?	Minimum Time Between Conversions	Able to Achieve the Specified Time Between Conversions? (Refer to the following Note)	Module Accuracy
NI 9201	Single	≥ 1.25 µs	Yes	Module accuracy not affected

Module	Sampling Data from a Singleor Channel or Multiple Channels?	Minimum Time Between Conversions	Able to Achieve the Specified Time Between Conversions? (Refer to the following Note)	Module Accuracy
< 1.25 µs	No	Module accuracy not affected		
Multiple	≥ 2 µs	Yes	Module accuracy not affected	
< 2 µs	No	Module accuracy degrades		
NI 9221	Single	≥ 1.25 µs	Yes	Module accuracy not affected
< 1.25 µs	No	Module accuracy not affected		
Multiple	≥ 1.25 µs	Yes	Module accuracy not affected	
< 1.25 µs	No	Module accuracy degrades		

Set the minimum time between conversions in the C Series Module Properties dialog box.



Note

The actual time between conversions depends on how you develop the FPGA VI:

- *Yes* indicates that it is possible to write an FPGA VI that can sustain the minimum time between conversions specified in the C Series Module Properties dialog box.
- *No* indicates that the minimum time between conversions specified in the C Series Module Properties dialog box is too low and you may not be able to write an FPGA VI that can sustain the specified time.

The following table lists the minimum time between conversions when you are using

the NI 9151 R Series Expansion chassis.

Table 187. Minimum Time Between Conversions when using the NI 9151 R Series Expansion Chassis

Module	Sampling Data from a Single Channel or Multiple Channels?	Minimum Time Between Conversions	Able to Achieve the Specified Time Between Conversions? (Refer to the following Note)	Module Accuracy
NI 9201	Single	≥ 2.1 µs	Yes	Module accuracy not affected
< 2.1 µs	No	Module accuracy not affected		
Multiple	≥ 2.1 µs	Yes	Module accuracy not affected	
< 2.1 µs and ≥ 2 µs	No	Module accuracy not affected		
< 2 µs	No	Module accuracy degrades		
NI 9221	Single	≥ 2.1 µs	Yes	Module accuracy not affected
< 2.1 µs	No	Module accuracy not affected		
Multiple	≥ 2.1 µs	Yes	Module accuracy not affected	
< 2.1 µs and ≥ 1.25 µs	No	Module accuracy not affected		
< 1.25 µs	No	Module accuracy degrades		

Set the minimum time between conversions in the C Series Module Properties dialog box.

Note

The actual time between conversions depends on how you develop the FPGA VI:

- *Yes* indicates that it is possible to write an FPGA VI that can sustain the minimum time between conversions specified in the C Series Module Properties dialog box.
- *No* indicates that the minimum time between conversions specified in the C Series Module Properties dialog box is too low and you may not be able to write an FPGA VI that can sustain the specified time.

Understanding NI 9201/9221 Scanning (FPGA Interface)

To scan the channels of the NI 9201/9221, configure an FPGA I/O Node with the channels you want to acquire from the module.

The module implements a pipeline that is automatically managed by the FPGA I/O Node. Channels within the FPGA I/O Node are sampled in numerical order regardless of the order they appear in the node.

The first time an FPGA I/O Node configured with channels on an NI 9201/9221 module executes, the module performs two setup conversions before converting the first channel. The two setup conversions prime the pipeline for subsequent FPGA I/O Node reads. The module does not repeat the setup conversions unless the FPGA I/O Node channel configuration changes and the pipeline needs to be primed again.

NI 9221 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9221 has the following channels.

Table 188. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9221 has AI channels 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

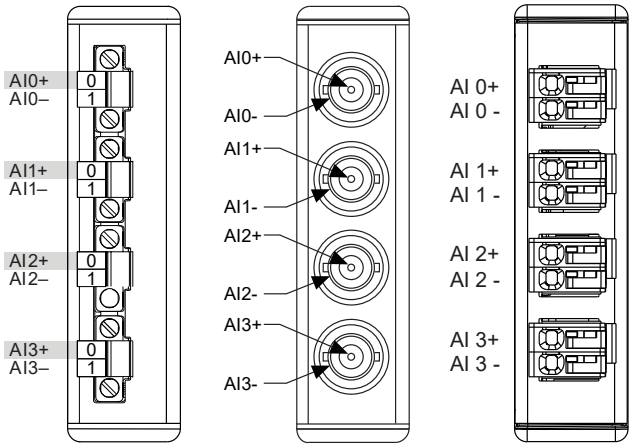
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9222

CompactRIO 4-Channel, ±10 V, 16-Bit Simultaneous, Channel-to-Channel Isolated Analog Input Module

NI 9222 Pinout

Figure 46. NI 9222 Pinout



NI 9222 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Note You can simultaneously read from multiple channels on the NI 9222. Refer to *Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)*.

User-Controlled I/O Sampling

You can use the following User-Controlled I/O Sampling functions to perform I/O with more specific control over the I/O hardware on the FPGA.

Table 189. User-Controlled I/O Sampling Functions

Function	Type
Generate I/O Sample Pulse method	Module function
Get I/O Read Status method	I/O function
Read I/O method	I/O function
Reset I/O method	Module function

Note Call the Reset I/O Method function first to prepare the NI 9222 to use the other User-Controlled I/O Sampling functions.

Terminals in Software

Use the FPGA I/O Node or User-Controlled I/O Sampling functions to access the following terminals for this device.

Table 190. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9222 has AI channels 0 to 3.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 191. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in nV/LSB for the channel. Use this value to convert and calibrate NI 9220 data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert and calibrate NI 9220 data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 192. Module Properties

Property	Description
Module ID	Returns the module ID, 0x7445.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Reading from C Series Channels
- Converting and Calibrating CompactRIO Analog Input Values (FPGA Interface)

NI 9222 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9222 has the following channels.

Table 193. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9222 has AI channels 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

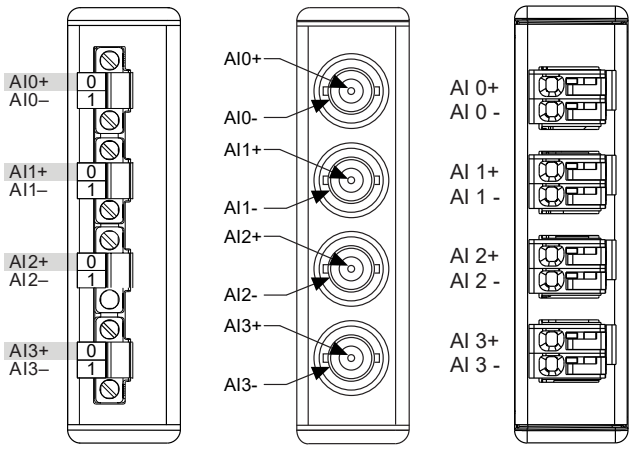
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9223

CompactRIO 4-Channel, ±10 V, 16-Bit Simultaneous, Channel-to-Channel Isolated Analog Input Module

NI 9223 Pinout

Figure 47. NI 9223 Pinout



NI 9223 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Note You can simultaneously read from multiple channels on the NI 9222. Refer to *Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)*.

User-Controlled I/O Sampling

You can use the following User-Controlled I/O Sampling functions to perform I/O with more specific control over the I/O hardware on the FPGA.

Table 194. User-Controlled I/O Sampling Functions

Function	Type
Generate I/O Sample Pulse method	Module function

Function	Type
Get I/O Read Status method	I/O function
Read I/O method	I/O function
Reset I/O method	Module function

Note Call the Reset I/O Method function first to prepare the NI 9223 to use the other User-Controlled I/O Sampling functions.

Terminals in Software

Use the FPGA I/O Node or User-Controlled I/O Sampling functions to access the following terminals for this device.

Table 195. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9223 has AI channels 0 to 3.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 196. I/O Properties

Property	Description
LSB	Returns the LSB weight in nV/LSB for the channel. Use this value to convert and calibrate

Property	Description
Weight	NI 9220 data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert and calibrate NI 9220 data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 197. Module Properties

Property	Description
Module ID	Returns the module ID, 0x74EC.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)

NI 9223 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9223 has the following channels.

Table 198. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9223 has AI channels 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

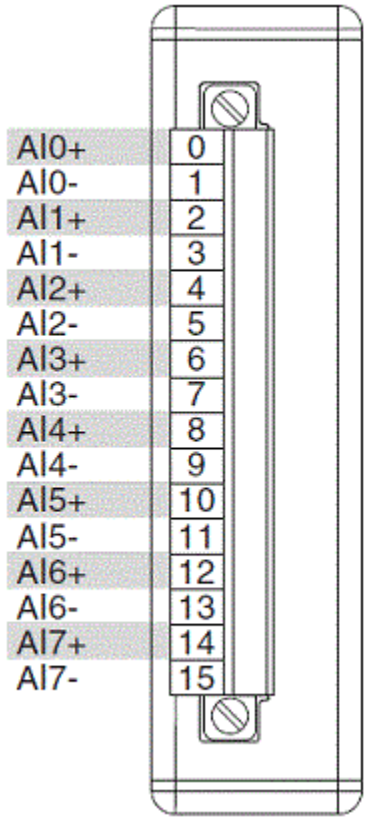
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9224

CompactRIO 8-Channel, ±10V 24-Bit Simultaneous Analog Input Module

NI 9224 Pinout

Figure 48. NI 9224 Pinout



NI 9224 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 199. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9224 has AI channels 0 to 7.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 200. Module Properties

Property	Description
Conversion Time	Sets the time it takes to acquire one point of data from one or more channels in a single FPGA I/O Node. Select High Speed, Best 60 Hz Rejection, Best 50 Hz Rejection, or High Resolution. This property overwrites the value configured in the C Series Module Properties dialog box. Returns the conversion time setting when configured to read.
Module ID	Returns the module ID, 0x780D.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Converting NI 9224 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9224 if you want the FPGA I/O Node to return calibrated, fixed-point data from the module in units of volts.

Set the **Calibration Mode** to Raw if you want the FPGA I/O Node to return calibrated, binary data from the module. If you set the **Calibration Mode** to **Raw**, convert the binary values to engineering units. Convert these values in the host VI.

Using an Equation to Convert Binary Values to Voltage

Use the following equation in the host VI to convert binary voltage values to volts:

$$Voltage = Binary\ Value \times 1257533\ pV/LSB$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.

NI 9224 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in units as specified on the C Series Module Properties dialog box.

Module Channels

The NI 9224 has the following channels.

Table 201. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9224 has AI channels 0 to 7.

C Series Module Properties Dialog Box

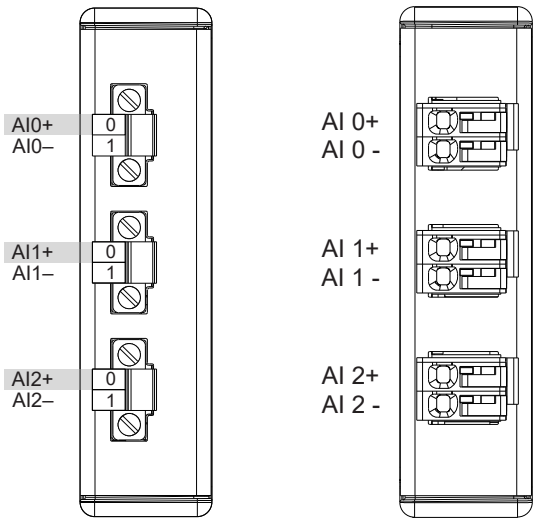
Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- Conversion Time—Determines the time it takes to acquire one point of data from one or more channels. Select Medium Speed, High Speed, Medium Resolution, or High Resolution. Refer to the NI 9224 hardware documentation for more information about the conversion times.

NI 9225

CompactRIO 3-Channel, 300 V_{rms}, 24-Bit Simultaneous Analog Input Module

NI 9225 Pinout



NI 9225 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Note You can simultaneously read from multiple channels on the NI 9225. Refer to *Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)*.

Terminals in Software

Table 202. Terminals in Software

Terminal	Description
AI*x*	AI input channel *x*, where *x* is the number of the channel. The NI 9225 has AI channels 0 to 2. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is12.8 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9225 to access this channel.
Start	Channel that controls when the module starts acquiring data. To acquire data, write *TRUE* to the Start channel.

Terminal	Description
	When the module is acquiring data, you must write *TRUE* to the Stop channel before you can access properties for the module. No operation is performed when writing *FALSE* to the Start channel.
Stop	Channel that controls when the module stops acquiring data. To stop acquiring data, write *TRUE* to the Stop channel. When the module is acquiring data, write *TRUE* to the Stop channel before you can access properties for the module. No operation is performed when writing *FALSE* to the Stop channel.

Note Refer to the NI 9225 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9225\NI 9225 Getting Started\NI 9225 Getting Started.lvproj` for an example of using the Start and Stop channels.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 203. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in pV/LSB for the channel. Use this value to convert and calibrate module data if you set the Calibration Mode to Raw in the C Series Module Properties

Property	Description
	dialog box.
Offset	Returns the calibration offset in µV for the channel. Use this value to convert and calibrate module data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 204. Module Properties

Property	Description
Data Rate	Sets the rate at which the module acquires data.
Module ID	Returns the module ID, 0x7383.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

Use the Onboard Clock channel in the single-cycle Timed Loop. You cannot use the other channels on the NI 9225 with the single-cycle Timed Loop. Refer to *Understanding Loop Timing (FPGA Interface)* for information about loop timing for this module.

Related concepts:

- Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)
- Understanding Loop Timing (FPGA Interface)

NI 9225 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9225 has the following channels.

Table 205. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9225, *x* 0 to 2.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

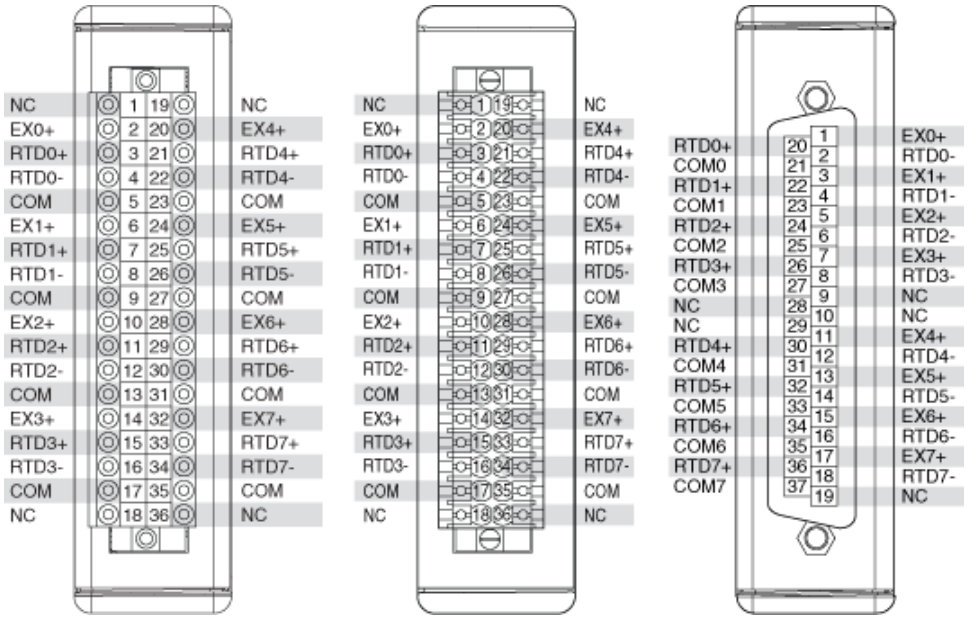
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9226

CompactRIO 8-Channel, 24-Bit, 1000 Ω RTD Analog Input Module

NI 9226 Pinout

Figure 49. NI 9226 Pinout



NI 9226 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 206. Terminals in Software

Terminal	Description
RTD*x*	RTD input channel *x*, where *x* is the number of the channel. The NI 9226 has RTD channels 0 to 7.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 207. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in pΩ/LSB for the channel. Use this value to convert and calibrate module data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in µΩ for the channel. Use this value to convert and calibrate module data if you set the **Calibration Mode** to **Raw** in the **C Series Module Properties** dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 208. Module Properties

Property	Description
Conversion Time	Sets the time in milliseconds it takes to acquire conversion data. This property overwrites the value you configure in the C Series Module Properties dialog box.
Module ID	Returns the module ID, 0x77C8 (NI 9226 with spring terminal) or 0x77E0 (NI 9226 with DSUB).
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Reading from C Series Channels
- Converting and Calibrating CompactRIO Analog Input Values (FPGA Interface)

Converting Nominal Values to Temperature Values for the NI 9226 (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9226 if you want the FPGA I/O Node to return calibrated, fixed-point data from the module in units of ohms.

When **Calibration Mode** is set to **Raw**, the FPGA I/O Node returns uncalibrated, binary values for the module. After converting these binary values to nominal values, convert the nominal resistance values into temperature values. Convert these values in the host VI.

Using a VI to Convert Values

For an example of converting nominal resistance values to temperature, refer to the RTD to Temp VI in the `labview\examples\CompactRIO\Module Specific\NI 9226\NI 9226 Getting Started\NI 9226 Getting Started.lvproj`. Use the RTD to Temp VI as a subVI in the host VI to convert nominal resistance values to temperature.

Use a linearization curve known as the Callendar-Van Dusen equation in the host VI to measure the temperature of RTDs.

NI 9226 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in units as specified on the C Series Module Properties dialog box.

Module Channels

The NI 9226 has the following channels.

Table 209. Module Channels

Channel	Description
RTD*x*	RTD input channel *x*, where *x* is the number of the channel. The NI 9226 has RTD channels 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

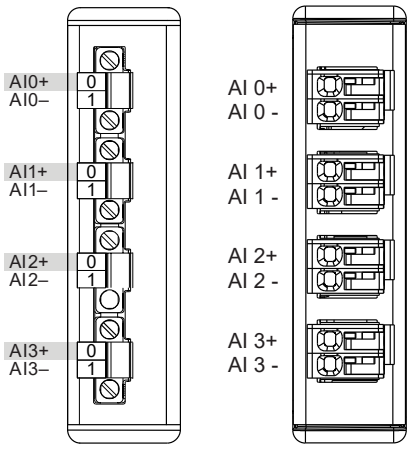
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- Conversion Time—Specifies the time it takes to acquire one point of data from all channels. Select 200 ms or 2.5 ms.
- Channels—Specifies the channel(s) for which you want to configure settings.
- Measurement Units—Specifies the units you want data to be returned in for the channel. You can select Raw Ohms, Degrees Kelvin, Degrees Celsius, Degrees Fahrenheit, or Degrees Rankine.
- RTD Type—Specifies the type of RTD connected to the channel by temperature coefficient of resistance (TCR). If you select Custom, enter the **Callendar-Van Dusen Coefficients** below.
  - **Callendar-Van Dusen Coefficients**—Specifies the coefficients to use in the Callendar-Van Dusen equation. the Callendar-Van Dusen equation is a linearization curve for measuring the temperature of an RTD.

NI 9227

CompactRIO 4-Channel, 5 A_{rms}, 24-Bit Simultaneous Analog Input Module

NI 9227 Pinout

Figure 50. NI 9227 Pinout



NI 9227 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Note You can simultaneously read from multiple channels on the NI 9227. Refer to *Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)*.

Terminals in Software

Table 210. Terminals in Software

Terminal	Description
AI*x*	AI input channel *x*, where *x* is the number of the channel. The NI 9227 has AI channels 0 to 3. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is12.8 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9227 to access this channel.
Start	Channel that controls when the module starts acquiring data. To acquire data, write *TRUE* to the Start channel.

Terminal	Description
	When the module is acquiring data, write *TRUE* to the Stop channel before you can access properties for the module. No operation is performed when writing *FALSE* to the Start channel.
Stop	Channel that controls when the module stops acquiring data. To stop acquiring data, write *TRUE* to the Stop channel. When the module is acquiring data, write *TRUE* to the Stop channel before you can access properties for the module. No operation is performed when writing *FALSE* to the Stop channel.



Note Refer to the NI 9227 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9227\NI 9227 Getting Started\NI 9227 Getting Started.lvproj` for an example of using the Start and Stop channels.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 211. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in pA/LSB for the channel. Use this value to convert and calibrate module data if you set the Calibration Mode to Raw in the C Series Module Properties

Property	Description
	dialog box.
Offset	Returns the calibration offset in nA for the channel. Use this value to convert and calibrate module data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 212. Module Properties

Property	Description
Data Rate	Sets the rate at which the module acquires data.
Module ID	Returns the module ID, 0x7416.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

Use the Onboard Clock channel in the single-cycle Timed Loop. You cannot use the other channels on the NI 9227 with the single-cycle Timed Loop. Refer to *Understanding Loop Timing (FPGA Interface)* for information about loop timing for this module.

Related concepts:

- Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)

NI 9227 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project**

Explorer window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in amps.

Module Channels

The NI 9227 has the following channels.

Table 213. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9227, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

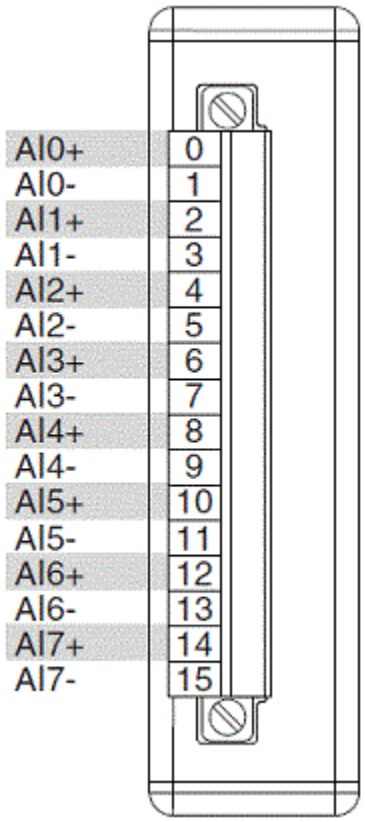
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9228

CompactRIO 8-Channel, ±60V 24-Bit Simultaneous Analog Input Module

NI 9228 Pinout

Figure 51. NI 9228 Pinout



NI 9228 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 214. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9228 has AI channels 0 to 7.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 215. Module Properties

Property	Description
Conversion Time	Sets the time it takes to acquire one point of data from one or more channels in a single FPGA I/O Node. Select High Speed (1k S/s), Medium Speed (100 S/s), Medium Resolution (50/60Hz rejection – 12 S/s), or High Resolution (50/60Hz rejection – 2 S/s). This property overwrites the value configured in the C Series Module Properties dialog box. Returns the conversion time setting when configured to read.
Module ID	Returns the module ID, 0x780E.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Converting NI 9228 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9228 if you want the FPGA I/O Node to return calibrated, fixed-point data from the module in units of volts.

When **Calibration Mode** is set to **Raw**, the FPGA I/O Node returns calibrated, binary data from the module. Convert the binary values to engineering units. Convert these values in the host VI.

Using an Equation to Convert Binary Values to Voltage

Use the following equation in the host VI to convert binary voltage values to volts:

$$Voltage = Binary\ Value \times 7602677\ pV/LSB$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.

NI 9228 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in units as specified on the C Series Module Properties dialog box.

Module Channels

The NI 9228 has the following channels.

Table 216. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9228, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

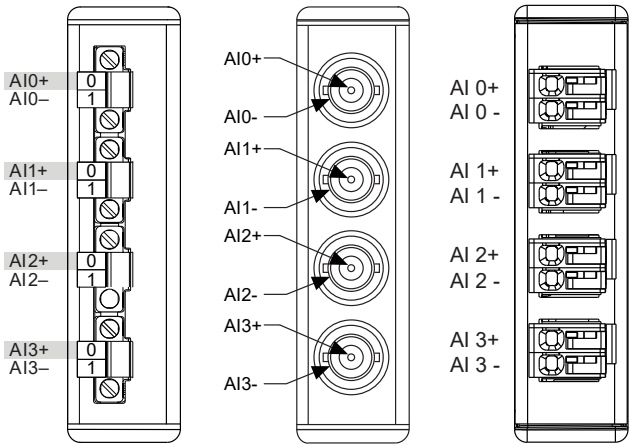
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. Use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- Conversion Time—Determines the time it takes to acquire one point of data from one or more channels. You can select Medium Speed, High Speed, Medium Resolution , or High Resolution. Refer to the NI 9228 hardware documentation for more information about the conversion times.

NI 9229

CompactRIO 4-Channel, ±60 V, 24-Bit Simultaneous Analog Input Module

NI 9229 Pinout

Figure 52. NI 9229 Pinout



NI 9229 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Note You can simultaneously read from multiple channels on the NI 9229. Refer to *Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)*. You also can synchronize the NI 9229 with other modules that have a selectable timebase source.

Terminals in Software

Table 217. Terminals in Software

Terminal	Description
AI*x*	AI input channel *x*, where *x* is the number of the channel. The NI 9229 has AI channels 0 to 3. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is12.8 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9229 to access this channel.
Start	Channel that controls when the module starts acquiring data. To acquire data, write *TRUE* to the Start channel. When the module is acquiring data, write *TRUE* to the Stop channel before you can access properties for the module. No operation is performed when writing *FALSE* to the Start channel.
Stop	Channel that controls when the module stops acquiring data. To stop acquiring data, write *TRUE* to the Stop channel. When the module is acquiring data, write *TRUE* to the Stop channel before you can access properties for the module. No operation is performed when writing *FALSE* to the Stop channel.

Note Refer to the NI 9229 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9229\NI 9229 Getting Started\NI 9229 Getting Started.lvproj` for an example of using the Start and Stop channels.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 218. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in pV/LSB for the channel. Use this value to convert and calibrate module data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert and calibrate module data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 219. Module Properties

Property	Description
Data Rate	Sets the rate at which the module acquires data.

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

Use the Onboard Clock channel in the single-cycle Timed Loop. You cannot use the other channels on the NI 9229 with the single-cycle Timed Loop. Refer to *Understanding Loop Timing (FPGA Interface)* for information about loop timing for this module.

Related concepts:

- Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)
- Understanding Loop Timing (FPGA Interface)

NI 9229 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9229 has the following channels.

Table 220. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9229, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

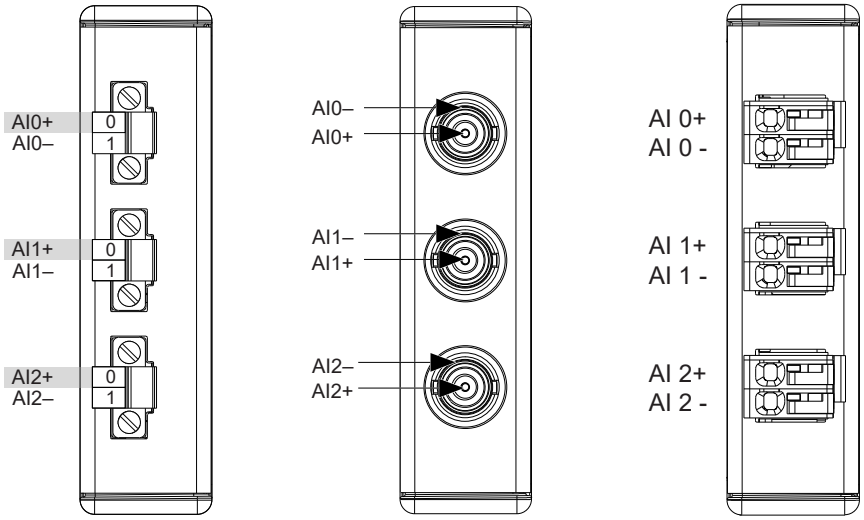
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9230

CompactRIO 3-Channel, ±30 V, 12.8 kS/s, 24-Bit Software Selectable IEPE and AC/DC Analog Input Module

NI 9230 Pinout

Figure 53. NI 9230 Pinout



NI 9230 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Note You can synchronize an NI 9230 with other modules that have a selectable timebase source.

Terminals in Software

Table 221. Terminals in Software

Terminal	Description
AI*x*	AI input channel *x*, where *x* is the number of the channel. The NI 9230 has AI channels 0 to 2. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is 13.1072 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9230 to access this channel. If you do not export the onboard clock and try to access the Onboard Clock channel, LabVIEW returns a code generation error when you try to compile the FPGA VI.
Start	Channel that controls when the module starts acquiring data. To acquire data, write *TRUE* to the Start channel. When the module is acquiring data, write *TRUE* to the Stop channel before you can access properties or TEDS information for the module. No operation is performed when writing *FALSE* to the Start channel.
Stop	Channel that controls when the module stops acquiring data. To stop acquiring data, write *TRUE* to the Stop channel. When the module is acquiring data, write *TRUE* to the Stop channel before you can access properties for the module. No operation is performed when writing *FALSE* to the Stop channel.

Note Refer to the NI 9230 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9230\NI 9230 Getting Started\NI 9230 Getting Started.lvproj` for an example of using the Start and Stop channels.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

To access the following method for this device, use the FPGA I/O Method Node.

Table 222. Module Methods

Method	Description
Check Cached Status	Returns Booleans for each channel that indicate whether there was a short circuit or open loop on the channel since the last execution of the Check Cached Status method. When the FPGA I/O Node reads the channels, the FPGA VI determines the state of the channels and caches any *TRUE* value until the Check Cached Status method executes. • **Short Circuit**—Returns an array of Boolean values. A value of *TRUE* in any index indicates that the channel sharing a number with that index detected a short circuit on the channel at some point after the last time that the Check Cached Status method executed. • **Open Loop**—Returns an array of Boolean values. A value of *TRUE* in any index indicates that the channel sharing a number with that index detected an open loop on the channel at some point after the last time that the Check Cached Status method executed.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 223. I/O Properties

Property	Description
Input Configuration	Sets the input configuration of the corresponding channel to one of three modes: AC coupled, DC coupled, or IEPE AC coupled.
LSB Weight	Returns the LSB weight in pV/LSB for the channel. Use this value to convert and calibrate module data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.

Property	Description
Offset	Returns the calibration offset in nV for the channel. Use this value to convert and calibrate module data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 224. Module Properties

Property	Description
Data Rate	Sets the rate at which the module acquires data.
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Configuring the NI 9230/9232/9234 Channel Input Modes (FPGA Interface)

You can configure the input mode for each channel of the NI 9230, NI 9232, or NI 9234 at edit time using the C Series Module Properties dialog box.

Programmatically change the input mode for each channel at run time using the FPGA I/O Property Node. The execution of an FPGA I/O Property Node that is configured with an **Input Configuration** property overwrites the value you configured in the C Series Module Properties dialog box for the corresponding channel.

Configuring the Channel Input Modes Using the C Series Module Properties Dialog Box

Complete the following steps to configure the channel input modes using the C Series

Module Properties dialog box.

1. Configure the CompactRIO system, and add an module.
2. Right-click the module in the **Project Explorer** window and select **Properties** to display the **C Series Module Properties** dialog box.
3. Select the channel(s) for which you want to configure the input mode from the **Channels** table. You can select more than one channel by holding the <Ctrl> or <Shift> key when selecting channels.
4. Select the input mode for the selected channel(s) from the **Input Configuration** pull-down menu.
5. Click the **OK** button.
6. Select File»Save All in the **Project Explorer** window.

Configuring the Channel Input Modes Using the FPGA I/O Property Node

Complete the following steps to configure the channel input modes using the FPGA I/O Property Node.

1. Create FPGA I/O items for the channels of the module for which you want to configure the **Input Configuration** property.
2. Place an FPGA I/O Property Node on the block diagram and configure it for the module channel for which you want to configure the input mode.
3. Click the **Property** section and select **Input Configuration** from the shortcut menu.
4. Right-click the **Input Configuration** input and select **Create»Control** from the shortcut menu.
5. On the front panel of the VI, select an input mode for the channel from the **Input Configuration** pull-down menu.

Change the channel input modes at run time by writing to the control from the host VI. For examples of how to configure the input modes using the FPGA I/O Property Node, refer to the following getting started VIs.

Table 225. Getting Started VI Projects

Module	Example File Path
NI 9230	labview\examples\CompactRIO\Module Specific\NI 9230\NI 9230 Getting Started\NI 9230 Getting Started.lvproj
NI 9232	labview\examples\CompactRIO\Module Specific\NI 9232\NI 9232

Module	Example File Path
	`Getting Started\NI 9232 Getting Started.lvproj`
NI 9234	`labview\examples\CompactRIO\Module Specific\NI 9234\NI 9234 Getting Started\NI 9234 Getting Started.lvproj`

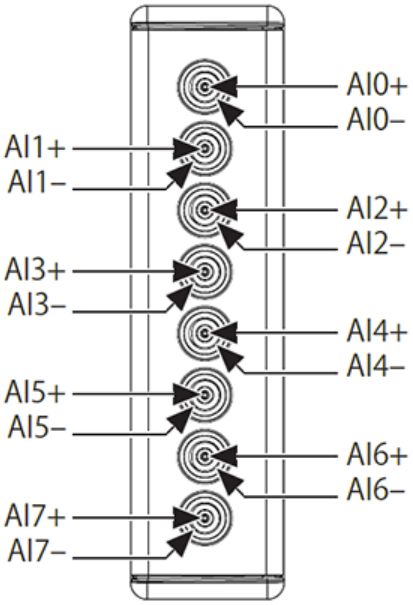
Note The **Input Configuration** property returns error 65538 if the module is acquiring data. Call Stop channel to stop acquiring data before writing properties to the modules.

NI 9231

8 AI, ±5 V, 24 Bit, 51.2 kS/s/ch Simultaneous, AC/DC Coupling, IEPE AC Coupling

NI 9231 Pinout

Figure 54. NI 9231 Pinout



NI 9231 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Note You can synchronize an NI 9231 with other modules that have a selectable timebase source.

Terminals in Software

Table 226. Terminals in Software

Terminal	Description
AI*x*	AI input channel *x*, where *x* is the number of the channel. The NI 9231 has AI channels 0 to 7. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is 13.1072 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9231 to access this channel.
Start	Channel that controls when the module starts acquiring data. To acquire data, write *TRUE* to the Start channel. When the module is acquiring data, write *TRUE* to the Stop channel before you can access properties or TEDS information for the module. No operation is performed when writing *FALSE* to the Start channel.
Stop	Channel that controls when the module stops acquiring data. To stop acquiring data, write *TRUE* to the Stop channel. When the module is acquiring data, write *TRUE* to the Stop channel before you can access properties for the module. No operation is performed when writing *FALSE* to the Stop channel.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

To access the following method for this device, use the FPGA I/O Method Node.

Table 227. Module Methods

Method	Description
Check Cached Status	Returns Booleans for each channel that indicate whether there was a short circuit or open loop on the channel since the last execution of the Check Cached Status method. When the FPGA I/O Node reads the channels, the FPGA VI determines the state of the channels and caches any *TRUE* value until the Check Cached Status method executes. • **Short Circuit**—Returns an array of Boolean values. A value of *TRUE* in any index indicates that the channel sharing a number with that index detected a short circuit on the channel at some point after the last time that the Check Cached Status method executed. • **Open Loop**—Returns an array of Boolean values. A value of *TRUE* in any index indicates that the channel sharing a number with that index detected an open loop on the channel at some point after the last time that the Check Cached Status method executed.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 228. I/O Properties

Property	Description
Input Configuration	Sets the input configuration of the corresponding channel to one of three modes: AC coupled, DC coupled, or IEPE AC coupled.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 229. Module Properties

Property	Description
Data Rate	Sets the rate at which the module acquires data.
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.

Property	Description
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Configuring the NI 9231 Channel Input Modes (FPGA Interface)

You can configure the input mode for each channel of the NI 9230, NI 9232, or NI 9234 at edit time using the C Series Module Properties dialog box.

Programmatically change the input mode for each channel at run time using the FPGA I/O Property Node. The execution of an FPGA I/O Property Node that is configured with an **Input Configuration** property overwrites the value you configured in the C Series Module Properties dialog box for the corresponding channel.

Configuring the Channel Input Modes Using the C Series Module Properties Dialog Box

Complete the following steps to configure the channel input modes using the **C Series Module Properties** dialog box.

1. Configure the CompactRIO system, and add an module.
2. Right-click the module in the **Project Explorer** window and select **Properties** to display the **C Series Module Properties** dialog box.
3. Select the channel(s) for which you want to configure the input mode from the **Channels** table. You can select more than one channel by holding the <Ctrl> or <Shift> key when selecting channels.
4. Select the input mode for the selected channel(s) from the **Input Configuration** pull-down menu.
5. Click the **OK** button.
6. Select **File»Save All** in the **Project Explorer** window.

Configuring the Channel Input Modes Using the FPGA I/O Property Node

Complete the following steps to configure the channel input modes using the FPGA I/O

Property Node.

1. Create FPGA I/O items for the channels of the module for which you want to configure the **Input Configuration** property.
2. Place an FPGA I/O Property Node on the block diagram and configure it for the module channel for which you want to configure the input mode.
3. Click the **Property** section and select **Input Configuration** from the shortcut menu.
4. Right-click the **Input Configuration** input and select **Create»Control** from the shortcut menu.
5. On the front panel of the VI, select an input mode for the channel from the **Input Configuration** pull-down menu.

You can change the channel input modes at run time by writing to the control from the host VI. Refer to the following getting started VIs for examples of how to configure the input modes using the FPGA I/O Property Node.

Table 230. Getting Started VI Project

Module	Example File Path
NI 9231	`labview\examples\CompactRIO\Module Specific\NI 9231\NI 9231 Getting Started\NI 9231 Getting Started.lvproj`

Note The **Input Configuration** property returns error 65538 if the module is acquiring data. Call Stop channel to stop acquiring data before writing properties to the modules.

Converting NI 9231 Data (FPGA Interface)

Set the **Calibration Mode** to Calibrated in the C Series Module Properties dialog box for the NI 9231 if you want the FPGA I/O Node to return calibrated, fixed-point data from the module.

Set the **Calibration Mode** to Raw if you want the FPGA I/O Node to return calibrated, binary data from the module. If you set the **Calibration Mode** to Raw, convert the binary values to engineering units. Convert these values in the host VI.

You can use the following equations in the host VI to convert binary voltage values to

volts:

$$Volts = Binary\ Value \times 610714.8\ pV/LSB$$

Where

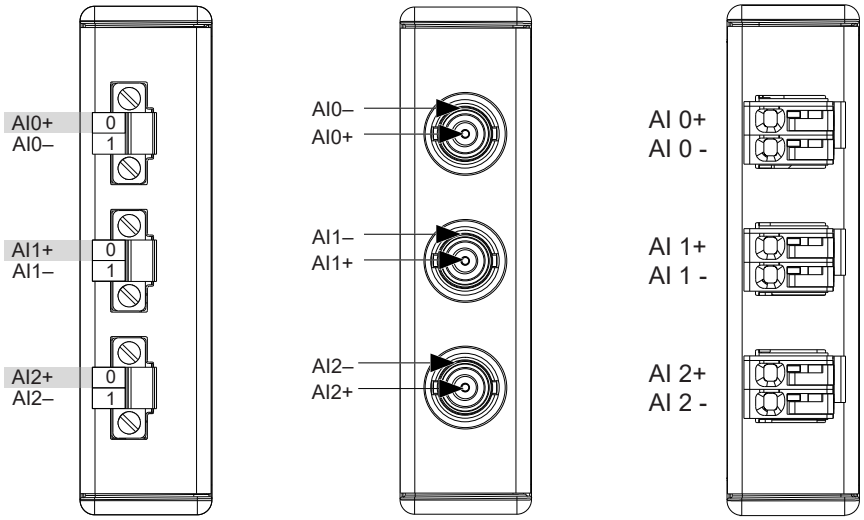
- *Binary Value* is the value returned by the FPGA I/O Node.

NI 9232

CompactRIO 3-Channel, ±30 V, 102.4 kS/s, 24-Bit Software Selectable IEPE and AC/DC Analog Input Module

NI 9232 Pinout

Figure 55. NI 9232 Pinout



NI 9232 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Note You can synchronize an NI 9232 with other modules that have a selectable timebase source.

Terminals in Software

Table 231. Terminals in Software

Terminal	Description
AI*x*	AI input channel *x*, where *x* is the number of the channel. The NI 9232 has AI channels 0 to 2. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is 13.1072 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9232 to access this channel. If you do not export the onboard clock and try to access the Onboard Clock channel, LabVIEW returns a code generation error when you try to compile the FPGA VI.
Start	Channel that controls when the module starts acquiring data. To acquire data, write *TRUE* to the Start channel. When the module is acquiring data, write *TRUE* to the Stop channel before you can access properties or TEDS information for the module. No operation is performed when writing *FALSE* to the Start channel.
Stop	Channel that controls when the module stops acquiring data. To stop acquiring data, write *TRUE* to the Stop channel. When the module is acquiring data, write *TRUE* to the Stop channel before you can access properties for the module. No operation is performed when writing *FALSE* to the Stop channel.

Note Refer to the NI 9232 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9232\NI 9232 Getting Started\NI 9232 Getting Started.lvproj` for an example of using the Start and Stop channels.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

To access the following method for this device, use the FPGA I/O Method Node.

Table 232. Module Methods

Method	Description
Check Cached Status	Returns Booleans for each channel that indicate whether there was a short circuit or open loop on the channel since the last execution of the Check Cached Status method. When the FPGA I/O Node reads the channels, the FPGA VI determines the state of the channels and caches any *TRUE* value until the Check Cached Status method executes. • **Short Circuit**—Returns an array of Boolean values. A value of *TRUE* in any index indicates that the channel sharing a number with that index detected a short circuit on the channel at some point after the last time that the Check Cached Status method executed. • **Open Loop**—Returns an array of Boolean values. A value of *TRUE* in any index indicates that the channel sharing a number with that index detected an open loop on the channel at some point after the last time that the Check Cached Status method executed.

I/O Properties

To access the following I/O properties for this device, use the FPGA I/O Property Node.

Table 233. I/O Properties

Property	Description
Input Configuration	Sets the input configuration of the corresponding channel to one of three modes: AC coupled, DC coupled, or IEPE AC coupled.
LSB Weight	Returns the LSB weight in pV/LSB for the channel. Use this value to convert and calibrate module data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert and calibrate module data if you set the Calibration Mode to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 234. Module Properties

Property	Description
Data Rate	Sets the rate at which the module acquires data.
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Configuring the NI 9230/9232/9234 Channel Input Modes (FPGA Interface)

You can configure the input mode for each channel of the NI 9230, NI 9232, or NI 9234 at edit time using the C Series Module Properties dialog box.

Programmatically change the input mode for each channel at run time using the FPGA I/O Property Node. The execution of an FPGA I/O Property Node that is configured with an **Input Configuration** property overwrites the value you configured in the C Series Module Properties dialog box for the corresponding channel.

Configuring the Channel Input Modes Using the C Series Module Properties Dialog Box

Complete the following steps to configure the channel input modes using the C Series Module Properties dialog box.

1. Configure the CompactRIO system, and add an module.
2. Right-click the module in the **Project Explorer** window and select **Properties** to display the **C Series Module Properties** dialog box.
3. Select the channel(s) for which you want to configure the input mode from the

Channels table. You can select more than one channel by holding the <Ctrl> or <Shift> key when selecting channels.
4. Select the input mode for the selected channel(s) from the **Input Configuration** pull-down menu.
5. Click the **OK** button.
6. Select File»Save All in the **Project Explorer** window.

Configuring the Channel Input Modes Using the FPGA I/O Property Node

Complete the following steps to configure the channel input modes using the FPGA I/O Property Node.

1. Create FPGA I/O items for the channels of the module for which you want to configure the **Input Configuration** property.
2. Place an FPGA I/O Property Node on the block diagram and configure it for the module channel for which you want to configure the input mode.
3. Click the **Property** section and select **Input Configuration** from the shortcut menu.
4. Right-click the **Input Configuration** input and select **Create»Control** from the shortcut menu.
5. On the front panel of the VI, select an input mode for the channel from the **Input Configuration** pull-down menu.

Change the channel input modes at run time by writing to the control from the host VI. For examples of how to configure the input modes using the FPGA I/O Property Node, refer to the following getting started VIs.

Table 235. Getting Started VI Projects

Module	Example File Path
NI 9230	labview\examples\CompactRIO\Module Specific\NI 9230\NI 9230 Getting Started\NI 9230 Getting Started.lvproj
NI 9232	labview\examples\CompactRIO\Module Specific\NI 9232\NI 9232 Getting Started\NI 9232 Getting Started.lvproj
NI 9234	labview\examples\CompactRIO\Module Specific\NI 9234\NI 9234 Getting Started\NI 9234 Getting Started.lvproj

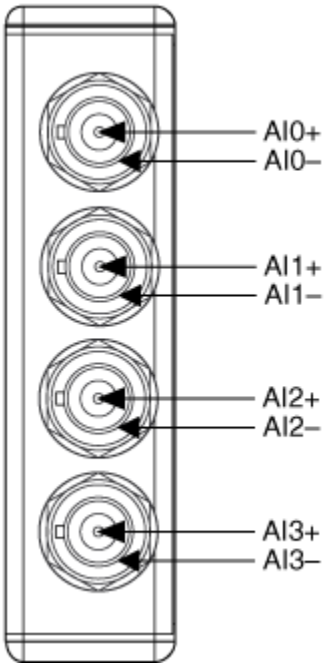
Note The **Input Configuration** property returns error 65538 if the module is acquiring data. Call Stop channel to stop acquiring data before writing properties to the modules.

NI 9234

CompactRIO 4-Channel, ±5 V, 51.2 KS/s, 24-Bit Software Selectable IEPE and AC/DC Analog Input Module

NI 9234 Pinout

Figure 56. NI 9234 Pinout



NI 9234 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Note You can synchronize the NI 9234 with other modules that have a selectable timebase source.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 236. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9234 has AI channels 0 to 3. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	13.1072 MHz internal clock of the NI 9234. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the onboard clock of the NI 9234 to access this channel. If you do not export the onboard clock, when you try to access the **Onboard Clock** channel, LabVIEW returns a code generation error when you compile the FPGA VI.
Start	Channel that controls when the NI 9234 starts acquiring data. If TRUE is written to the **Start** channel, the NI 9234 starts acquiring data. When the NI 9234 is acquiring data, write TRUE to the **Stop** channel before you can access properties or TEDS information for the module. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9234 stops acquiring data. If TRUE is written to the **Stop** channel, the NI 9234 stops acquiring data. When the NI 9234 is acquiring data, write TRUE to the **Stop** channel before you can access properties or TEDS information for the module. If FALSE is written to the **Stop** channel, no operation is performed.

Note For an example of using the **Start** and **Stop** channels, refer to the NI 9234 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9234\NI 9234 Getting Started\NI 9234 Getting Started.lvproj`.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 237. I/O Properties

Property	Description
Input Configuration	Sets the input configuration of the corresponding channel to one of three modes: AC coupled, DC coupled, or IEPE AC coupled.
LSB Weight	Returns the LSB weight in pV/LSB for the channel. Use this value to convert and calibrate NI 9234 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert and calibrate NI 9234 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 238. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9234 acquires data.
Module ID	Returns the module ID, 0x72B5.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

Use the **Onboard Clock** channel in the single-cycle Timed Loop. You cannot use the other channels on the NI 9234 with the single-cycle Timed Loop. For information about loop timing for this module, refer to *Understanding Loop Timing (FPGA Interface)*.

Related concepts:

- Understanding Loop Timing (FPGA Interface)

Configuring the NI 9230/9232/9234 Channel Input Modes (FPGA Interface)

You can configure the input mode for each channel of the NI 9230, NI 9232, or NI 9234 at edit time using the C Series Module Properties dialog box.

Programmatically change the input mode for each channel at run time using the FPGA I/O Property Node. The execution of an FPGA I/O Property Node that is configured with an **Input Configuration** property overwrites the value you configured in the C Series Module Properties dialog box for the corresponding channel.

Configuring the Channel Input Modes Using the C Series Module Properties Dialog Box

Complete the following steps to configure the channel input modes using the C Series Module Properties dialog box.

1. Configure the CompactRIO system, and add an module.
2. Right-click the module in the **Project Explorer** window and select **Properties** to display the **C Series Module Properties** dialog box.
3. Select the channel(s) for which you want to configure the input mode from the **Channels** table. You can select more than one channel by holding the <Ctrl> or <Shift> key when selecting channels.
4. Select the input mode for the selected channel(s) from the **Input Configuration** pull-down menu.
5. Click the **OK** button.
6. Select File»Save All in the **Project Explorer** window.

Configuring the Channel Input Modes Using the FPGA I/O Property Node

Complete the following steps to configure the channel input modes using the FPGA I/O Property Node.

1. Create FPGA I/O items for the channels of the module for which you want to configure the **Input Configuration** property.
2. Place an FPGA I/O Property Node on the block diagram and configure it for the module channel for which you want to configure the input mode.
3. Click the **Property** section and select **Input Configuration** from the shortcut menu.
4. Right-click the **Input Configuration** input and select **Create»Control** from the shortcut menu.
5. On the front panel of the VI, select an input mode for the channel from the **Input Configuration** pull-down menu.

Change the channel input modes at run time by writing to the control from the host VI. For examples of how to configure the input modes using the FPGA I/O Property Node, refer to the following getting started VIs.

Table 239. Getting Started VI Projects

Module	Example File Path
NI 9230	`labview\examples\CompactRIO\Module Specific\NI 9230\NI 9230 Getting Started\NI 9230 Getting Started.lvproj`
NI 9232	`labview\examples\CompactRIO\Module Specific\NI 9232\NI 9232 Getting Started\NI 9232 Getting Started.lvproj`
NI 9234	`labview\examples\CompactRIO\Module Specific\NI 9234\NI 9234 Getting Started\NI 9234 Getting Started.lvproj`

Note The **Input Configuration** property returns error 65538 if the module is acquiring data. Call Stop channel to stop acquiring data before writing properties to the modules.

NI 9234 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9234 has the following channels.

Table 240. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9234, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9234. Right-click the NI 9234 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

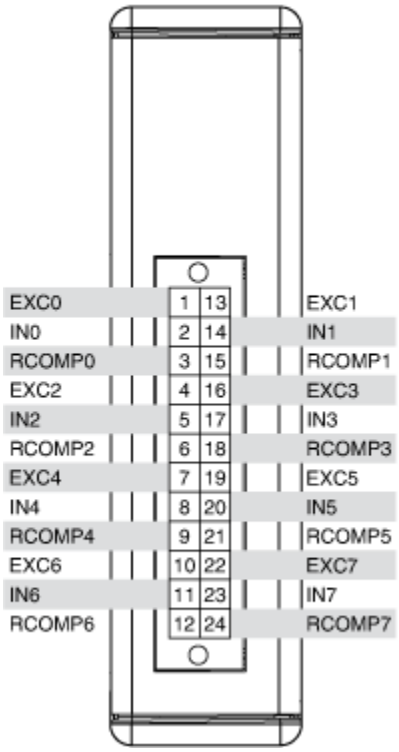
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.
- Channels—Specifies the channel(s) for which you want to select the input configuration.
- Input Configuration—Specifies the input configuration for the selected channel(s). You can select one of three modes: AC coupled, DC coupled, or IEPE AC coupled.

NI 9235

CompactRIO 8-Channel, 24-Bit Quarter-Bridge Analog Input Module

NI 9235 Pinout

Figure 57. NI 9235 Pinout



NI 9235 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Note You can synchronize the NI 9235 with other modules that have a selectable timebase source.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 241. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9235 has AI channels 0 to 7. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.

Terminal	Description
Onboard Clock	12.8 MHz internal clock of the NI 9235. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9235 to access this channel.
Start	Channel that controls when the NI 9235 starts *acquiring data*. If TRUE is written to the **Start** channel, the NI 9235 starts acquiring data. When the NI 9235 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9235 stops acquiring data. If TRUE is written to the **Stop** channel, the NI 9235 stops acquiring data. When the NI 9235 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Stop** channel, no operation is performed.



Note For an example of using the **Start** and **Stop** channels, refer to the NI 9235 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9235\NI 9235 Getting Started\NI 9235 Getting Started.lvproj`.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 242. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in units of 100 fV/V per LSB for the channel. 100 fV is equal to 10^{-13} volts. Use this value to convert and calibrate NI 9235 data if you set the **Calibration Mode** to **Raw** in the *C Series Module Properties* dialog box.
Offset	Returns the calibration offset in units of 100 nV/V for the channel. Use this value to convert and calibrate NI 9235 data if you set the **Calibration Mode** to **Raw** in the **C Series Module Properties** dialog box.
Shunt Cal Enable	Controls the shunt calibration switch for each channel. Refer to the NI 9235 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9235\NI 9235 Getting Started\NI 9235 Getting Started.lvproj` for an example of using the **Shunt Cal Enable** property.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 243. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9235 acquires data.
Module ID	Returns the module ID, 0x732A.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

Use the **Onboard Clock** channel in the single-cycle Timed Loop. You cannot use the other channels on the NI 9235 with the single-cycle Timed Loop. For information about loop timing for this module, refer to *Understanding Loop Timing (FPGA Interface)*.

Related concepts:

- Understanding Loop Timing (FPGA Interface)

NI 9235 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts/volts (channel voltage/excitation voltage).

Module Channels

The NI 9235 has the following channels.

Table 244. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9235, *x* is 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the NI 9235 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

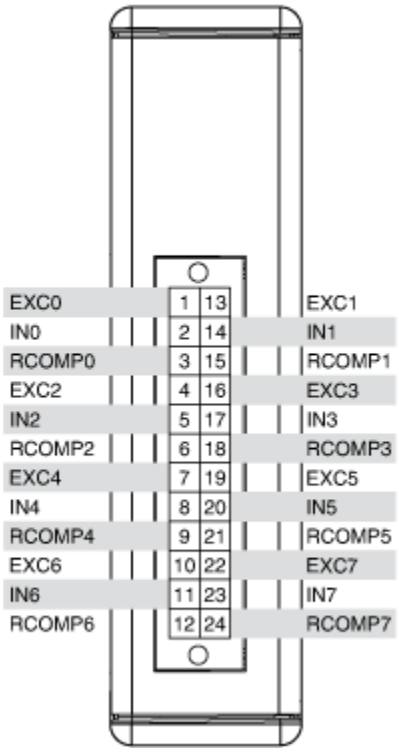
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.

NI 9236

CompactRIO 8-Channel, 24-Bit Quarter-Bridge Analog Input Module

NI 9236 Pinout

Figure 58. NI 9236 pinout



NI 9236 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Note You can synchronize the NI 9236 with other modules that have a selectable timebase source.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 245. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9236 has AI channels 0 to 7. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.

Terminal	Description
Onboard Clock	12.8 MHz internal clock of the NI 9236. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9236 to access this channel.
Start	Channel that controls when the NI 9236 starts acquiring data. If TRUE is written to the **Start** channel, the NI 9236 starts acquiring data. When the NI 9236 is acquiring data, write TRUE to the **Stop** channel before you can access properties or TEDS information for the module. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9236 stops acquiring data. If TRUE is written to the **Stop** channel, the NI 9236 stops acquiring data. When the NI 9236 is acquiring data, write TRUE to the **Stop** channel before you can access properties or TEDS information for the module. If FALSE is written to the **Stop** channel, no operation is performed.

Note For an example of using the **Start** and **Stop** channels, refer to the NI 9236 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9236\NI 9236 Getting Started\NI 9236 Getting Started.lvproj`.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 246. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in units of 100 fV/V per LSB for the channel. 100 fV is equal to 10^{-13} volts. Use this value to convert and calibrate NI 9236 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in units of 100 nV/V for the channel. Use this value to convert and calibrate NI 9236 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Shunt Cal Enable	Controls the shunt calibration switch for each channel. Refer to the NI 9236 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9236\NI 9236 Getting Started\NI 9236 Getting Started.lvproj` for an example of using the **Shunt Cal Enable** property.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 247. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9236 acquires data.
Module ID	Returns the module ID, 0x732B.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

Use the **Onboard Clock** channel in the single-cycle Timed Loop. You cannot use the other channels on the NI 9236 with the single-cycle Timed Loop. For information about loop timing for this module, refer to *Understanding Loop Timing (FPGA Interface)*.

Related concepts:

- Understanding Loop Timing (FPGA Interface)

NI 9236 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts/volts (channel voltage/excitation voltage).

Module Channels

The NI 9236 has the following channels.

Table 248. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9236, *x* is 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9236. Right-click the NI 9236 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

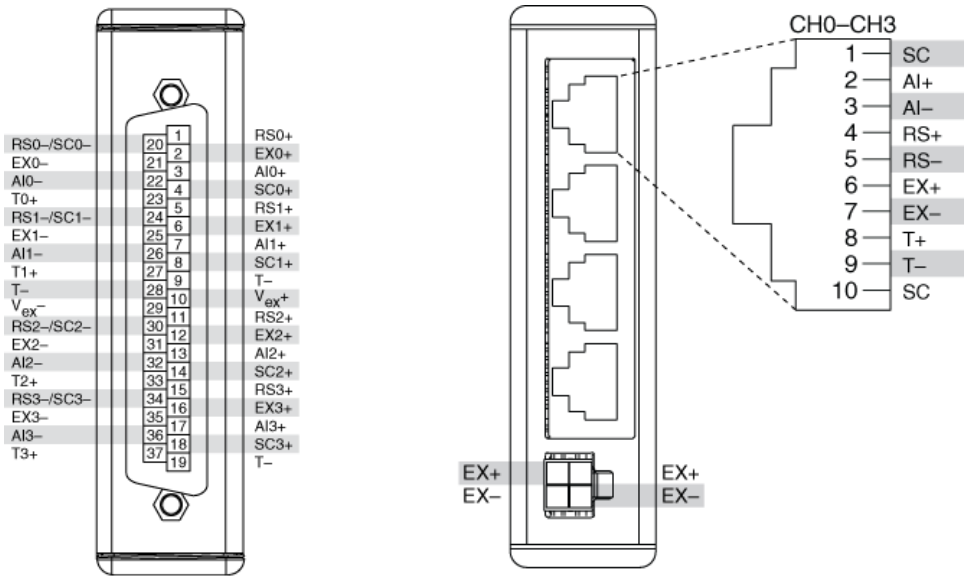
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.

NI 9237

CompactRIO 4-Channel, 24-Bit Half/Full-Bridge Analog Input Module

NI 9237 Pinout

Figure 59. NI 9237 pinout



NI 9237 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Note You can synchronize the NI 9237 with other modules that have a selectable timebase source.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 249. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9237 has AI channels 0 to 3. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	12.8 MHz internal clock of the NI 9237. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the

Terminal	Description
	Onboard Clock of the NI 9237 to access this channel.
Start	Channel that controls when the NI 9237 starts acquiring data. If TRUE is written to the **Start** channel, the NI 9237 starts acquiring data. When the NI 9237 is acquiring data, write TRUE to the **Stop** channel before you can access properties or TEDS information for the module. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9237 stops acquiring data. If TRUE is written to the **Stop** channel, the NI 9237 stops acquiring data. When the NI 9237 is acquiring data, write TRUE to the **Stop** channel before you can access properties or TEDS information for the module. If FALSE is written to the **Stop** channel, no operation is performed.

Note For an example of using the **Start** and **Stop** channels, refer to the NI 9237 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9237\NI 9237 Getting Started\NI 9234 Getting Started.lvproj`.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 250. I/O Properties

Property	Description
Half-Bridge Enable	Controls the half-bridge completion option for each channel.
LSB Weight	Returns the LSB weight in units of 100 fV/V per LSB for the channel. 100 fV is equal to 10^{-13} volts. Use this value to convert and calibrate NI 9237 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in units of 10 nV/V for the channel. Use this value to convert and calibrate NI 9237 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Offset Cal Enable	Enables offset calibration. This disconnects both signal input pins and forces the channel inputs to zero.
Shunt Cal Enable	Controls the shunt calibration switch for each channel.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 251. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9237 acquires data.
Excitation Voltage	Sets the excitation voltage level. All channels share the same excitation voltage.
Module ID	Returns the module ID, 0x73A4 (NI 9237 with DSUB) or 0x71C3 (NI 9237 with RJ-50).
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

Use the **Onboard Clock** channel in the single-cycle Timed Loop. You cannot use the other channels on the NI 9237 with the single-cycle Timed Loop. For information about loop timing for this module, refer to *Understanding Loop Timing (FPGA Interface)*.

Related concepts:

- Understanding Loop Timing (FPGA Interface)

NI 9237 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts/volts (channel voltage/excitation voltage).

Module Channels

The NI 9237 has the following channels.

Table 252. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9236, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9237. Right-click the NI 9237 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

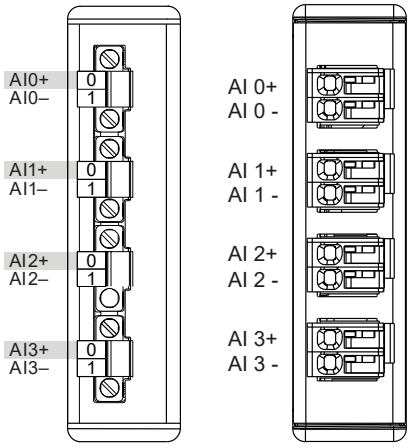
- Excitation Voltage—Specifies the excitation voltage for the module to output to bridges, or specifies external excitation.
- Enable Half-Bridge Completion—Enables half-bridge completion for individual channels.

NI 9238

CompactRIO 4-Channel, ±0.5 V, 24-Bit Simultaneous Analog Input Module

NI 9238 Pinout

Figure 60. NI 9238 Pinout



NI 9238 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Note You can simultaneously read from or write to multiple channels on the NI 9238. You can also synchronize an NI 9238 module with other modules that have a *selectable timebase source*.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 253. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9238 has AI channels 0 to 3. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is 12.8 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9238 to access this channel.
Start	Channel that controls when the NI 9238 starts acquiring data. If TRUE is written to the **Start** channel, the NI 9238 starts acquiring data. When the NI 9238 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9238 stops acquiring data. If TRUE is written to the **Stop** channel, the NI 9238 stops acquiring data. When the NI 9238 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Stop** channel, no operation is performed.

Note For an example of using the **Start** and **Stop** channels, refer to the NI 9238 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9238\NI 9238 Getting Started\NI 9234 Getting Started.lvproj`.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 254. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in fV/LSB for the channel. Use this value to convert and calibrate NI 9238 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in units of 100 nV/V for the channel. Use this value to convert and calibrate NI 9238 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 255. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9238 acquires data.
Module ID	Returns the module ID, 0x7750.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

Use the **Onboard Clock** channel in the single-cycle Timed Loop. You cannot use the other channels on the NI 9238 with the single-cycle Timed Loop. For information about loop timing for this module, refer to *Understanding Loop Timing (FPGA Interface)*.

Related concepts:

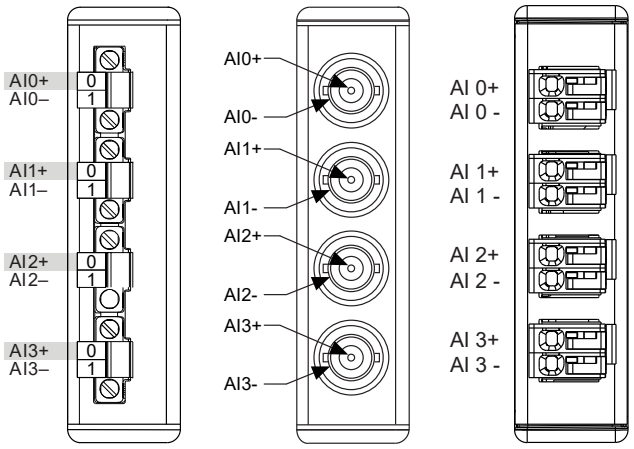
- Understanding Loop Timing (FPGA Interface)

NI 9239

CompactRIO 4-Channel, ±10 V, 24-Bit Simultaneous Analog Input Module

NI 9239 Pinout

Figure 61. NI 9239 pinout



NI 9239 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Note You can simultaneously read from or write to multiple channels on the NI 9239. You also can synchronize the NI 9239 module with other modules that have a selectable timebase source.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 256. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9239 has AI

Terminal	Description
	channels 0 to 3. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is 12.8 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9239 to access this channel.
Start	Channel that controls when the NI 9239 starts acquiring data. If TRUE is written to the **Start** channel, the NI 9239 starts acquiring data. When the NI 9239 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9239 stops acquiring data. If TRUE is written to the **Stop** channel, the NI 9239 stops acquiring data. When the NI 9239 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Stop** channel, no operation is performed.

Note For an example of using the **Start** and **Stop** channels, refer to the NI 9239 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9239\NI 9239 Getting Started\NI 9239 Getting Started.lvproj`.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 257. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in pV/LSB for the channel. Use this value to convert and calibrate NI 9239 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in units of 100 nV/V for the channel. Use this value to convert and calibrate NI 9239 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 258. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9239 acquires data.
Module ID	Returns the module ID, 0x71C2.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

Use the **Onboard Clock** channel in the single-cycle Timed Loop. You cannot use the other channels on the NI 9239 with the single-cycle Timed Loop. For information about loop timing for this module, refer to *Understanding Loop Timing (FPGA Interface)*.

Related concepts:

- Understanding Loop Timing (FPGA Interface)

NI 9239 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9239 has the following channels.

Table 259. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9239, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9239. Right-click the NI 9239 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

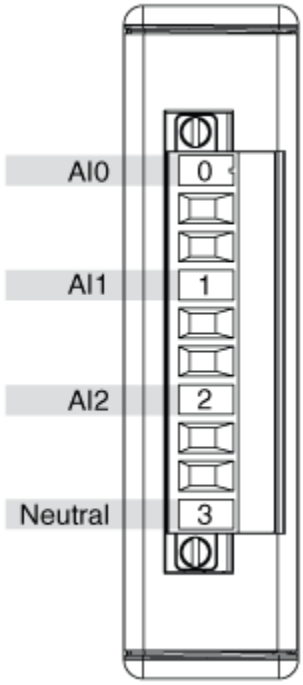
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9242

CompactRIO 4-Channel, 250 V_{rms}, 24-Bit Simultaneous Analog Input Module

NI 9242 Pinout

Figure 62. NI 9242 pinout



NI 9242 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Note You can simultaneously read from or write to multiple channels on the NI 9242. You can also synchronize an NI 9242 module with other modules that have a *selectable timebase source*.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 260. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9242 has AI channels 0 to 2. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.

Terminal	Description
Neutral	Neutral input channel. Do not access Neutral channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is 12.8 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9242 to access this channel.
Start	Channel that controls when the NI 9242 starts acquiring data. If TRUE is written to the **Start** channel, the NI 9242 starts acquiring data. When the NI 9242 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9242 stops acquiring data. If TRUE is written to the **Stop** channel, the NI 9242 stops acquiring data. When the NI 9242 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Stop** channel, no operation is performed.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

This device does not support any I/O 0roperties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 261. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9242 acquires data.
Module ID	Returns the module ID, 0x772A.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

Use the **Onboard Clock** channel in the single-cycle Timed Loop. You cannot use the other channels on the NI 9242 with the single-cycle Timed Loop. For information about loop timing for this module, refer to *Understanding Loop Timing (FPGA Interface)*.

Related concepts:

- Understanding Loop Timing (FPGA Interface)

Converting NI 9242 and NI 9244 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9242 and NI 9242 if you want the FPGA I/O Node to return calibrated, fixed-point data from the module.

When **Calibration Mode** is set to **Raw**, the FPGA I/O Node returns calibrated, binary data from the module. Convert the binary values to engineering units. You must convert these values in the host VI.

Using an Equation to Convert Binary Values to Voltage

You can use the following equations in the host VI to convert binary voltage values to volts: NI 9242

Use the following equation in the host VI to convert binary voltage values to volts:

NI 9242: $Volts = Binary\ Value \times 500\ V \div 8{,}388{,}607$

NI 9244: $Volts = Binary\ Value \times 997.5\ V \div 8{,}388{,}607$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.

NI 9242 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9242 has the following channels.

Table 262. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9242, *x* is 0 to 2.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9242. Right-click the NI 9242 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

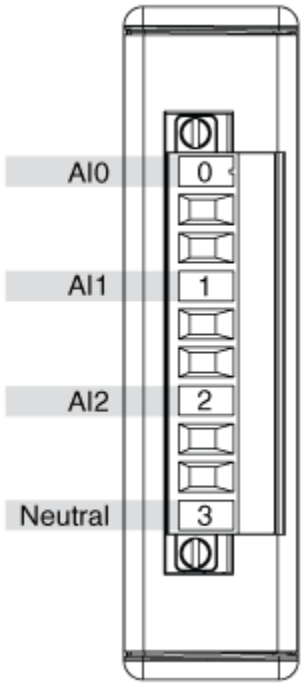
- **Data Rate**—Specifies the rate at which the NI 9242 acquires and returns data. The data rate must remain within the appropriate data rate range listed in NI 9242 hardware documentation on `ni.com/manuals`.

NI 9244

CompactRIO 4-Channel, 400 V_{rms} L-N, 800 V_{rms} L-L, 24-Bit Simultaneous Analog Input Module

NI 9244 Pinout

Figure 63. NI 9244 pinout



NI 9244 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Note You can simultaneously read from or write to multiple channels on the NI 9244. You can also synchronize an NI 9244 module with other modules that have a *selectable timebase source*.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 263. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9244 has AI channels 0 to 2. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Neutral	Neutral input channel. Do not access Neutral channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is 12.8 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9244 to access this channel.
Start	Channel that controls when the NI 9244 starts acquiring data. If TRUE is written to the **Start** channel, the NI 9244 starts acquiring data. When the NI 9244 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9244 stops acquiring data. If TRUE is written to the **Stop** channel, the NI 9244 stops acquiring data. When the NI 9244 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Stop** channel, no operation is performed.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

This device does not support any I/O 0roperties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 264. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9244 acquires data.
Module ID	Returns the module ID, 0x76E8.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

Use the **Onboard Clock** channel in the single-cycle Timed Loop. You cannot use the other channels on the NI 9244 with the single-cycle Timed Loop. For information about loop timing for this module, refer to *Understanding Loop Timing (FPGA Interface)*.

Related concepts:

- Understanding Loop Timing (FPGA Interface)

Converting NI 9242 and NI 9244 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9242 and NI 9242 if you want the FPGA I/O Node to return calibrated, fixed-point data from the module.

When **Calibration Mode** is set to **Raw**, the FPGA I/O Node returns calibrated, binary data from the module. Convert the binary values to engineering units. You must convert these values in the host VI.

Using an Equation to Convert Binary Values to Voltage

You can use the following equations in the host VI to convert binary voltage values to volts: NI 9242

Use the following equation in the host VI to convert binary voltage values to volts:

NI 9242: $Volts = Binary\ Value \times 500\ V \div 8,388,607$

NI 9244: $Volts = Binary\ Value \times 997.5\ V \div 8,388,607$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.

NI 9244 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9244 has the following channels.

Table 265. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9244, *x* is 0 to 2.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9244. Right-click the NI 9244 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

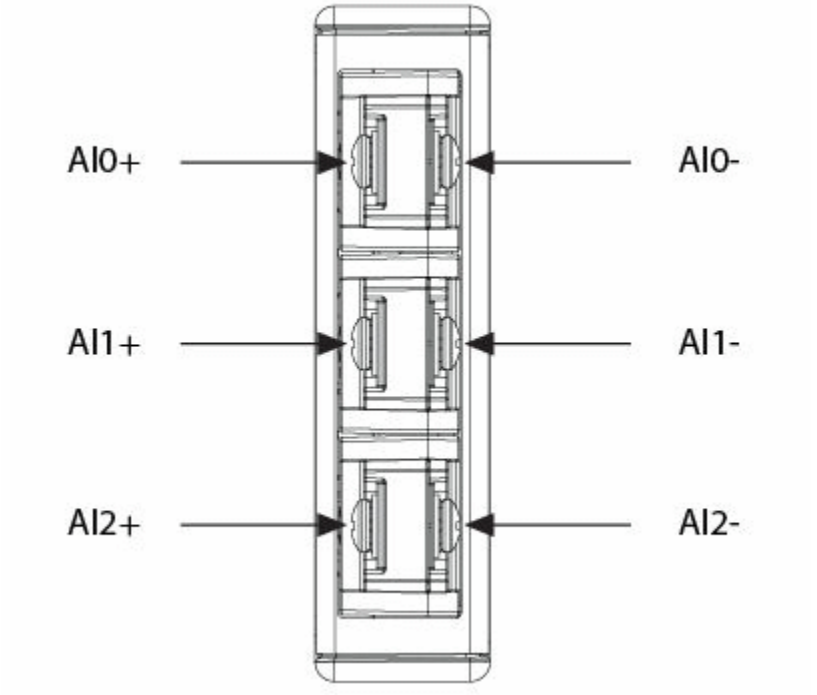
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- **Data Rate**—Specifies the rate at which the NI 9244 acquires and returns data. The data rate must remain within the appropriate data rate range listed in NI 9244 hardware documentation on `ni.com/manuals`.

NI 9246

CompactRIO 3-Channel, 20 A_{rms} Continuous, 24-Bit, Analog Input Module

NI 9246 Pinout

Figure 64. NI 9246 pinout



NI 9246 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Note You can simultaneously read from or write to multiple channels on the NI 9246. You can also synchronize an NI 9246 module with other modules that have a *selectable timebase source*.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 266. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9246 has AI channels 0 to 2. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is 13.1 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9246 to access this channel.
Start	Channel that controls when the NI 9246 starts acquiring data. If TRUE is written to the **Start** channel, the NI 9246 starts acquiring data. When the NI 9246 is acquiring data, write TRUE to the **Stop** channel before you can access properties information for the module. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9246 stops acquiring data. If TRUE is written to the **Stop** channel, the NI 9246 stops acquiring data. When the NI 9246 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module.

Terminal	Description
	If FALSE is written to the **Stop** channel, no operation is performed.

Note For an example of using the **Start** and **Stop** channels, refer to the NI 9246 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9246\NI 9246 Getting Started\NI 9234 Getting Started.lvproj`.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 267. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in pV/LSB for the channel. Use this value to convert and calibrate NI 9246 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in µV for the channel. Use this value to convert and calibrate NI 9246 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 268. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9246 acquires data.
Module ID	Returns the module ID, 0x77E3.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

Use the **Onboard Clock** channel in the single-cycle Timed Loop. You cannot use the other channels on the NI 9246 with the single-cycle Timed Loop. For information about loop timing for this module, refer to *Understanding Loop Timing (FPGA Interface)*.

Related concepts:

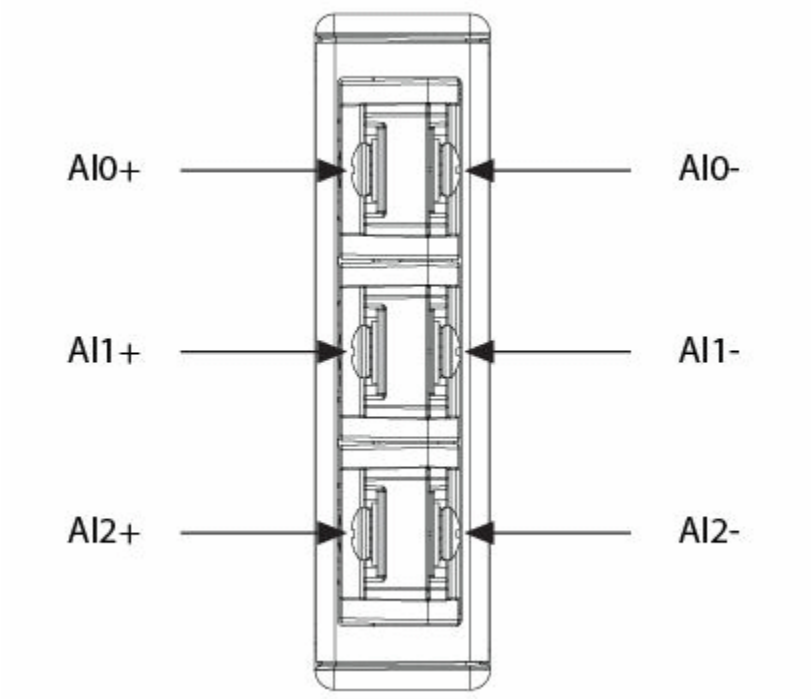
- Understanding Loop Timing (FPGA Interface)

NI 9247

CompactRIO 3-Channel, 50 A_{rms} Continuous, 24-Bit, Analog Input Module

NI 9247 Pinout

Figure 65. NI 9247 pinout



NI 9247 (FPGA Interface)

CompactRIO 3-Channel, 50 A_{rms} Continuous, 24-Bit, Analog Input Module

FPGA I/O Node

You can use an FPGA I/O Node, configured for *reading*, with this device.

Note You can *simultaneously read from or write to* multiple channels on the NI 9247. You can also *synchronize* an NI 9247 module with other modules that have a *selectable timebase source*.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 269.

Terminal	Description
AIx	Analog input channel x, where x is the number of the channel. The NI 9247 has AI

Terminal	Description
	channels 0 to 2. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is 13.1 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. You must *export the Onboard Clock of the NI 9247* to access this channel.
Start	Channel that controls when the NI 9247 starts *acquiring data*. If TRUE is written to the **Start** channel, the NI 9247 starts acquiring data. When the NI 9247 is acquiring data, you must write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Start** channel, no operation is performed. Refer to the NI 9247 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9247\NI 9247 Getting Started\NI 9247 Getting Started.lvproj` for an example of using the **Start** and **Stop** channels.
Stop	Channel that controls when the NI 9247 stops acquiring data. If TRUE is written to the **Stop** channel, the NI 9247 stops acquiring data. When the NI 9247 is acquiring data, you must write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Stop** channel, no operation is performed. Refer to the NI 9247 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9247\NI 9247 Getting Started\NI 9247 Getting Started.lvproj` for an example of using the **Start** and **Stop** channels.

Arbitration

This device supports only the Arbitrate if Multiple Requestors Only option for arbitration. You cannot configure arbitration settings for this device. Methods

This device does not support any methods. I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 270.

Property	Description
LSB Weight	Returns the LSB weight in pV/LSB for the channel. Use this value to *convert and calibrate* NI 9247 data if you set the **Calibration Mode** to **Raw** in the *C Series Module Properties* dialog box.
Offset	Returns the calibration offset in µV for the channel. Use this value to convert and calibrate NI 9247 data if you set the **Calibration Mode** to **Raw** in the **C Series Module**

Property	Description
	Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 271.

Property	Description
Data Rate	*Sets the rate* at which the NI 9247 acquires data.
Module ID	Returns the module ID, 0x77B8.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

You can use the **Onboard Clock** channel in the single-cycle Timed Loop. You cannot use the other channels on the NI 9247 with the single-cycle Timed Loop. Refer to the *Understanding Loop Timing (FPGA Interface)* topic for information about loop timing for this module.

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Note You can simultaneously read from or write to multiple channels on the NI 9247. You can also synchronize an NI 9247 module with other modules that have a selectable timebase source.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 272. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9247 has AI channels 0 to 2. Do not access AI channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is 13.1 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9247 to access this channel.
Start	Channel that controls when the NI 9247 starts acquiring data. If TRUE is written to the **Start** channel, the NI 9247 starts acquiring data. When the NI 9247 is acquiring data, write TRUE to the **Stop** channel before you can access properties information for the module. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9247 stops acquiring data. If TRUE is written to the **Stop** channel, the NI 9247 stops acquiring data. When the NI 9247 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Stop** channel, no operation is performed.

Note For an example of using the **Start** and **Stop** channels, refer to the NI 9247 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9247\NI 9247 Getting Started\NI 9234 Getting Started.lvproj`.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 273. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in pV/LSB for the channel. Use this value to convert and calibrate NI 9247 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in µV for the channel. Use this value to convert and calibrate NI 9247 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 274. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9247 acquires data.
Module ID	Returns the module ID, 0x77B8.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

Use the **Onboard Clock** channel in the single-cycle Timed Loop. You cannot use the other channels on the NI 9247 with the single-cycle Timed Loop. For information about loop timing for this module, refer to *Understanding Loop Timing (FPGA Interface)*.

Related concepts:

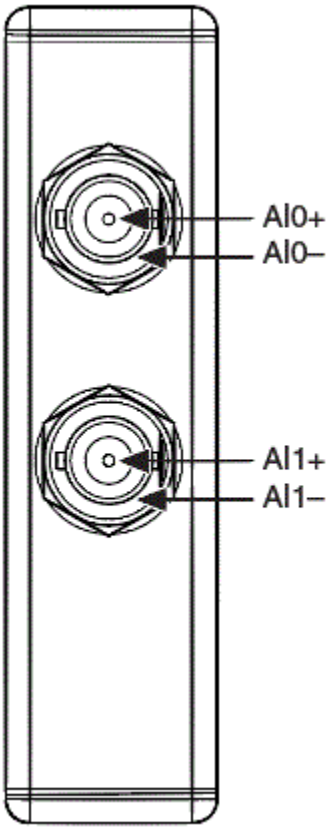
- Understanding Loop Timing (FPGA Interface)

NI 9250

CompactRIO 2-Ch, ±5 V, 24-Bit, Analog Input Software Selectable IEPE & AC/DC

NI 9250 Pinout

Figure 66. NI 9250 pinout



NI 9250 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Note You can synchronize the NI 9250 module with other modules that have

a selectable timebase source.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 275. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9250 has AI channels 0 to 1.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is 13.1072 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9250 to access this channel.
Start	Channel that controls when the NI 9250 starts acquiring data. If TRUE is written to the **Start** channel, the NI 9250 starts acquiring data. When the NI 9250 is acquiring data, write TRUE to the **Stop** channel before you can access properties information for the module. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9250 stops acquiring data. If TRUE is written to the **Stop** channel, the NI 9250 stops acquiring data. When the NI 9250 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Stop** channel, no operation is performed.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 276. I/O Properties

Property	Description
Input Configuration	Sets the input configuration of the corresponding channel to one of three modes: AC Coupled, DC Coupled, IEPE AC Coupled.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 277. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9250 acquires data.
Module ID	Returns the module ID, 0x77EA.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Configuring the NI 9250 Channel Input Modes (FPGA Interface)

Configure the input mode for each channel at edit time using the C Series Module Properties dialog box.

Programmatically change the input mode for each channel at run time using the FPGA I/O Property Node. The execution of an FPGA I/O Property Node that is configured with an **Input Configuration** property overwrites the value you configured in the C Series Module Properties dialog box for the corresponding channel.

Configuring the Channel Input Modes Using the C Series Module Properties Dialog Box

Complete the following steps to configure the channel input modes using the C Series Module Properties dialog box.

1. Configure the CompactRIO system, and add an module.
2. Right-click the module in the **Project Explorer** window and select **Properties** to display the **C Series Module Properties** dialog box.
3. Select the channel(s) for which you want to configure the input mode from the **Channels** table. You can select more than one channel by holding the <Ctrl> or <Shift> key when selecting channels.
4. Select the input mode for the selected channel(s) from the **Input Configuration** pull-down menu.
5. Click the **OK** button.
6. Select File»Save All in the **Project Explorer** window.

Configuring the Channel Input Modes Using the FPGA I/O Property Node

Complete the following steps to configure the channel input modes using the FPGA I/O Property Node.

1. Create FPGA I/O items for the channels of the module for which you want to configure the **Input Configuration** property.
2. Place an FPGA I/O Property Node on the block diagram and configure it for the module channel for which you want to configure the input mode.
3. Click the **Property** section and select **Input Configuration** from the shortcut menu.
4. Right-click the **Input Configuration** input and select **Create»Control** from the shortcut menu.
5. On the front panel of the VI, select an input mode for the channel from the **Input Configuration** pull-down menu.

Change the channel input modes at run time by writing to the control from the host VI. For an example of how to configure the input modes using the FPGA I/O Property Node, refer to labview\examples\CompactRIO\Module Specific\NI 9250\NI 9250 Getting Started\NI 9250 Getting Started.lvproj.

Note The **Input Configuration** property returns error 65538 if the module is acquiring data. Call Stop channel to stop acquiring data before writing properties to the modules.

Converting NI 9250 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9250 if you want the FPGA I/O Node to return calibrated, fixed-point data from the module.

When **Calibration Mode** is set to **Raw**, the FPGA I/O Node returns calibrated, binary data from the module. Convert the binary values to engineering units. You must convert these values in the host VI.

Using an Equation to Convert Binary Values to Voltage

Use the following equation in the host VI to convert binary voltage values to volts:

$$Volts = Binary\ Value \times 608896\ pV/LSB$$

Where

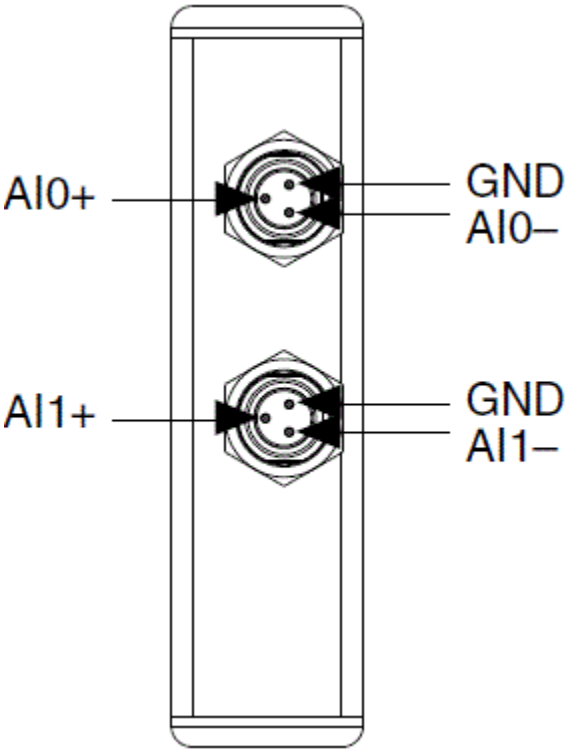
- *Binary Value* is the value returned by the FPGA I/O Node.

NI 9251

CompactRIO 2-Channel, 3 Vrms, 24-Bit Differential Analog Input Module

NI 9251 Pinout

Figure 67. NI 9251 pinout



NI 9251 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Note You can synchronize the NI 9251 module with other modules that have a selectable timebase source.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 278. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9251 has AI channels 0 to 1.
Onboard	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock

Terminal	Description
Clock	frequency is 13.1072 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9251 to access this channel.
Start	Channel that controls when the NI 9251 starts acquiring data. If TRUE is written to the **Start** channel, the NI 9251 starts acquiring data. When the NI 9251 is acquiring data, write TRUE to the **Stop** channel before you can access properties information for the module. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9251 stops acquiring data. If TRUE is written to the **Stop** channel, the NI 9251 stops acquiring data. When the NI 9251 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Stop** channel, no operation is performed.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 279. I/O Properties

Property	Description
Input Configuration	Sets the input configuration of the corresponding channel to one of three modes: AC Coupled, DC Coupled, IEPE AC Coupled.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 280. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9251 acquires data.
Module ID	Returns the module ID, 0x77E9.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Configuring the NI 9251 Channel Input Modes (FPGA Interface)

Configure the input mode for each channel at edit time using the C Series Module Properties dialog box.

Programmatically change the input mode for each channel at run time using the FPGA I/O Property Node. The execution of an FPGA I/O Property Node that is configured with an **Input Configuration** property overwrites the value you configured in the C Series Module Properties dialog box for the corresponding channel.

Configuring the Channel Input Modes Using the C Series Module Properties Dialog Box

Complete the following steps to configure the channel input modes using the C Series Module Properties dialog box.

1. Configure the CompactRIO system, and add an module.
2. Right-click the module in the **Project Explorer** window and select **Properties** to display the **C Series Module Properties** dialog box.
3. Select the channel(s) for which you want to configure the input mode from the

Channels table. You can select more than one channel by holding the <Ctrl> or <Shift> key when selecting channels.
4. Select the input mode for the selected channel(s) from the **Input Configuration** pull-down menu.
5. Click the **OK** button.
6. Select File»Save All in the **Project Explorer** window.

Configuring the Channel Input Modes Using the FPGA I/O Property Node

Complete the following steps to configure the channel input modes using the FPGA I/O Property Node.

1. Create FPGA I/O items for the channels of the module for which you want to configure the **Input Configuration** property.
2. Place an FPGA I/O Property Node on the block diagram and configure it for the module channel for which you want to configure the input mode.
3. Click the **Property** section and select **Input Configuration** from the shortcut menu.
4. Right-click the **Input Configuration** input and select **Create»Control** from the shortcut menu.
5. On the front panel of the VI, select an input mode for the channel from the **Input Configuration** pull-down menu.

Change the channel input modes at run time by writing to the control from the host VI. For an example of how to configure the input modes using the FPGA I/O Property Node, refer to labview\examples\CompactRIO\Module Specific\NI 9251\NI 9251 Getting Started\NI 9251 Getting Started.lvproj.

Note The **Input Configuration** property returns error 65538 if the module is acquiring data. Call Stop channel to stop acquiring data before writing properties to the modules.

Converting NI 9251 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9251 if you want the FPGA I/O Node to return calibrated, fixed-point data from the module.

When **Calibration Mode** is set to **Raw**, the FPGA I/O Node returns calibrated, binary data from the module. Convert the binary values to engineering units. You must convert these values in the host VI.

Using an Equation to Convert Binary Voltage Values to Volts

Use the following equation in the host VI to convert binary voltage values to volts:

$$Volts = Binary\ Value \times 515589\ pV/LSB$$

Where

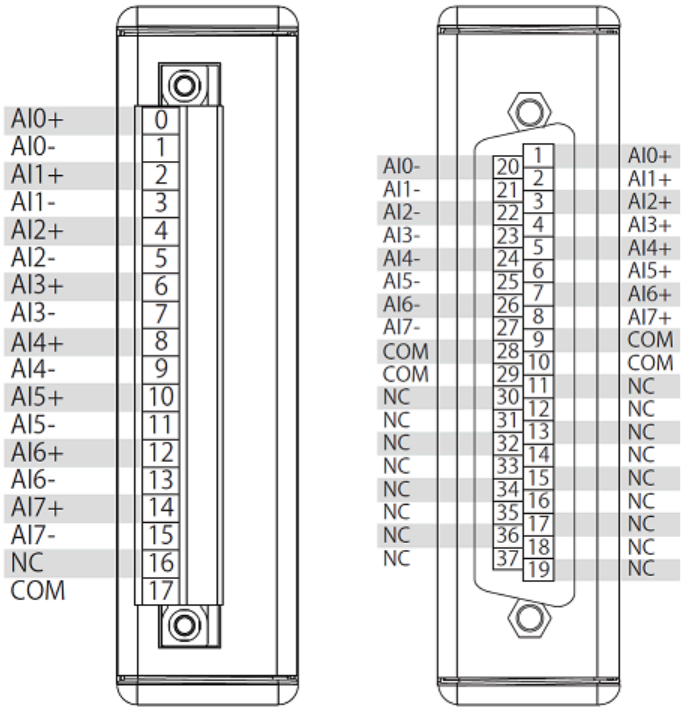
- *Binary Value* is the value returned by the FPGA I/O Node.

NI 9252

CompactRIO 8-Channel, ±10 V, 24-Bit Differential Analog Input Module

NI 9252 Pinout

Figure 68. NI 9252 Pinout



NI 9252 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 281. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9252 has channels 0 to 7.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is 12.8 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9252 to access this channel.
Start	Channel that controls when the NI 9252 starts acquiring data. If TRUE is written to the **Start** channel, the NI 9252 starts acquiring data. When the NI 9252 is acquiring data, write TRUE to the **Stop** channel before you can access properties information for the module. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9252 stops acquiring data. If TRUE is written to the **Stop** channel, the NI 9252 stops acquiring data. When the NI 9252 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Stop** channel, no operation is performed.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 282. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9252 acquires data.
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.
Filter Response	Sets the AI filter types supported by the device. The filters supported by this module are the Comb (default) and Butterworth filter.
Filter Order	Sets the Filter Order for the Butterworth filter. This setting would only take effect if the module is currently configured to the Butterworth filter.
Filter Frequency	Sets the AI filter's cutoff frequency.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Converting NI 9252 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9252 if you want the FPGA I/O Node to return calibrated, fixed-point data from the module.

When **Calibration Mode** is set to **Raw**, the FPGA I/O Node returns calibrated, binary

data from the module. Convert the binary values to engineering units. You must convert these values in the host VI.

Using an Equation to Convert Binary Voltage Values to Volts

Use the following equation in the host VI to convert binary voltage values to volts:

$$Volts = Binary\ Value \times 1261243\ pV/LSB$$

Where

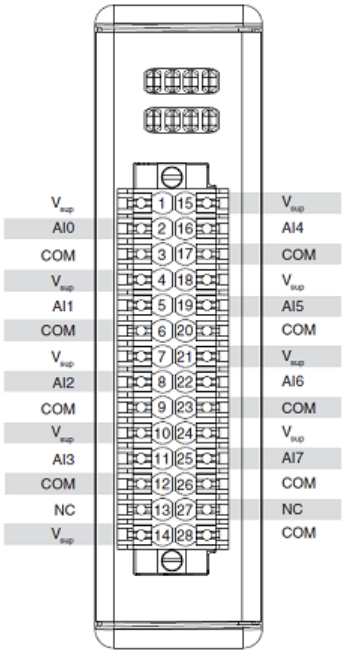
- *Binary Value* is the value returned by the FPGA I/O Node.

NI 9253

CompactRIO 8-Channel, ±20 mA, 24-Bit Differential Analog Input Module

NI 9253 Pinout

Figure 69. NI 9253 Pinout



NI 9253 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 283. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9253 has channels 0 to 7.
Onboard Clock	Gives access to the onboard clock in the LabVIEW block diagram. The Onboard Clock frequency is 12.8 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the Onboard Clock of the NI 9253 to access this channel.
Start	Channel that controls when the NI 9253 starts acquiring data. If TRUE is written to the **Start** channel, the NI 9253 starts acquiring data. When the NI 9253 is acquiring data, write TRUE to the **Stop** channel before you can access properties information for the module. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9253 stops acquiring data. If TRUE is written to the **Stop** channel, the NI 9253 stops acquiring data. When the NI 9253 is acquiring data, write TRUE to the **Stop** channel before you can access properties for the module. If FALSE is written to the **Stop** channel, no operation is performed.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Module Methods

Use the FPGA I/O Method Node to access the following module method for this device.

Table 284. Module Methods

Method	Description
Check Cached Status	Returns Booleans for each channel that indicate whether there was a input limits fault, overcurrent, or field side power fault on the channel since the last execution of the Check Cached Status method. When the FPGA I/O Node reads the channels, the FPGA VI determines the state of the channels and caches any TRUE value until the Check Cached Status method executes. • **Channel Input Limits Fault**—Returns an array of Boolean values. A value of TRUE in any index indicates that the channel sharing a number with that index detected the channel input is out of range of Lower Fault Detection Limit and Upper Fault Detection Limit on the channel at some point after the last time that the Check Cached Status method executed. • **Channel Overcurrent**—Returns an array of Boolean values. A value of TRUE in any index indicates that the channel sharing a number with that index detected an overcurrent on the channel at some point after the last time that the Check Cached Status method executed. • **Field-Side Power Fault**—Returns a single Boolean value. A value of TRUE indicates that the module detected Field-Side Power Fault on the module at some point after the last time that the Check Cached Status method executed.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 285. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9253 acquires data.
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.
Filter Response	Sets the AI filter types supported by the device. The filters supported by this module are the Comb (default) and Butterworth filter.
Filter Order	Sets the Filter Order for the Butterworth filter. This setting would only take effect if the module is currently configured to the Butterworth filter.
Filter Frequency	Sets the AI filter's cutoff frequency.
Field-Side Power Fault Detection Enable	Enables the module to detect Field-Side Power Supplied to the module. • Set the input to TRUE to enable Field-Side Power Fault Detection. • Set the input to FALSE (default) to disable Field-Side Power Fault Detection.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Converting NI 9253 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9253 if you want the FPGA I/O Node to return calibrated, fixed-point data from the module.

When **Calibration Mode** is set to Raw, the FPGA I/O Node returns calibrated, binary data from the module. Convert the binary values to engineering units. Convert these values in the host VI.

Using an Equation to Convert Binary Current Values to Amperes

Use the following equation in the host VI to convert binary current values to amperes:

$$Current = Binary\ Value \times 2615.02\ pA/LSB$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.

NI 9253 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in amps.

Module Channels

The NI 9253 has the following channels.

Table 286. Module Channels

Channel	Description
AIx	Analog input channel x, where x is the number of the channel. For the NI 9253, x is 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9253. Right-click the NI 9253 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.
- **Data Rate**—Sets the rate at which the module acquires data.
- **Filter Response**—Sets the AI filter types supported by the device. The filters supported by this module are the Comb filter (default) and Butterworth filter.
- **Filter Order**—Sets the Filter Order for the Butterworth filter. This setting takes

effect when the module is configured to the Butterworth filter.

- **Filter Frequency**—Sets the AI filter's cutoff frequency.
- **Diagnostic Settings**—You can set the following diagnostic functions for the module.
  - **Module Settings**
    - **Field-side Power Fault Detection Enable**—Enable the module to detect Field-Side Power Supplied to the module. Select the checkbox to enable Field-Side Power Fault Detection. Uncheck the checkbox to disable (default) Field-Side Power Fault Detection.
  - **Channel Configuration**
    - **Input Limits Fault Detection Enable**—Enable the module to detect an Input Limit Fault on the corresponding channel. Select the checkbox to enable Input Limits Fault Detection. Unselect (default) to disable Input Limits Fault Detection.
    - **Lower Fault Detection Limit**—Sets the Channel's Lower Fault Detection Limit of the corresponding channel. The range of valid current limit values is 0—0.0219 A (default value is 0.0 A).
    - **Upper Fault Detection Limit**—Sets the Channel's Upper Fault Detection Limit of the corresponding channel. The range of valid current limit values is 0—0.0219 A (default value is 0.0219 A).

Module Methods

Use the Invoke Node to access the following module properties for this device.

Table 287. Module Methods

Method	Description
Check Cached Status	Returns Booleans for each channel that indicate whether there was an input limits fault, overcurrent, or field side power fault on the channel since the last execution of the Check Cached Status method. When I/O variables or shared variables read the channels, the scan driver determines the state of the channels and caches any TRUE value until the Check Cached Status method executes. • **Channel Input Limits Fault**—Returns an array of Boolean values. A value of TRUE in any index indicates that the channel sharing a number with that index detected the channel input is out of range of Lower Fault Detection Limit and Upper Fault Detection Limit on the channel at some point after the last time that the Check Cached Status method executed.

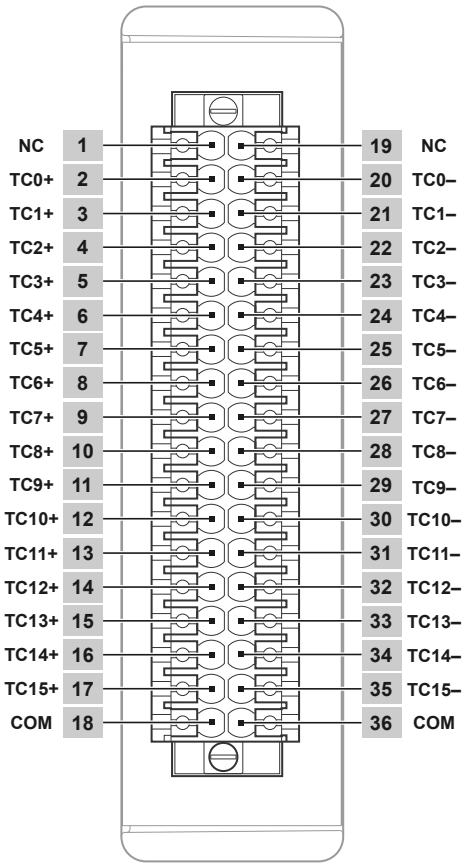
Method	Description
	• **Channel Overcurrent**—Returns an array of Boolean values. A value of TRUE in any index indicates that the channel sharing a number with that index detected an overcurrent on the channel at some point after the last time that the Check Cached Status method executed. • **Field-Side Power Fault**—Returns a single Boolean value. A value of TRUE indicates that the module detected Field-Side Power Fault on the module at some point after the last time that the Check Cached Status method executed.

NI 9313

CompactRIO 16 channel, 1 S/s per channel, ±78 mV Temperature Input Module

NI 9313 Pinout

Figure 70. NI 9313 Pinout



NI 9313 (FPGA Interface)

FPGA I/O Node

Use an **FPGA I/O Node** configured for reading with this device.

Terminals in Software

To access the following terminals for this device, use the FPGA I/O Node.

Table 288. Terminals in Software

Terminal	Description
TC*x*	Thermocouple input channel *x*, where *x* is the number of the channel. The NI 9313 has TC channels 0 to 15.
Autozero	Autozero channel. For the best accuracy, read the Autozero channel in the same FPGA I/O Node as the thermocouple input channels.
CJC	Cold-junction compensation channel. For the best accuracy, read the CJC channel in the same FPGA I/O Node as the thermocouple input channels. Convert the CJC data to temperature.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Module Methods

To access the following method for this device, use the FPGA I/O Method Node.

Table 289. Module Methods

Method	Description
Check Cached Status	Returns a Boolean value for each channel that indicates whether the channel was out of range or had an open thermocouple since the last execution of the Check Cached Status method. When the FPGA I/O Node reads the channels, the FPGA VI determines the state of the channels and caches any *TRUE* value until the Check Cached Status method executes. • **Force Status Read**—When the value of this input is FALSE, the method

Method	Description
	returns the cached status information since the last time that the Check Cached Status method executed. When the value is *TRUE*, the method forces the FPGA I/O Node to read all channels and update the status information. Forcing a status read can introduce jitter into an analog input loop. • **Out of Range**—Returns an array of Boolean values. A value of *TRUE* in any index indicates that the channel sharing a number with that index exceeded the common-mode voltage range at some point after the last time that the Check Cached Status method executed. • **Open**—Returns an array of Boolean values. A value of *TRUE* in any index indicates that the channel sharing a number with that index detected an open thermocouple on the channel at some point after the last time that the Check Cached Status method executed.

I/O Properties

This device does not support any I/O properties.

Module Properties

To access the following module properties for this device, use the FPGA I/O Property Node.

Table 290. Module Properties

Property	Description
Module ID	Returns the module ID, 0x7BA8.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Converting NI 9313 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box

for the NI 9313 if you want the FPGA I/O Node to return calibrated, fixed-point data from the module in units of volts. If you set the **Calibration Mode** to **Calibrated**, you must convert the CJC data from voltage to temperature.

Set the **Calibration Mode** to **Raw** if you want the FPGA I/O Node to return calibrated, binary data from the module. If you set the **Calibration Mode** to **Raw**, you must convert the binary thermocouple and CJC values to voltage and then convert the CJC data from voltage to temperature. You must convert these values in the host VI.

Using a VI to Convert Data to Temperature

Refer to the NI 9313 Convert to Temperature polymorphic VI in the `labview\examples\CompactRIO\Module Specific\NI 9313\NI 9313 Getting Started\NI 9313 Getting Started.lvproj` for an example of converting calibrated or raw data to temperature. Use the `NI 9313 Convert to Temperature` VI as a subVI in the host VI.

Using an Equation to Convert Binary Values to Voltage

Use the following equation in the host VI to convert the binary thermocouple and CJC values to voltage:

$$Voltage = Binary\ Value \times 78.125\ mV \div 8{,}388{,}607$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.

Using Equations to Convert CJC Data from Voltage to Temperature

Use the following equations in the host VI to convert CJC data from volts to temperature:

Calculate the resistance of the thermistor:

$$R_T = (10000 \times CJC\ Data \times 32) \div (2.5 - CJC\ Data \times 32)$$

Calculate the CJC temperature:

$$T = [\ 1 \div [A + B(\ln(R_T)) + C(\ln(R_T))^3]] - (273.15 + OffsetConstant)$$

Where

- T = temperature in °C
- $A = 1.2873851 \times 10^{-3}$
- $B = 2.3575235 \times 10^{-4}$
- $C = 9.4978060 \times 10^{-8}$
- R_T = thermistor resistance reading
- $OffsetConstant$[14] = 1

Refer to the National Institute of Standards and Technology (NIST) Monograph 175 thermocouple reference tables for more information about converting and adjusting thermocouple values.

NI 9313 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in units as specified on the C Series Module Properties dialog box.

Note The Scan Interface applies Autozero and cold-junction compensation to NI 9313 data returned in temperature units.

Module Channels

The NI 9313 has the following channels.

14. The *OffsetConstant* is the typical temperature gradient between the CJC sensor and the thermocouple cold junction.

Table 291. Module Channels

Channel	Description
TC*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9313, *x* is 0 to 15.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. To display this dialog box, right-click the module in the **Project Explorer** window and select **Properties**. You can configure the following options.

- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.
- **Channels**—Specifies the channel(s) for which you want to configure settings.
- **Thermocouple Type**—Specifies the type of thermocouple connected to the channel.
- **Measurement Units**—Specifies the units you want data to be returned in for the channel. You can select Raw Volts, Degrees Kelvin, Degrees Celsius, Degrees Fahrenheit, or Degrees Rankine.
- **Enable Open Thermocouple Detection**—If this box is checked, the NI 9313 returns a zero value and reports an error if it detects an open thermocouple on any channel. This box is checked by default. If you do not check this box, each channel of the NI 9313 independently returns full-scale data and the NI 9313 does not report an error if an open thermocouple is detected.

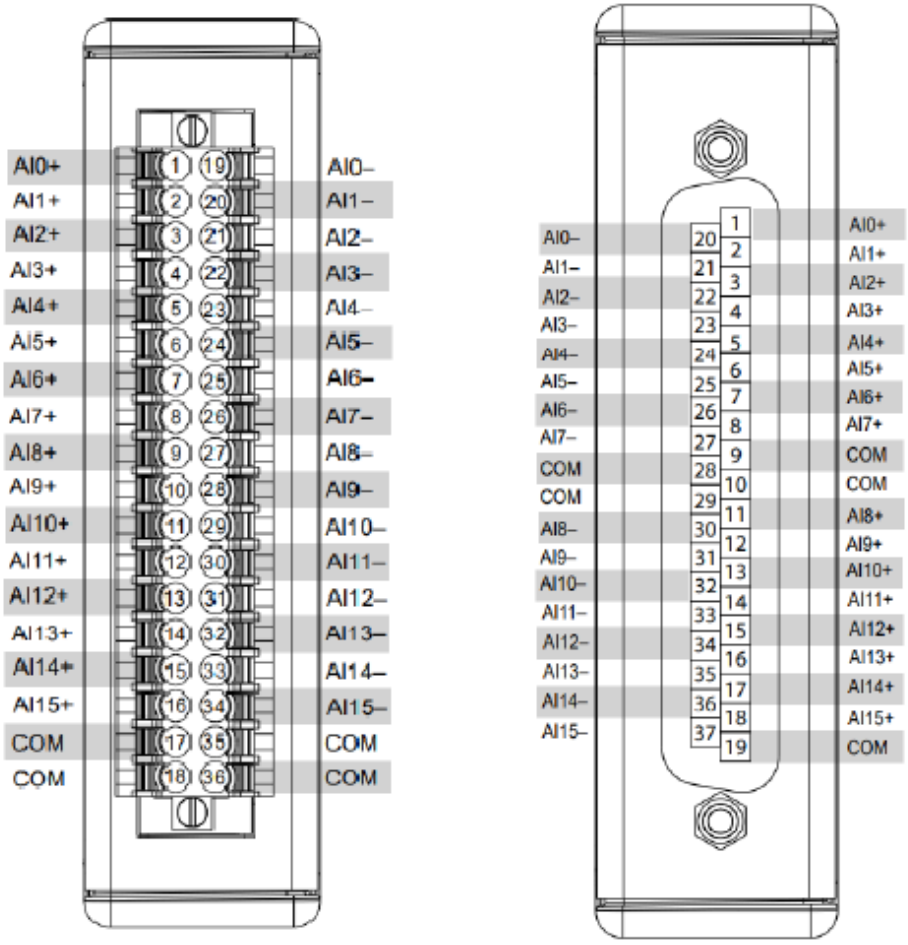
> **Note** Enabling or disabling open thermocouple detection on the NI 9313 has no effect on the accuracy of the measurement because this option on the module is implemented entirely from software.

NI 9320

CompactRIO 16-Channel, ±10 V, 16-Bit Simultaneous Analog Input Module

NI 9320 Pinout

Figure 71. NI 9320 Pinout



NI 9320 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Note You can simultaneously read from multiple channels on the NI 9320. Refer to *Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)*.

User-Controlled I/O Sampling

You can use the following User-Controlled I/O Sampling functions to perform I/O with

more specific control over the I/O hardware on the FPGA.

Table 292. User-Controlled I/O Sampling Functions

Function	Type
Generate I/O Sample Pulse Method	Module function
Get I/O Read Status Method	I/O function
Read I/O Method	I/O function
Reset I/O Method	Module function

Note Call the Reset I/O Method function first to prepare the NI 9320 to use the other User-Controlled I/O Sampling functions.

Terminals in Software

Use the FPGA I/O Node or User-Controlled I/O Sampling functions to access the following terminals for this device.

Table 293. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9320 has AI channels 0 to 15.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the **FPGA I/O Method Node** to access the following module methods for this

device.

Table 294. Module Methods

Method	Description
Read Module Temperature	Reads the module temperature, represented as -40 °C to 125 °C.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 295. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)

Converting NI 9320 Data (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the *NI 9320* if you want the FPGA I/O Node to return calibrated, fixed-point data from the module in units of volts.

Set the **Calibration Mode** to **Raw** if you want the FPGA I/O Node to return calibrated, binary data from the module. If you set the **Calibration Mode** to **Raw**, you must convert the binary values to engineering units in the host VI. Use the following equations to convert binary voltage values to volts in the host VI.

$$Volts = Binary\ Value \times 320,435 \frac{nV}{LSB}$$

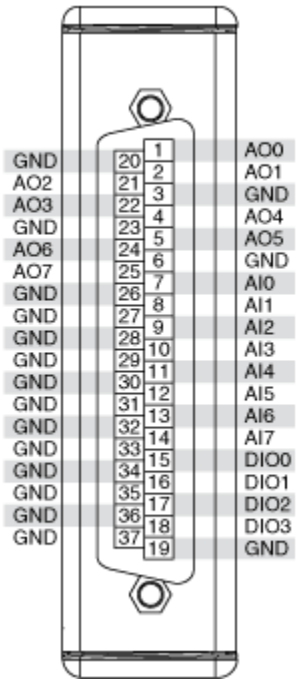
where `Binary Value` is the value returned by the FPGA I/O Node.

NI 9381

CompactRIO 0 V to 5 V, 12-bit, Multifunction Input/Output Module

NI 9381 Pinout

Figure 72. NI 9381 Pinout



Running AI and AO Operations Concurrently on the NI 9381

AI and AO operations that are run concurrently may exhibit jitter. Access both subsystems using a single I/O Node to avoid jitter in your application. Refer to the *Avoiding Timing Uncertainty* topic for more information.

NI 9381 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading and writing with this device.

Table 296. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9381 has AI channels 0 to 7.
AO*x*	Analog output channel *x*, where *x* is the number of the channel. The NI 9381 has AO channels 0 to 3.
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. The NI 9381 has DIO channels 0 to 3.
DIO3:0	Digital port consisting of channels 0 through 3. Channel 3 is returned in the MSB, and channel 0 is returned in the LSB.

Caution AI and AO operations that are run concurrently may exhibit jitter. Access both subsystems using a single I/O Node to avoid jitter in your application. Refer to *Avoiding Timing Uncertainty* for more information.

Arbitration

Analog input and analog output channels of this device support only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for analog input and analog output channels of this device.

You can configure the arbitration settings for digital output channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

Digital input channels of this device support only the **Never Arbitrate** option for arbitration. You cannot configure arbitration settings for the digital input channels of this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for the DIO channels of this device.

Table 297. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node (FPGA Module) topic for a description of this method. This method is available only when you set the DIO line direction to output. You can set the DIO line direction in the C Series Module Properties dialog box.
Set Output Enable	Sets the line direction of the digital channel or the DIO3:0 digital port. Refer to the FPGA I/O Method Node (FPGA Module) topic for more information on this method. This method can introduce jitter in an analog input or analog output loop. This method is available only when you enable programmable DIO line changes on the module. Enable programmable DIO line changes on the module in the C Series Module Properties dialog box.
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge.

Method	Description
	• A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Wait on Rising	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal.

Method	Description
Edge	The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.

Module Method

Use the FPGA I/O Method Node to access the following module method for this device.

Table 298. Module Method

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for AI and AO channels of this device.

Table 299. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in nV/LSB for the channel. Use this value to convert and calibrate NI 9381 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert and calibrate NI 9381 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 300. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop.

- Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.
- Configure the number of output synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box.

This device supports the Number of Synchronizing Registers for Output Data synchronizing register option when used in SCTL output. This option supports the same functionality as the **Number of Synchronizing Registers for Output Data** option described in the Advanced Code Generation FPGA I/O Properties Page (FPGA Module) topic, with the exception that you can use this option only in a single-cycle Timed Loop. You can implement either 0 or 1 synchronizing registers inside the single-cycle Timed Loop. I you configure 0 synchronizing registers outside of the single-cycle Timed Loop, the FPGA VI implements 1 synchronizing register by default.

When the device is within a single-cycle Timed Loop, it must be ready to perform digital I/O before a loop containing digital I/O starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready. The module also might ignore or delay digital output operations if it is not ready.

While the digital is performing digital I/O within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Performing any of these actions causes the module to be unable to perform digital I/O and the **Ready** output of the Check Status method to return *FALSE*.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as:

- 40 MHz
- 80 MHz
- 120 MHz

Related concepts:

- Converting Voltage Values to Binary Values for the NI 9381 (FPGA Interface)
- Reading from C Series Channels
- Writing to C Series Channels
- Configuring the Line Direction for the NI 9402 (FPGA Interface)
- Converting and Calibrating CompactRIO Analog Input Values (FPGA Interface)

Related information:

- FPGA I/O Node Properties Dialog Box (Advanced Code Generation Page)
- FPGA I/O Properties Dialog Box (Advanced Code Generation Page)

Avoiding Timing Uncertainty with the NI 9381 (FPGA Interface)

Follow the guidelines below to avoid introducing timing uncertainty into AI node, AO Node, DO Node, and Set Output Data method calls. Failing to follow these guidelines may result in lost data points for DO Node calls or Set Output Data method calls when either of them are made from inside a single-cycle Timed Loop.

- Use a single I/O Node to access AI and AO operations to ensure proper sequencing.
- Do not run the Set Output Enable method when an AI, AO, or DO subsystems is active.
- Do not perform the following operations concurrently:
  - AI Node call

- AO Node call
- Set Output Enable method call
- Automatic line direction change when a DO call is made under the following conditions:
  - The call is made from outside of a single-cycle Timed Loop.
  - The **Allow programmatic DIO line direction** checkbox is enabled in the C Series Module Properties dialog box.
  - The DIO channel is configured as an input at the time when the DO node is called.

For an example of the recommended way to use the AI, AO , and DIO subsystems, refer to the NI 9381 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9381\NI 9381 Advanced IO\NI 9381 Advanced IO.lvproj`.

Converting Voltage Values to Binary Values for the NI 9381 (FPGA Interface)

Set the **Calibration Mode** to Calibrated in the C Series Module Properties dialog box for the NI 9381 if you want the FPGA I/O Node to accept fixed-point data in units of volts when writing to the module.

When **Calibration Mode** is set to Raw when writing to the module, the FPGA I/O Node accepts only binary values. Convert output voltage values to binary values before writing them to the module. Convert these values in the host VI.

Using an Equation to Convert Voltage to Binary

Use the following equation in the host VI to convert the analog output values to binary values:

$$Binary\ Value = (Voltage\ Value - Offset) \div LSB\ Weight$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.
- *Voltage Value* is the voltage in *V* that you want the channel to output
- *Offset* is the value returned by the Offset property
- *LSB Weight* is the value returned by the LSB Weight property

NI recommends using calibrated values for analog output. To convert calibrated voltage values, use the FPGA I/O Property Node to read the LSB Weight and Offset properties. If you do not want to read the LSB Weight and Offset values from the module, convert uncalibrated voltage values by using the following values for *Offset* and *LSB Weight*:

$$Offset = -5249$$

$$LSB\ Weight = 5.105\ V \div 2^{DAC\ Resolution}$$

Where

- *DAC Resolution* is the DAC resolution value. Refer to the NI 9381 hardware documentation for the DAC resolution value.

Related concepts:

- NI 9381 (FPGA Interface)

NI 9381 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI.

- The I/O variables for the AI channels return floating-point data in volts.
- The I/O variables for the AO channels write floating-point data in volts.
- The I/O variables for the DIO channels read and write Boolean values.

Module Channels

The NI 9381 has the following channels.

Table 301. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9381, *x* is 0 to 7.
AO*x*	Analog output channel *x*, where *x* is the number of the channel. For the NI 9381, *x* is 0

Channel	Description
	to 7.
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. For the NI 9381, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9381. Right-click the NI 9381 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- Channels—Specifies the channel(s) for which you want to configure settings.
- Direction—Specifies the initial line direction of the selected channel(s) as input or output.

Note If you are using direct access to read from or write to AI or AO channels of the NI 9381, the conversion time is somewhat longer than 50 µs.

Analog Output Modules

Table 302. C Series Analog Output Modules

C Series Analog Output Module	Description
NI-9260	51.2 kS/s/ch Simultaneous, 3 Vrms, 2-Channel C Series Voltage Output Module
NI-9262	1 MS/s/ch Simultaneous, ±10 V, 6-Channel C Series Voltage Output Module
NI-9263	100 kS/s/ch Simultaneous, ±10 V, 4-Channel C Series Voltage Output Module
NI-9264	25 kS/s/ch Simultaneous, ±10 V, 16-Channel C Series Voltage Output Module
NI-9265	4-Channel C Series Current Output Module

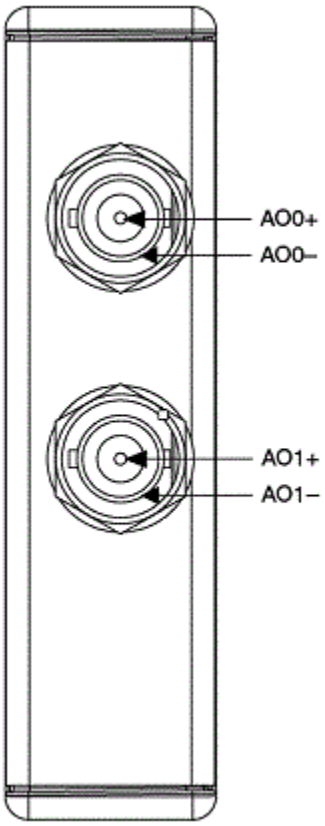
C Series Analog Output Module	Description
NI-9266	8-Channel C Series Current Output Module
NI-9269	100 kS/s/ch Simultaneous, ±10 V, Isolated, 4-Channel C Series Voltage Output Module
NI-9381	8 AI, 8 AO, 4 DIO, 0 V to 5 V, C Series Multifunction I/O Module

NI 9260

CompactRIO 2-Channel, 3 Vrms, 24-Bit Single-Ended Analog Output Module

NI 9260 Pinout

Figure 73. NI 9260 Pinout



NI 9260 (FPGA Interface)

FPGA I/O Node

You can use an FPGA I/O Node, configured for reading, with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 303. Terminals in Software

Terminal	Description
AOx	Analog output channel x, where x is the number of the channel. The NI 9260 has channels 0 to 1. Do not access AO channels on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Start	Channel that controls when the NI 9260 starts outputting data. If TRUE is written to the **Start** channel, the NI 9260 starts outputting data. The AOx will be reset to 0 volts every time this I/O node is executed. When the NI 9260 is outputting data, you must write TRUE to the **Stop** channel before you can access properties. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9260 stops outputting data. If TRUE is written to the **Stop** channel, the NI 9260 stops outputting data. When the NI 9260 is outputting data, you must write TRUE to the **Stop** channel before you can access properties. If FALSE is written to the **Stop** channel, no operation is performed.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device contains no I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following I/O properties for this device.

Table 304. Module Methods

Method	Description
Check Cached Status	Returns Booleans for each channel that indicates whether there was underflow or overvoltage on the channel since the last execution of the Check Cached Status method. When the FPGA I/O Node reads the channels, the FPGA VI determines the state of the channels and caches any TRUE value until the Check Cached Status method executes. • **Underflow**—Returns an array of Boolean values. A value of TRUE in any index indicates that the channel sharing a number with that index has missed one or more output samples at some point after the last time that the Check Cached Status method executed. • **Overvoltage**—Returns an array of Boolean values. A value of TRUE in any index indicates that the channel sharing a number with that index detected an overvoltage on the channel at some point after the last time that the Check Cached Status method executed.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 305. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in nV/LSB for the channel. Use this value to convert NI 9260 data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert NI 9264 data if you set the **Calibration Mode** to **Raw** in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 306. Module Properties

Property	Description
Data Rate	Sets the rate at which the NI 9260 outputs data.
Module ID	Returns the module ID, 0x7786 for BNC, 0x778C for mini-XLR.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Excitation Protection

The NI 9260 excitation circuit is protected from overvoltage fault conditions. The circuit is automatically disabled in the event of a fault condition. Whenever possible, a channel automatically recovers after the fault is removed. You must wire error terminals on the FPGA I/O Property Nodes to receive notification of overvoltage faults. LabVIEW returns *warning 65548* if there is an overvoltage fault on at least one channel. Only the channel(s) with the fault are affected and all other channels on the module continue to function properly without interruption.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Converting Voltage Values to Binary Values for the NI 9260 (FPGA Interface)

Set the **Calibration Mode** to Calibrated in the *C Series Module Properties* dialog box for the NI 9260 if you want the FPGA I/O Node to accept fixed-point data in units of volts when writing to the module. If you set the Calibration Mode to Raw, the FPGA I/O Node accepts only binary values when writing to the module. You must convert output voltage values to binary values before you write them to the module. You must convert these values in the host VI.

Note The equations below apply only to the NI 9260.

Using an Equation to Convert Voltage to Binary

You can use the following equation in the host VI to convert the analog output values to binary:

$$Binary\ Value = (Voltage\ Value - (Offset \times 10^{-9})) \times (LSB\ Weight \times 10^{-12})$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.
- *Voltage Value* is the voltage in *V* that you want the channel to output
- *Offset* is the value returned by the Offset property
- *LSB Weight* is the value returned by the LSB Weight property

NI recommends using calibrated values for analog output. To convert calibrated voltage values, use the FPGA I/O Property Node to read the LSB Weight and Offset properties. If you do not want to read the LSB Weight and Offset values from the module, you can convert uncalibrated voltage values by using the following values for *Offset* and *LSB Weight*:

$$Offset = 0$$

$$LSB\ Weight = Typical\ Output\ Span \div 2^{DAC\ Resolution} \times 10^{12}$$

Where

- *Typical Output Span* is 8.71 V for the NI 9260
- *DAC Resolution* is the DAC resolution of the module. Refer to the module hardware documentation for this value.

Outputting Data from an NI 9260 (FPGA Interface)

The NI 9260 has two analog output channels that simultaneously output at the data rate for which you have configured the module.

Use the **Start** and **Stop** channels of the NI 9260 to put the module in and out of data mode. In data mode, the NI 9260 can only output data. You can use the FPGA I/O Node to write the data to the module. You cannot access properties while the module is in data mode.

Note You must create FPGA I/O items for the NI 9260 before you can

> configure the items using the FPGA I/O Node.

Putting the Module in Data Mode

Configure an FPGA I/O Node with the **Start** channel of the NI 9260. Wire a Boolean constant set to TRUE to the **Start** input of the FPGA I/O Node to send a synchronization pulse to the module. The AOx will be reset to 0 volts every time the device is put back into data mode.

Writing Data to the Module

When the module is in data mode, you can use an FPGA I/O Node to write data to the module. You can connect the AO input of the FPGA I/O Node to various types of functions, including an FPGA Memory function or an FPGA FIFO function. If you write to multiple channels on the module, place the channels in the same FPGA I/O Node to ensure that the VI writes the data synchronously.

Because the NI 9260 internally outputs data at a specified rate, the FPGA I/O Node does not finish until new data has been written to the module. If the NI 9260 is not in data mode while an FPGA I/O Node is running, the FPGA I/O Node returns an I/O Not Started error.

Exiting Data Mode

Configure an FPGA I/O Node with the **Stop** channel of the NI 9260. Write a TRUE to the **Stop** input. The module is no longer in data mode and you can now access NI 9260 properties.

NI recommends you ramp down to 0 volts before exiting data mode to prevent overshooting and ringing of the outputs due to the **Start** behavior. See **Putting the Module in Data Mode** above for the **Start** behavior.

Related concepts:

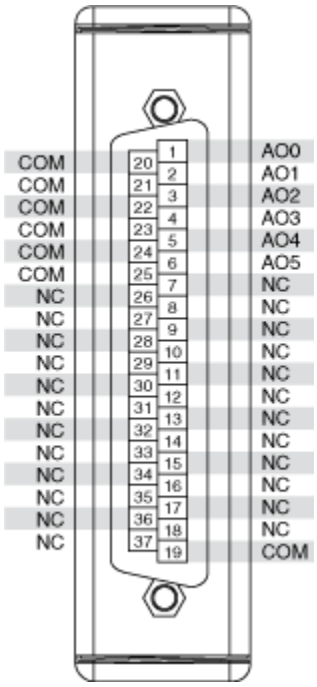
- Configuring the Data Rate for a Module (FPGA Interface)

NI 9262

CompactRIO 1 MS/s/ch Simultaneous, ±10 V, 6-Channel C Series Voltage Output Module

NI 9262 Pinout

Figure 74. NI 9262 Pinout



NI 9262 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for writing with this device.

Note You can simultaneously write to multiple channels on the NI 9262.

User-Controlled I/O Sampling

You can use the following User-Controlled I/O Sampling functions to perform I/O with more specific control over the I/O hardware on the FPGA. For information about these functions, refer to the LabVIEW FPGA Module Programming Reference Manual.

Table 307. User-Controlled I/O Sampling Functons

Function	Type
Generate I/O Sample Pulse Method	Module function
Get I/O Write Status Method	I/O function
Write I/O Method	I/O function
Reset I/O Method Method	Module function

Note You must call the Reset I/O Method function first to prepare the NI 9262 to use the other User-Controlled I/O Sampling functions.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 308. Terminals in Software

Terminal	Description
AO*x*	Analog output channel *x*, where *x* is the number of the channel. The NI 9262 has AO channels 0 to 5.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Methods

This device does not support any methods.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 309. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in nV/LSB for the channel. Use this value to convert NI 9262 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert NI 9262 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 310. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Converting Voltage Values to Binary Values for the NI 9262/9263/9264/9269 (FPGA Interface)
- Writing to C Series Channels
- Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)

Converting Voltage Values to Binary Values for the NI 9262/9263/9264/9269 (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9262/NI 9263/NI 9264/NI 9269 if you want the FPGA I/O Node to accept fixed-point data in units of volts when writing to the module.

If you set the **Calibration Mode** to Raw, the FPGA I/O Node accepts only binary values when writing to the module. Convert output voltage values to binary values before you write them to the module. Convert these values in the host VI.

Using an Equation to Convert Voltage to Binary

You can use the following equation in the host VI to convert the analog output values to binary values:

$$Binary\ Value = (Voltage\ Value \times 10^{9} - Offset) \div LSB\ Weight$$

Where

- *Binary Value* is the value you write to the FPGA I/O Node
- *Voltage Value* is the voltage in *V* that you want the channel to output
- *Offset* is the value returned by the Offset property
- *LSB Weight* is the value returned by the LSB Weight property.

NI recommends using calibrated values for analog output. To convert calibrated voltage values, use the FPGA I/O Property Node to read the LSB Weight and Offset properties.

If you do not want to read the LSB Weight and Offset values from the module, convert uncalibrated voltage values by using the following values for *Offset* and *LSB Weight*:

$$Offset = 0$$

$$LSB\ Weight = Typical\ Output\ Span \div 2^{DAC\ Resolution} \times 10^{9}$$

Where

- *Typical Output Span* is 21.4 V for the NI 9263, 21 V for the NI 9264, and 20.98 V for the NI 9269.
- *DAC Resolution* is the DAC resolution of the module. Refer to the module hardware documentation for this value.

Related concepts:

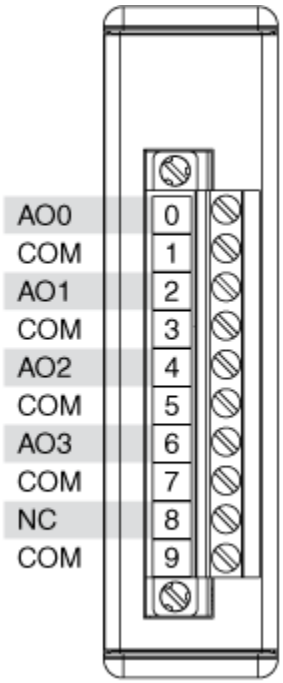
- NI 9262 (FPGA Interface)
- NI 9263 (FPGA Interface)
- NI 9264 (FPGA Interface)
- NI 9269 (FPGA Interface)

NI 9263

CompactRIO 4-Channel, ±10 V, 16-Bit Simultaneous Analog Output Module

NI 9263 Pinout

Figure 75. NI 9263 Pinout



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9263 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for writing with this device.

Note You can simultaneously write to multiple channels on the NI 9263.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 311. Terminals in Software

Terminal	Description
AO*x*	Analog output channel *x*, where *x* is the number of the channel. The NI 9263 has AO channels 0 to 3.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following module methods for this device.

Use these three methods in conjunction with each other.

- The Write Data method stores data values before writing the data to a channel,
- The Update method writes the stored data values to the output channel.
- The Wait for Update method waits until the Update method updates the output channels.

Refer to the NI 9263 Advanced Write Methods VI in the `labview\examples\CompactRIO\Module Specific\NI 9263\NI 9263 Advanced Write Methods\NI 9263 Advanced Write Methods.lvproj` for an example of using these methods.

Table 312. Module Methods

Method	Description
Update	Writes stored data values to the output channel. • **Overrun**—Returns a Boolean value. A value of TRUE indicates an overrun warning. The overrun warning means that the Update method is trying to run while the Write Data method is running, typically in a different loop. Either the Write Data method is writing data to the module too quickly or the Update method is running too slowly. • **Underflow**—Returns a Boolean value. A value of TRUE indicates an underflow warning. The underflow warning means that the Write Data method has not written new data to the module since the last time the Update method ran. Either the Write Data method is writing data to the module too slowly or the Update method is running too quickly.
Wait for Update	Waits until the Update method updates the output channels. • **Timeout (Ticks)**—Specifies in FPGA clock ticks how long the Wait for Update method waits for the Update method to update the output channels. A value of 0 causes the Wait for Update method to time out immediately. A negative value causes the Wait for Update method to wait indefinitely. A positive value causes the Wait for Update method to wait for that number of clock ticks before timing out. • **Timeout Occurred?**—Returns a Boolean value. A value of TRUE indicates that a timeout occurred.
Write Data	Stores data values before writing the data to a channel. • **Channel**—Specifies the channel to which you want to write data. • **Data**—Specifies the data you want to write to the channel.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 313. I/O Properties

Property	Description
LSB	Returns the LSB weight in nV/LSB for the channel.

Property	Description
Weight	Use this value to convert NI 9263 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert NI 9263 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 314. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Converting Voltage Values to Binary Values for the NI 9262/9263/9264/9269 (FPGA Interface)
- Writing to C Series Channels
- Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)

Converting Voltage Values to Binary Values for the NI 9262/9263/9264/9269 (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9262/NI 9263/NI 9264/NI 9269 if you want the FPGA I/O Node to accept fixed-point data in units of volts when writing to the module.

If you set the **Calibration Mode** to Raw, the FPGA I/O Node accepts only binary values when writing to the module. Convert output voltage values to binary values before you write them to the module. Convert these values in the host VI.

Using an Equation to Convert Voltage to Binary

You can use the following equation in the host VI to convert the analog output values to binary values:

$$Binary\ Value = (Voltage\ Value \times 10^{9} - Offset) \div LSB\ Weight$$

Where

- *Binary Value* is the value you write to the FPGA I/O Node
- *Voltage Value* is the voltage in *V* that you want the channel to output
- *Offset* is the value returned by the Offset property
- *LSB Weight* is the value returned by the LSB Weight property.

NI recommends using calibrated values for analog output. To convert calibrated voltage values, use the FPGA I/O Property Node to read the LSB Weight and Offset properties.

If you do not want to read the LSB Weight and Offset values from the module, convert uncalibrated voltage values by using the following values for *Offset* and *LSB Weight*:

$$Offset = 0$$

$$LSB\ Weight = Typical\ Output\ Span \div 2^{DAC\ Resolution} \times 10^{9}$$

Where

- *Typical Output Span* is 21.4 V for the NI 9263, 21 V for the NI 9264, and 20.98 V for the NI 9269.
- *DAC Resolution* is the DAC resolution of the module. Refer to the module hardware documentation for this value.

NI 9263 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables write floating-point values to the channels in volts.

Module Channels

The NI 9263 has the following channels.

Table 315. Module Channels

Channel	Description
AO*x*	Analog output channel *x*, where *x* is the number of the channel. For the NI 9263, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

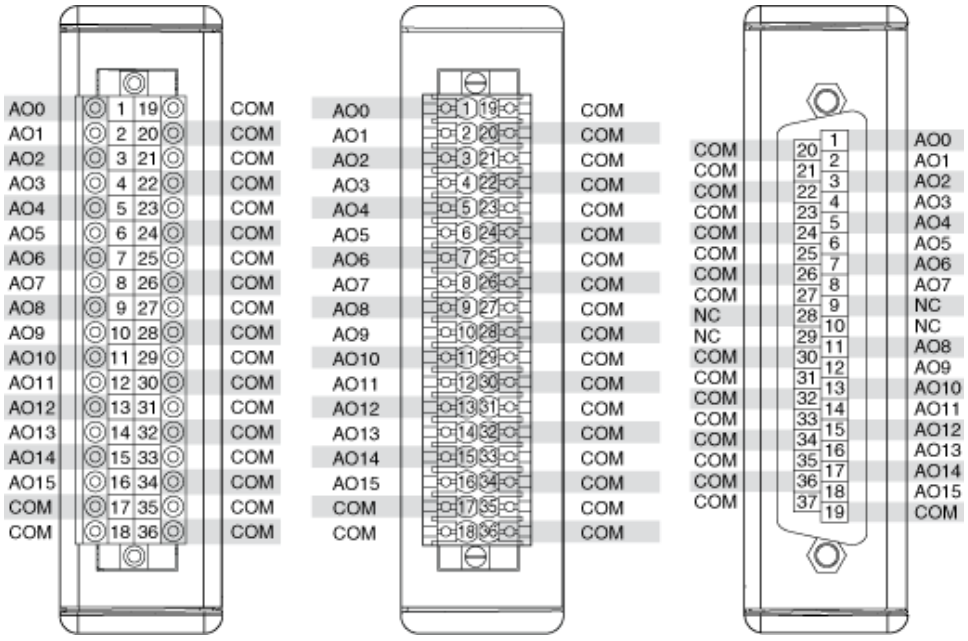
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9264

CompactRIO 16-Channel, ±10 V, 16-Bit Simultaneous Analog Voltage Output Module

NI 9264 Pinout

Figure 76. NI 9264 Pinout



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9264 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for writing with this device.

Note You can simultaneously write to multiple channels on the NI 9264.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 316. Terminals in Software

Terminal	Description
AO*x*	Analog output channel *x*, where *x* is the number of the channel. The NI 9264 has AO channels 0 to 15.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following module methods for this device.

Use these three methods in conjunction with each other.

- The Write Data method stores data values before writing the data to a channel,
- The Update method writes the stored data values to the output channel.
- The Wait for Update method waits until the Update method updates the output channels.

Refer to the NI 9264 Advanced Write Methods VI in the `labview\examples\CompactRIO\Module Specific\NI 9264\NI 9264 Advanced Write Methods\NI 9264 Advanced Write Methods.lvproj` for an example of using these methods.

Table 317. Module Methods

Method	Description
Update	Writes stored data values to the output channel. • **Overrun**—Returns a Boolean value. A value of TRUE indicates an overrun warning. The overrun warning means that the Update method is trying to run while the Write Data method is running, typically in a different loop. Either the Write Data method is writing data to the module too quickly or the Update method is running too slowly. • **Underflow**—Returns a Boolean value. A value of TRUE indicates an underflow warning. The underflow warning means that the Write Data method has not written new data to the module since the last time the Update method ran. Either the Write Data method is writing data to the module too slowly or the

Method	Description
	Update method is running too quickly.
Wait for Update	Waits until the Update method updates the output channels. • **Timeout (Ticks)**—Specifies in FPGA clock ticks how long the Wait for Update method waits for the Update method to update the output channels. A value of 0 causes the Wait for Update method to time out immediately. A negative value causes the Wait for Update method to wait indefinitely. A positive value causes the Wait for Update method to wait for that number of clock ticks before timing out. • **Timeout Occurred?**—Returns a Boolean value. A value of TRUE indicates that a timeout occurred.
Write Data	Stores data values before writing the data to a channel. • **Channel**—Specifies the channel to which you want to write data. • **Data**—Specifies the data you want to write to the channel.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 318. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in nV/LSB for the channel. Use this value to convert module data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert module data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 319. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Converting Voltage Values to Binary Values for the NI 9262/9263/9264/9269 (FPGA Interface)
- Writing to C Series Channels
- Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)

Converting Voltage Values to Binary Values for the NI 9262/9263/9264/9269 (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9262/NI 9263/NI 9264/NI 9269 if you want the FPGA I/O Node to accept fixed-point data in units of volts when writing to the module.

If you set the **Calibration Mode** to Raw, the FPGA I/O Node accepts only binary values when writing to the module. Convert output voltage values to binary values before you write them to the module. Convert these values in the host VI.

Using an Equation to Convert Voltage to Binary

You can use the following equation in the host VI to convert the analog output values to binary values:

$$Binary\ Value = (Voltage\ Value \times 10^{9} - Offset) \div LSB\ Weight$$

Where

- *Binary Value* is the value you write to the FPGA I/O Node
- *Voltage Value* is the voltage in *V* that you want the channel to output
- *Offset* is the value returned by the Offset property
- *LSB Weight* is the value returned by the LSB Weight property.

NI recommends using calibrated values for analog output. To convert calibrated voltage values, use the FPGA I/O Property Node to read the LSB Weight and Offset properties.

If you do not want to read the LSB Weight and Offset values from the module, convert uncalibrated voltage values by using the following values for *Offset* and *LSB Weight*:

$$Offset = 0$$

$$LSB\ Weight = Typical\ Output\ Span \div 2^{DAC\ Resolution} \times 10^{9}$$

Where

- *Typical Output Span* is 21.4 V for the NI 9263, 21 V for the NI 9264, and 20.98 V for the NI 9269.
- *DAC Resolution* is the DAC resolution of the module. Refer to the module hardware documentation for this value.

NI 9264 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables write floating-point values to the channels in volts.

Module Channels

The NI 9264 has the following channels.

Table 320. Module Channels

Channel	Description
AO*x*	Analog output channel *x*, where *x* is the number of the channel. For the NI 9264, *x* is 0 to 15.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

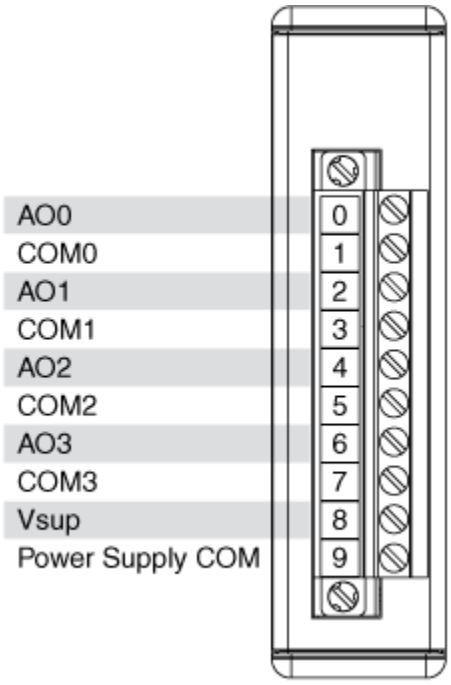
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9265

CompactRIO 4-Channel, 0–20 mA, 16-Bit Simultaneous Analog Current Output Module

NI 9265 Pinout

Figure 77.



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9265 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for writing with this device.

Note You can simultaneously write to multiple channels on the NI 9265.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 321. Terminals in Software

Terminal	Description
AO*x*	Analog output channel *x*, where *x* is the number of the channel. The NI 9264 has AO

Terminal	Description
	channels 0 to 3.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following module methods for this device.

Refer to the NI 9265 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9265\NI 9265 Getting Started\NI 9265 Getting Started.lvproj` for an example of using the Check Output Status methods.

Use the Update, Wait for Update and Write Data methods in conjunction with each other.

- The Write Data method stores data values before writing the data to a channel,
- The Update method writes the stored data values to the output channel.
- The Wait for Update method waits until the Update method updates the output channels.

Refer to the NI 9264 Advanced Write Methods VI in the `labview\examples\CompactRIO\Module Specific\NI 9264\NI 9264 Advanced Write Methods\NI 9264 Advanced Write Methods.lvproj` for an example of using the Update, Wait for Update, and Write Data methods.

Table 322. Module Methods

Method	Description
Check Cached Output Status	Returns Booleans that indicate whether the module or channels reported a power supply fault or an open current loop since the last execution of the Check Cached Output Status method. When the FPGA I/O Node communicates with the module, the FPGA VI samples the states of the module and channels and caches any TRUE value until the Check Cached Output Status method executes. • **Force Status Read**—When the value of this input is FALSE, the method returns the cached status information since the last time that the Check Cached Output Status method executed. When the value is TRUE, the method forces the FPGA I/O Node to communicate with the module and update the status information. Forcing a status read can introduce jitter into an analog output loop. • **Any Fault**—Returns a Boolean value. A value of TRUE indicates that the module has a power supply fault or at least one channel has an open current loop. • **Power Supply Fault**—Returns a Boolean value. A value of TRUE indicates that the external power supply is out of the expected range. • **Open Current Loop**—Returns an array of Boolean values. A value of TRUE in any index indicates that the channel sharing a number with that index has an open current loop and is configured to output a nonzero current value.
Check Output Status	Returns Booleans that indicate whether the module or channels reported a power supply fault or an open current loop during the last time the FPGA I/O Node communicated with the module. When the FPGA I/O Node communicates with the module, the FPGA VI samples the states of the module and channels and overwrites the previous output status with the latest state of the module and channels. • **Force Status Read**—When the value of this input is FALSE, the method returns the status information from the last time the FPGA I/O Node communicated with the module. When the value is TRUE, the method forces the FPGA I/O Node to communicate with the module and update the status information. Forcing a status read can introduce jitter into an analog output loop. • **Any Fault**—Returns a Boolean value. A value of TRUE indicates that the module has a power supply fault or at least one channel has an open current loop. • **Power Supply Fault**—Returns a Boolean value. A value of TRUE indicates that the external power supply is out of the expected range.

Method	Description
	• **Open Current Loop**—Returns an array of Boolean values. A value of TRUE in any index indicates that the channel sharing a number with that index has an open current loop and is configured to output a nonzero current value.
Update	Writes stored data values to the output channel. • **Overrun**—Returns a Boolean value. A value of TRUE indicates an overrun warning. The overrun warning means that the Update method is trying to run while the Write Data method is running, typically in a different loop. Either the Write Data method is writing data to the module too quickly or the Update method is running too slowly. • **Underflow**—Returns a Boolean value. A value of TRUE indicates an underflow warning. The underflow warning means that the Write Data method has not written new data to the module since the last time the Update method ran. Either the Write Data method is writing data to the module too slowly or the Update method is running too quickly.
Wait for Update	Waits until the Update method updates the output channels. • **Timeout (Ticks)**—Specifies in FPGA clock ticks how long the Wait for Update method waits for the Update method to update the output channels. A value of 0 causes the Wait for Update method to time out immediately. A negative value causes the Wait for Update method to wait indefinitely. A positive value causes the Wait for Update method to wait for that number of clock ticks before timing out. • **Timeout Occurred?**—Returns a Boolean value. A value of TRUE indicates that a timeout occurred.
Write Data	Stores data values before writing the data to a channel. • **Channel**—Specifies the channel to which you want to write data. • **Data**—Specifies the data you want to write to the channel.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 323. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in pA/LSB for the channel. Use this value to convert module data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Property	Description
Offset	Returns the calibration offset in pA for the channel. Use this value to convert module data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 324. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Converting Current Values to Binary Values for the NI 9265 (FPGA Interface)
- Writing to C Series Channels
- Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)

Converting Current Values to Binary Values for the NI 9265 (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9265 if you want the FPGA I/O Node to accept fixed-point data in units of amps when writing to the module.

If you set the **Calibration Mode** to Raw, the FPGA I/O Node accepts only binary values

when writing to the module. Convert output voltage values to binary values before you write them to the module. Convert these values in the host VI.

Using an Equation to Convert Voltage to Binary

You can use the following equation in the host VI to convert the analog output values to binary values:

$$Binary\ Value = (Current\ Value \times 10^{12} - Offset) \div LSB\ Weight$$

Where

- *Binary Value* is the value you write to the FPGA I/O Node.
- *Current Value* is the current in mA that you want the channel to output.
- *Offset* is the value returned by the Offset property.
- *LSB Weight* is the value returned by the LSB Weight property.

NI recommends using calibrated values for analog output. To convert calibrated voltage values, use the FPGA I/O Property Node to read the LSB Weight and Offset properties.

If you do not want to read the LSB Weight and Offset values from the module, convert uncalibrated voltage values by using the following values for *Offset* and *LSB Weight*:

$$Offset = 0$$

$$LSB\ Weight = 20.6\ mA \div 2^{DAC\ Resolution} \times 10^{12}$$

Where

- *DAC Resolution* is the DAC resolution of the module. Refer to the module hardware documentation for this value.

Related concepts:

- NI 9265 (FPGA Interface)

NI 9265 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables write floating-point values to the channels in volts.

Module Channels

The NI 9265 has the following channels.

Table 325. Module Channels

Channel	Description
AO*x*	Analog output channel *x*, where *x* is the number of the channel. For the NI 9265, *x* is 0 to 3.

Module-Specific Errors

The NI 9265 can return the following module-specific errors.

Table 326. Module-Specific Errors

Error Code	Description
65542	One or more channels have detected an open current loop. Check the module connections.
65543	The power supply voltage level is out of range. Check the supply voltage and the module connections.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project**

Explorer window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.

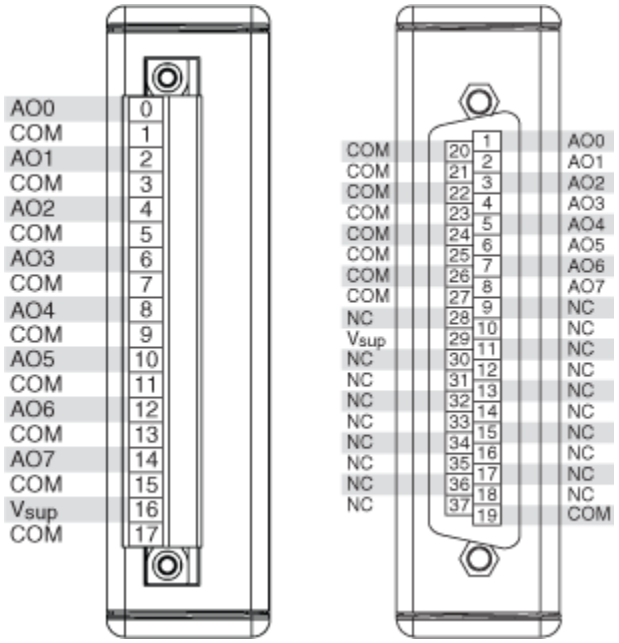
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9266

CompactRIO 8 AO, 0-20 mA, 16-Bit Simultaneous Analog Current Output Module

NI 9266 Pinout

Figure 78. NI 9266 Pinout



NI 9266 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 327. Terminals in Software

Terminal	Description
AO*x*	Analog output channel *x*, where *x* is the number of the channel. The NI 9266 has AO channels 0 to 7.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following module methods for this device.

Refer to the NI 9266 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9266\NI 9266 Getting Started\NI 9266 Getting Started.lvproj` for an example of using the Check Output Status methods.

Use the Update, Wait for Update and Write Data methods in conjunction with each other.

- The Write Data method stores data values before writing the data to a channel,
- The Update method writes the stored data values to the output channel.
- The Wait for Update method waits until the Update method updates the output channels.

Refer to the NI 9264 Advanced Write Methods VI in the `labview\examples\CompactRIO\Module Specific\NI 9264\NI 9264 Advanced Write Methods\NI 9264 Advanced Write Methods.lvproj` for an example of using the Update, Wait for Update, and Write Data methods.

Table 328. Module Methods

Method	Description
Check Cached Output Status	Returns Booleans that indicate whether the module or channels reported a power supply fault or an open current loop since the last execution of the Check Cached Output Status method. When the FPGA I/O Node communicates with the module, the FPGA VI samples the states of the module and channels and caches any TRUE value until the Check Cached Output Status method executes. • **Force Status Read**—When the value of this input is FALSE, the method returns the cached status information since the last time that the Check Cached Output Status method executed. When the value is TRUE, the method forces the FPGA I/O Node to communicate with the module and update the status information. Forcing a status read can introduce jitter into an analog output loop. • **Any Fault**—Returns a Boolean value. A value of TRUE indicates that the module has a power supply fault or at least one channel has an open current loop. • **Power Supply Fault**—Returns a Boolean value. A value of TRUE indicates that the external power supply is out of the expected range. • **Open Current Loop**—Returns an array of Boolean values. A value of TRUE in any index indicates that the channel sharing a number with that index has an open current loop and is configured to output a nonzero current value.
Check Output Status	Returns Booleans that indicate whether the module or channels reported a power supply fault or an open current loop during the last time the FPGA I/O Node communicated with the module. When the FPGA I/O Node communicates with the module, the FPGA VI samples the states of the module and channels and overwrites the previous output status with the latest state of the module and channels. • **Force Status Read**—When the value of this input is FALSE, the method returns the status information from the last time the FPGA I/O Node communicated with the module. When the value is TRUE, the method forces the FPGA I/O Node to communicate with the module and update the status information. Forcing a status read can introduce jitter into an analog output loop. • **Any Fault**—Returns a Boolean value. A value of TRUE indicates that the module has a power supply fault or at least one channel has an open current loop. • **Power Supply Fault**—Returns a Boolean value. A value of TRUE indicates that the external power supply is out of the expected range.

Method	Description
	• **Open Current Loop**—Returns an array of Boolean values. A value of TRUE in any index indicates that the channel sharing a number with that index has an open current loop and is configured to output a nonzero current value.
Update	Writes stored data values to the output channel. • **Overrun**—Returns a Boolean value. A value of TRUE indicates an overrun warning. The overrun warning means that the Update method is trying to run while the Write Data method is running, typically in a different loop. Either the Write Data method is writing data to the module too quickly or the Update method is running too slowly. • **Underflow**—Returns a Boolean value. A value of TRUE indicates an underflow warning. The underflow warning means that the Write Data method has not written new data to the module since the last time the Update method ran. Either the Write Data method is writing data to the module too slowly or the Update method is running too quickly.
Wait for Update	Waits until the Update method updates the output channels. • **Timeout (Ticks)**—Specifies in FPGA clock ticks how long the Wait for Update method waits for the Update method to update the output channels. A value of 0 causes the Wait for Update method to time out immediately. A negative value causes the Wait for Update method to wait indefinitely. A positive value causes the Wait for Update method to wait for that number of clock ticks before timing out. • **Timeout Occurred?**—Returns a Boolean value. A value of TRUE indicates that a timeout occurred.
Write Data	Stores data values before writing the data to a channel. • **Channel**—Specifies the channel to which you want to write data. • **Data**—Specifies the data you want to write to the channel.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 329. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in pA/LSB for the channel. Use this value to convert module data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Property	Description
Offset	Returns the calibration offset in pA for the channel. Use this value to convert module data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 330. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Converting Current Values to Binary Values for the NI 9266 (FPGA Interface)
- Writing to C Series Channels

Converting Current Values to Binary Values for the NI 9266 (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9266 if you want the FPGA I/O Node to accept fixed-point data in units of amps when writing to the module.

If you set the **Calibration Mode** to Raw, the FPGA I/O Node accepts only binary values when writing to the module. Convert output voltage values to binary values before you write them to the module. Convert these values in the host VI.

Using an Equation to Convert Voltage to Binary

Use the following equation in the host VI to convert the analog output values to binary values:

$$Binary\ Value = (Current\ Value \times 10^{12} - Offset) \div LSB\ Weight$$

Where

- *Binary Value* is the value you write to the FPGA I/O Node.
- *Current Value* is the current in mA that you want the channel to output.
- *Offset* is the value returned by the Offset property.
- *LSB Weight* is the value returned by the LSB Weight property.

NI recommends using calibrated values for analog output. To convert calibrated voltage values, use the FPGA I/O Property Node to read the LSB Weight and Offset properties.

If you do not want to read the LSB Weight and Offset values from the module, convert uncalibrated voltage values by using the following values for *Offset* and *LSB Weight*:

$$Offset = 0$$

$$LSB\ Weight = 20.89\ mA \div 2^{DAC\ Resolution} \times 10^{12}$$

Where

- *DAC Resolution* is the DAC resolution of the module. Refer to the module hardware documentation for this value.

Related concepts:

- NI 9266 (FPGA Interface)

NI 9266 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables write floating-point values to the channels in volts.

Module Channels

The NI 9266 has the following channels.

Table 331. Module Channels

Channel	Description
AO*x*	Analog output channel *x*, where *x* is the number of the channel. For the NI 9266, *x* is 0 to 7.

Module-Specific Errors

The NI 9266 can return the following module-specific errors.

Table 332. Module-Specific Errors

Error Code	Description
65542	One or more channels have detected an open current loop. Check the module connections.
65543	The power supply voltage level is out of range. Check the supply voltage and the module connections.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project**

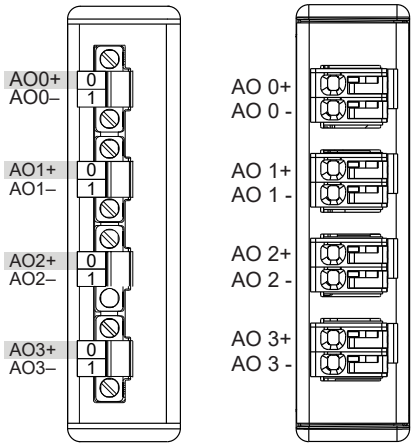
Explorer window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9269

CompactRIO 4-Channel, ±10 V, 16-Bit Simultaneous Isolated Analog Output Module

NI 9269 Pinout

Figure 79. NI 9269 Pinout



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9269 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for writing with this device.

Note You can simultaneously write to multiple channels on the NI 9269.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 333. Terminals in Software

Terminal	Description
AO*x*	Analog output channel *x*, where *x* is the number of the channel. The NI 9269 has AO channels 0 to 3.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following module methods for this device.

Refer to the NI 9269 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9269\NI 9269 Getting Started\NI 9269 Getting Started.lvproj` for an example of using the Check Output Status methods.

Use the Update, Wait for Update and Write Data methods in conjunction with each other.

- The Write Data method stores data values before writing the data to a channel,
- The Update method writes the stored data values to the output channel.
- The Wait for Update method waits until the Update method updates the output channels.

Refer to the NI 9264 Advanced Write Methods VI in the `labview\examples\CompactRIO\Module Specific\NI 9264\NI 9264 Advanced Write Methods\NI 9264 Advanced Write Methods.lvproj` for an example of using the Update, Wait for Update, and Write Data methods.

Table 334. Module Methods

Method	Description
Check Cached Output Status	Returns Booleans that indicate whether the module or channels reported a power supply fault or an open current loop since the last execution of the Check Cached Output Status method. When the FPGA I/O Node communicates with the module, the FPGA VI samples the states of the module and channels and caches any TRUE value until the Check Cached Output Status method executes. • **Force Status Read**—When the value of this input is FALSE, the method returns the cached status information since the last time that the Check Cached Output Status method executed. When the value is TRUE, the method forces the FPGA I/O Node to communicate with the module and update the status information. Forcing a status read can introduce jitter into an analog output loop. • **Any Fault**—Returns a Boolean value. A value of TRUE indicates that the module has a power supply fault or at least one channel has an open current loop. • **Power Supply Fault**—Returns a Boolean value. A value of TRUE indicates that the external power supply is out of the expected range. • **Open Current Loop**—Returns an array of Boolean values. A value of TRUE in any index indicates that the channel sharing a number with that index has an open current loop and is configured to output a nonzero current value.
Check Output Status	Returns Booleans that indicate whether the module or channels reported a power supply fault or an open current loop during the last time the FPGA I/O Node communicated with the module. When the FPGA I/O Node communicates with the module, the FPGA VI samples the states of the module and channels and overwrites the previous output status with the latest state of the module and channels. • **Force Status Read**—When the value of this input is FALSE, the method returns the status information from the last time the FPGA I/O Node communicated with the module. When the value is TRUE, the method forces the FPGA I/O Node to communicate with the module and update the status information. Forcing a status read can introduce jitter into an analog output loop. • **Any Fault**—Returns a Boolean value. A value of TRUE indicates that the module has a power supply fault or at least one channel has an open current loop. • **Power Supply Fault**—Returns a Boolean value. A value of TRUE indicates that the external power supply is out of the expected range.

Method	Description
	• **Open Current Loop**—Returns an array of Boolean values. A value of TRUE in any index indicates that the channel sharing a number with that index has an open current loop and is configured to output a nonzero current value.
Update	Writes stored data values to the output channel. • **Overrun**—Returns a Boolean value. A value of TRUE indicates an overrun warning. The overrun warning means that the Update method is trying to run while the Write Data method is running, typically in a different loop. Either the Write Data method is writing data to the module too quickly or the Update method is running too slowly. • **Underflow**—Returns a Boolean value. A value of TRUE indicates an underflow warning. The underflow warning means that the Write Data method has not written new data to the module since the last time the Update method ran. Either the Write Data method is writing data to the module too slowly or the Update method is running too quickly.
Wait for Update	Waits until the Update method updates the output channels. • **Timeout (Ticks)**—Specifies in FPGA clock ticks how long the Wait for Update method waits for the Update method to update the output channels. A value of 0 causes the Wait for Update method to time out immediately. A negative value causes the Wait for Update method to wait indefinitely. A positive value causes the Wait for Update method to wait for that number of clock ticks before timing out. • **Timeout Occurred?**—Returns a Boolean value. A value of TRUE indicates that a timeout occurred.
Write Data	Stores data values before writing the data to a channel. • **Channel**—Specifies the channel to which you want to write data. • **Data**—Specifies the data you want to write to the channel.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 335. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in nV/LSB for the channel. Use this value to convert module data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Property	Description
Offset	Returns the calibration offset in nV for the channel. Use this value to convert module data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 336. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Converting Voltage Values to Binary Values for the NI 9262/9263/9264/9269 (FPGA Interface)
- Writing to C Series Channels
- Simultaneously Reading From or Writing to Multiple CompactRIO Channels (FPGA Interface)

Converting Voltage Values to Binary Values for the NI 9262/9263/9264/9269 (FPGA Interface)

Set the **Calibration Mode** to **Calibrated** in the C Series Module Properties dialog box for the NI 9262/NI 9263/NI 9264/NI 9269 if you want the FPGA I/O Node to accept fixed-point data in units of volts when writing to the module.

If you set the **Calibration Mode** to Raw, the FPGA I/O Node accepts only binary values when writing to the module. Convert output voltage values to binary values before you write them to the module. Convert these values in the host VI.

Using an Equation to Convert Voltage to Binary

You can use the following equation in the host VI to convert the analog output values to binary values:

$$Binary\ Value = (Voltage\ Value \times 10^{9} - Offset) \div LSB\ Weight$$

Where

- *Binary Value* is the value you write to the FPGA I/O Node
- *Voltage Value* is the voltage in *V* that you want the channel to output
- *Offset* is the value returned by the Offset property
- *LSB Weight* is the value returned by the LSB Weight property.

NI recommends using calibrated values for analog output. To convert calibrated voltage values, use the FPGA I/O Property Node to read the LSB Weight and Offset properties.

If you do not want to read the LSB Weight and Offset values from the module, convert uncalibrated voltage values by using the following values for *Offset* and *LSB Weight*:

$$Offset = 0$$

$$LSB\ Weight = Typical\ Output\ Span \div 2^{DAC\ Resolution} \times 10^{9}$$

Where

- *Typical Output Span* is 21.4 V for the NI 9263, 21 V for the NI 9264, and 20.98 V for the NI 9269.
- *DAC Resolution* is the DAC resolution of the module. Refer to the module hardware documentation for this value.

NI 9269 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables write floating-point values to the channels in volts.

Module Channels

The NI 9269 has the following channels.

Table 337. Module Channels

Channel	Description
AO*x*	Analog output channel *x*, where *x* is the number of the channel. For the NI 9269, *x* is 0 to 3.

Module-Specific Errors

The NI 9269 can return the following module-specific errors.

Table 338. Module-Specific Errors

Error Code	Description
65548	One or more channels are in overcurrent or overvoltage protection mode. Check the terminals for any fault condition that could be causing an out-of-range voltage or current on the channels.
65579	The module is in overcurrent protection mode. Check the terminals for any fault condition that could be causing an out-of-range current.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

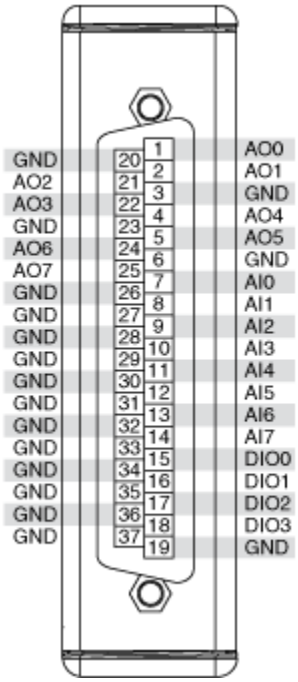
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9381

CompactRIO 0 V to 5 V, 12-bit, Multifunction Input/Output Module

NI 9381 Pinout

Figure 80. NI 9381 Pinout



Running AI and AO Operations Concurrently on the NI 9381

AI and AO operations that are run concurrently may exhibit jitter. Access both subsystems using a single I/O Node to avoid jitter in your application. Refer to the *Avoiding Timing Uncertainty* topic for more information.

NI 9381 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading and writing with this device.

Table 339. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9381 has AI channels 0 to 7.
AO*x*	Analog output channel *x*, where *x* is the number of the channel. The NI 9381 has AO channels 0 to 3.
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. The NI 9381 has DIO channels 0 to 3.
DIO3:0	Digital port consisting of channels 0 through 3. Channel 3 is returned in the MSB, and channel 0 is returned in the LSB.

Caution AI and AO operations that are run concurrently may exhibit jitter. Access both subsystems using a single I/O Node to avoid jitter in your application. Refer to *Avoiding Timing Uncertainty* for more information.

Arbitration

Analog input and analog output channels of this device support only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for analog input and analog output channels of this device.

You can configure the arbitration settings for digital output channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

Digital input channels of this device support only the **Never Arbitrate** option for arbitration. You cannot configure arbitration settings for the digital input channels of this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for the DIO channels of this device.

Table 340. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node (FPGA Module) topic for a description of this method. This method is available only when you set the DIO line direction to output. You can set the DIO line direction in the C Series Module Properties dialog box.
Set Output Enable	Sets the line direction of the digital channel or the DIO3:0 digital port. Refer to the FPGA I/O Method Node (FPGA Module) topic for more information on this method. This method can introduce jitter in an analog input or analog output loop. This method is available only when you enable programmable DIO line changes on the module. Enable programmable DIO line changes on the module in the C Series Module Properties dialog box.
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge.

Method	Description
	• A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Wait on Rising	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal.

Method	Description
Edge	The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.

Module Method

Use the FPGA I/O Method Node to access the following module method for this device.

Table 341. Module Method

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for AI and AO channels of this device.

Table 342. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in nV/LSB for the channel. Use this value to convert and calibrate NI 9381 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert and calibrate NI 9381 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 343. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop.

- Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.
- Configure the number of output synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box.

This device supports the Number of Synchronizing Registers for Output Data synchronizing register option when used in SCTL output. This option supports the same functionality as the **Number of Synchronizing Registers for Output Data** option described in the Advanced Code Generation FPGA I/O Properties Page (FPGA Module) topic, with the exception that you can use this option only in a single-cycle Timed Loop. You can implement either 0 or 1 synchronizing registers inside the single-cycle Timed Loop. I you configure 0 synchronizing registers outside of the single-cycle Timed Loop, the FPGA VI implements 1 synchronizing register by default.

When the device is within a single-cycle Timed Loop, it must be ready to perform digital I/O before a loop containing digital I/O starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready. The module also might ignore or delay digital output operations if it is not ready.

While the digital is performing digital I/O within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Performing any of these actions causes the module to be unable to perform digital I/O and the **Ready** output of the Check Status method to return *FALSE*.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as:

- 40 MHz
- 80 MHz
- 120 MHz

Avoiding Timing Uncertainty with the NI 9381 (FPGA Interface)

Follow the guidelines below to avoid introducing timing uncertainty into AI node, AO Node, DO Node, and Set Output Data method calls. Failing to follow these guidelines may result in lost data points for DO Node calls or Set Output Data method calls when either of them are made from inside a single-cycle Timed Loop.

- Use a single I/O Node to access AI and AO operations to ensure proper sequencing.
- Do not run the Set Output Enable method when an AI, AO, or DO subsystems is active.
- Do not perform the following operations concurrently:
  - AI Node call
  - AO Node call
  - Set Output Enable method call
  - Automatic line direction change when a DO call is made under the following conditions:
    - The call is made from outside of a single-cycle Timed Loop.
    - The **Allow programmatic DIO line direction** checkbox is enabled in the C Series Module Properties dialog box.
    - The DIO channel is configured as an input at the time when the DO node is called.

For an example of the recommended way to use the AI, AO , and DIO subsystems, refer to the NI 9381 Advanced IO VI in the `labview\examples\CompactRIO\Module`

`Specific\NI 9381\NI 9381 Advanced IO\NI 9381 Advanced IO.lvproj`.

Converting Voltage Values to Binary Values for the NI 9381 (FPGA Interface)

Set the **Calibration Mode** to Calibrated in the C Series Module Properties dialog box for the NI 9381 if you want the FPGA I/O Node to accept fixed-point data in units of volts when writing to the module.

When **Calibration Mode** is set to Raw when writing to the module, the FPGA I/O Node accepts only binary values. Convert output voltage values to binary values before writing them to the module. Convert these values in the host VI.

Using an Equation to Convert Voltage to Binary

Use the following equation in the host VI to convert the analog output values to binary values:

$$Binary\ Value = (Voltage\ Value - Offset) \div LSB\ Weight$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.
- *Voltage Value* is the voltage in *V* that you want the channel to output
- *Offset* is the value returned by the Offset property
- *LSB Weight* is the value returned by the LSB Weight property

NI recommends using calibrated values for analog output. To convert calibrated voltage values, use the FPGA I/O Property Node to read the LSB Weight and Offset properties. If you do not want to read the LSB Weight and Offset values from the module, convert uncalibrated voltage values by using the following values for *Offset* and *LSB Weight*:

$$Offset = -5249$$

$$LSB\ Weight = 5.105\ V \div 2^{DAC\ Resolution}$$

Where

- *DAC Resolution* is the DAC resolution value. Refer to the NI 9381 hardware documentation for the DAC resolution value.

NI 9381 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI.

- The I/O variables for the AI channels return floating-point data in volts.
- The I/O variables for the AO channels write floating-point data in volts.
- The I/O variables for the DIO channels read and write Boolean values.

Module Channels

The NI 9381 has the following channels.

Table 344. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9381, *x* is 0 to 7.
AO*x*	Analog output channel *x*, where *x* is the number of the channel. For the NI 9381, *x* is 0 to 7.
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. For the NI 9381, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9381. Right-click the NI 9381 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

- Channels—Specifies the channel(s) for which you want to configure settings.
- Direction—Specifies the initial line direction of the selected channel(s) as input or output.

Note If you are using direct access to read from or write to AI or AO channels of the NI 9381, the conversion time is somewhat longer than 50 µs.

Counter Input Modules

Table 345. C Series Counter Input Modules

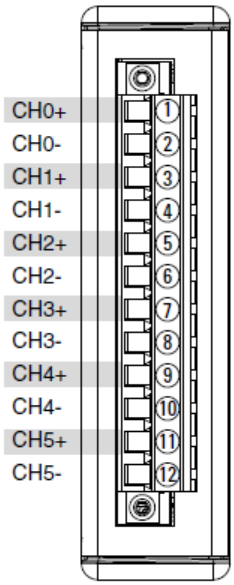
C Series Counter Input Module	Description
NI-9326	6-Channel, 150 V, 128 kHz C Series Frequency Input Module
NI-9361	32-Bit, 8-Channel C Series Counter Input Module

NI 9326

CompactRIO 6-Channel, Frequency Input Module

NI 9326 Pinout

Figure 81. NI 9326 Pinout



NI 9326 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 346. Terminals in Software

Terminal	Description
CTR*x*	Counter input channel *x*, where *x* is the number of the channel. The counter I/O node returns a cluster of three items, consisting of two unsigned 32-bit integers called A and B as well as the counter status enum. The **Start** input I/O node must be executed before counter or DI data can be read. The NI 9326 has counters 0 through 5.
DI*x*	Digital input channel *x*, where *x* is the number of the channel. The NI 9326 has DI channels 0 through 5. Supported in *Digital Input Mode*.
Start	Channel that controls when the counters begin monitoring their input terminals. If TRUE is written to the **Start** channel, the NI 9326 counters are reset, armed, and begin monitoring their input terminals to perform the currently configured measurement (such as Edge Counting). Once the NI 9326 counters are armed, execute the **CTR*x*** and **DI*x*** I/O nodes to read the counter data. You may not execute any NI 9326 I/O Method or Property nodes while the counters are armed. To disarm the counters, use the **Stop** channel. If FALSE is written to the **Start** channel, no operation is performed. The **Start** channel is also used to synchronize multiple NI 9326 modules.
Stop	Channel that controls when the counters are disarmed. If TRUE is written to the **Stop** channel, the counters are stopped. When the counters are stopped, you may change the counter configuration by executing

Terminal	Description
	I/O Method and Property nodes. If FALSE is written to the **Stop** channel, no operation is performed.

The definition of **A** and **B** values vary according to the selected measurement mode . The following table defines the **A** and **B** values for each measurement mode option.

Table 347. A and B values per Measurement Mode

Measurement Mode	A	B	Status
Edge Counting	Number of edges.	Unused. Always returns 0.	Not applicable.
Period	Amount of time elapsed during **B** signal periods, measured in units of counter timebase ticks.	Scaling factor for **A** value. Divide **A** by **B** to calculate the measured period.	Applicable.

Period measurements are returned based on ticks of the module's 100 MHz timebase. Each tick corresponds to 10 ns. The actual measured period is calculated using the following equation:

```
Measured Period = (Number of Ticks * Timebase)/Scaling
Factor = (A * 10 ns)/B
```

Example measurement of 1 kHz input signal:

- **A** = 3,200,000
- **B** = 32

```
Measured Period = (3,200,000 * 10 ns)/32 = 1 ms
```

Counter statuses are applicable for selected measurement modes. The following table describes the different statuses.

Table 348. Counter Status

Status	Description
No Sample	The first measurement data is not ready yet for this **CTR*x*** channel. This occurs as a result of the following factors: the amount of time elapsed since the counter was armed via the **Start** channel, the current counter settings, and the input signal behavior. In this case, the **A** and **B** values will each return a default value of 0.
New	A new, valid measurement for this **CTR*x*** channel.
Repeated	The same measurement returned the last time a read was performed on this **CTR*x*** channel. This measurement is valid, and occurs when the **CTR*x*** I/O Node reads are performed faster than the counter can produce new measurements.
Timeout	A timeout has occurred.
New and Missed	A new, valid measurement for this **CTR*x*** channel. The counter also produced one or more new measurements that were never read. This occurs when the counter produces measurements faster than the **CTR*x*** I/O Node reads are performed.
Timeout and Missed	A timeout has occurred. In addition, the counter produced one or more new measurements that were never read.

Timeout is applicable for period measurement mode only. The definition of Timeout varies according to the setting of the Butterworth filter. The following table defines Timeout for each option.

Table 349. Timeout Definition per Measurement Mode

Measurement Mode	Timeout Definition
Period (Butterworth filter enabled)	Timeout occurs when the actual input signal period is greater than the internal counter limit at 2^{32} x 10 ns and the counter returns a value of 0.
Period (Butterworth filter disabled)	Timeout occurs when the actual input signal period is greater than the maximum measurable period and the counter returns a value of 0.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device. You can also use the properties dialog box to configure the measurement modes for this device.

- **Configure Edge Counting**—Configures the corresponding counter for any edge counting application. The Method Node returns an error if invoked while the counters are armed. The Initial Count and Reset Count are in units of number of edges.

  Table 350. Configure Edge Counting I/O Method Parameters

Parameters	Description
Terminal	Source input terminal of the counter.
Initial Count	Initial counter value once the counter is armed.
Direction	Sets the count direction to either count up, count down or to be externally controlled.
Direction Terminal	Configures the terminal that is used to control the count direction. This needs to be set when Direction is configured to "Externally Controlled". If not set, the parameter value defaults to DI0 terminal.
Reset Enable	Enables or disables Hardware Reset.
Reset Terminal	Configures the terminal that is used to reset the counter. This needs to be set when Hardware Reset is enabled. If not set, the parameter value defaults to DI0 terminal.
Reset Count	Sets the count value to load to the counter upon reset. If not set, the parameter value defaults to zero.
Pause Enable	Enables or disables the pause trigger.
Pause Terminal	Configures the terminal that is used to pause the counter. This needs to be set when pause trigger is enabled. If not set, the parameter value defaults to DI0 terminal.

- **Configure Edge Counting Simple**—Configures the corresponding counter for simple edge counting. The simple configuration for edge counting sets Direction to

Count Up and disables both Reset and Pause. The Method Node returns an error if invoked while the counters are armed. The Initial Count is in units of number of edges.

Table 351. Configure Edge Counting Simple I/O Method Parameters

Parameters	Description
Terminal	Source input terminal of the counter.
Initial Count	Initial counter value once the counter is armed.

- **Configure Period**—Configures the corresponding counter for period measurement. The Method Node returns an error if invoked while the counters are armed.

Table 352. Configure Period I/O Method Parameters

Parameters	Description
Terminal	Source input terminal of the counter.
Enable Butterworth Filter	Enables or disables Butterworth filter. If Butterworth Filter is enabled, set the Filter Frequency parameter accordingly to filter out the higher frequency noise. If disabled, configure the counter settings based on Divisor, Measurement Time and Maximum Measurable Period before the start of an acquisition.
Filter Frequency	Configures the Butterworth filter's cut-off frequency to filter out variations in the period measurements.
Measurement Time	Configures the amount of time over which to measure and average to determine the input signal period. This value is specified in units of counter timebase ticks. Set to 0 to disable the **Measurement Time**, in which case the measurement will be completed using only the **Divisor** setting.
Divisor	Configures the number of periods of the input signal to measure and average to determine the input signal period. Set to 0 to disable the **Divisor**, in which case the measurement will be completed using only the **Measurement Time** setting.
Maximum Measurable Period	Configures the maximum (slowest) period of the input signal that can be measured. If the actual input signal period is longer than this value, the measurement will timeout and the counter returns a value of 0. This value is specified in units of counter timebase ticks. Set this value to 0 to disable enforcing the maximum period.

Note Setting both **Measurement Time** and **Divisor** to 0 is invalid, and will cause the counter to always return a value of 0.

- **Configure Period Simple**—Configures the corresponding counter for simple period measurement. The simple configuration for period measurement enables the Butterworth filter. The Method Node returns an error if invoked while the counters are armed.

Table 353. Configure Period Simple I/O Method Patameters

Parameters	Description
Terminal	Source input terminal of the counter.
Filter Frequency	Configures the Butterworth filter's cut-off frequency to filter out variations in the period measurements.

Module Methods

Use the FPGA I/O Method Node to access the following module methods for this device.

Table 354. Module Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready. Supported in Digital Input Mode.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 355. I/O Properties

Property	Description
Digital Filter Enable	Enables the digital glitch filter for the corresponding digital line.
Digital Filter Minimum Pulse Width[15]	Configures the minimum pulse width for the digital glitch filter of the corresponding digital line in increments of 80 ns, up to 5.24 ms.

15. Unit in ticks. 1 tick = 80 ns. For example, the Minimum Pulse Width = 1 ms. Therefore, input to the

Property	Description
Digital Logic Threshold	Configures the digital logic threshold for the channel.
Hysteresis Value	Configures the hysteresis value for the channel.
Active Edge	Configures the triggering edge of the input signal.

Note The definition of Active Edge varies according to the measurement mode. The following table defines Active Edge for each option.

Table 356. Active Edge Behavior per Measurement Mode

Measurement Mode	Active Edge Description
Edge Counting	• **Input Terminal:** Configures the triggering edge of the input signal. Rising indicates the counter would count on every rising edge and vice versa. • **Direction Terminal:** Configures the terminal that is used to control the count direction. When set to rising edge, the counter counts up when signal on the direction terminal is high. This applies only when the Direction parameter is set to Externally Controlled. If the Direction Terminal is not set and the Direction parameter is set to Externally Controlled, the default setting is to count up when signal is high. • **Reset Terminal:** Configures the triggering edge of the reset signal. This applies only when the Reset Enable parameter is set to true. If the Reset Terminal is not set and the Reset Enable parameter is set to true, the default setting is to reset on the rising edge of the signal. • **Pause Terminal:** Configures the terminal that is used to pause the counter. When set to rising edge, the counter pauses when the signal on the pause terminal is high. This applies only when the Pause Enable parameter is set to true. If the Pause Terminal is not set and the Pause Enable parameter is set to true, the default setting is to pause when signal is high.
Period (Butterworth filter enabled)	**Input Terminal:** Configures the triggering edge of the input signal.
Period (Butterworth filter	**Input Terminal:** Configures the triggering edge of the input signal.

Digital Filter Minimum Pulse Width property node is 1 ms / 80 ns = 12500 ticks.

Measurement Mode	Active Edge Description
disabled)	

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 357.

Property	Description
Module ID	Returns the module ID, 0x7A5E.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop when Digital Input Mode is set in the properties dialog box. The input channels of this device are synchronized with two synchronizing registers.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital input before a loop containing digital input starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return null data if the module is not ready.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates that are multiples of 40 MHz:

- 40 MHz
- 80 MHz
- 120 MHz
- 160 MHz.

Related concepts:

- Reading from C Series Channels

NI 9326 I/O Method Parameters

Use the following I/O Methods to configure the NI 9326 measurement modes.

Note The counters operate based on the module's timebase of 100 MHz. Each timebase tick corresponds to 10 ns.

- **Configure Edge Counting**—Configures the corresponding counter for any edge counting application. The Method Node returns an error if invoked while the module is in running state. The Initial Count and Reset Count are in edges.

Table 358. Configure Edge Counting Method Parameters

Parameters	Description
Terminal	Source input terminal of the counter.
Initial Count	Initial counter value once the counter is armed.
Direction	Sets the count direction to either count up, count down or to be externally controlled.
Direction Terminal	Configures the external terminal that is used to control the count direction. This needs to be set when Direction is configured to "Externally Controlled". If not set, the parameter value defaults to DI0 terminal.
Reset Enable	Enables or disables Hardware Reset.
Reset Terminal	Configures the reset terminal that is used to reset the counter. This needs to be set when Hardware Reset is enabled. If not set, the parameter value defaults to DI0 terminal.
Reset Count	Sets the count value to load to the counter upon reset. If not set, the parameter value defaults to zero.
Pause Enable	Enables or disables the pause trigger.
Pause Terminal	Configures the pause terminal that is used to pause the counter. This needs to be set when pause trigger is enabled. If not set, the parameter value defaults to DI0

Parameters	Description
	terminal.

- **Configure Edge Counting Simple**—Configures the corresponding counter for simple edge counting. The simple configuration for edge counting sets Direction to Count Up, and disables both Reset and Pause. The Method Node returns an error if invoked while the module is in running state. The Initial Count is in edges.

Table 359. Configure Edge Counting Simple Method Parameters

Parameters	Description
Terminal	Source input terminal of the counter.
Initial Count	Initial counter value once the counter is armed.

- **Configure Period**—Configures the corresponding counter for period measurement. The Method Node returns an error if invoked while the module is in running state.

Table 360. Configure Period Method Parameters

Parameters	Description
Terminal	Source input terminal of the counter.
Enable Butterworth Filter	Enables or disables Butterworth filter. If Butterworth Filter is enabled, set the Filter Frequency parameter accordingly to filter out the higher frequency noise. If disabled, configure the counter settings based on Divisor, Measurement Time and Maximum Measurable Period before the start of an acquisition.
Filter Frequency	Configures the Butterworth cut-off frequency to filter out variations in the period measurements.
Measurement Time	Configures the amount of time over which to measure and average the period of the input signal. This value is specified in units of counter timebase ticks. Set to 0 to disable the **Measurement Time**, in which case the measurement will be completed using only the **Divisor** setting.
Divisor	Configures the number of periods of the input signal to measure and average to determine the input signal period. Set to 0 to disable the **Divisor**, in which case the measurement will be completed using only the **Measurement Time** setting.
Maximum Measurable	Configures the maximum (slowest) period input signal that can be measured. If the actual input signal period is longer than this value, the measurement will

Parameters	Description
Period	timeout and the counter returns a value of 0. This value is specified in units of counter timebase ticks. Set this value to 0 to disable enforcing the maximum period.

Note Setting both **Measurement Time** and **Divisor** to 0 is invalid, and will cause the counter to always return a value of 0.

- **Configure Period Simple**—Configures the corresponding counter for simple period measurement. The Butterworth filter is enabled by default. The Method Node returns an error if invoked while the module is in running state.

Table 361. Configure Period Simple Method Parameters

Parameters	Description
Terminal	Source input terminal of the counter.
Filter Frequency	Configures the Butterworth filter's cut-off frequency to filter out variations in the period measurements.

Synchronizing Multiple NI 9326 Modules (FPGA Interface)

Complete the following steps to synchronize multiple NI 9326 modules that are connected to the same FPGA device.

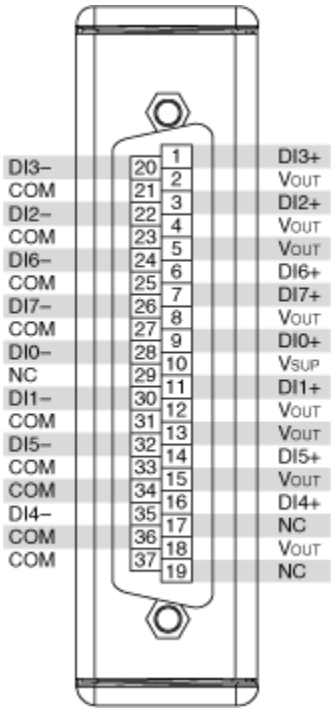
1. Create the FPGA I/O items with all the I/O channels to be synchronized within a single FPGA I/O Node. Otherwise, the FPGA I/O Node will not return synchronized data.
2. Configure an FPGA I/O Node with **Start** channels for the NI 9326 modules you want to synchronize.
3. Wire a Boolean constant set to *TRUE* to each **Start** channel.

NI 9361

CompactRIO 8-Channel, Counter Input Module

NI 9361 Pinout

Figure 82. NI 9361 Pinout



NI 9361 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 362. Terminals in Software

Terminal	Description
CTR*x*	Counter input channel *x*, where *x* is the number of the channel. The counter I/O node returns a cluster of three items, consisting of two unsigned 32-bit integers called A and B as well as the counter status enum. The **Start** input I/O node must be executed before counter or DI data can be read. The NI 9361 has counters 0 through 7.
DI*x*	Digital input channel *x*, where *x* is the number of the channel. The NI 9361 has DI channels 0 through 7.
DI7:0	Digital Port consisting of channels 0 through 7. Channel 7 is returned in the Most

Terminal	Description
	Significant Bit (MSB), and channel 0 is returned in the Least Significant Bit (LSB).
Start	Channel that controls when the counters begin monitoring their input terminals. If TRUE is written to the **Start** channel, the NI 9361 counters are reset, armed, and begin monitoring their input terminals to perform the currently configured measurement (such as Edge Counting). Once the NI 9361 counters are armed, you may execute the **CTR*x*** and **DI*x*** I/O nodes to read the counter data. You may not execute any NI 9361 I/O Method or Property nodes while the counters are armed. To disarm the counters, use the **Stop** channel. If FALSE is written to the **Start** channel, no operation is performed. The **Start** channel is also used to synchronize multiple NI 9361 modules.
Stop	Channel that controls when the counters are disarmed. If TRUE is written to the **Stop** channel, the counters are stopped. When the counters are stopped, you may change the counter configuration by executing I/O Method and Property nodes. If FALSE is written to the **Stop** channel, no operation is performed.

The definition of **A** and **B** values vary according to the measurement mode you select. The following table defines the **A** and **B** values for each measurement mode option.

Table 363. A and B values per Measurement Mode

Measurement Mode	A	B	Status
Edge Counting	Number of edges.	Unused. Always returns 0.	Not applicable.
Period	Amount of time elapsed during **B** signal periods, measured in units of counter	Number of signal periods measured.	Applicable.

Measurement Mode	A	B	Status
	timebase ticks.		
Pulse	Width of High portion of input signal period, measured in units of counter timebase ticks.	Width of Low portion of input signal period, measured in units of counter timebase ticks.	Applicable.
Pulse Width	Width of input pulse, measured in units of counter timebase ticks.	Unused. Always returns 0.	Applicable.
Two Edge Separation	Amount of time between two input edges, measured in units of counter timebase ticks.	Unused. Always returns 0.	Applicable.
Encoder Position	Number of encoder counts.	Unused. Always returns 0.	Not applicable.
Encoder Velocity	Amount of time elapsed during **B** encoder counts, measured in units of counter	Signed number of counter encoder counts and direction. Positive values indicate counting in the positive direction, and negative values indicate counting in the opposite direction. You must use the **To Long Integer** function to convert the unsigned **B** value (U32) to a signed value (I32) when performing an Encoder Velocity measurement.	Applicable.

Measurement Mode	A	B	Status
	timebase ticks.		

All values that are based on ticks use the module's timebase of 100 MHz. Each tick corresponds to 10 ns. The actual value is calculated as:

```
Actual = Number of ticks * 10 ns
```

Examples:

- **Period Measurement Mode**—A = 100000. B = 1. `Measured period = (A * 10 ns)/B = 1 ms`
- **Encoder Velocity Measurement Mode**—A = 100000. B = 1. `Measured Encoder Velocity = B/(A * 10 ns) = 1000 count/s`
- **Pulse Width and Two Edge Separation Measurement Mode**—A is 100000. `Measured Pulse Width or Two Edge Separation = A * 10 ns = 1 ms`
- **Pulse Measurement Mode**—A = 100000. B = 100000. `Measured High Pulse = A * 10 ns = 1 ms. Measured Low Pulse = B * 10 ns = 1 ms`

Counter statuses are applicable for selected measurement modes. The following table describes the different statuses.

Table 364. Counter Status

Status	Description
No Sample	The first measurement data is not ready yet for this **CTR*x*** channel. This occurs as a result of the following factors: the amount of time elapsed since the counter was armed via the **Start** channel, the current counter settings, and the input signal behavior. In this case, the **A** and **B** values will each return a default value of 0.
New	A new, valid measurement for this **CTR*x*** channel.
Repeated	The same measurement returned the last time a read was performed on this **CTR*x*** channel. This measurement is valid, and occurs when the **CTR*x*** I/O Node reads are performed faster than the counter can produce new measurements.
Timeout	A timeout has occurred.

Status	Description
New and Missed	A new, valid measurement for this **CTR*x*** channel. The counter also produced one or more new measurements that were never read. This occurs when the counter produces measurements faster than the **CTR*x*** I/O Node reads are performed.
Timeout and Missed	A timeout has occurred. In addition, the counter produced one or more new measurements that were never read.

Timeout is applicable for selected measurement modes, and the definition of Timeout varies according to the measurement mode you select. The following table defines Timeout for each measurement mode option.

Table 365. Timeout Definition per Measurement Mode

Measurement Mode	Timeout Definition
Period	Timeout occurs when the actual input signal period is greater than the maximum measurable period and the counter returns a value of 0.
Pulse	Timeout occurs when the actual input signal period is greater than the maximum measurable period. If the input signal remains high when timeout occurs, the counter returns a non-zero value for the high pulse and a value of 0 for the low pulse. If the input signal remains low when timeout occurs, the counter returns a non-zero value for the low pulse and a value of 0 for the high pulse.
Pulse Width	Timeout occurs when the actual input signal period is greater than the maximum measurable period and the counter returns a value of 0.
Two Edge Separation	Timeout occurs when the actual input signal period is greater than the maximum measurable separation and the counter returns a value of 0.
Quadrature Encoder Velocity	Timeout occurs when the actual input signal period is greater than the maximum measurable count period and the counter returns a value of 0.
Two Pulse Encoder Velocity	Timeout occurs when the actual input signal period is greater than the maximum measurable count period and the counter returns a value of 0.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for

arbitration. You cannot configure arbitration settings for this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device. You can also use the properties dialog box to configure the measurement modes for this device.

Module Methods

This device does not support any module methods.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 366. I/O Properties

Property	Description
Digital Filter Enable	Enables the digital filter for the corresponding digital line.
Digital Filter Minimum Pulse Width[16]	Configures the minimum pulse width for the digital filter of the corresponding digital line in increments of 80 ns, up to 5.24 ms.
Terminal Mode	Configures the terminal mode of the corresponding digital line (Differential, Single-ended, and Single-ended with Pull-up).

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 367. Module Properties

Property	Description
Module ID	Returns the module ID, 0x777E.
Serial Number	Returns the unique serial number of the module.

16. Unit in ticks. 1 tick = 80 ns. For example, Minimum Pulse Width = 1 ms. Therefore, input to the Digital Filter Minimum Pulse Width property node is 1 ms / 80 ns = 12500 ticks.

Property	Description
Vendor ID	Returns the NI vendor ID, 0x1093.
Digital Logic Threshold	Configures the digital logic threshold for the module. Does not apply to terminals that are configured as Differential.

Single-Cycle Timed Loop

This device does not support any single-cycled timed loop.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates that are multiples of 40 MHz:

- 40 MHz
- 80 MHz
- 120 MHz
- 160 MHz.

Related concepts:

- Reading from C Series Channels

NI 9361 I/O Method Parameters

Refer to the I/O Methods, that includes the following parameters, which you can use to configure the NI 9361 measurement modes.

Note The counters operate based on the module's timebase of 100 MHz. Each timebase tick corresponds to 10 ns.

- **Configure Edge Counting**—Configures the corresponding counter for any edge counting application. The Method Node returns an error if invoked while the module is in running state. The Initial Count and Reset Count are in edges.

Table 368. Configure Edge Counting Method Parameters

Parameters	Description
Terminal	Source input terminal of the counter.
Initial Count	Initial counter value once the counter is armed.
Active Edge	Sets the triggering edge of the input signal. Rising indicates the counter would count on every rising edge and vice versa.
Direction	Sets the count direction to either count up, count down or to be externally controlled.
Direction Terminal	Configures the external terminal that is used to control the count direction. This needs to be set when Direction is configured to "Externally Controlled". If not set, the parameter value defaults to DI0 terminal.
External Direction Count Up When	Configures whether the counter would count up when the Direction Terminal is 'HIGH' or to count up when the Direction Terminal is 'LOW'. If not set, the parameter value defaults to count up when high.
Reset Enable	Enables or disables Hardware Reset.
Reset Terminal	Configures the reset terminal that is used to reset the counter. This needs to be set when Hardware Reset is enabled. If not set, the parameter value defaults to DI0 terminal.
Reset Count	Sets the count value to load to the counter upon reset. If not set, the parameter value defaults to zero.
Reset Active Edge	Sets the triggering edge of the reset signal. If not set, the parameter value defaults to reset on the rising edge.
Pause Enable	Enables or disables the pause trigger.
Pause Terminal	Configures the pause terminal that is used to pause the counter. This needs to be set when pause trigger is enabled. If not set, the parameter value defaults to DI0 terminal.
Pause When	Configures the pause trigger to either pause the counter when the pause terminal is 'HIGH' or 'LOW'. If not set, the parameter value defaults to pause when high.

- **Configure Edge Counting Simple**—Configures the corresponding counter for simple edge counting. The simple configuration for edge counting sets Direction to

Count Up and disables both Reset and Pause. The Method Node returns an error if invoked while the module is in running state. The Initial Count is in edges.

Table 369. Configure Edge Counting Simple Method Parameters

Parameters	Description
Terminal	Source input terminal of the counter.
Initial Count	Initial counter value once the counter is armed.
Active Edge	Sets the triggering edge of the input signal. Rising indicates the counter would count on every rising edge and vice versa.

- **Configure Period**—Configures the corresponding counter for period measurement. The Method Node returns an error if invoked while the module is in running state.

Table 370. Configure Period Method Parameters

Parameters	Description
Terminal	Source input terminal of the counter.
Starting Edge	Configures the triggering edge of the input signal.
Measurement Time	Configures the amount of time over which to measure and average the period of the input signal. This value is specified in units of counter timebase ticks. Set to 0 to disable the **Measurement Time**, in which case the measurement will be completed using only the **Divisor** setting.
Divisor	Configures the number of periods of the input signal to measure and average to determine the input signal period. Set to 0 to disable the **Divisor**, in which case the measurement will be completed using only the **Measurement Time** setting.
Maximum Measurable Period	Configures the maximum (slowest) period input signal that can be measured. If the actual input signal period is longer than this value, the measurement will timeout and the counter returns a value of 0. This value is specified in units of counter timebase ticks. Set this value to 0 to disable enforcing the maximum period.

Note Setting both **Measurement Time** and **Divisor** to 0 is invalid, and will cause the counter to always return a value of 0.

- **Configure Period Simple**—Configures the corresponding counter for simple period

measurement. The simple configuration for period measurement sets **Measurement Time** to 0 and **Divisor** to 1. The Method Node returns an error if invoked while the module is in running state.

Table 371. Configure Period Simple Method Parameters

Parameters	Description
Terminal	Source input terminal of the counter.
Starting Edge	Configures the triggering edge of the input signal.
Maximum Measurable Period	Configures the maximum (slowest) period input signal that can be measured. If the actual input signal period is longer than this value, the measurement will timeout and the counter returns a value of 0. This value is specified in units of counter timebase ticks. Set this value to 0 to disable enforcing the maximum period.

- **Configure Pulse**—Configures the corresponding counter for pulse measurement. The Method Node returns an error if invoked while the module is in running state.

Table 372. Configure Pulse Method Parameters

Parameters	Description
Terminal	Source input terminal of the counter.
Starting Edge	Configures the triggering edge of the input signal.
Maximum Measurable Period	Configures the maximum (slowest) period input signal that can be measured. If the actual input signal period is longer than this value, the measurement will timeout and the counter returns a non-zero value for the high pulse and a value of 0 for the low pulse if the actual signal is high during timeout. The counter returns a non-zero value for the low pulse and a value of 0 for the high pulse if the actual signal is low during timeout. This value is specified in units of counter timebase ticks. Set this value to 0 to disable enforcing the maximum period.

- **Configure Pulse Width**—Configures the corresponding counter for pulse width measurement. The Method Node returns an error if invoked while the module is in running state.

Table 373. Configure Pulse Width Method Parameters

Parameters	Description
Terminal	Source input terminal of the counter.
Starting Edge	Configures the triggering edge of the input signal.
Maximum Measurable Period	Configures the maximum (slowest) period input signal that can be measured. If the actual input signal period is longer than this value, the measurement will timeout and the counter returns a value of 0. This value is specified in units of counter timebase ticks. Set this value to 0 to disable enforcing the maximum period.

- **Configure Two Edge Separation**—Configures the corresponding counter for two edge separation measurement. The Method Node returns an error if invoked while the module is in running state.

Table 374. Configure Two Edge Separation Method Parameters

Parameters	Description
First Terminal	First source input terminal of the counter.
First Edge	Configures the triggering edge of the first signal.
Second Terminal	Second source input terminal of the counter.
Second Edge	Configures the triggering edge of the second signal.
Maximum Measurable Separation	Configures the maximum separation between the first input signal and the second input signal that can be measured. If the actual input signal separation is longer than this value, the measurement will timeout and the counter returns a value of 0. This value is specified in units of counter timebase ticks. Set this value to 0 to disable enforcing the maximum separation.

- **Configure Quadrature Encoder Position**—Configure the corresponding counter for quadrature encoder measurement. **Z** indexing can be enabled or disabled. The Method Node returns an error if invoked while the module is in running state.

Table 375. Configure Quadrature Encoder Position Method Parameters

Parameters	Description
A Terminal	**A** input terminal of the counter.
A Invert Polarity	Configures the module to invert the **A** signal.
B Terminal	**B** input terminal of the counter.
B Invert Polarity	Configures the module to invert the **B** signal.
Initial Position	Initial counter value once the counter is armed.
Decoding Type	Configures the decoding type to either x1, x2 or x4.
Z Index Enable	Enables or disables the **Z** Index.
Z Terminal	Configures the **Z** terminal for the **Z** index. This needs to be set when **Z** Index is enabled. If not set, the parameter value defaults to DI0 terminal.
Z Invert Polarity	Configures the module to invert the **Z** signal. If not set, the parameter value defaults to not inverting the **Z** signal.
Z Index Value	Configures the counter value to be loaded when the **Z** Index is triggered. If not set, the parameter value defaults to not zero.
Z Index Phase	Configures the **Z** Index phase in which the **Z** index would be triggered. If not set, the parameter value defaults to trigger when **A** signal is low and **B** signal is low.

- **Configure Two Pulse Encoder**—Configures the corresponding counter for any two pulse encoder measurement. The Method Node returns an error if invoked while the module is in running state. The Initial Position and Reset Position values are in encoder counts.

Table 376. Configure Two Pulse Encoder Method Parameters

Parameters	Description
A Terminal	**A** input terminal of the counter.
A Invert Polarity	Configures the module to invert the **A** signal.
B Terminal	**B** input terminal of the counter.

Parameters	Description
B Invert Polarity	Configures the module to invert the **B** signal.
Initial Position	Initial counter value once the counter is armed.
Reset Enable	Enables or disables Hardware Reset.
Reset Terminal	Configures the reset terminal that is used to reset the counter. This needs to be set when Hardware Reset is enabled. If not set, the parameter value defaults to DI0 terminal
Reset Position	Sets the count value that would be loaded to the counter upon reset. If not set, the parameter value defaults to zero.
Reset Active Edge	Sets the triggering edge of the reset signal. If not set, the parameter value defaults to reset on rising edge.

- **Configure Two Pulse Encoder Simple**—Configures the corresponding counter for any two pulse encoder measurement. The simple configuration for two pulse encoder measurement disables Reset. The Method Node returns an error if invoked while the module is in running state. The Initial Position value is in encoder counts.

Table 377. Configure Two Pulse Encoder Simple Method Parameters

Parameters	Description
A Terminal	**A** input terminal of the counter.
A Invert Polarity	Configures the module to invert the **A** signal.
B Terminal	**B** input terminal of the counter.
B Invert Polarity	Configures the module to invert the **B** signal.
Initial Position	Initial counter value once the counter is armed.

- **Configure Quadrature Encoder Velocity**—Configures the corresponding counter for quadrature encoder velocity measurement. The Method Node returns an error if invoked while the module is in running state.

Table 378. Configure Quadrature Encoder Velocity Method Parameters

Parameters	Description
A Terminal	**A** input terminal of the counter.
A Invert Polarity	Configures the module to invert the **A** signal.
B Terminal	**B** input terminal of the counter.
B Invert Polarity	Configures the module to invert the **B** signal.
Decoding Type	Configures the decoding type to either x1, x2 or x4.
Measurement Time	Configures the amount of time over which to measure and average the period of the input signal. This value is specified in units of counter timebase ticks. Set to 0 to disable the **Measurement Time**, in which case the measurement will be completed using only the **Divisor** setting.
Divisor	Configures the number of encoder count periods to measure and average to determine the encoder input period (velocity). Set to 0 to disable the **Divisor**, in which case the measurement will be completed using only the **Measurement Time** setting.
Maximum Measurable Count Period	Configures the maximum (slowest) period between encoder input counts that can be measured. If the actual time between encoder counts is longer than this value, the measurement will timeout and the counter returns a value of 0. This value is specified in units of counter timebase ticks. Set this value to 0 to disable enforcing the maximum period.

Note Setting both **Measurement Time** and **Divisor** to 0 is invalid, and will cause the counter to always return a value of 0.

- **Configure Two Pulse Encoder Velocity**—Configures the corresponding counter for two pulse encoder velocity measurement. The Method Node returns an error if invoked while the module is in running state.

Table 379. Configure Two Pulse Encoder Velocity Method Parameters

Parameters	Description
A Terminal	**A** input terminal of the counter.
A Invert	Configures the module to invert the **A** signal.

Parameters	Description
Polarity	
B Terminal	**B** input terminal of the counter.
B Invert Polarity	Configures the module to invert the **B** signal.
Measurement Time	Configures the amount of time over which to measure and average the period of the input signal. This value is specified in units of counter timebase ticks. Set to 0 to disable the **Measurement Time**, in which case the measurement will be completed using only the **Divisor** setting.
Divisor	Configures the number of encoder count periods to measure and average to determine the encoder input period (velocity). Set to 0 to disable the **Divisor**, in which case the measurement will be completed using only the **Measurement Time** setting.
Maximum Measurable Count Period	Configures the maximum (slowest) period between encoder input counts that can be measured. If the actual time between encoder counts is longer than this value, the measurement will timeout and the counter returns a value of 0. This value is specified in units of counter timebase ticks. Set this value to 0 to disable enforcing the maximum period.

Synchronizing Multiple NI 9361 Modules (FPGA Interface)

Complete the following steps to synchronize multiple NI 9361 modules that are connected to the same FPGA device.

1. Create the FPGA I/O items with all the I/O channels to be synchronized, within a single FPGA I/O Node. Otherwise, the FPGA I/O Node will not return synchronized data.
2. Configure an FPGA I/O Node with **Start** channels for the NI 9361 modules you want to synchronize.
3. Wire a Boolean constant set to *TRUE* to each **Start** channel.

Digital Input Modules

Table 380. C Series Digital Input Modules

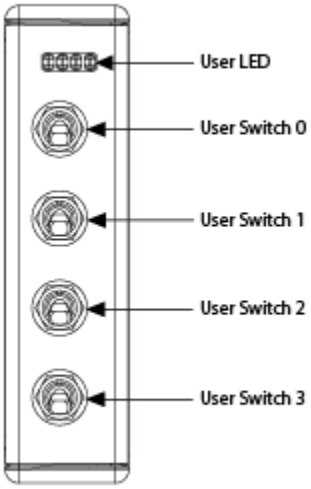
C Series Digital Input Module	Description
NI-9344	4-Channel C Series User Interface Module
NI-9375	30 V, 32-Channel (Sinking Input, Sourcing Output), 7 µs (Input)/500 µs (Output) C Series Digital Module
NI-9381	8 AI, 8 AO, 4 DIO, 0 V to 5 V, C Series Multifunction I/O Module
NI-9401	5 V/TTL, 8 Bidirectional Channels, 100 ns C Series Digital Module
NI-9402	LVTTL, 4 Bidirectional Channels, 55 ns C Series Digital Module
NI-9403	5 V/TTL, 32 Bidirectional Channels, 7 µs C Series Digital Module
NI-9411	±5 V to 24 V, 6 Differential/Single-Ended Channels, 500 ns C Series Digital Module
NI-9421	24 V, 8-Channel (Sinking Input), 100 µs C Series Digital Module
NI-9422	24 V to 60 V, 8 Channel (Sinking/Sourcing Input), 250 µs C Series Digital Module
NI-9423	24 V, 8 Channel (Sinking Input), 1 µs C Series Digital Module
NI-9425	24 V, 32-Channel (Sinking Input), 7 µs C Series Digital Module
NI-9426	24 V, 32 Channel (Sourcing Input), 7 µs C Series Digital Module
NI-9435	±5 VDC to ±250 VDC/10 VAC to 250 VAC, 4-Channel (Sinking/Sourcing Input), 3 ms C Series Digital Module
NI-9436	120 VDC to 240 VDC/120 VAC to 240 VAC, 8 Channel (Sinking/Sourcing Input), 20 ms C Series Digital Module
NI-9437	24 VDC to 250 VDC, 8-Channel (Sinking Input), 1 µs C Series Digital Module

NI 9344

CompactRIO 4-Channel Switch Input and 4-Channel LED Output Module

NI 9344 Front Panel

Figure 83. NI 9344 Front Panel



NI 9344 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading and writing with this device. When you write to an NI 9344 channel, the FPGA I/O Node does not automatically enable the channel for output.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 381. Terminals in Software

Terminal	Description
User LED*x*	LED output channel *x*, where *x* is the number of the channel. The NI 9344 has LED channels 0 to 3.
User Switch x	Switch input channel *x*, where *x* is the number of the channel. The NI 9344 has Switch channels 0 to 3.
User LED 3:0	Digital output port consisting of channels 0 through 3. Channel 3 is returned in the MSB, and channel 0 is returned in the LSB.
User	Digital input port consisting of channels 0 through 3. Channel 3 is returned in the MSB,

Terminal	Description
Switch 3:0	and channel 0 is returned in the LSB.

Arbitration

You can configure the arbitration settings for digital output channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

Digital input channels of this device support only the **Never Arbitrate** option for arbitration. You cannot configure arbitration settings for the digital input channels of this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Method	Description
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.

Method	Description
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.

Module Methods

Use the FPGA I/O Method Node to access the following module methods for this device.

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

Note During the first 2 seconds after you reset the FPGA VI, the error terminals on this method may not correctly report certain types of errors.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Property	Description
Module ID	Returns the module ID, 0x77C5.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the channels of this device in the Advanced Configuration dialog box. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital I/O before a loop containing digital I/O starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready. The module also might ignore or delay digital output operations if it is not ready.

While the module is performing digital I/O within a single-cycle Timed Loop, do not perform property reads, use the Set Line Direction method, or remove the module from the chassis. Doing any of these actions causes the module to be unable to perform digital I/O and the **Ready** output of the Check Status method to return *FALSE*.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates that are multiples of 40 MHz, such as:

- 40 MHz
- 80 MHz
- 120 MHz
- 160 MHz

Related concepts:

- Reading from C Series Channels
- Writing to C Series Channels

NI 9344 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI.

The I/O variables for the channels read and write Boolean values.

Module Channels

The NI 9344 has the following channels.

Table 382. Module Channels

Channel	Description
User LED*x*	LED output channel *x*, where *x* is the number of the channel. The NI 9344 has LED channels 0 to 3.
User Switch *x*	Switch input channel *x*, where *x* is the number of the channel. The NI 9344 has Switch channels 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project**

Explorer window and select **Properties** to display this dialog box. You can configure the following options.

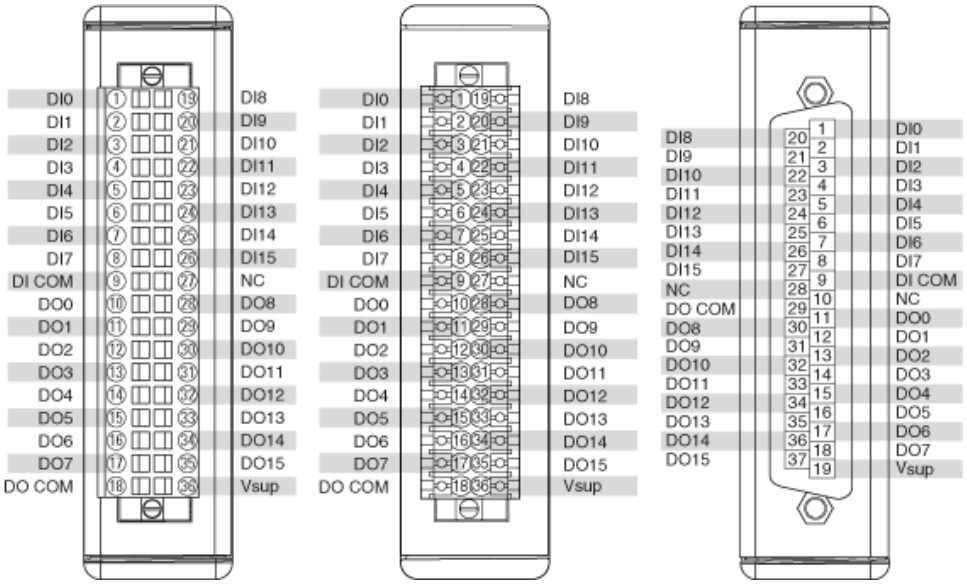
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9375

CompactRIO 16-Channel Digital Input and 16-Channel Digital Output

NI 9375 Pinout

Figure 84. NI 9375 Pinout



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9375 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading and writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 383. Terminals in Software

Terminal	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. The NI 9375 has DI channels 0 to 15.
DI7:0	Digital port consisting of DI channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
DI15:8	Digital port consisting of DI channels 8 through 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
DI15:0	Digital port consisting of DI channels 0 through 15. Channel 15 is returned in the MSB, and channel 0 is returned in the LSB.
DO*x*	Digital output channel *x*, where *x* is the number of the channel. The NI 9375 has DO channels 0 to 15.
DO7:0	Digital port consisting of DO channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
DO15:8	Digital port consisting of DO channels 8 through 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
DO15:0	Digital port consisting of DO channels 0 through 15. Channel 15 is returned in the MSB, and channel 0 is returned in the LSB.

Caution DI and DO operations that are run concurrently may exhibit jitter. Access both subsystems using a single I/O Node to avoid jitter in your application. Refer to *Avoiding Timing Uncertainty* for more information.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

This device does not support any module methods.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Reading from C Series Channels
- Writing to C Series Channels

Avoiding Timing Uncertainty with the NI 9375 (FPGA Interface)

To avoid introducing timing uncertainty into DI and DO Node calls, adhere to the following guidelines:

- Use a single I/O Node to access DI and DO operations to ensure proper sequencing.
- Do not perform the following operations concurrently:
  - DI Node call
  - DO Node call

For an example of the recommended way to use the DIO subsystems, refer to the NI 9375 Digital Port Input Output VI in `labview\examples\CompactRIO\Module Specific\NI 9375\NI 9375 Digital Port Input Output\NI 9375 Digital Port Input Output.lvproj`.

NI 9375 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI.

The I/O variables for the channels read and write Boolean values.

Module Channels

The NI 9375 has the following channels.

Table 384. Module Channels

Channel	Description
DIx	Digital input channel x, where x is the number of the channel. For the NI 9375, x is 0 to 15.
DOx	Digital output channel x, where x is the number of the channel. For the NI 9375, x is 0 to 15.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9375. Right-click the NI 9375 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

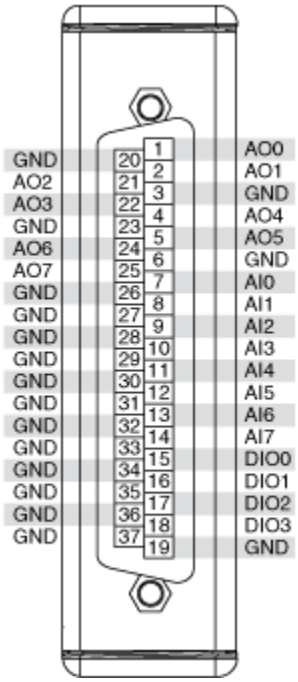
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9381

CompactRIO 0 V to 5 V, 12-bit, Multifunction Input/Output Module

NI 9381 Pinout

Figure 85. NI 9381 Pinout



Running AI and AO Operations Concurrently on the NI 9381

AI and AO operations that are run concurrently may exhibit jitter. Access both subsystems using a single I/O Node to avoid jitter in your application. Refer to the *Avoiding Timing Uncertainty* topic for more information.

NI 9381 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading and writing with this device.

Table 385. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9381 has AI

Terminal	Description
	channels 0 to 7.
AO*x*	Analog output channel *x*, where *x* is the number of the channel. The NI 9381 has AO channels 0 to 3.
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. The NI 9381 has DIO channels 0 to 3.
DIO3:0	Digital port consisting of channels 0 through 3. Channel 3 is returned in the MSB, and channel 0 is returned in the LSB.

Caution AI and AO operations that are run concurrently may exhibit jitter. Access both subsystems using a single I/O Node to avoid jitter in your application. Refer to *Avoiding Timing Uncertainty* for more information.

Arbitration

Analog input and analog output channels of this device support only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for analog input and analog output channels of this device.

You can configure the arbitration settings for digital output channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

Digital input channels of this device support only the **Never Arbitrate** option for arbitration. You cannot configure arbitration settings for the digital input channels of this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for the DIO channels of this device.

Table 386. I/O Methods

Method	Description
Set	Refer to the FPGA I/O Method Node (FPGA Module) topic for a description of this method.

Method	Description
Output Data	This method is available only when you set the DIO line direction to output. You can set the DIO line direction in the C Series Module Properties dialog box.
Set Output Enable	Sets the line direction of the digital channel or the DIO3:0 digital port. Refer to the FPGA I/O Method Node (FPGA Module) topic for more information on this method. This method can introduce jitter in an analog input or analog output loop. This method is available only when you enable programmable DIO line changes on the module. Enable programmable DIO line changes on the module in the C Series Module Properties dialog box.
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until

Method	Description
	the module is identified.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.

Method	Description
	Note This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.

Module Method

Use the FPGA I/O Method Node to access the following module method for this device.

Table 387. Module Method

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for AI and AO channels of this device.

Table 388. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in nV/LSB for the channel. Use this value to convert and calibrate NI 9381 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert and calibrate NI 9381 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 389. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop.

- Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.
- Configure the number of output synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box.

This device supports the Number of Synchronizing Registers for Output Data synchronizing register option when used in SCTL output. This option supports the same functionality as the **Number of Synchronizing Registers for Output Data** option described in the Advanced Code Generation FPGA I/O Properties Page (FPGA Module) topic, with the exception that you can use this option only in a single-cycle Timed Loop. You can implement either 0 or 1 synchronizing registers inside the single-cycle Timed Loop. I you configure 0 synchronizing registers outside of the single-cycle Timed Loop, the FPGA VI implements 1 synchronizing register by default.

When the device is within a single-cycle Timed Loop, it must be ready to perform digital I/O before a loop containing digital I/O starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready. The module also might ignore or delay digital output operations if it is not ready.

While the digital is performing digital I/O within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Performing any of these actions causes the module to be unable to perform digital I/O and the **Ready** output of the Check Status method to return *FALSE*.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as:

- 40 MHz
- 80 MHz
- 120 MHz

Avoiding Timing Uncertainty with the NI 9381 (FPGA Interface)

Follow the guidelines below to avoid introducing timing uncertainty into AI node, AO Node, DO Node, and Set Output Data method calls. Failing to follow these guidelines may result in lost data points for DO Node calls or Set Output Data method calls when either of them are made from inside a single-cycle Timed Loop.

- Use a single I/O Node to access AI and AO operations to ensure proper sequencing.
- Do not run the Set Output Enable method when an AI, AO, or DO subsystems is active.
- Do not perform the following operations concurrently:
  - AI Node call
  - AO Node call
  - Set Output Enable method call
  - Automatic line direction change when a DO call is made under the following conditions:
    - The call is made from outside of a single-cycle Timed Loop.
    - The **Allow programmatic DIO line direction** checkbox is enabled in the C Series Module Properties dialog box.
    - The DIO channel is configured as an input at the time when the DO node is called.

For an example of the recommended way to use the AI, AO , and DIO subsystems, refer to the NI 9381 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9381\NI 9381 Advanced IO\NI 9381 Advanced IO.lvproj`.

Converting Voltage Values to Binary Values for the NI 9381 (FPGA Interface)

Set the **Calibration Mode** to Calibrated in the C Series Module Properties dialog box for the NI 9381 if you want the FPGA I/O Node to accept fixed-point data in units of volts when writing to the module.

When **Calibration Mode** is set to Raw when writing to the module, the FPGA I/O Node accepts only binary values. Convert output voltage values to binary values before writing them to the module. Convert these values in the host VI.

Using an Equation to Convert Voltage to Binary

Use the following equation in the host VI to convert the analog output values to binary values:

$$Binary\ Value = (Voltage\ Value - Offset) \div LSB\ Weight$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.
- *Voltage Value* is the voltage in *V* that you want the channel to output
- *Offset* is the value returned by the Offset property
- *LSB Weight* is the value returned by the LSB Weight property

NI recommends using calibrated values for analog output. To convert calibrated voltage values, use the FPGA I/O Property Node to read the LSB Weight and Offset properties. If you do not want to read the LSB Weight and Offset values from the module, convert uncalibrated voltage values by using the following values for *Offset* and *LSB Weight*:

$$Offset = -5249$$

$$LSB\ Weight = 5.105\ V \div 2^{DAC\ Resolution}$$

Where

- *DAC Resolution* is the DAC resolution value. Refer to the NI 9381 hardware documentation for the DAC resolution value.

NI 9381 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI.

- The I/O variables for the AI channels return floating-point data in volts.
- The I/O variables for the AO channels write floating-point data in volts.
- The I/O variables for the DIO channels read and write Boolean values.

Module Channels

The NI 9381 has the following channels.

Table 390. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9381, *x* is 0 to 7.
AO*x*	Analog output channel *x*, where *x* is the number of the channel. For the NI 9381, *x* is 0 to 7.
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. For the NI 9381, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9381. Right-click the NI 9381 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- Channels—Specifies the channel(s) for which you want to configure settings.
- Direction—Specifies the initial line direction of the selected channel(s) as input or output.

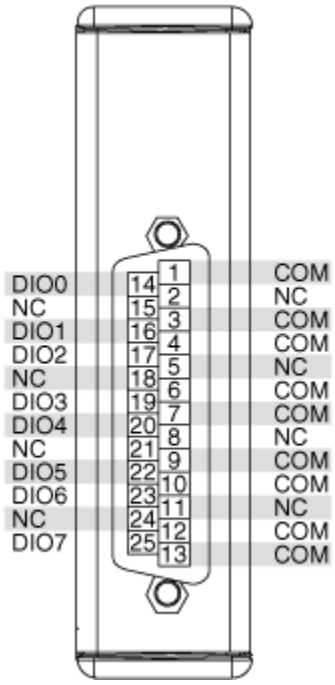
Note If you are using direct access to read from or write to AI or AO channels of the NI 9381, the conversion time is somewhat longer than 50 µs.

NI 9401

CompactRIO 8-Channel, TTL Digital Input/Output Module

NI 9401 Pinout

Figure 86. NI 9401 Pinout



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9401 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading and writing with this device.

When you write to an NI 9401 channel, the FPGA I/O Node does not automatically enable the channel for output.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Terminal	Description
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. The NI 9401 has DIO channels 0 to 7.
DIO3:0	Digital port consisting of channels 0 through 3. Channel 3 is returned in the MSB, and channel 0 is returned in the LSB.
DIO7:4	Digital port consisting of channels 4 through 7. Channel 7 is returned in the MSB, and channel 4 is returned in the LSB.
DIO7:0	Digital port consisting of channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.

Arbitration

You can configure the arbitration settings for digital output channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

Digital input channels of this device support only the **Never Arbitrate** option for arbitration. You cannot configure arbitration settings for the digital input channels of this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Method	Description
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Falling	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge

Method	Description
Edge	method waits for the next falling edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Module Methods

Use the FPGA I/O Method Node to access the following module methods for this device.

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.  **Note** During the first 2 seconds after you reset the FPGA VI, the error terminals on this method may not correctly report certain types of errors.
Set Line Direction	Sets the direction of one port to input or output.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the channels of this device in the Advanced Configuration dialog box. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital I/O before a loop containing digital I/O starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready. The module also might ignore or delay digital output operations if it is not ready.

While the module is performing digital I/O within a single-cycle Timed Loop, do not perform property reads, use the Set Line Direction method, or remove the module from the chassis. Doing any of these actions causes the module to be unable to perform digital I/O and the **Ready** output of the Check Status method to return *FALSE*.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as:

- 40 MHz
- 80 MHz

- 120 MHz

Related concepts:

- Reading from C Series Channels
- Writing to C Series Channels
- Configuring the Initial Line Direction for the NI 9401 (FPGA Interface)

Configuring the Initial Line Direction for the NI 9401 (FPGA Interface)

Each digital port on the NI 9401 is initially configured as a digital input. You can configure the initial line direction for each port on the NI 9401 at edit time using the C Series Module Properties dialog box.

Programmatically change the line direction for each port at run time using the FPGA I/O Method Node. The execution of an I/O Method Node that is configured with a Set Line Direction method overwrites the value you configured in the C Series Module Properties dialog box.

Configuring Line Direction Using the C Series Module Properties Dialog Box

Complete the following steps to configure the line direction of each digital port using the C Series Module Properties dialog box.

1. Configure the CompactRIO system, and add an NI 9401.
2. Right-click the NI 9401 in the **Project Explorer** window and select **Properties** from the shortcut menu to display the C Series Module Properties dialog box.
3. Select the direction for each port from the **Initial Line Direction** pull-down menus.
4. Click the **OK** button.
5. Select **File»Save All** in the **Project Explorer** window.

Configuring Line Direction Using the FPGA I/O Method Node

Complete the following steps to configure the line direction of each digital port using the FPGA I/O Method Node.

1. Place an FPGA I/O Method Node on the block diagram and configure it for the

NI 9401.
2. Click the **Method** section and select the Set Line Direction method from the shortcut menu.
3. Right-click each digital port input and select **Create**»**Control** from the shortcut menu.
4. On the front panel of the VI, select the direction for each port from the digital port pull-down menus.

Related concepts:

- NI 9401 (FPGA Interface)

NI 9401 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels read and write Boolean values.

Module Channels

The NI 9401 has the following channels.

Table 391. Module Channels

Channel	Description
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. For the NI 9401, *x* is 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9401. Right-click the NI 9401 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the Project Explorer window. LabVIEW assigns a default name to the module based on the slot

number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- Initial Line Direction—Specifies the initial line direction of each four-channel port as input or output.

Specialty Digital Configuration

Use the Specialty Digital Configuration page of the C Series Module Properties dialog box to configure channels of this module for counter input, counter-driven output, pulse-width modulation output, or quadrature input.

Related concepts:

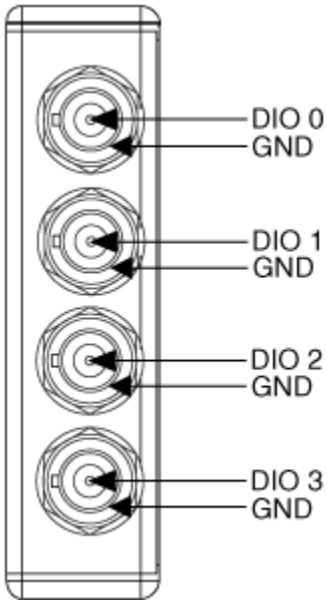
- Configuring a Counter to Measure Frequency
- Configuring Specialty Digital Functions
- Configuring Counters
- Configuring Counter-Driven Outputs
- Configuring a Channel for Pulse-Width Modulation
- Configuring a Module for Quadrature Input

NI 9402

CompactRIO 4-Channel, LVTTL Digital Input/Output Module

NI 9402 Pinout

Figure 87. NI 9402 Pinout



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9402 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading and writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 392. Terminals in Software

Terminal	Description
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. The NI 9402 has DIO channels 0 to 3.
DIO3:0	Digital port consisting of channels 0 through 3. Channel 3 is returned in the MSB, and channel 0 is returned in the LSB.

Arbitration

You can configure the arbitration settings for digital output channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

Digital input channels of this device support only the **Never Arbitrate** option for arbitration. You cannot configure arbitration settings for the digital input channels of this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 393. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node (FPGA Module) topic for a description of this method.
Set Output Enable	Sets the line direction of the digital channel or the DIO3:0 digital port. Refer to the FPGA I/O Method Node (FPGA Module) topic for more information on this method.
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Method	Description
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Module Method

Use the FPGA I/O Method Node to access the following module method for this device.

Table 394. Module Method

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready. **Note** During the first 2 seconds after you reset the FPGA VI, the error terminals on this method may not correctly report certain types of errors.

I/O properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 395. Module Properties

Property	Description
Module ID	Returns the module ID.

Property	Description
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box. Configure the number of output synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box.

The NI 9402 supports the following output synchronizing register options:

- **Number of Synchronizing Registers for Output Data when used in SCTL**
- **Number of Synchronizing Registers for Output Enable when used in SCTL**

These two options support the same functionality as the **Number of Synchronizing Registers for Output Data** and **Number of Synchronizing Registers for Output Enable** options described in the Advanced Code Generation FPGA I/O Properties Page (FPGA Module) topic, with the exception that you can use these options only in a single-cycle Timed Loop. You can implement either 0 or 1 synchronizing registers inside the single-cycle Timed Loop, however, if you configure 0 synchronizing registers outside of the single-cycle Timed Loop, the FPGA VI implements 1 synchronizing register by default.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital I/O before a loop containing digital I/O starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready. The module also might ignore or delay digital output operations if it is not ready.

While the module is performing digital I/O within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Doing any of these actions causes the module to be unable to perform digital I/O and the **Ready** output of the Check Status method to return FALSE. FPGA

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as:

- 40 MHz
- 80 MHz
- 120 MHz

Related concepts:

- Reading from C Series Channels
- Writing to C Series Channels

Configuring the Line Direction for the NI 9402 (FPGA Interface)

Each digital port on the NI 9402 is initially configured as a digital input. You can configure the initial line direction for each port on the NI 9402 at edit time using the C Series Module Properties dialog box.

Programmatically change the line direction for each port at run time using the FPGA I/O Method Node. The execution of an I/O Method Node that is configured with a Set Line Direction method overwrites the value you configured in the C Series Module Properties dialog box.

The execution of an FPGA I/O Node configured for output automatically configures the line for output and overwrites the values you configured in the C Series Module Properties dialog box or using the Set Output Enable method.

Configuring Line Direction Using the C Series Module Properties Dialog Box

Complete the following steps to configure the line direction of each digital port using the C Series Module Properties dialog box.

1. Configure the CompactRIO system, and add an NI 9402.
2. Right-click the NI 9402 in the **Project Explorer** window and select **Properties** from the shortcut menu to display the **C Series Module Properties** dialog box.

3. Select the channel(s) for which you want to configure the line direction from the **Channels** table. You can select more than one channel by holding the <Ctrl> or <Shift> key when selecting channels.
4. Select the direction for the channel(s) from the **Direction** pull-down menu.
5. Click the **OK** button.
6. Select **File»Save All** in the **Project Explorer** window.

Configuring Line Direction Using the FPGA I/O Method Node

Complete the following steps to configure the line direction of each digital port using the FPGA I/O Method Node.

1. Create FPGA I/O items for the channel of the NI 9402 for which you want to configure the line direction.
2. Place an FPGA I/O Method Node on the block diagram and configure it for this channel.
3. Click the **Method** section and select the Set Line Direction method from the shortcut menu.
4. Right-click the **Enable** input and select **Create»Control** from the shortcut menu.
5. On the front panel of the VI, click the **Enable** Boolean control to set it to *TRUE* if you want to set the line direction of the channel to digital output.

Configuring Line Direction of Multiple Channels Using the FPGA I/O Method Node

Complete the following steps to set the line direction of multiple channels using the FPGA I/O Method Node.

1. Create FPGA I/O items for the DIO3:0 digital port of the NI 9402.
2. Place an FPGA I/O Method Node on the block diagram and configure it for the DIO3:0 digital port.
3. Click the Method section and select the **Set Output Enable** method from the shortcut menu.
4. Right-click the **Enable** input and select **Create»Control** from the shortcut menu. The **Enable** control appears as an unsigned 8-bit integer. Each bit in the integer represents the line direction of one channel of the NI 9402.
5. On the front panel of the VI, use the **Enable** control to enter the line direction for

each channel in the port. Change a bit to 1 to set the line direction of the corresponding channel to digital output. Leave a bit as 0 to set the line direction of the corresponding channel to digital input. Refer to the table below for examples of what to enter in the **Enable** control.

Table 396. Enable Control Values

Channel Line Direction Configuration	Enable Control(Floating Point)	Enable Control(Binary)	Enable Control(Hex)
Change all channels to input	0	0000	0x00
Change all channels to output	15	1111	0x0F
Change channel 0 to output	1	0001	0x01

Related concepts:

- NI 9402 (FPGA Interface)

NI 9402 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels read and write Boolean values.

Module Channels

The NI 9402 has the following channels.

Table 397. Module Channels

Channel	Description
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. For the NI 9402, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project**

Explorer window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the Project Explorer window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- Direction—Specifies the initial line direction of the selected channel(s) as input or output.

Specialty Digital Configuration

Use the Specialty Digital Configuration page of the C Series Module Properties dialog box to configure channels of this module for counter input, counter-driven output, pulse-width modulation output, or quadrature input.

Related concepts:

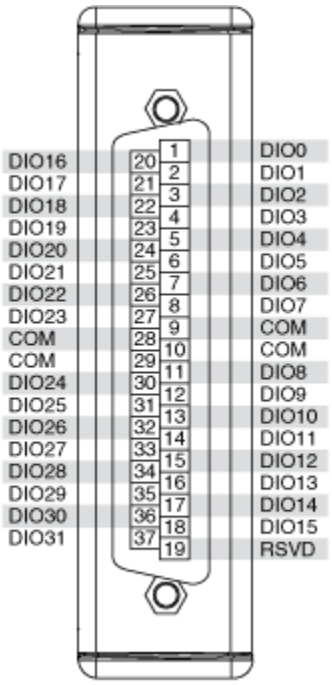
- Configuring Specialty Digital Functions
- Configuring Counters
- Configuring Counter-Driven Outputs
- Configuring a Channel for Pulse-Width Modulation
- Configuring a Module for Quadrature Input

NI 9403

CompactRIO 32-Channel, TTL Digital Input/Output Module

NI 9403 Pinout

Figure 88. NI 9403 Pinout



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9403 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading and writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 398. Terminals in Software

Terminal	Description
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. The NI 9403 has DIO channels 0 to 31.
DIO7:0	Digital port consisting of channels 0 through 7. Channel 7 is returned in the MSB, and

Terminal	Description
	channel 0 is returned in the LSB.
DIO15:8	Digital port consisting of channels 8 through 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
DIO23:16	Digital port consisting of channels 16 through 23. Channel 23 is returned in the MSB, and channel 16 is returned in the LSB.
DIO31:24	Digital port consisting of channels 24 through 31. Channel 31 is returned in the MSB, and channel 24 is returned in the LSB.
DIO31:0	Digital port consisting of channels 0 through 31. Channel 31 is returned in the MSB, and channel 0 is returned in the LSB.

Arbitration

This device supports only the Arbitrate if Multiple Requestors Only and Never Arbitrate options for arbitration. The default arbitration setting is **Arbitrate if Multiple Requestors Only**.

If you are sure that the design of the FPGA VI will never allow more than one digital function to execute at the same time, even on different channels, enable the **Disable Arbitration** checkbox on the C Series Module Properties dialog box to disable arbitration and reduce the amount of FPGA logic used by VIs. If more than one digital function could execute simultaneously in the FPGA VI, leave the box unchecked.

I/O methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 399. I/O methods

Method	Description
Read and Set Output Data	First returns the data read from the digital line or port, then writes data to the line or port without enabling it. You can use the Read and Set Output Data method to optimize performance when performing successive reads and writes on a DIO resource. For the NI 9403, this method is faster than using a read FPGA I/O Node and a Set Output Data method. The data type of the output and input depends on the I/O item.

Method	Description
	• If the I/O item is a digital line, **Data to Set** requires a Boolean data type. • If the I/O item is a digital port, **Data to Set** requires a numeric data type.
Set Output Data	Refer to the FPGA I/O Method Node (FPGA Module) in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Set Output Enable	Sets the line direction of the digital channel or port. Refer to the FPGA I/O Method Node (FPGA Module) in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.

Module Methods

This device does not support any module methods.

I/O properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 400. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Reading from C Series Channels
- Writing to C Series Channels

Configuring the Line Direction for the NI 9403 (FPGA Interface)

Each digital port on the NI 9403 is initially configured as a digital input. You can configure the initial line direction for each port on the NI 9403 at edit time using the C Series Module Properties dialog box.

Programmatically change the line direction for each port at run time using the FPGA I/O Method Node. The execution of an I/O Method Node that is configured with a Set Line Direction method overwrites the value you configured in the C Series Module Properties dialog box.

The execution of an FPGA I/O Node configured for output automatically configures the line for output and overwrites the values you configured in the C Series Module Properties dialog box or using the Set Output Enable method.

Configuring Line Direction Using the C Series Module Properties Dialog Box

Complete the following steps to configure the line direction of each digital port using the C Series Module Properties dialog box.

1. Configure the CompactRIO system, and add an NI 9403.
2. Right-click the NI 9403 in the **Project Explorer** window and select **Properties** from the shortcut menu to display the **C Series Module Properties** dialog box.
3. Select the channel(s) for which you want to configure the line direction from the **Channels** table. You can select more than one channel by holding the <Ctrl> or <Shift> key when selecting channels.
4. Select the direction for the channel(s) from the **Direction** pull-down menu.
5. Click the **OK** button.
6. Select **File**»**Save All** in the **Project Explorer** window.

Configuring Line Direction Using the FPGA I/O Method Node

Complete the following steps to configure the line direction of each digital port using the FPGA I/O Method Node.

1. Create FPGA I/O items for the channel of the NI 9403 for which you want to configure the line direction.
2. Place an FPGA I/O Method Node on the block diagram and configure it for this channel.
3. Click the **Method** section and select the Set Output Enable method from the shortcut menu.
4. Right-click the **Enable** input and select **Create»Control** from the shortcut menu. Each bit in the integer represents the line direction of one channel of the NI 9403.
5. On the front panel of the VI, click the **Enable** Boolean control to set it to *TRUE* if you want to set the line direction of the channel to digital output.

Configuring Line Direction of Multiple Channels Using the FPGA I/O Method Node

Complete the following steps to set the line direction of multiple channels using the FPGA I/O Method Node.

1. Create FPGA I/O items for the digital port of the NI 9403 that contains the channels you want to configure.
2. Place an FPGA I/O Method Node on the block diagram and configure it for this digital port.
3. Click the **Method** section and select the **Set Output Enable** method from the shortcut menu.
4. Right-click the **Enable** input and select **Create»Control** from the shortcut menu.
   - If you configured the FPGA I/O Method Node for the DIO7:0, DIO15:8, DIO23:16, or DIO31:24 digital port, the Enable control appears as an unsigned 8-bit integer.
   - If you configured the FPGA I/O Method Node for the DIO31:0 digital port, the Enable control appears as an unsigned 32-bit integer.

   Each bit in the integer represents the line direction of one channel of the NI 9403.
5. On the front panel of the VI, use the **Enable** control to enter the line direction for each channel in the port. Change a bit to 1 to set the line direction of the corresponding channel to digital output. Leave a bit as 0 to set the line direction of the corresponding channel to digital input. Refer to the table below for examples of what to enter in the **Enable** control if you configured the FPGA I/O Method Node for the DIO7:0 digital port.

Table 401. Enable Control Values (DIO7:0)

Channel Line Direction Configuration	Enable Control (Hex)	Enable Control (Binary)
Change all channels to input	0x00	0b00000000
Change all channels to output	0xFF	0b11111111
Change channel 0 to output	0x01	0b00000001
Change channels 0 through 5 to output	0x3F	0b00111111
Change channels 3 and 7 to output	0x88	0b10001000

Refer to the table below for examples of what to enter in the **Enable** control if you configured the FPGA I/O Method Node for the DIO31:0 digital port.

Table 402. Enable Control Values (DIO31:0)

Channel Line Direction Configuration	Enable Control (Hex)	Enable Control (Binary)
Change all channels to input	0x00000000	0b0000000000000000 0000000000000000
Change all channels to output	0xFFFFFFFF	0b1111111111111111 1111111111111111
Change channel 0 to output	0x00000001	0b0000000000000000 0000000000000001
Change channels 0 through 5 to output	0x0000003F	0b0000000000000000 0000000000111111
Change channels 3 and 7 to output	0x00000088	0b0000000000000000 0000000010001000

Related concepts:

- NI 9403 (FPGA Interface)

NI 9403 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project**

Explorer window to the block diagram of the VI. The I/O variables for the channels read and write Boolean values.

Module Channels

The NI 9402 has the following channels.

Table 403. Module Channels

Channel	Description
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. For the NI 9403, *x* is 0 to 31.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select Properties to display this dialog box. You can configure the following options.

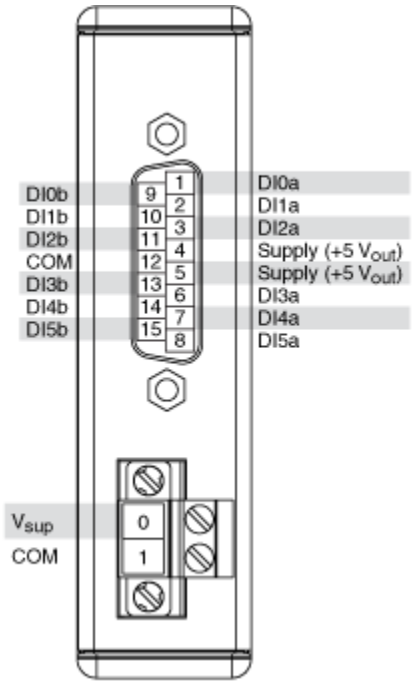
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- Channels—Specifies the channel(s) for which you want to configure settings.
- Initial Line Direction—Specifies the initial line direction of the selected channel(s) as input or output.

NI 9411

CompactRIO 6-Channel, Differential or TTL Digital Input Module

NI 9411 Pinout

Figure 89. NI 9411 Pinout



NI 9411 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 404. Terminals in Software

Terminal	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. The NI 9411 has DI channels 0 to 5.
DI5:0	Digital port consisting of channels 0 through 5. • Channel 5 is returned in bit 5. • Channel 0 is returned in bit 0. • Bits 6 and 7 return a zero.

Arbitration

This device supports only the Never Arbitrate option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 405. I/O Methods

Method	Description
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Module Methods

Table 406. Module Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

Note During the first 2 seconds after you reset the FPGA VI, the error terminals on the Check Status method may not correctly report certain types of errors.

I/O properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 407. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital input before a loop containing digital input starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready.

While the module is performing digital input within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Doing either of these actions causes the module to be unable to perform digital input and the **Ready** output of the Check Status method to return FALSE.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

Related concepts:

- Reading from C Series Channels

NI 9411 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return data as Boolean values.

Module Channels

The NI 9411 has the following channels.

Table 408. Module Channels

Channel	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. For the NI 9411, *x* is 0 to 5.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9411. Right-click the NI 9411 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

Specialty Digital Configuration

Use the Specialty Digital Configuration page of the C Series Module Properties dialog box to configure channels of this module for counter input or quadrature input.

Related concepts:

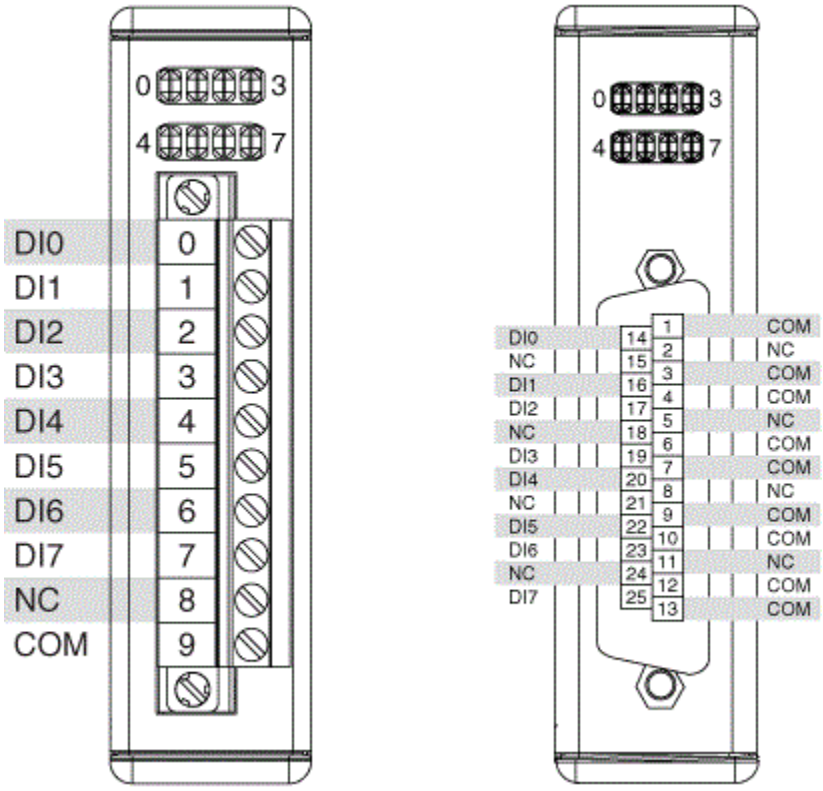
- Configuring Specialty Digital Functions
- Configuring Counters
- Configuring a Module for Quadrature Input

NI 9421

CompactRIO 8-Channel, 24 V, Sinking Digital Input Module

NI 9421 Pinout

Figure 90. NI 9421 Pinout



NI 9421 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 409. Terminals in Software

Terminal	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. The NI 9421 has DI channels 0 to 7.
DI7:0	Digital port consisting of channels 0 through 7.

Terminal	Description
	• Channel 7 is returned in the MSB. • Channel 0 is returned in the LSB.

Arbitration

This device supports only the Never Arbitrate option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 410. I/O Methods

Method	Description
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Method	Description
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Module Methods

Table 411. Module Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

Note During the first 2 seconds after you reset the FPGA VI, the error terminals on the Check Status method may not correctly report certain types of errors.

I/O properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 412. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital input before a loop containing digital input starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready.

While the module is performing digital input within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Doing either of these actions causes the module to be unable to perform digital input and the **Ready** output of the Check Status method to return FALSE.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

Related concepts:

- Reading from C Series Channels

NI 9421 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return data as Boolean values.

Module Channels

The NI 9421 has the following channels.

Table 413. Module Channels

Channel	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. For the NI 9421, *x* is 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9421. Right-click the NI 9421 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

Specialty Digital Configuration

Use the Specialty Digital Configuration page of the C Series Module Properties dialog box to configure channels of this module for counter input or quadrature input.

Related concepts:

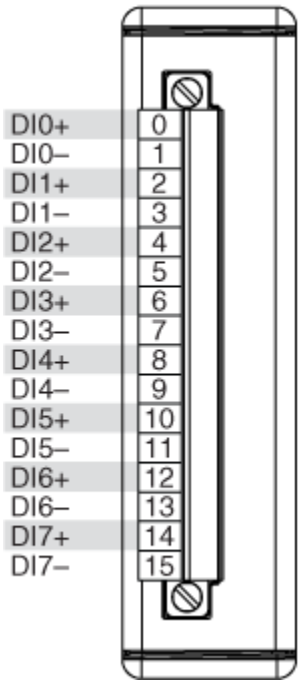
- Configuring Specialty Digital Functions
- Configuring Counters
- Configuring a Module for Quadrature Input

NI 9422

CompactRIO 8-Channel, Differential or TTL Digital Input Module

NI 9422 Pinout

Figure 91. NI 9422 Pinout



NI 9422 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 414. Terminals in Software

Terminal	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. The NI 9422 has DI channels 0 to 7.
DI7:0	Digital port consisting of channels 0 through 7. • Channel 7 is returned in the MSB. • Channel 0 is returned in the LSB.

Arbitration

This device supports only the Never Arbitrate option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 415. I/O Methods

Method	Description
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Module Methods

Table 416. Module Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

Note During the first 2 seconds after you reset the FPGA VI, the error terminals on the Check Status method may not correctly report certain types of errors.

I/O properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 417. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital input before a loop containing digital input starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready.

While the module is performing digital input within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Doing either of these actions causes the module to be unable to perform digital input and the **Ready** output of the Check Status method to return FALSE.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

Related concepts:

- Reading from C Series Channels

NI 9422 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return data as Boolean values.

Module Channels

The NI 9422 has the following channels.

Table 418. Module Channels

Channel	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. For the NI 9422, *x* is 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9422. Right-click the NI 9422 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

Specialty Digital Configuration

Use the Specialty Digital Configuration page of the C Series Module Properties dialog box to configure channels of this module for counter input or quadrature input.

Related concepts:

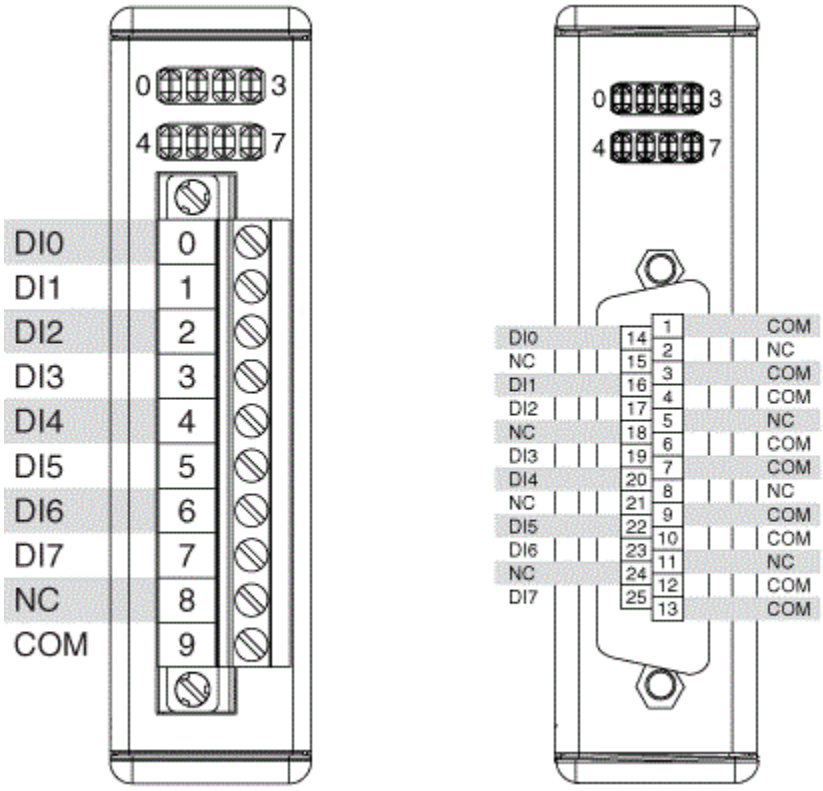
- Configuring Specialty Digital Functions
- Configuring Counters
- Configuring a Module for Quadrature Input

NI 9423

CompactRIO 8-Channel, 24 V, High-Speed Digital Input Module

NI 9423 Pinout

Figure 92. NI 9423 Pinout



NI 9423 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 419. Terminals in Software

Terminal	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. The NI 9423 has DI channels 0 to 7.
DI7:0	Digital port consisting of channels 0 through 7.

Terminal	Description
	• Channel 7 is returned in the MSB. • Channel 0 is returned in the LSB.

Arbitration

This device supports only the Never Arbitrate option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 420. I/O Methods

Method	Description
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Method	Description
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Module Methods

Table 421. Module Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

Note During the first 2 seconds after you reset the FPGA VI, the error terminals on the Check Status method may not correctly report certain types of errors.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 422. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital input before a loop containing digital input starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready.

While the module is performing digital input within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Doing either of these actions causes the module to be unable to perform digital input and the **Ready** output of the Check Status method to return FALSE.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

Related concepts:

- Reading from C Series Channels

NI 9423 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return data as Boolean values.

Module Channels

The NI 9423 has the following channels.

Table 423. Module Channels

Channel	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. For the NI 9423, *x* is 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9423. Right-click the NI 9423 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

Specialty Digital Configuration

Use the Specialty Digital Configuration page of the C Series Module Properties dialog box to configure channels of this module for counter input or quadrature input.

Related concepts:

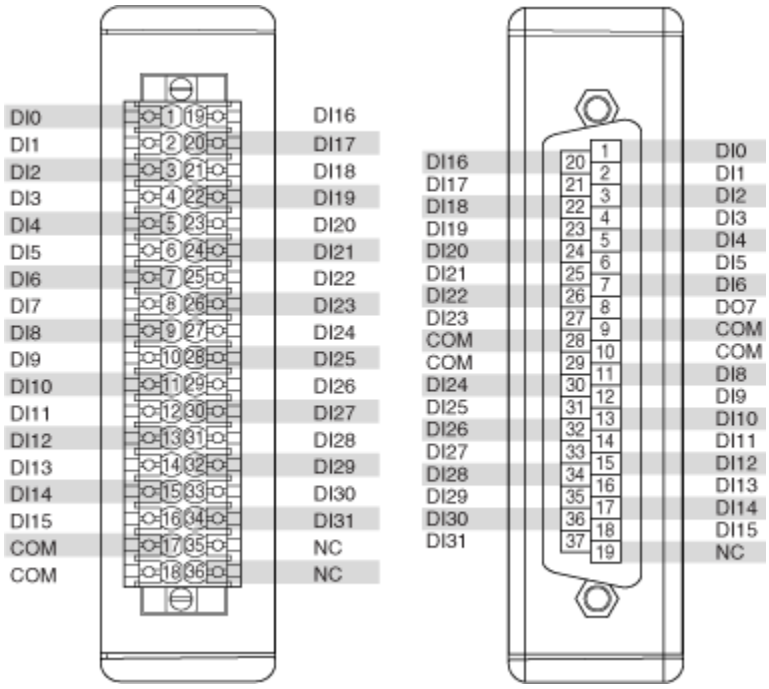
- Configuring Specialty Digital Functions
- Configuring Counters
- Configuring a Module for Quadrature Input

NI 9425

CompactRIO 32-Channel, 24 V, Sinking Digital Input Module

NI 9425 Pinout

Figure 93. NI 9425 Pinout



NI 9425 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 424. Terminals in Software

Terminal	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. The NI 9425 has DI channels 0 to 31.
DI7:0	Digital port consisting of channels 0 through 7. • Channel 7 is returned in the MSB. • Channel 0 is returned in the LSB.

Terminal	Description
DI15:8	Digital port consisting of channels 8 through 15. • Channel 15 is returned in the MSB. • Channel 8 is returned in the LSB.
DI23:16	Digital port consisting of channels 16 through 23. • Channel 23 is returned in the MSB. • Channel 16 is returned in the LSB.
DI31:24	Digital port consisting of channels 24 through 31. • Channel 31 is returned in the MSB. • Channel 24 is returned in the LSB.
DI31:0	Digital port consisting of channels 0 through 31. • Channel 31 is returned in the MSB. • Channel 0 is returned in the LSB.

Arbitration

This device supports only the Arbitrate if Multiple Requestors Only option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 425. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Reading from C Series Channels

NI 9425 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return data as Boolean values.

Module Channels

The NI 9425 has the following channels.

Table 426. Module Channels

Channel	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. For the NI 9425, *x* is 0 to 31.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9425. Right-click the NI 9425 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

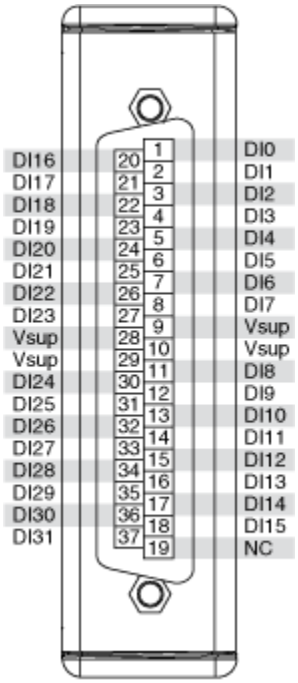
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9426

CompactRIO 32-Channel, 24 V, Sourcing Digital Input Module

NI 9426 Pinout

Figure 94. NI 9426 Pinout



NI 9426 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 427. Terminals in Software

Terminal	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. The NI 9426 has DI channels 0 to 31.
DI7:0	Digital port consisting of channels 0 through 7. • Channel 7 is returned in the MSB. • Channel 0 is returned in the LSB.
DI15:8	Digital port consisting of channels 8 through 15. • Channel 15 is returned in the MSB. • Channel 8 is returned in the LSB.
DI23:16	Digital port consisting of channels 16 through 23. • Channel 23 is returned in the MSB. • Channel 16 is returned in the LSB.
DI31:24	Digital port consisting of channels 24 through 31. • Channel 31 is returned in the MSB. • Channel 24 is returned in the LSB.
DI31:0	Digital port consisting of channels 0 through 31. • Channel 31 is returned in the MSB. • Channel 0 is returned in the LSB.

Arbitration

This device supports only the Arbitrate if Multiple Requestors Only option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 428. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Reading from C Series Channels

NI 9426 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return data as Boolean values.

Module Channels

The NI 9426 has the following channels.

Table 429. Module Channels

Channel	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. For the NI 9426, *x* is 0 to 31.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9426. Right-click the NI 9426 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

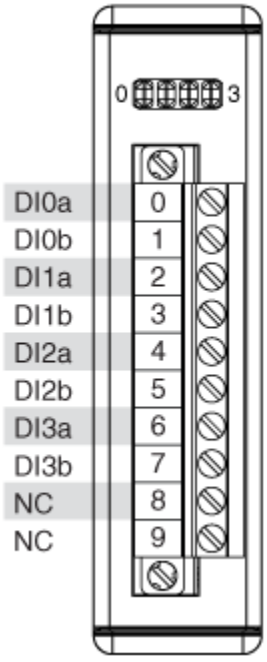
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9435

CompactRIO 4-Channel, AC/DC Universal Digital Input Module

NI 9435 Pinout

Figure 95. NI 9435 Pinout



NI 9435 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 430. Terminals in Software

Terminal	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. The NI 9435 has DI channels 0 to 3.
DI3:0	Digital port consisting of channels 0 through 3. and . • Channel 3 is returned in bit 3. • Channel 0 is returned in bit 0. • Bits 4 through 7 return a zero.

Arbitration

This device supports only the Never Arbitrate option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 431. I/O Methods

Method	Description
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Method	Description
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Module Methods

Table 432. Module Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

Note During the first 2 seconds after you reset the FPGA VI, the error terminals on the Check Status method may not correctly report certain types of errors.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 433. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital input before a loop containing digital input starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready.

While the module is performing digital input within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Doing either of these actions causes the module to be unable to perform digital input and the **Ready** output of the Check Status method to return FALSE.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

Related concepts:

- Reading from C Series Channels

NI 9435 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return data as Boolean values.

Module Channels

The NI 9435 has the following channels.

Table 434. Module Channels

Channel	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. For the NI 9435, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9435. Right-click the NI 9435 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

Specialty Digital Configuration

Use the Specialty Digital Configuration page of the C Series Module Properties dialog box to configure channels of this module for counter input or quadrature input.

Related concepts:

- Configuring Specialty Digital Functions
- Configuring Counters

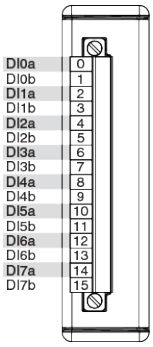
- Configuring a Module for Quadrature Input

NI 9436

CompactRIO 8 DI, ±100 VDC to ±250 VDC/100 VAC to 250 VAC, Sinking/Sourcing, 10 ms Input Module

NI 9436 Pinout

Figure 96. NI 9436 Pinout



NI 9436 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 435. Terminals in Software

Terminal	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. The NI 9436 has DI channels 0 to 7.
DI7:0	Digital port consisting of channels 0 through 7. • Channel 7 is returned in the MSB. • Channel 0 is returned in the LSB.

Arbitration

This device supports only the Never Arbitrate option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 436. I/O Methods

Method	Description
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Module Methods

Table 437. Module Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

Note During the first 2 seconds after you reset the FPGA VI, the error terminals on the Check Status method may not correctly report certain types of errors.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 438. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital input before a loop containing digital input starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready.

While the module is performing digital input within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Doing either of these actions causes the module to be unable to perform digital input and the **Ready** output of the Check Status method to return FALSE.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

Related concepts:

- Reading from C Series Channels

NI 9436 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels read and write Boolean values.

Module Channels

The NI 9436 has the following channels.

Table 439. Module Channels

Channel	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. For the NI 9436, *x* is 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9436. Right-click the NI 9436 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

Specialty Digital Configuration

Use the Specialty Digital Configuration page of the C Series Module Properties dialog box to configure channels of this module for counter input or quadrature input.

Related concepts:

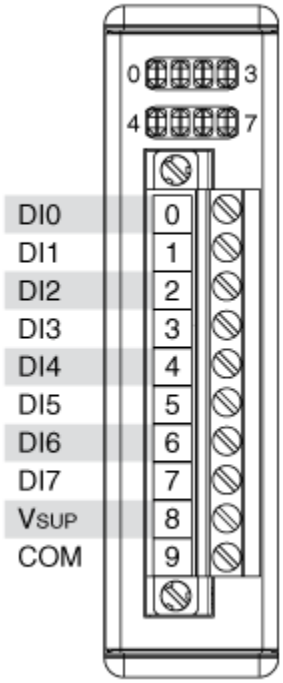
- Configuring Specialty Digital Functions
- Configuring Counters
- Configuring a Module for Quadrature Input

NI 9437

CompactRIO 8-Channel, 250 V, Sinking Digital Input Module

NI 9437 Pinout

Figure 97. NI 9437 Pinout



NI 9437 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 440. Terminals in Software

Terminal	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. The NI 9437 has DI channels 0 to 7.
DI7:0	Digital port consisting of channels 0 through 7. • Channel 7 is returned in the MSB. • Channel 0 is returned in the LSB.

Arbitration

This device supports only the Never Arbitrate option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 441. I/O Methods

Method	Description
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Module Methods

Table 442. Module Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

Note During the first 2 seconds after you reset the FPGA VI, the error terminals on the Check Status method may not correctly report certain types of errors.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 443. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital input before a loop containing digital input starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready.

While the module is performing digital input within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Doing either of these actions causes the module to be unable to perform digital input and the **Ready** output of the Check Status method to return FALSE.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

Related concepts:

- Reading from C Series Channels

NI 9437 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels return calibrated floating-point data in volts.

Module Channels

The NI 9437 has the following channels.

Table 444. Module Channels

Channel	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. For the NI 9437, *x* is 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9437. Right-click the NI 9437 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

Digital Output Modules

Table 445. C Series Digital Output Modules

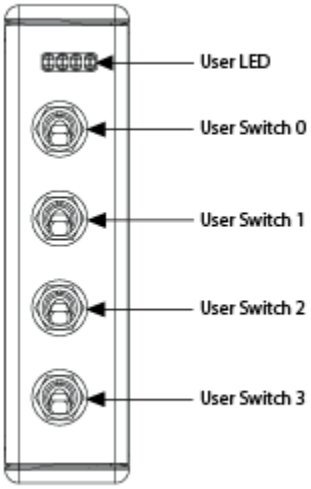
C Series Digital Output Module	Description
NI-9344	4-Channel C Series User Interface Module
NI-9375	30 V, 32-Channel (Sinking Input, Sourcing Output), 7 µs (Input)/500 µs (Output) C Series Digital Module
NI-9381	8 AI, 8 AO, 4 DIO, 0 V to 5 V, C Series Multifunction I/O Module
NI-9401	5 V/TTL, 8 Bidirectional Channels, 100 ns C Series Digital Module
NI-9402	LVTTL, 4 Bidirectional Channels, 55 ns C Series Digital Module
NI-9403	5 V/TTL, 32 Bidirectional Channels, 7 µs C Series Digital Module
NI-9470	5 VDC to 30 VDC, 8-Channel (Sourcing Output), 2 kHz C Series Digital Module
NI-9472	24 V, 8-Channel (Sourcing Output), 100 µs C Series Digital Module
NI-9474	30 V, 8-Channel (Sourcing Output), 1 µs C Series Digital Module
NI-9475	60 V, 8-Channel (Sourcing Output), 1 µs C Series Digital Module
NI-9476	36 V, 32-Channel (Sourcing Output), 500 µs C Series Digital Module
NI-9477	60 V, 32-Channel (Sinking Output), 8 µs C Series Digital Module
NI-9478	50 V, 16-Channel (Sinking Output), 50 µs C Series Digital Module
NI-9481	4-Channel, SPST Relay, 60 VDC(1 A)/ 250 Vrms (2 A) C Series Relay Output Module
NI-9482	4-Channel, SPST Relay, 60 VDC (1 A)/250 VAC (1.5 A) C Series Relay Output Module
NI-9485	8-Channel, SSR Relay, 60 VDC/30 Vrms, 750 mA C Series Relay Output Module

NI 9344

CompactRIO 4-Channel Switch Input and 4-Channel LED Output Module

NI 9344 Front Panel

Figure 98. NI 9344 Front Panel



NI 9344 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading and writing with this device. When you write to an NI 9344 channel, the FPGA I/O Node does not automatically enable the channel for output.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 446. Terminals in Software

Terminal	Description
User LED*x*	LED output channel *x*, where *x* is the number of the channel. The NI 9344 has LED channels 0 to 3.
User	Switch input channel *x*, where *x* is the number of the channel. The NI 9344 has Switch

Terminal	Description
Switch x	channels 0 to 3.
User LED 3:0	Digital output port consisting of channels 0 through 3. Channel 3 is returned in the MSB, and channel 0 is returned in the LSB.
User Switch 3:0	Digital input port consisting of channels 0 through 3. Channel 3 is returned in the MSB, and channel 0 is returned in the LSB.

Arbitration

You can configure the arbitration settings for digital output channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

Digital input channels of this device support only the **Never Arbitrate** option for arbitration. You cannot configure arbitration settings for the digital input channels of this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Method	Description
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge.

Method	Description
	• A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.

Module Methods

Use the FPGA I/O Method Node to access the following module methods for this device.

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

Note During the first 2 seconds after you reset the FPGA VI, the error terminals on this method may not correctly report certain types of errors.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Property	Description
Module ID	Returns the module ID, 0x77C5.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the channels of this device in the Advanced Configuration dialog box. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital I/O before a loop containing digital I/O starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready. The module also might ignore or delay digital output operations if it is not ready.

While the module is performing digital I/O within a single-cycle Timed Loop, do not perform property reads, use the Set Line Direction method, or remove the module from the chassis. Doing any of these actions causes the module to be unable to perform digital I/O and the **Ready** output of the Check Status method to return *FALSE*.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates that are multiples of 40 MHz, such as:

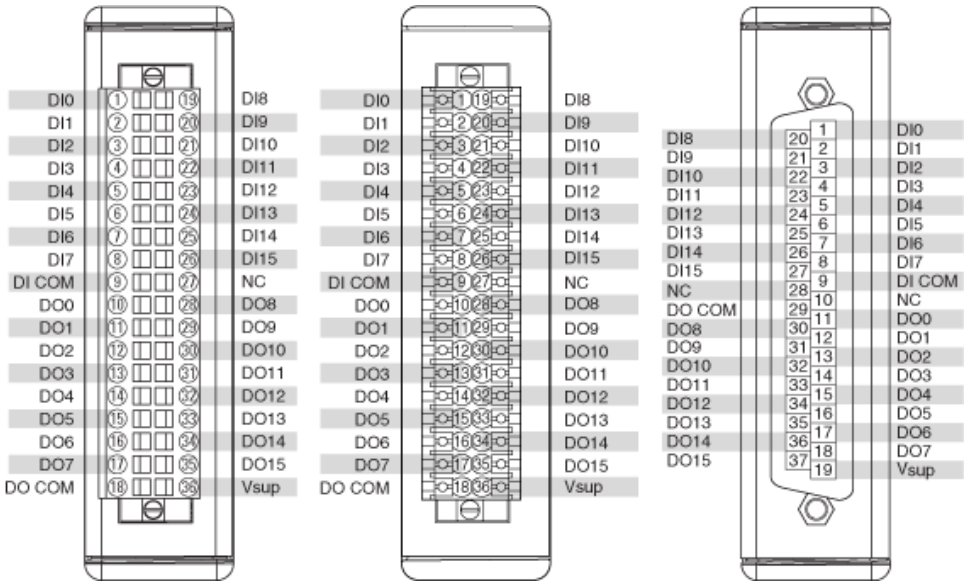
- 40 MHz
- 80 MHz
- 120 MHz
- 160 MHz

NI 9375

CompactRIO 16-Channel Digital Input and 16-Channel Digital Output

NI 9375 Pinout

Figure 99. NI 9375 Pinout



NI 9375 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading and writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 447. Terminals in Software

Terminal	Description
DI*x*	Digital input channel *x*, where *x* is the number of the channel. The NI 9375 has DI channels 0 to 15.
DI7:0	Digital port consisting of DI channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
DI15:8	Digital port consisting of DI channels 8 through 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
DI15:0	Digital port consisting of DI channels 0 through 15. Channel 15 is returned in the MSB, and channel 0 is returned in the LSB.
DO*x*	Digital output channel *x*, where *x* is the number of the channel. The NI 9375 has DO channels 0 to 15.
DO7:0	Digital port consisting of DO channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
DO15:8	Digital port consisting of DO channels 8 through 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
DO15:0	Digital port consisting of DO channels 0 through 15. Channel 15 is returned in the MSB, and channel 0 is returned in the LSB.

Caution DI and DO operations that are run concurrently may exhibit jitter. Access both subsystems using a single I/O Node to avoid jitter in your application. Refer to *Avoiding Timing Uncertainty* for more information.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

This device does not support any module methods.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Avoiding Timing Uncertainty with the NI 9375 (FPGA Interface)

To avoid introducing timing uncertainty into DI and DO Node calls, adhere to the following guidelines:

- Use a single I/O Node to access DI and DO operations to ensure proper sequencing.
- Do not perform the following operations concurrently:
  - DI Node call
  - DO Node call

For an example of the recommended way to use the DIO subsystems, refer to the NI 9375 Digital Port Input Output VI in `labview\examples\CompactRIO\Module Specific\NI 9375\NI 9375 Digital Port Input Output\NI 9375 Digital Port Input Output.lvproj`.

NI 9375 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI.

The I/O variables for the channels read and write Boolean values.

Module Channels

The NI 9375 has the following channels.

Table 448. Module Channels

Channel	Description
DIx	Digital input channel x, where x is the number of the channel. For the NI 9375, x is 0 to 15.
DOx	Digital output channel x, where x is the number of the channel. For the NI 9375, x is 0 to 15.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9375. Right-click the NI 9375 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot

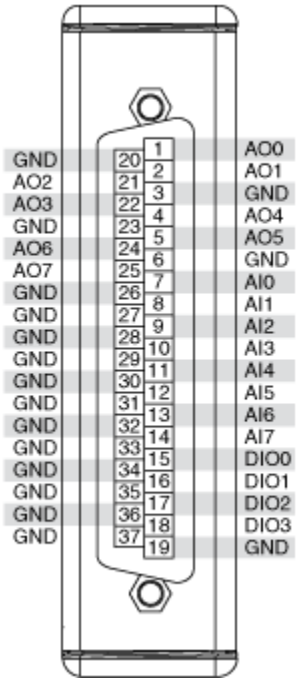
number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9381

CompactRIO 0 V to 5 V, 12-bit, Multifunction Input/Output Module

NI 9381 Pinout

Figure 100. NI 9381 Pinout



Running AI and AO Operations Concurrently on the NI 9381

AI and AO operations that are run concurrently may exhibit jitter. Access both subsystems using a single I/O Node to avoid jitter in your application. Refer to the *Avoiding Timing Uncertainty* topic for more information.

NI 9381 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading and writing with this device.

Table 449. Terminals in Software

Terminal	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. The NI 9381 has AI channels 0 to 7.
AO*x*	Analog output channel *x*, where *x* is the number of the channel. The NI 9381 has AO channels 0 to 3.
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. The NI 9381 has DIO channels 0 to 3.
DIO3:0	Digital port consisting of channels 0 through 3. Channel 3 is returned in the MSB, and channel 0 is returned in the LSB.

Caution AI and AO operations that are run concurrently may exhibit jitter. Access both subsystems using a single I/O Node to avoid jitter in your application. Refer to *Avoiding Timing Uncertainty* for more information.

Arbitration

Analog input and analog output channels of this device support only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for analog input and analog output channels of this device.

You can configure the arbitration settings for digital output channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

Digital input channels of this device support only the **Never Arbitrate** option for arbitration. You cannot configure arbitration settings for the digital input channels of this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for the DIO channels of this device.

Table 450. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node (FPGA Module) topic for a description of this method. This method is available only when you set the DIO line direction to output. You can set the DIO line direction in the C Series Module Properties dialog box.
Set Output Enable	Sets the line direction of the digital channel or the DIO3:0 digital port. Refer to the FPGA I/O Method Node (FPGA Module) topic for more information on this method. This method can introduce jitter in an analog input or analog output loop. This method is available only when you enable programmable DIO line changes on the module. Enable programmable DIO line changes on the module in the C Series Module Properties dialog box.
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.

Method	Description
	Note This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely.

Method	Description
	• A positive value causes the method to wait for that number of clock ticks before timing out. **Note** This method is not available on the DIO3:0 port. For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.

Module Method

Use the FPGA I/O Method Node to access the following module method for this device.

Table 451. Module Method

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for AI and AO channels of this device.

Table 452. I/O Properties

Property	Description
LSB Weight	Returns the LSB weight in nV/LSB for the channel. Use this value to convert and calibrate NI 9381 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.
Offset	Returns the calibration offset in nV for the channel. Use this value to convert and calibrate NI 9381 data if you set the **Calibration Mode** to Raw in the C Series Module Properties dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 453. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop.

- Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.
- Configure the number of output synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box.

This device supports the Number of Synchronizing Registers for Output Data synchronizing register option when used in SCTL output. This option supports the same functionality as the **Number of Synchronizing Registers for Output Data** option described in the Advanced Code Generation FPGA I/O Properties Page (FPGA Module) topic, with the exception that you can use this option only in a single-cycle Timed Loop. You can implement either 0 or 1 synchronizing registers inside the single-cycle Timed Loop. I you configure 0 synchronizing registers outside of the single-cycle Timed Loop, the FPGA VI implements 1 synchronizing register by default.

When the device is within a single-cycle Timed Loop, it must be ready to perform digital I/O before a loop containing digital I/O starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready. The module also might ignore or delay digital output operations if it is not ready.

While the digital is performing digital I/O within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Performing any of these actions causes the module to be unable to perform digital I/O and the **Ready** output of the Check Status method to return *FALSE*.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as:

- 40 MHz
- 80 MHz
- 120 MHz

Avoiding Timing Uncertainty with the NI 9381 (FPGA Interface)

Follow the guidelines below to avoid introducing timing uncertainty into AI node, AO Node, DO Node, and Set Output Data method calls. Failing to follow these guidelines may result in lost data points for DO Node calls or Set Output Data method calls when either of them are made from inside a single-cycle Timed Loop.

- Use a single I/O Node to access AI and AO operations to ensure proper sequencing.
- Do not run the Set Output Enable method when an AI, AO, or DO subsystems is active.
- Do not perform the following operations concurrently:
  - AI Node call
  - AO Node call
  - Set Output Enable method call
  - Automatic line direction change when a DO call is made under the following conditions:
    - The call is made from outside of a single-cycle Timed Loop.
    - The **Allow programmatic DIO line direction** checkbox is enabled in the C Series Module Properties dialog box.
    - The DIO channel is configured as an input at the time when the DO node is called.

For an example of the recommended way to use the AI, AO , and DIO subsystems, refer to the NI 9381 Advanced IO VI in the `labview\examples\CompactRIO\Module Specific\NI 9381\NI 9381 Advanced IO\NI 9381 Advanced IO.lvproj`.

Converting Voltage Values to Binary Values for the NI 9381 (FPGA Interface)

Set the **Calibration Mode** to Calibrated in the C Series Module Properties dialog box for the NI 9381 if you want the FPGA I/O Node to accept fixed-point data in units of volts when writing to the module.

When **Calibration Mode** is set to Raw when writing to the module, the FPGA I/O Node accepts only binary values. Convert output voltage values to binary values before writing them to the module. Convert these values in the host VI.

Using an Equation to Convert Voltage to Binary

Use the following equation in the host VI to convert the analog output values to binary values:

$$Binary\ Value = (Voltage\ Value - Offset) \div LSB\ Weight$$

Where

- *Binary Value* is the value returned by the FPGA I/O Node.
- *Voltage Value* is the voltage in *V* that you want the channel to output
- *Offset* is the value returned by the Offset property
- *LSB Weight* is the value returned by the LSB Weight property

NI recommends using calibrated values for analog output. To convert calibrated voltage values, use the FPGA I/O Property Node to read the LSB Weight and Offset properties. If you do not want to read the LSB Weight and Offset values from the module, convert uncalibrated voltage values by using the following values for *Offset* and *LSB Weight*:

$$Offset = -5249$$

$$LSB\ Weight = 5.105\ V \div 2^{DAC\ Resolution}$$

Where

- *DAC Resolution* is the DAC resolution value. Refer to the NI 9381 hardware documentation for the DAC resolution value.

NI 9381 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI.

- The I/O variables for the AI channels return floating-point data in volts.
- The I/O variables for the AO channels write floating-point data in volts.
- The I/O variables for the DIO channels read and write Boolean values.

Module Channels

The NI 9381 has the following channels.

Table 454. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9381, *x* is 0 to 7.
AO*x*	Analog output channel *x*, where *x* is the number of the channel. For the NI 9381, *x* is 0 to 7.
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. For the NI 9381, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9381. Right-click the NI 9381 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- Channels—Specifies the channel(s) for which you want to configure settings.
- Direction—Specifies the initial line direction of the selected channel(s) as input or output.

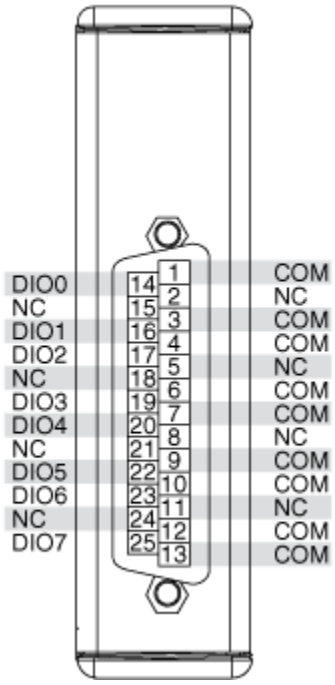
Note If you are using direct access to read from or write to AI or AO channels of the NI 9381, the conversion time is somewhat longer than 50 µs.

NI 9401

CompactRIO 8-Channel, TTL Digital Input/Output Module

NI 9401 Pinout

Figure 101. NI 9401 Pinout



NI 9401 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading and writing with this device.

When you write to an NI 9401 channel, the FPGA I/O Node does not automatically enable the channel for output.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Terminal	Description
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. The NI 9401 has DIO channels 0 to 7.
DIO3:0	Digital port consisting of channels 0 through 3. Channel 3 is returned in the MSB, and channel 0 is returned in the LSB.
DIO7:4	Digital port consisting of channels 4 through 7. Channel 7 is returned in the MSB, and channel 4 is returned in the LSB.
DIO7:0	Digital port consisting of channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.

Arbitration

You can configure the arbitration settings for digital output channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

Digital input channels of this device support only the **Never Arbitrate** option for arbitration. You cannot configure arbitration settings for the digital input channels of this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Method	Description
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout**

Method	Description
High Level	input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Module Methods

Use the FPGA I/O Method Node to access the following module methods for this device.

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready. **Note** During the first 2 seconds after you reset the FPGA VI, the error terminals on this method may not correctly report certain types of errors.
Set Line Direction	Sets the direction of one port to input or output.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this

device.

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the channels of this device in the Advanced Configuration dialog box. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital I/O before a loop containing digital I/O starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready. The module also might ignore or delay digital output operations if it is not ready.

While the module is performing digital I/O within a single-cycle Timed Loop, do not perform property reads, use the Set Line Direction method, or remove the module from the chassis. Doing any of these actions causes the module to be unable to perform digital I/O and the **Ready** output of the Check Status method to return *FALSE*.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as:

- 40 MHz
- 80 MHz
- 120 MHz

Configuring the Initial Line Direction for the NI 9401 (FPGA Interface)

Each digital port on the NI 9401 is initially configured as a digital input. You can configure the initial line direction for each port on the NI 9401 at edit time using the C Series Module Properties dialog box.

Programmatically change the line direction for each port at run time using the FPGA I/O Method Node. The execution of an I/O Method Node that is configured with a Set Line Direction method overwrites the value you configured in the C Series Module Properties dialog box.

Configuring Line Direction Using the C Series Module Properties Dialog Box

Complete the following steps to configure the line direction of each digital port using the C Series Module Properties dialog box.

1. Configure the CompactRIO system, and add an NI 9401.
2. Right-click the NI 9401 in the **Project Explorer** window and select **Properties** from the shortcut menu to display the C Series Module Properties dialog box.
3. Select the direction for each port from the **Initial Line Direction** pull-down menus.
4. Click the **OK** button.
5. Select **File»Save All** in the **Project Explorer** window.

Configuring Line Direction Using the FPGA I/O Method Node

Complete the following steps to configure the line direction of each digital port using the FPGA I/O Method Node.

1. Place an FPGA I/O Method Node on the block diagram and configure it for the NI 9401.
2. Click the **Method** section and select the Set Line Direction method from the shortcut menu.
3. Right-click each digital port input and select **Create»Control** from the shortcut menu.
4. On the front panel of the VI, select the direction for each port from the digital port pull-down menus.

NI 9401 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels read and write Boolean values.

Module Channels

The NI 9401 has the following channels.

Table 455. Module Channels

Channel	Description
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. For the NI 9401, *x* is 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9401. Right-click the NI 9401 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the Project Explorer window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- Initial Line Direction—Specifies the initial line direction of each four-channel port as input or output.

Specialty Digital Configuration

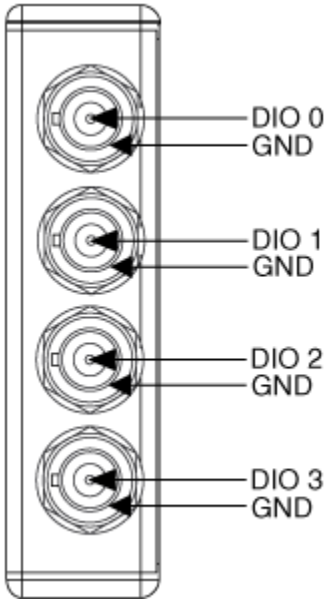
Use the Specialty Digital Configuration page of the C Series Module Properties dialog box to configure channels of this module for counter input, counter-driven output, pulse-width modulation output, or quadrature input.

NI 9402

CompactRIO 4-Channel, LVTTL Digital Input/Output Module

NI 9402 Pinout

Figure 102. NI 9402 Pinout



NI 9402 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading and writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 456. Terminals in Software

Terminal	Description
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. The NI 9402 has DIO channels 0 to 3.
DIO3:0	Digital port consisting of channels 0 through 3. Channel 3 is returned in the MSB, and channel 0 is returned in the LSB.

Arbitration

You can configure the arbitration settings for digital output channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

Digital input channels of this device support only the **Never Arbitrate** option for arbitration. You cannot configure arbitration settings for the digital input channels of this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 457. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node (FPGA Module) topic for a description of this method.
Set Output Enable	Sets the line direction of the digital channel or the DIO3:0 digital port. Refer to the FPGA I/O Method Node (FPGA Module) topic for more information on this method.
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Method	Description
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.

Module Method

Use the FPGA I/O Method Node to access the following module method for this device.

Table 458. Module Method

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready. **Note** During the first 2 seconds after you reset the FPGA VI, the error terminals on this method may not correctly report certain types of errors.

I/O properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 459. Module Properties

Property	Description
Module ID	Returns the module ID.

Property	Description
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box. Configure the number of output synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box.

The NI 9402 supports the following output synchronizing register options:

- **Number of Synchronizing Registers for Output Data when used in SCTL**
- **Number of Synchronizing Registers for Output Enable when used in SCTL**

These two options support the same functionality as the **Number of Synchronizing Registers for Output Data** and **Number of Synchronizing Registers for Output Enable** options described in the Advanced Code Generation FPGA I/O Properties Page (FPGA Module) topic, with the exception that you can use these options only in a single-cycle Timed Loop. You can implement either 0 or 1 synchronizing registers inside the single-cycle Timed Loop, however, if you configure 0 synchronizing registers outside of the single-cycle Timed Loop, the FPGA VI implements 1 synchronizing register by default.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital I/O before a loop containing digital I/O starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready. The module also might ignore or delay digital output operations if it is not ready.

While the module is performing digital I/O within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Doing any of these actions causes the module to be unable to perform digital I/O and the **Ready** output of the Check Status method to return FALSE. FPGA

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as:

- 40 MHz
- 80 MHz
- 120 MHz

Configuring the Line Direction for the NI 9402 (FPGA Interface)

Each digital port on the NI 9402 is initially configured as a digital input. You can configure the initial line direction for each port on the NI 9402 at edit time using the C Series Module Properties dialog box.

Programmatically change the line direction for each port at run time using the FPGA I/O Method Node. The execution of an I/O Method Node that is configured with a Set Line Direction method overwrites the value you configured in the C Series Module Properties dialog box.

The execution of an FPGA I/O Node configured for output automatically configures the line for output and overwrites the values you configured in the C Series Module Properties dialog box or using the Set Output Enable method.

Configuring Line Direction Using the C Series Module Properties Dialog Box

Complete the following steps to configure the line direction of each digital port using the C Series Module Properties dialog box.

1. Configure the CompactRIO system, and add an NI 9402.
2. Right-click the NI 9402 in the **Project Explorer** window and select **Properties** from the shortcut menu to display the **C Series Module Properties** dialog box.
3. Select the channel(s) for which you want to configure the line direction from the **Channels** table. You can select more than one channel by holding the <Ctrl> or <Shift> key when selecting channels.
4. Select the direction for the channel(s) from the **Direction** pull-down menu.
5. Click the **OK** button.

6. Select **File»Save All** in the **Project Explorer** window.

Configuring Line Direction Using the FPGA I/O Method Node

Complete the following steps to configure the line direction of each digital port using the FPGA I/O Method Node.

1. Create FPGA I/O items for the channel of the NI 9402 for which you want to configure the line direction.
2. Place an FPGA I/O Method Node on the block diagram and configure it for this channel.
3. Click the **Method** section and select the Set Line Direction method from the shortcut menu.
4. Right-click the **Enable** input and select **Create»Control** from the shortcut menu.
5. On the front panel of the VI, click the **Enable** Boolean control to set it to *TRUE* if you want to set the line direction of the channel to digital output.

Configuring Line Direction of Multiple Channels Using the FPGA I/O Method Node

Complete the following steps to set the line direction of multiple channels using the FPGA I/O Method Node.

1. Create FPGA I/O items for the DIO3:0 digital port of the NI 9402.
2. Place an FPGA I/O Method Node on the block diagram and configure it for the DIO3:0 digital port.
3. Click the Method section and select the **Set Output Enable** method from the shortcut menu.
4. Right-click the **Enable** input and select **Create»Control** from the shortcut menu. The **Enable** control appears as an unsigned 8-bit integer. Each bit in the integer represents the line direction of one channel of the NI 9402.
5. On the front panel of the VI, use the **Enable** control to enter the line direction for each channel in the port. Change a bit to 1 to set the line direction of the corresponding channel to digital output. Leave a bit as 0 to set the line direction of the corresponding channel to digital input. Refer to the table below for examples of what to enter in the **Enable** control.

Table 460. Enable Control Values

Channel Line Direction Configuration	Enable Control(Floating Point)	Enable Control(Binary)	Enable Control(Hex)
Change all channels to input	0	0000	0x00
Change all channels to output	15	1111	0x0F
Change channel 0 to output	1	0001	0x01

NI 9402 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels read and write Boolean values.

Module Channels

The NI 9402 has the following channels.

Table 461. Module Channels

Channel	Description
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. For the NI 9402, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the Project Explorer window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

- Direction—Specifies the initial line direction of the selected channel(s) as input or output.

Specialty Digital Configuration

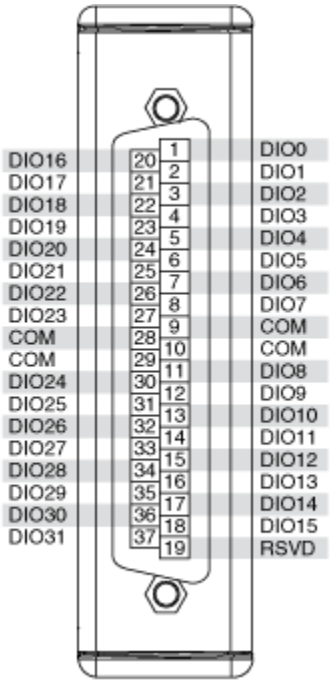
Use the Specialty Digital Configuration page of the C Series Module Properties dialog box to configure channels of this module for counter input, counter-driven output, pulse-width modulation output, or quadrature input.

NI 9403

CompactRIO 32-Channel, TTL Digital Input/Output Module

NI 9403 Pinout

Figure 103. NI 9403 Pinout



NI 9403 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for reading and writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 462. Terminals in Software

Terminal	Description
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. The NI 9403 has DIO channels 0 to 31.
DIO7:0	Digital port consisting of channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
DIO15:8	Digital port consisting of channels 8 through 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
DIO23:16	Digital port consisting of channels 16 through 23. Channel 23 is returned in the MSB, and channel 16 is returned in the LSB.
DIO31:24	Digital port consisting of channels 24 through 31. Channel 31 is returned in the MSB, and channel 24 is returned in the LSB.
DIO31:0	Digital port consisting of channels 0 through 31. Channel 31 is returned in the MSB, and channel 0 is returned in the LSB.

Arbitration

This device supports only the Arbitrate if Multiple Requestors Only and Never Arbitrate options for arbitration. The default arbitration setting is **Arbitrate if Multiple Requestors Only**.

If you are sure that the design of the FPGA VI will never allow more than one digital function to execute at the same time, even on different channels, enable the **Disable Arbitration** checkbox on the C Series Module Properties dialog box to disable arbitration and reduce the amount of FPGA logic used by VIs. If more than one digital function could execute simultaneously in the FPGA VI, leave the box unchecked.

I/O methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 463. I/O methods

Method	Description
Read and Set Output Data	First returns the data read from the digital line or port, then writes data to the line or port without enabling it. You can use the Read and Set Output Data method to optimize performance when performing successive reads and writes on a DIO resource. For the NI 9403, this method is faster than using a read FPGA I/O Node and a Set Output Data method. The data type of the output and input depends on the I/O item. • If the I/O item is a digital line, **Data to Set** requires a Boolean data type. • If the I/O item is a digital port, **Data to Set** requires a numeric data type.
Set Output Data	Refer to the FPGA I/O Method Node (FPGA Module) in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Set Output Enable	Sets the line direction of the digital channel or port. Refer to the FPGA I/O Method Node (FPGA Module) in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.

Module Methods

This device does not support any module methods.

I/O properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 464. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Configuring the Line Direction for the NI 9403 (FPGA Interface)

Each digital port on the NI 9403 is initially configured as a digital input. You can configure the initial line direction for each port on the NI 9403 at edit time using the C Series Module Properties dialog box.

Programmatically change the line direction for each port at run time using the FPGA I/O Method Node. The execution of an I/O Method Node that is configured with a Set Line Direction method overwrites the value you configured in the C Series Module Properties dialog box.

The execution of an FPGA I/O Node configured for output automatically configures the line for output and overwrites the values you configured in the C Series Module Properties dialog box or using the Set Output Enable method.

Configuring Line Direction Using the C Series Module Properties Dialog Box

Complete the following steps to configure the line direction of each digital port using the C Series Module Properties dialog box.

1. Configure the CompactRIO system, and add an NI 9403.
2. Right-click the NI 9403 in the **Project Explorer** window and select **Properties** from the shortcut menu to display the **C Series Module Properties** dialog box.
3. Select the channel(s) for which you want to configure the line direction from the **Channels** table. You can select more than one channel by holding the <Ctrl> or <Shift> key when selecting channels.
4. Select the direction for the channel(s) from the **Direction** pull-down menu.
5. Click the **OK** button.
6. Select **File»Save All** in the **Project Explorer** window.

Configuring Line Direction Using the FPGA I/O Method Node

Complete the following steps to configure the line direction of each digital port using

the FPGA I/O Method Node.

1. Create FPGA I/O items for the channel of the NI 9403 for which you want to configure the line direction.
2. Place an FPGA I/O Method Node on the block diagram and configure it for this channel.
3. Click the **Method** section and select the Set Output Enable method from the shortcut menu.
4. Right-click the **Enable** input and select **Create»Control** from the shortcut menu. Each bit in the integer represents the line direction of one channel of the NI 9403.
5. On the front panel of the VI, click the **Enable** Boolean control to set it to *TRUE* if you want to set the line direction of the channel to digital output.

Configuring Line Direction of Multiple Channels Using the FPGA I/O Method Node

Complete the following steps to set the line direction of multiple channels using the FPGA I/O Method Node.

1. Create FPGA I/O items for the digital port of the NI 9403 that contains the channels you want to configure.
2. Place an FPGA I/O Method Node on the block diagram and configure it for this digital port.
3. Click the **Method** section and select the **Set Output Enable** method from the shortcut menu.
4. Right-click the **Enable** input and select **Create»Control** from the shortcut menu.
   - If you configured the FPGA I/O Method Node for the DIO7:0, DIO15:8, DIO23:16, or DIO31:24 digital port, the Enable control appears as an unsigned 8-bit integer.
   - If you configured the FPGA I/O Method Node for the DIO31:0 digital port, the Enable control appears as an unsigned 32-bit integer.

   Each bit in the integer represents the line direction of one channel of the NI 9403.
5. On the front panel of the VI, use the **Enable** control to enter the line direction for each channel in the port. Change a bit to 1 to set the line direction of the corresponding channel to digital output. Leave a bit as 0 to set the line direction of the corresponding channel to digital input. Refer to the table below for examples of what to enter in the **Enable** control if you configured the FPGA I/O Method Node

for the DIO7:0 digital port.

Table 465. Enable Control Values (DIO7:0)

Channel Line Direction Configuration	Enable Control (Hex)	Enable Control (Binary)
Change all channels to input	0x00	0b00000000
Change all channels to output	0xFF	0b11111111
Change channel 0 to output	0x01	0b00000001
Change channels 0 through 5 to output	0x3F	0b00111111
Change channels 3 and 7 to output	0x88	0b10001000

Refer to the table below for examples of what to enter in the **Enable** control if you configured the FPGA I/O Method Node for the DIO31:0 digital port.

Table 466. Enable Control Values (DIO31:0)

Channel Line Direction Configuration	Enable Control (Hex)	Enable Control (Binary)
Change all channels to input	0x00000000	0b0000000000000000 0000000000000000
Change all channels to output	0xFFFFFFFF	0b1111111111111111 1111111111111111
Change channel 0 to output	0x00000001	0b0000000000000000 0000000000000001
Change channels 0 through 5 to output	0x0000003F	0b0000000000000000 0000000000111111
Change channels 3 and 7 to output	0x00000088	0b0000000000000000 0000000010001000

NI 9403 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables for the channels read and write Boolean values.

Module Channels

The NI 9402 has the following channels.

Table 467. Module Channels

Channel	Description
DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. For the NI 9403, *x* is 0 to 31.

C Series Module Properties Dialog Box

Use this dialog box to configure the module. Right-click the module in the **Project Explorer** window and select Properties to display this dialog box. You can configure the following options.

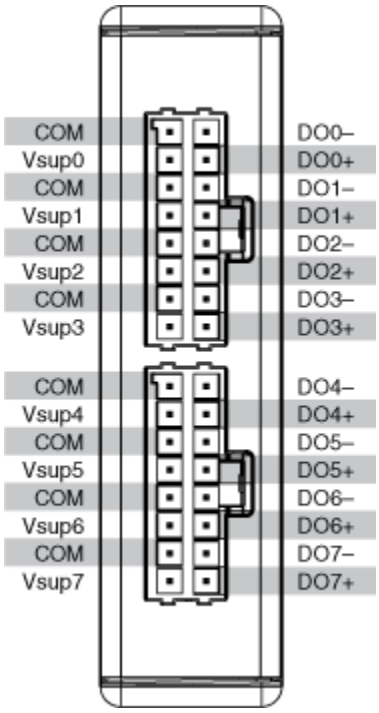
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- Channels—Specifies the channel(s) for which you want to configure settings.
- Initial Line Direction—Specifies the initial line direction of the selected channel(s) as input or output.

NI 9470

CompactRIO 8-Channel Digital Output Module with Current Readback

NI 9470 Pinout

Figure 104. NI 9470 Pinout



NI 9470 (FPGA Interface)

FPGA I/O Node

You can use an FPGA I/O Node configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 468. Terminals in Software

Terminal	Description
CH*x*	Output setpoint for channel *x*, where *x* is the number of the channel. The NI 9470 has channels 0 to 7. The output setpoint depends on the Output Mode setting for the channel. Open Loop Mode supports output setpoints from 0% to 100%. Closed Loop Mode support output setpoints from 0 A to 3.999 A.
Start	Channel that controls when the NI 9470 starts outputting PWM and acquiring readback data. If TRUE is written to the **Start** channel, the NI 9470 starts outputting PWM and acquiring readback data. The DO*x* that have been enabled

Terminal	Description
	will be reset to 0 V every time this I/O node is executed. When the NI 9470 is outputting PWM and acquiring readback data, you must write TRUE to the **Stop** channel before you can access properties. If FALSE is written to the **Start** channel, no operation is performed.
Stop	Channel that controls when the NI 9470 stops outputting PWM and acquiring readback data. If TRUE is written to the **Stop** channel, the NI 9470 stops outputting PWM and acquiring readback data. When the NI 9470 is outputting PWM and acquiring readback data, you must write TRUE to the **Stop** channel before you can access properties. If FALSE is written to the **Stop** channel, no operation is performed.
AverageCurrent	Average current readback. Readback is functional after issuing a **Start** and depends on channel enable. If channel is not enabled, the current reading will be 0. Represented as -4 A to 3.999 A.
DutyCycle	Duty cycle readback. Readback is functional after issuing a **Start** and is independent of channel enable. Represented as 0% to 100%.
Voltage	Channel voltage readback. Readback is functional after issuing a **Start** and is independent of the channel enable. Voltage is represented as 0 V to 40 V. Voltage readback nodes run at a constant data rate independent of the channel PWM frequencies and should not be placed in the same loop as the other NI 9470 I/O nodes.

Arbitration

This device supports only the Arbitrate if Multiple Requestors Only option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 469. I/O Methods

Method	Description
Set Channel Enable	Enables or disables each channel. All channels are disabled and DO*x* outputs are tristate when the module powers on. You can execute the **Set Channel Enable** method before or after **Start**. To turn on the channels before acquisition, execute the method before **Start**. If the method is executed before **Start**, the channel will be enabled or disabled

Method	Description
	immediately, and the DO*x* output will output 0 V. If the method is executed after **Start**, the channel will be enabled or disabled on the next write sequence. • To **Enable** a channel, wire a Boolean constant set to TRUE. • To **Disable** a channel, wire a Boolean constant set to FALSE.

Module Methods

Use the FPGA I/O Method Node to access the following module methods for this device.

Table 470. Module Methods

Method	Description
Check Output Status	Returns Booleans for each channel that indicates whether the module reported an over temperature fault or any channel reported a power supply fault or overcurrent fault during the last time the FPGA I/O Node communicated with the module. When the FPGA I/O Node communicates with the module, the FPGA VI samples the states of the module and channels and overwrites the previous output status with the latest state of the module and channels. • **Module Over Temperature**—Returns a Boolean value. A value of TRUE indicates that the module exceeded its temperature rating. When a Module Over Temperature occurs, all enabled channels will be disabled, and their respective DO*x* outputs will drive to 0 V. Use the **Set Channel Enable** I/O Method after the over temperature fault condition is removed to re-enable the channel without stopping the task, and the latched status will clear. • **Power Supply Fault**—Returns an array of Boolean values. A value of TRUE in any index indicates that a power fault has been detected after the last status read on the channel with the same number as the index. A value of FALSE indicates that no power fault was detected on that channel after the last status read. When a Power Supply Fault occurs, the channel with the same number as the index will be disabled, and DO*x* will drive to 0 V. Use the **Set Channel Enable** I/O Method after the Power Supply Fault condition is removed to re-enable the channel without stopping the task, and the latched status will clear. • **Channel Overcurrent**—Returns an array of Boolean values. A value of TRUE in any index indicates that the channel with the same number as the index exceeded the current limit after the last status read. A value of FALSE indicates

Method	Description
	that the channel did not exceed the current limit after the last status read. When a Channel Overcurrent fault occurs, the channel with the same number as the index will be disabled, and DO*x* will drive to 0 V. Use the **Set Channel Enable** I/O Method after the Channel Overcurrent fault condition is removed to re-enable the channel without stopping the task, and the latched status will clear.
Read Module Temperature	Reads the average measured temperature on the module. • **Module Temperature**—Returns the module temperature. Temperature readback is functional after issuing **Start**. Represented as -40 °C to 125 °C

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device. You must configure all channel properties before starting acquisition. Changing the properties during acquisition will not work.

Table 471. I/O Properties

Property	Description
Output Mode	Sets the channel to operate in either open loop mode (duty cycle setpoint) or closed loop mode (average current setpoint).
PWM Divisor	Configures the channel PWM frequency as a divisor of the module's overall data rate. Divisor values range from 1 to 511. **Note** A divisor value of 1 is only supported when the data rate is set to 2,000 S/s or 2,048 S/s.
Proportional Gain	Configures the proportional gain (*duty cycle %*/Amps) coefficient of the closed loop PID control. **Note** This property only applies to channels configured to Closed Loop Mode (Average Current Setpoint).
Integral Gain	Configures the integral gain (*duty cycle %*/Amps) coefficient of the closed loop PID control.

Property	Description
	Note This property only applies to channels configured to Closed Loop Mode (Average Current Setpoint).
Derivative gain	Configures the derivative gain (*duty cycle* %/Amps) coefficient of the closed loop PID control. **Note** This property only applies to channels configured to Closed Loop Mode (Average Current Setpoint).
Dither Amplitude	Configures the channel dither amplitude. The value depends on the output mode to which the channel is set. In open loop mode, the peak amplitude is duty cycle (percentage) with a range of 0% to 100%. In closed loop mode, the peak amplitude (amps) is current with a range of 0 A to 3.999 A. **Note** In closed loop mode, dither performance depends on the PID response.
Dither Divisor	Configures the channel dither frequency as a divisor of the channel's PWM frequency. Divisor values range from 4 to 4,095. **Note** In closed loop mode, dither performance depends on the PID response.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 472. Module Properties

Property	Description
Data Rate	Sets the overall data rate. All I/O will be synchronized to this rate. You must configure the data rate before starting acquisition. There are four data rates based on the master timebase source:

Property	Description
	<table><tr><th>12.8 MHz OCLK</th><th>13.1072 MHz OCLK</th></tr><tr><td>3.200 kS/s</td><td>3.277 kS/s</td></tr><tr><td>3.125 kS/s</td><td>3.200 kS/s</td></tr><tr><td>2.560 kS/s</td><td>2.621 kS/s</td></tr><tr><td>2.000 kS/s</td><td>2.048 kS/s</td></tr></table>
Module ID	Returns the module ID, 0x7A59.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop. Example

Refer to `LabVIEW\examples\CompactRIO\Module Specific\NI 9470\NI 9470 Getting Started.lvproj` for an example on getting started with this module.

Related concepts:

- Reading from C Series Channels

Configuring Channels on the NI 9470

Complete the following steps to configure the channels of the NI 9470.

1. Right-click an NI 9470 C Series module in the **Project Explorer** window and select **Properties** from the shortcut menu to display this dialog box.
2. Click the **Channel Configuration** button to open the Channel Configuration dialog box.

3. Select a channel to configure.
4. Configure the following settings.
   - **Output Mode**—Specifies the output mode for the selected channel.
     - **Duty Cycle**—Sets the channel to open loop mode without any internal control mechanism. You will have to implement your own control loop in your application.
     - **Average Current**—Sets the channel to closed loop mode using an internal PID control loop. This PID control executes based on the PID coefficients.
   - **PWM Divisor**—Specifies the PWM divisor value for the selected channel. Each channel runs based on the PWM frequency, which is derived by the formula *Data Rate ÷ PWM Divisor*. Supported values range 2 to 511. You can only set the PWM Divisor to a value of 1 if the data rate is set to 2,000 S/s or 2,048 S/s.
   - **PID Response**—Specifies the PID response for the selected channel. This option is only available if the channel's **Output Mode** is set to **Average Current**.
     - **Slow**—The proportional, integral, and derivative gain coefficients are pre-set to provide a slow PID response speed. The output current will arrive at the intended setpoint at a slower rate.
     - **Medium**—The proportional, integral, and derivative gain coefficients are pre-set to provide a medium PID response speed. The output current will arrive at the intended setpoint at a moderate rate.
     - **Fast**—The proportional, integral, and derivative gain coefficients are pre-set to provide a fast PID response speed. The output current will arrive at the intended setpoint at a faster rate.
     - **Custom**—Customize your own PID response by entering your own **Proportional Gain**, **Integral Gain**, and **Derivative Gain**.
       - **Proportional Gain**—Specifies the proportional gain coefficient for the selected channel. Specified in units of *(duty cycle %) ÷ Amps*.
       - **Integral Gain**—Specifies the integral gain coefficient for the selected channel. Specified in units of *(duty cycle %) ÷ Amps*.
       - **Derivative Gain**—Specifies the derivative gain coefficient for the selected channel. Specified in units of *(duty cycle %) ÷ Amps*.
   - **Dither Amplitude**—Specifies the dither amplitude for the selected channel. Represents the dither peak amplitude for the channel. You can disable dither by setting this parameter to 0. The range depends on which **Output Mode** the channel is set.

Output Mode	Range
Duty Cycle	0% to 100%
Average Current	0 A to 3.999 A

    - **Dither Divisor**—Specifies the Dither Divisor for the selected channel, which controls the dither frequency. Dither frequency is derived using the formula *PWM Frequency ÷ Dither Divisor*.
5. Click **OK** to close the Channel Configuration dialog box.
6. Click **OK** to close the Module Properties Dialog Box.

Related concepts:

- NI 9470 (FPGA Interface)

Synchronizing the NI 9470 with Other Modules (FPGA Interface)

You can synchronize the NI 9470 with other modules that are connected to the same FPGA device if your application meets the following requirements:

- The modules must use the same master timebase source
- The modules must start acquisition mode at the same time
- Each channel of the NI 9470 must be in its own FPGA I/O Node residing in its own loop.

Create FPGA I/O items for the module before you can configure the items using the FPGA I/O Node. Develop the FPGA VI to meet the guidelines described in the following table.

For delta-sigma modules, you will need to synchronize multiple sample rates.

Table 473. FPGA VI Development Guidelines

Guideline	Details
Share a master timebase source	Configure the modules to share master timebase source.

Guideline	Details
Start the synchronized acquisition	• Configure an FPGA I/O Node with **Start** channels for the module you want to synchronize. • Wire a Boolean constant set to TRUE to each **Start** channel. • Ensure that all **Start** channels are in the same FPGA I/O Node. The data will not be synchronized if they are not.
Acquire data from synchronized modules	• Create separate loops for each channel of the NI 9470 that runs at a different data rate. Channels that are configured to run at the same data rate can either be in the same loop or be in a separate loop. • For channels that are configured to run at different data rates, configure an FPGA I/O Node in each loop. For channels that are configured to run at the same data rate, you can either configure one FPGA I/O Node for the all the channels or configure an FPGA I/O Node for each channel. If you place channels that are configured for different data rates in the same loop, LabVIEW returns an overrun warning, error 65539, from the FPGA I/O Node for the AI channels and an underflow warning, error 65676, from the FPGA I/O Node for the AO channels. • A delay occurs before the FPGA I/O Node returns the first data point. The length of the delay depends on the data rate and the PWM Divisor for each NI 9470 channel. • Refer to *Synchronizing Multiple Modules (FPGA Interface)* for how to synchronize other DSA modules that are synchronized with the NI 9470.
Understand the maximum sample rate when synchronizing multiple modules	—

Table 474. Equation for Synchronizing Multiple Sample Rates

12.8 MHz OCLK	13.1072 MHz OCLK	Time to First Channel Sample(s)
3.200 kS/s	3.277 kS/s	(14,472,838 + 4,000 * *n*) * (*Master Timebase Period* ± 1 *Master Timebase Period*)
3.125	3.200 kS/s	(14,472,838 + 4,096 * *n*) * (*Master Timebase Period*

12.8 MHz OCLK	13.1072 MHz OCLK	Time to First Channel Sample(s)
kS/s		± 1 *Master Timebase Period*)
2.560 KS/s	2.621 KS/s	(14,472,838 + 5,000 * *n*) * (*Master Timebase Period* ± 1 *Master Timebase Period*)
2.000 KS/s	2.048 KS/s	(14,473,006 + 6,400 * *n*) * (*Master Timebase Period* ± 1 *Master Timebase Period*)

Note *Master Timebase Period* = period of the internal or external clock that the module uses (1/12.8 MHz or 1/13.072 MHz). *n* = channel PWN divisor. Refer to the *PWM Divisor I/O Property* for valid values of *n* for each data rate divisor option. Only data rates of 3.125 kS/s or 2.000 kS/s with 12.8 MHz OCLK or 3.200 kS/s or 2.048 kS/s with 13.1072 MHz OCLK are compatible with other DSA modules for *synchronizing multiple modules*. The formulas above apply to each individual channel of the NI 9470 based on each channel's PWM divisor.

Related concepts:

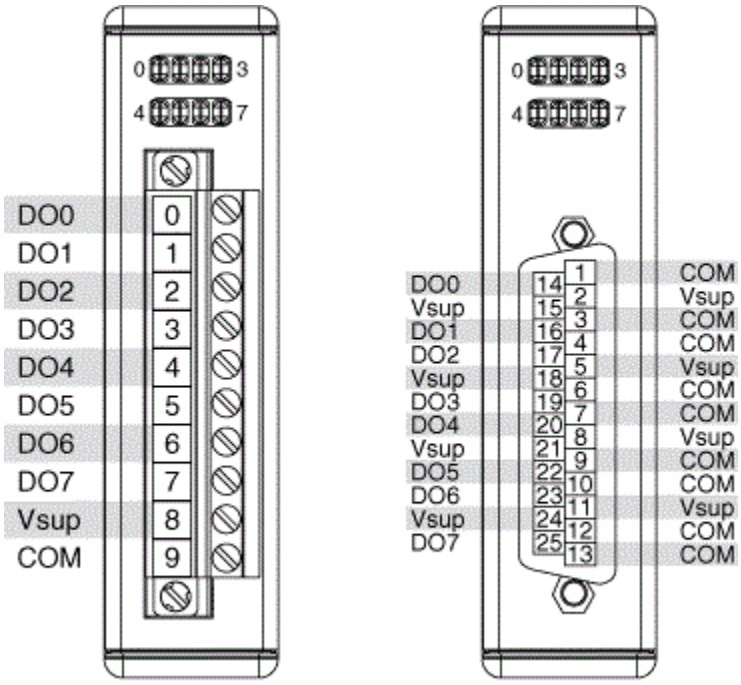
- Configuring the Master Timebase Source for a C Series Module (FPGA Interface)
- Synchronizing Multiple Modules (FPGA Interface)
- Understanding the Maximum Sample Rate When Synchronizing Multiple Modules (FPGA Interface)
- NI 9470 (FPGA Interface)

NI 9472

CompactRIO 8-Channel, 24 V, Sourcing Digital Output Module

NI 9472 Pinout

Figure 105. NI 9472 Pinout



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9472 (FPGA Interface)

FPGA I/O Node

You can use an FPGA I/O Node, configured for writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 475. Terminals in Software

Terminal	Description
DOx	Digital output channel x, where x is the number of the channel. The NI 9472 has DO channels 0 to 7.
DO7:0	Digital port consisting of channels 0 through 7. Channel 7 is returned in the MSB, and

Terminal	Description
	channel 0 is returned in the LSB.

Arbitration

You can configure the arbitration settings for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following module method for this device.

Table 476. I/O Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

Note During the first 2 seconds after you reset the FPGA VI, the error terminals on this method may not correctly report certain types of errors.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 477. Module Properties

Property	Description
Module ID	Returns the *module ID*.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the channels of this device in the Advanced Configuration dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital output before a loop containing digital output starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. The module might ignore or delay digital output operations if it is not ready.

While the module is performing digital output within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Doing either of these actions causes the module to be unable to perform digital output and the **Ready** output of the Check Status method to return FALSE.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

Related concepts:

- Writing to C Series Channels

NI 9472 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables write Boolean values to the channels.

Module Channels

The NI 9472 has the following channels.

Table 478. Module Channels

Channel	Description
DO*x*	Digital output channel *x*, where *x* is the number of the channel. For the NI 9472, *x* is 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9472. Right-click the NI 9472 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.

Specialty Digital Configuration

Use the Specialty Digital Configuration page of the **C Series Module Properties** dialog box to configure channels of this module for counter-driven output or pulse-width modulation output.

Related concepts:

- Configuring Specialty Digital Functions

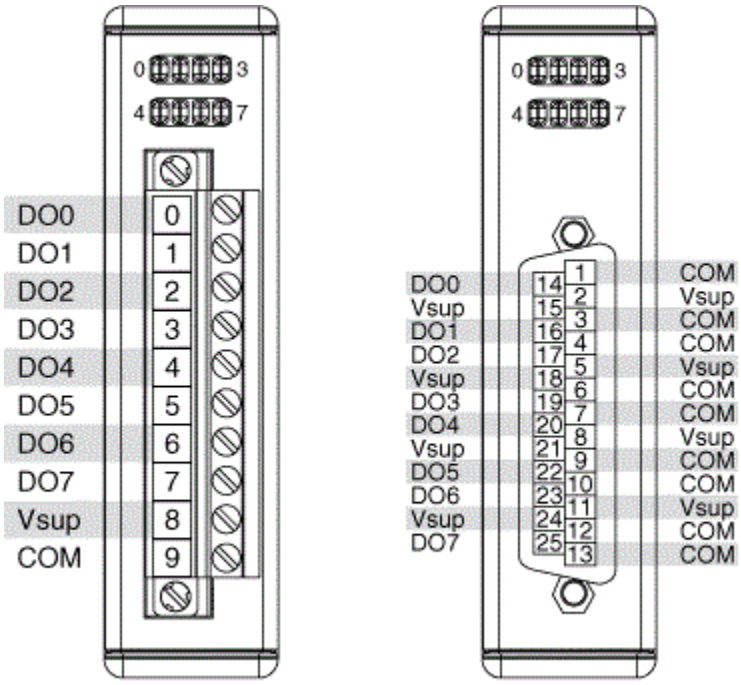
- Configuring Counter-Driven Outputs
- Configuring a Channel for Pulse-Width Modulation

NI 9474

CompactRIO 8-Channel, 24 V, High-Speed, Sourcing Digital Output Module

NI 9474 Pinout

Figure 106. NI 9474 Pinout



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9474 (FPGA Interface)

FPGA I/O Node

You can use an FPGA I/O Node, configured for writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 479. Terminals in Software

Terminal	Description
DO*x*	Digital output channel *x*, where *x* is the number of the channel. The NI 9474 has DO channels 0 to 7.
DO7:0	Digital port consisting of channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.

Arbitration

You can configure the arbitration settings for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following module method for this device.

Table 480. Module Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

Note During the first 2 seconds after you reset the FPGA VI, the error terminals on this method may not correctly report certain types of errors.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 481. Module Properties

Property	Description
Module ID	Returns the *module ID*.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the channels of this device in the Advanced Configuration dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital output before a loop containing digital output starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. The module might ignore or delay digital output operations if it is not ready.

While the module is performing digital output within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Doing either of these actions causes the module to be unable to perform digital output and the **Ready** output of the Check Status method to return FALSE.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

Related concepts:

- Writing to C Series Channels

NI 9474 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables write Boolean values to the channels.

Module Channels

The NI 9474 has the following channels.

Table 482. Module Channels

Channel	Description
DO*x*	Digital output channel *x*, where *x* is the number of the channel. For the NI 9474, *x* is 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9474. Right-click the NI 9474 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.

Specialty Digital Configuration

Use the Specialty Digital Configuration page of the **C Series Module Properties** dialog box to configure channels of this module for counter-driven output or pulse-width modulation output.

Related concepts:

- Configuring Specialty Digital Functions

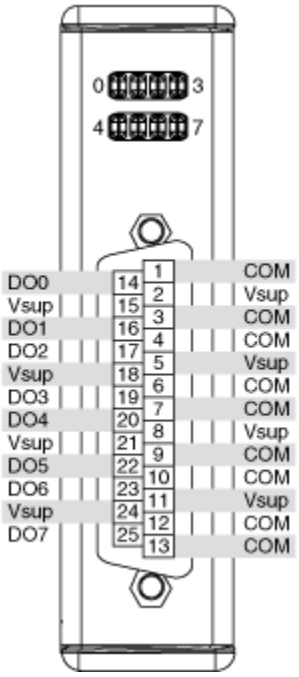
- Configuring Counter-Driven Outputs
- Configuring a Channel for Pulse-Width Modulation

NI 9475

CompactRIO 8-Channel, 60 V, High-Speed, Sourcing Digital Output Module

NI 9475 Pinout

Figure 107. NI 9475 Pinout



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9475 (FPGA Interface)

FPGA I/O Node

You can use an FPGA I/O Node, configured for writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 483. Terminals in Software

Terminal	Description
DO*x*	Digital output channel *x*, where *x* is the number of the channel. The NI 9475 has DO channels 0 to 7.
DO7:0	Digital port consisting of channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.

Arbitration

You can configure the arbitration settings for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following module method for this device.

Table 484. Module Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

Note During the first 2 seconds after you reset the FPGA VI, the error terminals on this method may not correctly report certain types of errors.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this

device.

Table 485. Module Properties

Property	Description
Module ID	Returns the *module ID*.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the channels of this device in the Advanced Configuration dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital output before a loop containing digital output starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. The module might ignore or delay digital output operations if it is not ready.

While the module is performing digital output within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Doing either of these actions causes the module to be unable to perform digital output and the **Ready** output of the Check Status method to return FALSE.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

Related concepts:

- Writing to C Series Channels

NI 9475 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables write Boolean values to the channels.

Module Channels

The NI 9475 has the following channels.

Table 486. Module Channels

Channel	Description
DO*x*	Digital output channel *x*, where *x* is the number of the channel. For the NI 9475, *x* is 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9475. Right-click the NI 9475 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- **Module Type**—Specifies the type of C Series module. You cannot change this value.
- **Location**—Specifies a slot in the chassis for the C Series module.

Specialty Digital Configuration

Use the Specialty Digital Configuration page of the **C Series Module Properties** dialog box to configure channels of this module for counter-driven output or pulse-width modulation output.

Related concepts:

- Configuring Specialty Digital Functions

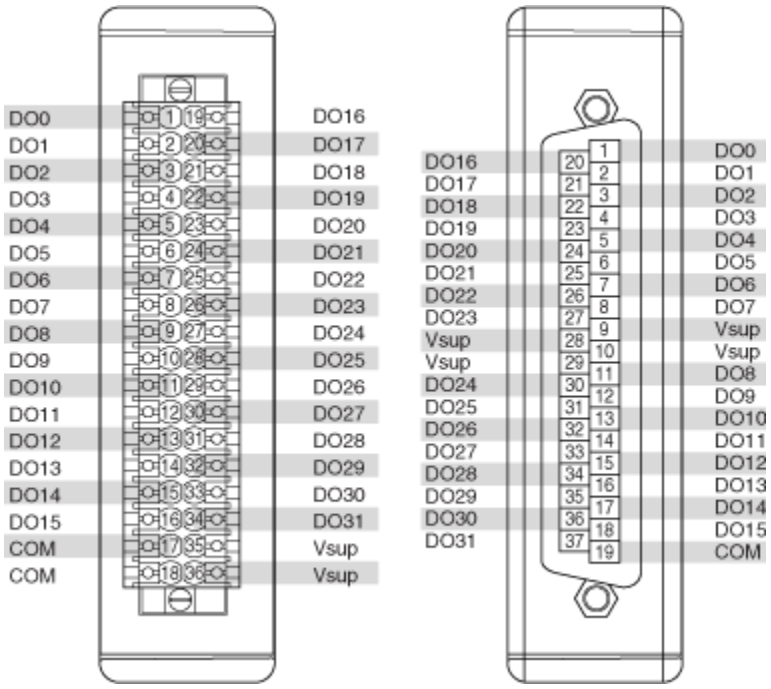
- Configuring Counter-Driven Outputs
- Configuring a Channel for Pulse-Width Modulation

NI 9476

36 V, 32-Channel (Sourcing Output), 500 µs C Series Digital Module

NI 9476 Pinout

Figure 108. NI 9476 Pinout



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9476 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 487. Terminals in Software

Terminal	Description
DO*x*	Digital output channel *x*, where *x* is the number of the channel. The NI 9476 has DO channels 0 to 31.
DO7:0	Digital port consisting of channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
DO15:8	Digital port consisting of channels 8 through 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
DO23:16	Digital port consisting of channels 16 through 23. Channel 23 is returned in the MSB, and channel 16 is returned in the LSB.
DO31:24	Digital port consisting of channels 24 through 31. Channel 31 is returned in the MSB, and channel 24 is returned in the LSB.
DO31:0	Digital port consisting of channels 0 through 31. Channel 31 is returned in the MSB, and channel 0 is returned in the LSB.

Arbitration

This device supports only the Arbitrate if Multiple Requestors Only and Never Arbitrate options for arbitration. The default arbitration setting is Arbitrate if Multiple Requestors Only.

If you are sure that the design of the FPGA VI will never allow more than one digital function to execute at the same time, even on different channels, enable the **Disable Arbitration** checkbox on the *C Series Module Properties* dialog box to disable arbitration and reduce the amount of FPGA logic used by VIs. If more than one digital function could execute simultaneously in the FPGA VI, leave the box unchecked.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following module method for this device.

Table 488. Module Methods

Method	Description
Check Output Status	Returns Booleans that indicate whether any channel reported an overcurrent fault during the last time the FPGA I/O Node communicated with the module. When the FPGA I/O Node communicates with the module, the FPGA VI samples the states of the channels and overwrites the previous output status with the latest state of the channels. • **Force Status Read**—When the value of this input is FALSE, the method returns the cached status information from the last status read. When the value is TRUE, the method gets the current status information from the module. Forcing a status read can introduce jitter into a digital output loop. • **Any Overcurrent**—Returns a Boolean value. A value of TRUE indicates an overcurrent condition on at least one channel. • **Channel Overcurrent**—Returns an array of Boolean values. A value of TRUE in any index indicates that the channel sharing a number with that index is in an overcurrent condition.

I/O properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 489. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Writing to C Series Channels

NI 9476 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables write Boolean values to the channels.

Module Channels

The NI 9476 has the following channels.

Table 490. Module Channels

Channel	Description
DO*x*	Digital output channel *x*, where *x* is the number of the channel. For the NI 9476, *x* is 0 to 31.

Module-Specific Errors

The NI 9476 can return the following module-specific errors.

Table 491. Module-Specific Error Codes

Error Code	Description
65548	One or more channels are in overcurrent or overvoltage protection mode. Check the terminals for any fault condition that could be causing an out-of-range voltage or current on the channels.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9476. Right-click the NI 9476 in the Project Explorer window and select Properties to display this dialog box. You can configure the following options.

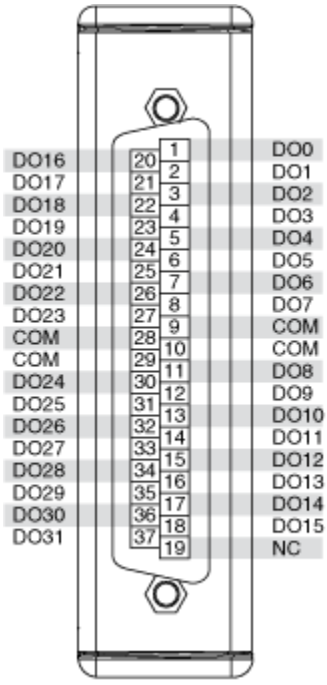
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9477

CompactRIO 32-Channel, 5–60 V, Sinking Digital Output Module

NI 9477 Pinout

Figure 109. NI 9477 Pinout



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9477 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 492. Terminals in Software

Terminal	Description
DO*x*	Digital output channel *x*, where *x* is the number of the channel. The NI 9477 has DO channels 0 to 31.
DO7:0	Digital port consisting of channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
DO15:8	Digital port consisting of channels 8 through 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
DO23:16	Digital port consisting of channels 16 through 23. Channel 23 is returned in the MSB, and channel 16 is returned in the LSB.
DO31:24	Digital port consisting of channels 24 through 31. Channel 31 is returned in the MSB, and channel 24 is returned in the LSB.
DO31:0	Digital port consisting of channels 0 through 31. Channel 31 is returned in the MSB, and channel 0 is returned in the LSB.

Arbitration

This device supports only the Arbitrate if Multiple Requestors Only and Never Arbitrate options for arbitration. The default arbitration setting is Arbitrate if Multiple Requestors Only.

If you are sure that the design of the FPGA VI will never allow more than one digital function to execute at the same time, even on different channels, enable the **Disable Arbitration** checkbox on the C Series Module Properties dialog box to disable arbitration and reduce the amount of FPGA logic used by VIs. If more than one digital function could execute simultaneously in the FPGA VI, leave the box unchecked.

Methods

This device does not support any methods.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 493. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Writing to C Series Channels

NI 9477 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables write Boolean values to the channels.

Module Channels

The NI 9477 has the following channels.

Table 494. Module Channels

Channel	Description
DO*x*	Digital output channel *x*, where *x* is the number of the channel. For the NI 9477, *x* is 0 to 31.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9477. Right-click the NI 9477 in the Project Explorer window and select Properties to display this dialog box. You can configure the following options.

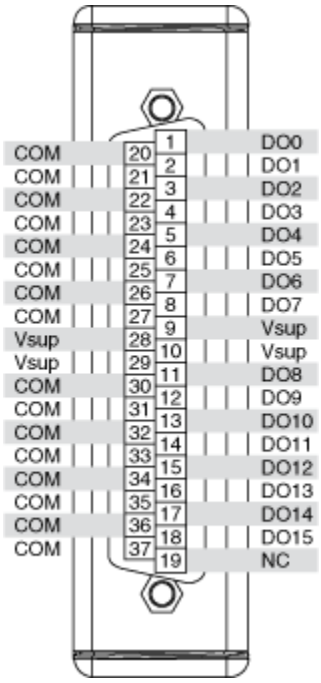
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9478

CompactRIO 16-Channel, 0–50 V Sinking Digital Output Module with Programmable Current Limits

NI 9478 Pinout

Figure 110. NI 9478 Pinout



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9478 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 495. Terminals in Software

Terminal	Description
DO*x*	Digital output channel *x*, where *x* is the number of the channel. The NI 9478 has DO channels 0 to 15.
DO7:0	Digital port consisting of channels 0 through 7. Channel 7 is returned in the MSB, and

Terminal	Description
	channel 0 is returned in the LSB.
DO15:8	Digital port consisting of channels 8 through 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
DO15:0	Digital port consisting of channels 0 through 15. Channel 15 is returned in the MSB, and channel 0 is returned in the LSB.

Arbitration

This device supports only the Arbitrate if Multiple Requestors Only option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following module method for this device.

Table 496. Module Methods

Method	Description
Check Output Status	Returns Booleans that indicate whether the module reported an overtemp fault or any channel reported an overcurrent fault during the last time the FPGA I/O Node communicated with the module. When the FPGA I/O Node communicates with the module, the FPGA VI samples the states of the module and channels and overwrites the previous output status with the latest state of the module and channels. • **Limit Selection**—Specifies which current limit status information the module reads. • Select **Active Limit** when you want the **Channel Overcurrent** output to return *TRUE* for any channel that exceeds the current limit threshold specified for the active current limit set for that channel. • Select **Limit A** to return *TRUE* for any channel that exceeds the current limit threshold specified for Current Limit A. • Select **Limit B** to return *TRUE* for any channel that exceeds the current

Method	Description
	limit threshold specified for Current Limit B. • **Module Overtemp**—Returns a Boolean value. • Returns *TRUE* when the module exceeds its temperature rating. • **Channel Overcurrent**—Returns an array of Boolean values. • Returns *TRUE* in any index when the channel sharing a number with that index exceeded the current limit specified by the **Limit Selection** input after the last status read. • Returns *FALSE* when the channel did not exceed the current limit after the last status read.

I/O properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 497. Module Properties

Property	Description
Active Current Limit	Sets the active current limit for each channel. The value in an index of the input array controls the active current limit for the channel sharing a number with that index. • Select No Change to leave the active current limit unchanged for the channel at run time. • Select Limit A to set the active current limit for the channel to Current Limit A. • Select Limit B to set the active current limit for the channel to Current Limit B. • Select No Limit when you want the channel to have no current limit.
Current Limit A	Sets the current limit threshold for Current Limit A in binary values. Use the following equation to calculate the binary value from amperes: `Binary Value = (Current Value / 5.12 A) × 256` The range of valid current limit values is 0–5.1 A, which corresponds to binary values 0–255.

Property	Description
Current Limit B	Sets the current limit threshold for Current Limit B in binary values. Use the following equation to calculate the binary value from amperes: *Binary Value* = (*Current Value* / 5.12 A) × 256 The range of valid current limit values is 0–5.1 A, which corresponds to binary values 0–255.
Module ID	Returns the module ID.
Overcurrent Refresh Period	Sets the time in µs it takes a channel to automatically recover if the channel exceeds its active current limit threshold. Values 2 to 255 enable overcurrent refresh and specify the time in tens of µs. For example, a value of 3 sets an overcurrent refresh period of 30 µs. A value of 0 disables overcurrent refresh so the channel remains disabled after exceeding its active current limit until you write to the channel using an FPGA I/O Node. A value of 1 is not supported and sets an overcurrent refresh period of 20 µs.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Writing to C Series Channels
- Configuring Current Limits for the NI 9478 (FPGA Interface)

Configuring Current Limits for the NI 9478 (FPGA Interface)

You can configure the active current limit for each channel and the current limit thresholds at edit time using the *C Series Module Properties* dialog box. You can programmatically change these settings at run time using the FPGA I/O Property Node.

The execution of an I/O Property Node that is configured with a **Current Limit** property overwrites the value you configured in the **C Series Module Properties** dialog box.
Configuring Current Limits Using the C Series Module Properties Dialog Box

Complete the following steps to configure the active current limit for each channel and the current limit thresholds for the NI 9478 using the **C Series Module Properties** dialog box.

1. *Configure* the CompactRIO system, and add an NI 9478.
2. Right-click the NI 9478 in the **Project Explorer** window and select **Properties** from the shortcut menu to display the **C Series Module Properties** dialog box.
3. Select the channel(s) for which you want to configure the active current limit from the **Channels** table. You can select more than one channel by holding the <Ctrl> or <Shift> key when selecting channels.
4. Select the active current limit for the channel(s) from the **Active Limit** pull-down menu.
5. Enter a value between 0 and 5.1 in the **Current Limit** text box that corresponds to the active current limit for the selected channel(s).
6. Place a checkmark in the **Enable Overcurrent Refresh** checkbox and set the time in the **Overcurrent Refresh Period** text box if you want to enable a channel to automatically recover if the channel exceeds its active current limit threshold.
7. Click the **OK** button.
8. Select **File»Save All** in the **Project Explorer** window.

Configuring Current Limits Using the FPGA I/O Property Node

Complete the following steps to configure the active current limit for each channel and the current limit thresholds using the FPGA I/O Property Node.

1. Place three FPGA I/O Property Nodes on the block diagram and configure them for the NI 9478.
2. Click the **Property** section on the first Property Node and select the **Active Current Limit** property from the shortcut menu.
3. Right-click the **Active Current Limit** input and select **Create»Control** from the shortcut menu.
4. On the front panel of the VI, select the active current limit for each channel from the **Active Current Limit** array.
5. Click the **Property** section on the second Property Node and select the **Current**

Limit A property from the shortcut menu.
6. Right-click the **Current Limit A** input and select **Create»Control** from the shortcut menu.
7. Click the **Property** section on the third Property Node and select the **Current Limit B** property from the shortcut menu.
8. Right-click the **Current Limit B** input and select **Create»Control** from the shortcut menu.
9. On the front panel of the VI, enter binary values in the **Current Limit** controls to set the current limit thresholds. You can use the following equation in the host VI to calculate the binary value from amperes: Binary Value = (Current Value / 5.12 A) × 256

Related concepts:

- Configuring a Project

NI 9478 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables write Boolean values to the channels. Module Channels

Module Channels

The NI 9478 has the following channels.

Table 498. Module Channels

Channel	Description
DO*x*	Digital output channel *x*, where *x* is the number of the channel. For the NI 9478, *x* is 0 to 15.

Module-Specific Errors

The NI 9478 can return the following module-specific errors.

Table 499. Module-Specific Error Codes

Error Code	Description
65548	One or more channels are in overcurrent or overvoltage protection mode. Check the terminals for any fault condition that could be causing an out-of-range voltage or current on the channels.
65571	Serial module over-temperature error: An over-temperature error has occurred since the last time the module was accessed. Check for fault conditions or extraneous voltages on the I/O port.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9478. Right-click the NI 9478 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

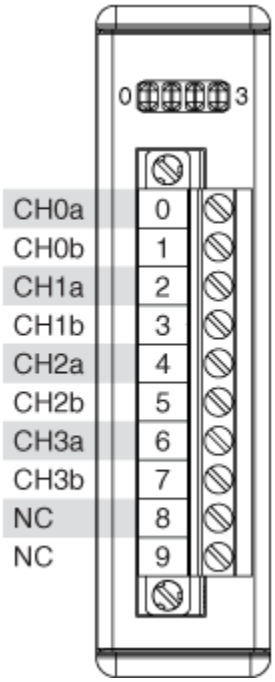
- **Name**—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.
- Channels—Specifies the channels for which you want to configure an active current limit.
- Active Limit—Specifies the current limit for the selected channel(s). You can select **Limit A**, **Limit B**, or **No Limit**.
- Current Limit A—Specifies one of two current limit values in amps for channels on the module. You can enter a value from 0 to 5.1.
- Current Limit B—Specifies one of two current limit values in amps for channels on the module. You can enter a value from 0 to 5.1.
- Enable Overcurrent Refresh—If you check this box, a channel that exceeds the current limit automatically recovers after a specified amount of time, the overcurrent refresh period. If you leave this box unchecked, a channel that exceeds the current limit remains disabled until you write to it.
- Overcurrent Refresh Period—Specifies the time in µs it takes a channel to automatically recover after exceeding the current limit. You can enter a value from 20 to 2,550.

NI 9481

CompactRIO 4-Channel, Form A Electromechanical Relay Module

NI 9481 Pinout

Figure 111. NI 9481 Pinout



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9481 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 500. Terminals in Software

Terminal	Description
CH*x*	channel *x*, where *x* is the number of the channel. The NI 9481 has channels 0 to 3.
CH3:0	Digital port consisting of channels 0 through 3. • Channel 3 is returned in bit 3. • Channel 0 is returned in bit 0.

Arbitration

Configure the arbitration settings for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following module method for this device.

Table 501. Module Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

Note During the first 2 seconds after you reset the FPGA VI, the error terminals on the Check Status method may not correctly report certain types of errors.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 502. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the channels of this device in the Advanced Configuration dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital output before a loop containing digital output starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. The module might ignore or delay digital output operations if it is not ready.

While the module is performing digital output within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Doing either of these actions causes the module to be unable to perform digital output and the **Ready** output of the Check Status method to return FALSE.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

Related concepts:

- Writing to C Series Channels

NI 9481 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables write Boolean values to the channels.

Module Channels

The NI 9481 has the following channels.

Table 503. Module Channels

Channel	Description
DO*x*	Digital output channel *x*, where *x* is the number of the channel. For the NI 9481, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9481. Right-click the NI 9481 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

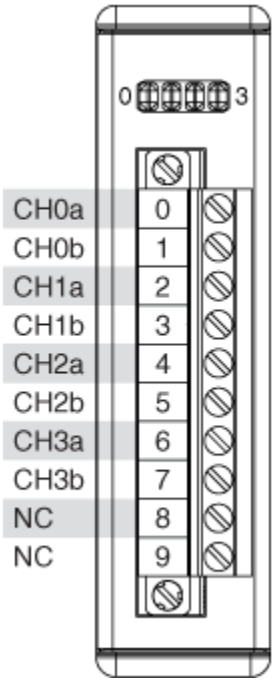
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9482

CompactRIO 4-Channel, Form A Electromechanical Relay Module

NI 9482 Pinout

Figure 112. NI 9482 Pinout



NI 9482 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 504. Terminals in Software

Terminal	Description
CHx	channel x, where x is the number of the channel. The NI 9482 has channels 0 to 3.
CH3:0	Digital port consisting of channels 0 through 3. • Channel 3 is returned in bit 3. • Channel 0 is returned in bit 0.

Arbitration

Configure the arbitration settings for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following module method for this device.

Table 505. Module Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

Note During the first 2 seconds after you reset the FPGA VI, the error terminals on the Check Status method may not correctly report certain types of errors.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 506. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.

Property	Description
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the channels of this device in the Advanced Configuration dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital output before a loop containing digital output starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. The module might ignore or delay digital output operations if it is not ready.

While the module is performing digital output within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Doing either of these actions causes the module to be unable to perform digital output and the **Ready** output of the Check Status method to return FALSE.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

Related concepts:

- Writing to C Series Channels
- C Series Module IDs

NI 9482 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables write Boolean values to the channels.

Module Channels

The NI 9482 has the following channels.

Table 507. Module Channels

Channel	Description
DO*x*	Digital output channel *x*, where *x* is the number of the channel. For the NI 9482, *x* is 0 to 3.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9482. Right-click the NI 9482 in the Project Explorer window and select **Properties** to display this dialog box. You can configure the following options.

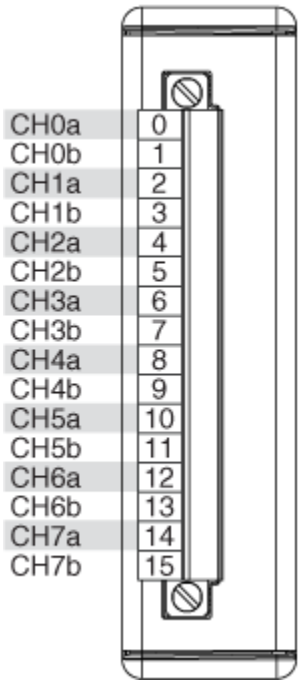
- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

NI 9485

CompactRIO 8-Channel, Solid-State Relay (SSR) Module

NI 9485 Pinout

Figure 113. NI 9485 Pinout



Related concepts:

- Understanding Power-On and Startup Output States for CompactRIO Output Modules (FPGA Interface)

NI 9485 (FPGA Interface)

FPGA I/O Node

Use an FPGA I/O Node configured for writing with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 508. Terminals in Software

Terminal	Description
CHx	channel x, where x is the number of the channel. The NI 9485 has channels 0 to 7.
CH7:0	Digital port consisting of channels 0 through 7.

Terminal	Description
	• Channel 7 is returned in the MSB. • Channel 0 is returned in the LSB.

Arbitration

Configure the arbitration settings for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following module method for this device.

Table 509. Module Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the module is ready.

Note During the first 2 seconds after you reset the FPGA VI, the error terminals on the Check Status method may not correctly report certain types of errors.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 510. Module Properties

Property	Description
Module ID	Returns the module ID.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the channels of this device in the Advanced Configuration dialog box.

When the module is within a single-cycle Timed Loop, it must be ready to perform digital output before a loop containing digital output starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. The module might ignore or delay digital output operations if it is not ready.

While the module is performing digital output within a single-cycle Timed Loop, do not perform property reads or remove the module from the chassis. Doing either of these actions causes the module to be unable to perform digital output and the **Ready** output of the Check Status method to return FALSE.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

Related concepts:

- Writing to C Series Channels

NI 9485 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI. The I/O variables write Boolean values to the channels.

Module Channels

The NI 9485 has the following channels.

Table 511. Module Channels

Channel	Description
DO*x*	Digital output channel *x*, where *x* is the number of the channel. For the NI 9485, *x* is 0 to 7.

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9485. Right-click the NI 9485 in the Project Explorer window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Specifies the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Specifies the type of C Series module. You cannot change this value.
- Location—Specifies a slot in the chassis for the C Series module.

Functional Safety Modules

Table 512. C Series Functional Safety Modules

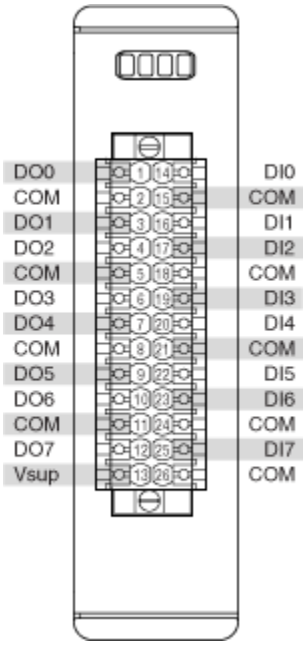
C Series Functional Safety Module	Description
NI-9350	8-Channel C Series Functional Safety Module
NI-9351	4-Channel C Series Functional Safety Module

NI 9350

8-Channel 24 V Sinking DI, 8-Channel 24 V Sourcing DO SIL 3 Capable C Series Functional Safety Module

NI 9350 Pinout

Figure 114. NI 9350 Pinout



Related information:

- C Series Functional Safety Manual

NI 9350 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI.

- **Digital input**—reads the Boolean state of that channel.
- **Digital output**—reads the Boolean state of that channel. When the User Program sets the channel to passthrough, the User Program will use the value in the output scan buffer. When the channel is not set to passthroughin the User Program, the User Program ignores the value in the output scan buffer.

- **State machine variable**—reads the Boolean value of that variable in the User Program.
- **User LED0**—reads the Boolean value of the LED in the User Program.

Module Channels

The NI 9350 has the following channels.

Table 513. Module Channels

Channel	Description
DIx	Digital input channel x, where x is the number of the channel. For the NI 9350, x is 0 to 7.
DOx	Digital output channel x, where x is the number of the channel. For the NI 9350, x is 0 to 7.
Varx	State machine variable x, where x is the number of the variable. For the NI 9350, x is 0 to 23.
User LED0	User-configurable LED

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9350. Right-click the NI 9350 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Indicates the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Indicates the type of C Series module. You cannot change this value.
- Location—Indicates a slot in the chassis for the C Series module.
- Read Module Information—Reads the Build Number and Program GUID of the current User Program installed on the C Series module.
- Build Number—Indicates the build number of the current User Program installed on the C Series module.
- Program GUID—Indicates the program GUID of the current User Program installed on the C Series module.
- Mode—Indicates the current operating mode of the C Series module.

  - **Power Up**—Indicates module is powering up.
  - **Unprogrammed**—Indicates User Program is not written to the C Series module.
  - **User Program Download**—Indicates the User Program is downloading to the C Series module.
  - **Module Firmware Download**—Indicates firmware is downloading to the C Series module.
  - **Verification**—Indicates User Program has successfully downloaded. Requires user verification.
  - **Operational**—Indicates C Series module is running the User Program.
  - **Reserved**—Indicates C Series module is in Reserved mode.
  - **Fail-safe**—Indicates User Program has detected a fault that triggered Fail-safe mode. User must address fault and cycle power to return the C Series module to Operational mode.
- Change Mode to X—Updates C Series module from Verification mode to Operational mode or from Operational mode to Verification mode.
- Path to New User Program—Specifies the file path of the User Program selected for download.
- Build Number—Indicates the build number of the User Program selected for download.
- Program GUID in New Program—Indicates the program GUID of the User Program selected for download.
- Download—Downloads the selected User Program to the C Series module.

Module Methods

Use the Invoke Node to access the following module properties for this device.

Table 514. Module Methods

Method	Description
Check Channel Status	Returns Boolean arrays that indicate the channel status, including various faults. The index number in the array corresponds to the channel number on the module, so that index 0 indicates the channel status for DI0 or DO0. The following parameters are available:

Method	Description
	• DO Internal Readback—Indicates whether the DO signal is high or low. • DO Channel Overcurrent Fault—Indicates an overcurrent fault has been detected. • DO Channel Open Circuit Fault—Indicates an open circuit fault has been detected. • DI Channel Discrepancy Fault—Indicates a discrepancy fault has been detected. • Channel Test Pulse Fault—Indicates a test pulse fault has been detected. • Channel Readback Fault—Indicates a readback fault has been detected.
Check Module Status	Returns the module status, including digital output values and various faults. The following parameters are available: • Mode—Returns an enum value that reads the operating mode of the NI 9350. ◦ Power Up—Indicates module is powering up. ◦ Unprogrammed—Indicates User Program is not written to the C Series module. ◦ User Program Download—Indicates the User Program is downloading to the C Series module. ◦ Module Firmware Download—Indicates firmware is downloading to the C Series module. ◦ Verification—Indicates User Program has successfully downloaded. Requires user verification. ◦ Operational—Indicates C Series module is running the User Program. ◦ Reserved—Indicates C Series module is in Reserved mode. ◦ Fail-safe—Indicates the User Program has detected a fault that triggered Fail-safe mode. Address the fault and cycle power to return the module to **Operational** mode. • User Program Running—Returns Boolean value to indicate the status of the User Program. • Internal Fault—Returns Boolean value to indicate whether an internal fault has been detected. • Temperature Fault—Returns Boolean value to indicate whether a temperature fault has been detected. • Power Supply Fault—Returns Boolean value to indicate whether a power supply fault has been detected.
Start Program	Starts the User Program.
Set Mode	Sets the operating mode for the NI 9350 as an enum value. Valid values are Operational

Method	Description
	or Verification.
Read Current State	Returns an array of U8 values. The array index corresponds to the state machine number in the User Program and the U8 value indicates the current state of that state machine. Refer to the `report.log` file generated by the Functional Safety Editor to index the state machines and identify the state number.

Module Properties

Use the Property Node to access the following module properties for this device.

Table 515. Module Properties

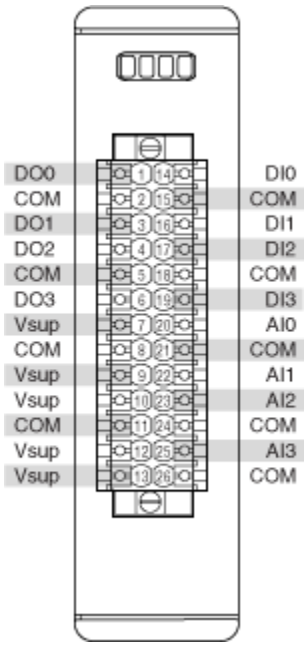
Property	Description
Firmware Version Number	Returns U32 value to indicate the version of the module firmware.
User Program GUID	Returns a string to indicate the GUID assigned to the User Program.
User Program Version Number	Returns U32 value to indicate the build number of the User Program.

NI 9351

4-Channel 0-20 mA 16-bit AI, 4-Channel 24 V Sinking DI, 4-Channel 24 V Sourcing DO SIL 3 Capable C Series Functional Safety Module

NI 9351 Pinout

Figure 115. NI 9351 Pinout



Related information:

- C Series Functional Safety Manual

NI 9351 (Scan Interface)

Module I/O Variables

To use I/O from this module in a VI, drag and drop I/O variables from the **Project Explorer** window to the block diagram of the VI.

- **Analog input**—Reads the analog signal of that channel.
- **Digital input**—Reads the Boolean state of that channel.
- **Digital output**—Reads the Boolean state of that channel. Writes a Boolean value to the output scan buffer. When the channel is set to passthrough in the User Program, the User Program will use the value in the output scan buffer. When the channel is not set to passthrough in the User Program, the User Program ignores the value in the output scan buffer.
- **State machine variable**—Reads the Boolean value of that variable in the User Program.
- **User LED0**—Reads the Boolean value of the LED in the User Program.

Module Channels

The NI 9351 has the following channels.

Table 516. Module Channels

Channel	Description
AI*x*	Analog input channel *x*, where *x* is the number of the channel. For the NI 9351, *x* is 0 to 3.
DI*x*	Digital input channel *x*, where *x* is the number of the channel. For the NI 9351, *x* is 0 to 3.
DO*x*	Digital output channel *x*, where *x* is the number of the channel. For the NI 9351, *x* is 0 to 3.
Var*x*	State machine variable *x*, where *x* is the number of the variable. For the NI 9351, *x* is 0 to 23.
User LED0	User-configurable LED

C Series Module Properties Dialog Box

Use this dialog box to configure the NI 9351. Right-click the NI 9351 in the **Project Explorer** window and select **Properties** to display this dialog box. You can configure the following options.

- Name—Indicates the name of the C Series module, which appears in the **Project Explorer** window. LabVIEW assigns a default name to the module based on the slot number. You can use this field to give the module a descriptive name.
- Module Type—Indicates the type of C Series module. You cannot change this value.
- Location—Indicates a slot in the chassis for the C Series module.
- Read Module Information—Reads the Build Number and Program GUID of the current User Program installed on the C Series module.
- Build Number—Indicates the build number of the current User Program installed on the C Series module.
- Program GUID—Indicates the program GUID of the current User Program installed on the C Series module.
- Mode—Indicates the current operating mode of the C Series module.
  - **Power Up**—Indicates module is powering up.
  - **Unprogrammed**—Indicates User Program is not written to the C Series module.

  - **User Program Download**—Indicates the User Program is downloading to the C Series module.
  - **Module Firmware Download**—Indicates firmware is downloading to the C Series module.
  - **Verification**—Indicates User Program has successfully downloaded. Requires user verification.
  - **Operational**—Indicates C Series module is running the User Program.
  - **Reserved**—Indicates C Series module is in Reserved mode.
  - **Fail-safe**—Indicates User Program has detected a fault that triggered Fail-safe mode. User must address fault and cycle power to return the C Series module to Operational mode.
- Change Mode to X—Updates C Series module from Verification mode to Operational mode or from Operational mode to Verification mode.
- Path to New User Program—Specifies the file path of the User Program selected for download.
- Build Number—Indicates the build number of the User Program selected for download.
- Program GUID in New Program—Indicates the program GUID of the User Program selected for download.
- Download—Downloads the selected User Program to the C Series module.

Module Methods

Use the Invoke Node to access the following module properties for this device.

Table 517. Module Methods

Method	Description
Check Channel Status	Returns Boolean arrays that indicate the channel status, including various faults. The index number in the array corresponds to the channel number on the module, so that index 0 indicates the channel status for DI0 or DO0. The following parameters are available: • DO Internal Readback—Indicates whether the DO signal is high or low. • DO Channel Overcurrent Fault—Indicates an overcurrent fault has been detected. • DO Channel Open Circuit Fault—Indicates an open circuit fault has been detected. • DI Channel Discrepancy Fault—Indicates a discrepancy fault has been detected.

Method	Description
	• Channel Test Pulse Fault—Indicates a test pulse fault has been detected. • Channel Readback Fault—Indicates a readback fault has been detected. • AI Discrepancy Fault—Indicates a discrepancy fault has been detected. • AI Overcurrent Fault—Indicates a overcurrent fault has been detected. • AI Discrepancy Warning—Indicates which AI channel in a 2oo3 configuration is out of discrepancy range.
Check Module Status	Returns the module status, including digital output values and various faults. The following parameters are available: • Mode—Returns an enum value that reads the operating mode of the NI 9351. ◦ Power Up—Indicates module is powering up. ◦ Unprogrammed—Indicates User Program is not written to the C Series module. ◦ User Program Download—Indicates the User Program is downloading to the C Series module. ◦ Module Firmware Download—Indicates firmware is downloading to the C Series module. ◦ Verification—Indicates User Program has successfully downloaded. Requires user verification. ◦ Operational—Indicates C Series module is running the User Program. ◦ Reserved—Indicates C Series module is in Reserved mode. ◦ Fail-safe—Indicates the User Program has detected a fault that triggered Fail-safe mode. Address the fault and cycle power to return the module to **Operational** mode. • User Program Running—Returns Boolean value to indicate the status of the User Program. • Internal Fault—Returns Boolean value to indicate whether an internal fault has been detected. • Temperature Fault—Returns Boolean value to indicate whether a temperature fault has been detected. • Power Supply Fault—Returns Boolean value to indicate whether a power supply fault has been detected.
Start Program	Starts the User Program.
Set Mode	Sets the operating mode for the NI 9351 as an enum value. Valid values are Operational

Method	Description
	or Verification.
Read Current State	Returns an array of U8 values. The array index corresponds to the state machine number in the User Program and the U8 value indicates the current state of that state machine. Refer to the `report.log` file generated by the Functional Safety Editor to index the state machines and identify the state number.

Module Properties

Use the Property Node to access the following module properties for this device.

Table 518. Module Properties

Property	Description
Firmware Version Number	Returns U32 value to indicate the version of the module firmware.
User Program GUID	Returns a string to indicate the GUID assigned to the User Program.
User Program Version Number	Returns U32 value to indicate the build number of the User Program.

Modular Instruments Modules

Table 519. C Series Modular Instruments Modules

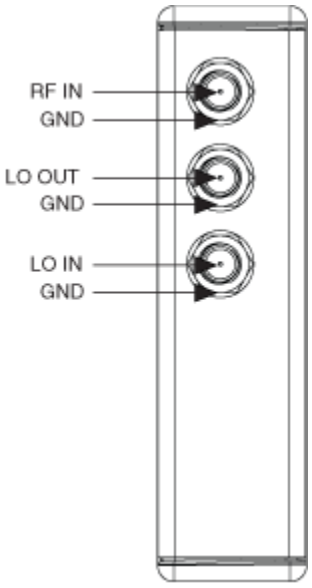
C Series Modular Instruments Module	Description
NI-9770	30 kHz to 100 MHz, C Series RF Receiver Module
NI-9775	±10 V, Up to 20 MS/s/ch, 14-Bit, 4-Channel C Series Digitizer Module

NI 9770

CompactRIO 30 kHz to 100 MHz, 50 Ω, AC Coupled RF Receiver Module

NI 9770 Pinout

Figure 116. NI 9770 Pinout



NI 9770 (FPGA Interface)

FPGA I/O Node

You can use an FPGA I/O Node configured for reading with this device.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 520. Terminals in Software

Terminal	Description
RF In/I	Returns the *I* data that comes from the RF IN connector. The *I* and *Q* data from each module must be accessed from within the same FPGA I/O Node. Do not access I and Q data on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
RF In/Q	Returns the *Q* data that comes from the RF IN connector.

Terminal	Description
	The *I* and *Q* data from each module must be accessed from within the same FPGA I/O Node. Do not access I and Q data on multiple modules in the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.
Onboard Clock	Provides access to the onboard clock in LabVIEW. The onboard clock frequency is 12.288 MHz. Use the FPGA I/O Node in a single-cycle Timed Loop to access this channel. Export the **Onboard Clock** of the module to access this channel.
Start	Channel that controls when the module starts acquiring IQ data. If you write TRUE to the **Start** channel, the module starts acquiring data. When the module is acquiring data, you must write TRUE to the **Stop** channel before you can access properties for this module. If you write FALSE to the **Start** channel, no operation is performed.
Stop	Channel that controls when the module stops acquiring data. If you write *TRUE* to the **Stop** channel, the module stops acquiring data. When the module is acquiring data, you must write *TRUE* to the **Stop** channel before you can access properties for the module. If you write *FALSE* to the **Stop** channel, no operation is performed.

Note For an example of using the **Start** and **Stop** channels, refer to the NI 9770 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9770\NI 9770 Getting Started\NI 9770 Getting Started.lvproj` directory.

Arbitration

This device supports only the Arbitrate if Multiple Requestors Only option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following module methods for this device.

Table 521. Module Methods

Method	Description
Configure RF In	Configures the RF center frequency of the module and input signal reference level. • **Center Frequency**—Specifies in Hz the desired RF center frequency that the module should tune to. • **Reference Level**—Specifies in dBm the expected total power of the RF input signal. • **LO Source**—Optional parameter that specifies the source of the LO signal that the module uses for RF downconversion. Options are **Internal** or **LO In**. When **Internal**, the module uses the internal LO signal that originates from within the module. When **LO In**, the module uses the signal that is connected to the LO IN connector on the module. • **LO In Frequency**—Optional parameter that specifies the frequency in Hz of the signal connected to the LO IN connector on the module. When you set the **LO Source** to **LO In**, you must connect a signal to the LO IN connector on the module and use this parameter to specify the frequency of that signal. • **Downconverter Gain**—Returns the net signal gain in dB of the RF downconverter. Use this parameter to scale the IQ data to the proper power level, as seen at the RF IN connector on the module. **Note** For an example of how to use Downconverter Gain to scale the IQ data, refer to the NI 9770 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9770\NI 9770 Getting Started\NI 9770 Getting Started.lvproj` directory.
Check Cached Status	Returns the following status information that reports whether various conditions occurred on the module since the last execution of the **Check Cached Status** method: • **IQ Data Overflow**—Returns a Boolean value. A value of TRUE indicates that the RF input subsystem tried to acquire a signal that is larger in amplitude than the ADC can accurately measure using the current hardware gain configuration. This problem may result from using a reference level that is too low for the supplied

Method	Description
	input signal. • **LO PLL Unlocked**—Returns a Boolean value. A value of TRUE indicates that the LO PLL is in an unlocked state. The LO PLL must be locked in order to ensure that correct data is acquired from the module. • **Sample Clock PLL Unlocked**—Returns a Boolean value. A value of TRUE indicates that the sample clock PLL is in an unlocked state. When synchronizing multiple modules, the sample clock PLL must be locked in order to ensure that the modules are synchronized properly.

I/O Properties

This device does not support any I/O properties.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 522. Module Properties

Property	Description
Data Rate	Sets the rate at which the module acquires IQ data.
Peak RF In Power	Returns the peak signal power level in dBm that was measured at the RF IN connector across the full RF frequency range of the module. The peak is cleared when this property is read. **Note** This value only updates while the module is acquiring IQ data. When the acquisition stops, this property returns the last measured peak signal power level.
LO Out Enabled	When set to TRUE, the module drives the LO signal to the LO OUT connector on the module. This property is used to *share the LO signal* across multiple NI 9770 modules. This property cannot be set to FALSE when the **LO Source** is set to **LO In**. The module always drives LO OUT when using an external LO signal on the LO IN connector on the module.

Property	Description
Downconverter Gain	Returns the net signal gain in dB of the RF downconverter. Use this parameter to scale the IQ data to the proper power level, as seen at the RF IN connector on the module. **Note** For an example of how to use Downconverter Gain to scale the IQ data, refer to the NI 9770 Getting Started VI in the `labview\examples\CompactRIO\Module Specific\NI 9770\NI 9770 Getting Started\NI 9770 Getting Started.lvproj` directory.
RF Attenuation	Configures the amount of RF attenuation in dB that is applied to the input signal. The valid parameter range is 0 dB to 30 dB. You can read or write to this property. If you write to this property, you must read Downconverter Gain again.
IF Attenuation	Configures the amount of IF attenuation in dB that is applied to the input signal. The valid parameter range is 0 dB to 30 dB. You can read or write to this property. If you write to this property, you must read Downconverter Gain again.
Module ID	Returns the Module ID, 0x7827.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.

Single-Cycle Timed Loop

Use the **Onboard Clock** channel in the single-cycle Timed Loop. You cannot use the other channels on the NI 9770 with the single-cycle Timed Loop. For information about loop timing for this module, refer to the *Understanding Loop Timing (FPGA Interface)* topic.

Related concepts:

- Reading from C Series Channels
- Synchronizing Multiple NI 9770 Modules (FPGA Interface)

- Understanding Loop Timing (FPGA Interface)

Synchronizing Multiple NI 9770 Modules (FPGA Interface)

You can synchronize multiple NI 9770 modules that are connected to the same FPGA device if your application meets the following requirements:

- The modules use the same master timebase source.
- The modules start acquisition mode at the same time.
- A single FPGA I/O Node function reads the synchronous data.

Create FPGA I/O items for the NI 9770 before configuring the items using the FPGA I/O Node. Develop the FPGA VI to meet the guidelines described in this topic.

Note The NI 9770 does not support synchronization with non-NI 9770 modules.

Sharing a Master Timebase Source

Configure the modules to share the same master timebase source.

Sharing the LO Signal

To enable a repeatable phase relationship between the master module and the slave modules, share a local oscillator (LO) signal between a master module and one or more slave modules.

Complete the following steps to configure the modules to share the same LO signal.

1. Designate one module as the master module, which will drive its LO signal to a slave module.
2. Connect the LO OUT terminal on the master module to the LO IN terminal on the slave module.

Note You can add multiple slave modules in a daisy-chain configuration by connecting the LO OUT terminal of one slave module to the LO IN terminal of the next slave module.

Note Add 50 Ω termination to the LO OUT terminal of the last slave module in the chain.

3. Configure the master module to export its LO signal on the LO OUT terminal by setting the **LO Out Enabled** property of the master module to TRUE.
4. Read the **LO Frequency** property of the master module. This value is used to configure the frequency of the LO IN signal on the slave module.
5. Configure the slave module to import its LO signal from the LO IN connector. Use the **Configure RF In** method of the slave module to set the following values:

Parameter	Value
LO Source	LO In
LO In Frequency	The value returned by the **LO Frequency** property of the master module

Starting the Synchronized Acquisition

Configure an FPGA I/O Node with **Start** channels for the NI 9770 modules you want to synchronize and wire a Boolean constant set to TRUE to each **Start** channel.

Note Ensure that all I/O channels are in the same FPGA I/O Node. Otherwise, the FPGA I/O Node will not return synchronized data.

Acquiring Data from Synchronized NI 9770 Modules with the Same Data Rate

Configure an FPGA I/O Node with all of the channels from which you want to synchronously sample.

Note Ensure that all I/O channels are in the same FPGA I/O Node. Otherwise, the FPGA I/O Node will not return synchronized data.

For an example of synchronizing multiple modules with the same data rate, refer to the Synchronizing NI 9770 Modules (FPGA) VI in `labview\examples\CompactRIO\Module Specific\NI 9770\NI 9770 Multi-Module Synchronization\NI 9770 Multi-Module Synchronization.lvproj`.

Acquiring Data from Synchronized NI 9770 Modules with Different Data Rates

If you synchronize NI 9770 modules that are configured for different data rates, create a separate loop for each data rate in the FPGA VI. In each loop, configure an FPGA I/O Node with all of the channels that are configured for the data rate of that loop. If you place NI 9770 channels that are configured for different data rates in the same loop, LabVIEW returns an overrun warning from the FPGA I/O Node (error 65539) when you run the VI. There is a delay before the FPGA I/O Node returns the first data point. The length of the delay depends on the data rate of the NI 9770.

Related concepts:

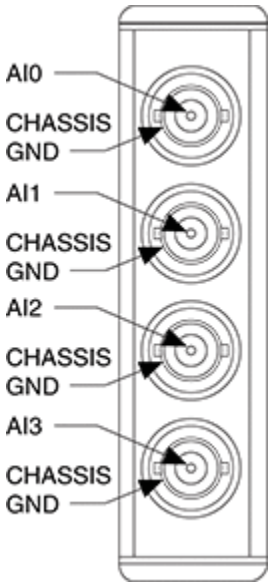
- Configuring the Master Timebase Source for a C Series Module (FPGA Interface)
- Understanding Loop Timing (FPGA Interface)

NI 9775

CompactRIO 4-Ch, ±10 V, 20 MS/s, 14-Bit Digitizer

NI 9775 Pinout

Figure 117. NI 9775 Pinout



NI 9775 (FPGA Interface)

FPGA I/O Node

You can use an FPGA I/O Node configured for reading with this device.

Note You can synchronize an NI 9775 module with other modules that have a selectable timebase source.

Note Use only the 12.8 MHz exported clock of the NI 9469 as the reference clock of the NI 9775. You cannot use the 13.1072 MHz clock with the NI 9775.

Note Use only another NI 9775 or the NI 9469 configured to export its clock as the reference clock for the NI 9775.

Terminals in Software

Use the FPGA I/O Node to access the following terminals for this device.

Table 523. Terminals in Software

Terminal	Description
AI*x*	Analog input channel x, where x is the number of the channel. The NI 9775 has AI channels 0 to 3.
Reference Clock	Gives access to the reference clock in the LabVIEW block diagram. Export the onboard clock of the NI 9775 and use the FPGA I/O Node in a single-cycle timed loop to access this channel.
Start	Controls when the NI 9775 starts acquiring data.
Stop	Controls when the NI 9775 stops acquiring data. When the 9775 is acquiring data, you must write TRUE to this channel before you can access properties of the module.

Caution Do not access AI channels on multiple modules from the same FPGA I/O Node if the modules are not synchronized or do not use the same data rate.

Arbitration

This device supports only the Arbitrate if Multiple Requestors Only option for arbitration. You cannot configure arbitration settings for this device.

I/O Methods

This device does not support any I/O methods.

Module Methods

Use the FPGA I/O Method Node to access the following module methods for this device.

Table 524. Module Methods

Method	Description
Fetch Record Data	Fetches the next available sample from each channel in the Triggered Records. Updates the number of remaining elements of the current record, and the number of remaining records. This method cannot be executed at the same time as the AI*x* I/O node or any of the Trigger Record Method Nodes. • **Records Remaining**—Indicates the number of records available to be fetched from the module. • **Elements Remaining**—Indicates the number of samples available to be fetched from the current record. • **Timeout**—Specifies the minimum number of ticks the Method Node waits before timing out. The Method Node times out if the data is not yet available to access. Set this parameter to -1 if you want the Method Node to wait indefinitely.
Trigger Record on HW Event	Triggers a record event whenever a hardware event occurs on a single channel. Any hardware event that occurs outside of this execution will be lost. Once there are no more records available to trigger, you cannot send any more triggers. • **AI Channel**—Specifies which input channel the hardware event will come from. • **Trigger Slope**—Specifies the slope direction of the hardware event. Can be either "Positive" or "Negative". • **Trigger Level**—Specifies the voltage level used to trigger a record. • **Hysteresis Value**—Specifies the hysteresis value for the trigger. • If the Trigger Slope is positive, the Hysteresis Value is below the Trigger Level. • If the Trigger Slope is negative, the Hysteresis Value is above the Trigger Level. • **Timeout**—Specifies the minimum number of ticks the Method Node waits before timing out. The Method Node times out if the module cannot be triggered or if the hardware event condition has not been met. Set this parameter to -1 if you want

Method	Description
	the Method Node to wait indefinitely. The hardware event may be missed if timeout is set to a small number of ticks.
Trigger Record On HW Event (Multiple Sources)	Triggers a record event whenever a hardware event occurs on any channel. Any hardware event that occurs outside of this execution will be lost. Once there are no more records available to trigger, you cannot send any more triggers. • **Trigger Configuration**—Specifies the trigger configurations for each channel. Each of these contains the following channels: Enable, Trigger Slope, Trigger Level, and Hysteresis Value. • **Timeout**—Specifies the minimum number of ticks the Method Node waits before timing out. The Method Node times out if the module cannot be triggered or if the hardware event condition has not been met. Set this parameter to -1 if you want the Method Node to wait indefinitely. The hardware event may be missed if timeout is set to a small number of ticks.
Trigger Record	Signals the module to trigger a record. Once there are no more records available to trigger, you can not send any more triggers. • **Timeout**—Specifies the minimum number of ticks the Method Node waits before timing out. The Method Node times out if the module cannot be triggered. Set this parameter to -1 if you want the Method Node to wait until the module can trigger. Set this parameter to 0 if you want the Method Node to trigger immediately if there is not already a record being acquired.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 525. I/O Properties

Property	Description
Enable Analog Filter	Changes the configuration of the analog filter
LSB Weight (High-Resolution)	Returns the LSB weight in nV/LSB.
LSB Weight (High-Speed, Analog Filter Disabled)	Returns the LSB weight in nV/LSB.
LSB Weight (High-Speed, Analog Filter Enabled)	Returns the LSB weight in nV/LSB.
Offset (High-Resolution)	Returns the calibration offset in µV.
Offset (High-Speed, Analog Filter Disabled)	Returns the calibration offset in µV.
Offset (High-Speed, Analog Filter Enabled)	Returns the calibration offset in µV.

Module Properties

Use the FPGA I/O Property Node to access the following module properties for this device.

Table 526. Module Properties

Property	Description
Module ID	Returns the Module ID, 0x7889.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the NI vendor ID, 0x1093.
Record Data Rate	Specifies the rate at which the module acquires data for the Record Acquisition.
Record Pre-Trigger Samples	Specifies the number of acquired samples prior to the trigger of a record.
Continuous Data Rate	Specifies the rate at which the module acquires data in continuous mode.
Records Available to Fetch	Returns the number of records available to fetch in the module.
Records Available to Trigger	Returns the number of records available to trigger in memory.
Total Record Samples	Specifies the number of samples per record.
Timing Mode	Changes the module configuration between High-Speed and High-Resolution

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Related concepts:

- Reading from C Series Channels
- Synchronizing Multiple Modules (FPGA Interface)

NI 9775 Acquisition Modes (FPGA Interface)

The NI 9775 has three different acquisition modes:

- Continuous Mode
- Record Mode
- Advanced Mode.

Continuous Mode

In continuous mode, the NI 9775 transfers real-time data to the chassis at an aggregate rate of 4MS/s across all channels. The NI 9775 defaults to continuous mode after executing the Start I/O Node. Configure the data rate through the Module Property **Continuous Data Rate**. The maximum value of the data rate varies based on the configuration. Refer to the module hardware documentation for the data rate.

To implement Continuous Mode, complete the following steps.

1. Configure the **Timing Mode**, **Continuous Data Rate** and **Analog Filter Configuration**.
2. Run the **Start** I/O Node.
3. Run the **AI*x*** I/O Node as many times as needed.
4. Run the **Stop** I/O Node.

Record Mode

In record mode, the NI 9775 stores samples into onboard memory at up to 20 MS/s then transfers the data to the chassis at a slower rate. The start of the data collection is based on a trigger signal sent to the device. This signal can be digital or analog. The collected data is stored in the buffer until the entire acquisition is restarted.

To implement Record Mode, complete the following steps.

1. Configure the **Timing Mode**, **Record Data Rate**, **Record Samples**, **Record Pre-Trigger Samples**, and **Analog Configuration**.
2. Run the **Start** I/O Node.
3. Trigger a record using any of the following nodes: **Trigger Record**, **Trigger Record on HW Event**, or **Trigger Record on HW Event (Multiple Sources)**.

4. Run the **Fetch Record Data** Method Node until there are no more samples available to fetch.
5. Run the **Stop** I/O Node.

Advanced Mode

In advanced mode, the NI 9775 combines the functionality of continuous mode and record mode to enable more complex triggering. Advanced mode can use live data from continuous mode to trigger a faster collection rate using record mode. This can be used for applications such as pattern detection or frequency triggering, and gives you more advanced triggering options.

To implement Advanced Mode, complete the following steps.

1. Configure **Timing Mode**, **Record Data Rate**, **Record Samples**, **Record Pre-Trigger Samples**, **Analog Filter Configuration**, and **Continuous Data Rate**.
2. Run the **Start** I/O Node.
3. Use the **AI*x*** I/O Node to build a custom trigger condition.
4. Trigger on the custom trigger condition using any Trigger Record Method Node for as many times as needed, up to the number of available records.
5. Run the **Fetch Record Data** Method Node until there are no more samples available to fetch.
6. Run the **Stop** I/O Node.

Timing and Synchronization Module Reference

Table 527. C Series Timing and Synchronization Modules

Timing and Synchronization Module	Description
NI-9467	GPS C Series Synchronization Module
NI-9469	Chassis C Series Synchronization Module

NI 9467

Figure 118. NI 9467 front panel



NI 9467 (FPGA Interface)

FPGA I/O Node

This device does not support any FPGA I/O Nodes.

Module Methods

Use the **FPGA I/O Method Node** to access the following I/O methods for this device.

Table 528. Module Methods

Method	Description
Wait for PPS	Waits until a new PPS is detected, indicating the start of a new second. The NI 9467 Module does not generate a PPS at start-up. The module begins generating a PPS only after it obtains the first location fix. This takes 30 seconds or more of access to GPS satellite signals with a clear view of the sky. This method has the following parameters: • **Timeout (Ticks)**—Specifies in FPGA clock ticks how long the Wait for PPS method waits for the PPS to occur. A value of 0 causes the Wait for PPS method to time out immediately, a negative value causes the Wait for PPS method to wait indefinitely, and a positive value causes the Wait for PPS method to wait for that number of clock ticks before timing out. • **Timed Out**—Returns a Boolean value. A value of TRUE indicates that a timeout occurred. The timeout counter starts counting only when module startup is complete.
Wait for Data Update	Waits until new data is received for the last PPS and the properties are updated. At start-up, when the PPS generation has not started, this method indicates the periodic update of the GPS Status, Antenna Status, and Number of Satellites properties. In the time between the PPS and when this method completes, the properties update at different times. Therefore, do not read properties during this time. Instead, read them after the Wait for Data Update method completes. This method has the following parameters: • **Timeout (Ticks)**—Specifies in FPGA clock ticks how long the Wait for Data Update method waits for the update to occur. A value of 0 causes the Wait for Data Update method to time out immediately, a negative value causes the Wait for Data Update method to wait indefinitely, and a positive value causes the Wait for Data Update method to wait for that number of clock ticks before timing out. • **Timed Out**—Returns a Boolean value. A value of TRUE indicates that a timeout occurred.

Method	Description
	The timeout counter starts counting only when module startup is complete.
Reset Module	This method makes the module reinitialize to its start-up state. When the module is reset, it stops generating a PPS. The module starts generating a PPS again after it obtains the first location fix. This takes 30 seconds or more of access to GPS satellite signals with a clear view of the sky.

Module Properties

Use the **FPGA I/O Property Node** to access the following module properties for this device.

Note In the time between the PPS and when the Wait for Data Update method completes, the properties update at different times. Therefore, do not read properties during this time. Instead, read them after the Wait for Data Update method completes.

Table 529. Module Properties

Property	Type	Description
PPS Timestamp (TAI)	Read	Returns a U64 value representing the TAI timestamp in nanoseconds of the last PPS.
UTC Offset	Read	Returns an I8 value representing the current offset between TAI time and UTC time in seconds. To compute UTC time, the UTC Offset must be subtracted from the TAI time `(UTC_Time = TAI_Time – UTC_Offset)`. The GPS satellites transmit the UTC Offset information only once every 12.5 minutes. This property reports invalid values for up to 12.5 minutes after start-up. Use the UTC Offset Valid property to determine when this property starts returning valid values.

Property	Type	Description
UTC Offset Valid	Read	Returns a Boolean indicating if the UTC Offset and leap second information has been received from the GPS satellites, making the UTC Offset, Leap Second Pending, Leap Second Direction, and Leap Second Occurred properties valid.
GPS Status	Read	Reports the current GPS status: • **Normal**—The module is operating properly. • **Unable to get fix**—The module is unable to compute a location fix and generate a precise PPS. This occurs during start-up and when the satellite signals are obstructed. • **Initializing**—This status is reported at start-up or after a module reset until the module is initialized.
Self-Survey Complete	Read	Returns *True* if the self-survey is complete. Returns *False* if the self-survey is in progress.
Antenna Status	Read	Reports the current antenna status: • **Normal**—Antenna is detected (present) and there is no overcurrent. • **Absent**—Antenna is not detected. • **Overcurrent**—An overcurrent condition has been detected. • **Unknown**—Status reported while the module is initializing.
Leap Second Pending	Read	Boolean that returns *True* for up to 60 seconds before the leap second occurs, and cleared the second after the leap second occurs. The GPS satellites transmit the Leap Second information only once every 12.5 minutes. This property reports invalid values for up to 12.5 minutes after start-up. Use the UTC Offset Valid property to determine when this property starts returning valid values.
Leap Second Occurred	Read	Boolean that returns *True* starting the first second after the leap second occurs and remains True for 24 hours. The GPS satellites transmit the Leap Second information only once every 12.5 minutes.

Property	Type	Description
		This property reports invalid values for up to 12.5 minutes after start-up. Use the UTC Offset Valid property to determine when this property starts returning valid values.
Leap Second Direction	Read	Returns *Add* or *Delete* depending on the kind of leap second event. Only valid when Leap Second Pending or Leap Second Occurred Properties are *True*. Disregard at other times. The GPS satellites transmit the Leap Second information only once every 12.5 minutes. This property reports invalid values for up to 12.5 minutes after start-up. Use the UTC Offset Valid property to determine when this property starts returning valid values.
Altitude (m)	Read	Return the current location coordinates in degrees and altitude in meters. The Altitude property reports altitude in meters HAE (height above ellipsoid). When the GPS receiver cannot compute a location fix during the self-survey (as reported by the GPS Status property), the location properties repeat the information obtained in the last successful location fix. When the self-survey is complete, the location properties return the precise location computed during the self-survey.
Latitude (°)	Read	
Longitude (°)	Read	
Satellites Available	Read	Returns the number of satellites the GPS receiver currently is tracking.
Firmware	Read	Returns the current GPS receiver firmware version.
Module ID	Read	Returns the *module ID*.
Serial Number	Read	Returns the unique module serial number.
Vendor ID	Read	Returns the National Instruments vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Glossary

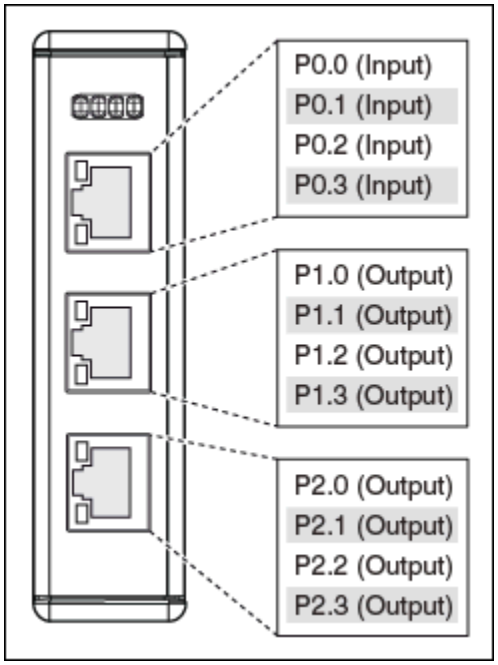
- **Clear view of the sky**—There is no strict definition for a clear view of the sky, but a suitable guideline is that the GPS antenna should have a clear line of sight to the sky in all directions (360°) down to an imaginary line making a 30° angle with the ground. Locations far from trees and tall buildings that could block or reflect GPS satellite signals are best.
- **GPS**—Global Positioning System
- **Leap second**—A one-second adjustment to UTC (with respect to TAI) with the intent of keeping the UTC timescale close to mean solar time. Leap seconds can be added or subtracted, though to date, they have only been added. The occurrence of leap seconds is not regular; they are announced up to 6 months prior to the occurrence, and they can only occur at 23:59:59 UTC on 30/June or 31/December.
- **Location fix**—Computation of latitude, longitude, and altitude of a location based on the signals received from GPS satellites.
- **PPS**—Pulse per second
- **Self-survey**—Method used by the GPS module to determine its precise geographic location. During the self-survey period, the GPS module acquires a new location fix every second. At the end of the self-survey period, all location fixes are averaged to obtain a more precise estimate of the current location.
- **TAI**—International Atomic Time (TAI, from the French name Temps Atomique International). Timescale based on atomic clocks distributed around the world, and thus not adjusted by leap seconds. The reference for TAI is 1/Jan/1970 00:00:00.
- **UTC**—Coordinated Universal Time. Time used in most civilian clocks, based on TAI and adjusted with leap seconds.

NI 9469

NI 9469 is a Delta-Sigma Synchronization Module.

You can use the NI 9469 in FPGA Interface mode.

Figure 119. NI 9469 Port Assignments



NI 9469 (FPGA Interface)

FPGA I/O Node

This device does not support the FPGA I/O Node. You can use the **FPGA I/O Method Node** and **FPGA I/O Property Node** to access methods and properties for the device.

Note If you add the NI 9469 to the project, the module's support compiles into the FPGA when you compile the FPGA VI, which can take up space on the FPGA. If you are not actively using the NI 9469, you may want to remove the module item from the project.

Note Trig0, Trig1, Trig2, and Trig3 are default possible Reference Trigger Sources for other modules unless the clock (Trig3 only) or another module's trigger is selected as the source. You can set the source for Trig0, Trig1, Trig2, and Trig3 through the Custom Configuration Dialog Box.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Trigger Methods

Use the **FPGA I/O Method Node** to access the following methods with this device.

Table 530. Trigger Methods

Method	Description
Wait On Trigger	Pauses the execution of the I/O Method Node until a trigger is received at the selected I/O item. The **Timeout** input specifies in FPGA clock ticks how long the Wait On Trigger method waits for the next trigger. A value of 0 causes the method to time out immediately, a negative value causes the method to wait indefinitely, and a positive value causes the method to wait for that number of clock ticks before timing out.  **Note** For the first two seconds after resetting the FPGA VI, the time out will not start counting until the module is identified.
Send Trigger	Sends a trigger at the specified trigger selected as the I/O item. The Send Trigger method will return as soon as the trigger is sent, however, once the trigger is sent, the next Send Trigger of the same trigger line will hold the execution of the I/O Method Node until the module is ready to send a new trigger. If the module is being re-configured, the Send Trigger method will hold its execution until the module is configured.

Module Methods

Use the **FPGA I/O Property Node** to access the following methods for this device.

Table 531. Module Methods

Method	Description
Set Line Source	Configures specified line to be connected to the specified source of the NI 9469. This method will check if the connection is valid and will not configure the module if such a connection is invalid. Refer to *NI 9469 error codes* for more details about the errors this method may produce.
Restore Configuration	Restores the entire configuration of the NI 9469 defined at the module **Configuration** sub panel.
Set Clock Source	Configures the module to use the specified Clock Source as the new clock reference. This checks if the requested Clock Source is valid and will not configure the module if such a request is invalid. Refer to *NI 9469 error codes* for more details about the errors this method may produce.

Method	Description
Set Internal Timebase Frequency	Configures the frequency that the internal oscillator will be generating. To select this frequency as the Clock Source, use the Set Clock Source method selecting Internal Timebase as the Clock Source.
Check Clock Status	Returns a Boolean value that indicates whether the clock generated from the NI 9469 is present and ready to be used.

Module Properties

Use the **FPGA I/O Property Node** to access the following properties with this device. These properties have lower priority than port-specific methods or properties. In arbitration, the port-specific methods and properties will be executed before these properties.

Table 532. Module Properties

Property	Description
Module ID	Returns the *module ID*.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the National Instruments vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the single-cycle Timed Loop.

Controller Area Network (CAN) Interface Module Reference

Table 533. C Series CAN Interface Modules

CAN Interface Module	Description
NI-9852	2-Port, Low-Speed/Fault Tolerant C Series CAN Interface Module
NI-9853	2-Port, High-Speed C Series CAN Interface Module

In this document, *CAN module* refers to the NI-9852 and NI-9853 C Series modules. Within CompactRIO software, the NI-9852 is designated as cRIO-9852, and the NI-9853 is designated as cRIO-9853.

C Series CAN Interface Modules

Table 534. CAN Interface Modules for CompactRIO

CAN Interface Module	Description
NI-9852	2-Port, Low-Speed/Fault Tolerant C Series CAN Interface Module
NI-9853	2-Port, High-Speed C Series CAN Interface Module

NI 9852

The NI 9852 is a 2-port, low-speed/fault-tolerant CAN Module

FPGA I/O Node

You can use an FPGA I/O Node for either CAN Input or CAN Output.

Terminals in Software

You can select the following terminals for this device.

Table 535. CAN Terminals

Terminal	Description
CAN0	CAN Port 0
CAN1	CAN Port 1

I/O methods

This device supports the following I/O methods.

Table 536. I/O Methods

Method	Description
Abort Transmit	Abort a pending CAN frame transmission.
Reset	Reset the CAN port to the same state as when the FPGA VI started running.
Start	Start communication.
Stop	Stop communication.
Wait on Comm State Change	Wait for a change in the communication state of the CAN port (*Comm State* property).
Wait on Transceiver Wakeup	Wait for the *Transceiver Mode* property to change from Sleep to Normal mode due to a remote wakeup (bus activity) or local wakeup.
Wait on Transmit Complete	Wait for all frames written to CAN Output to complete transmission.

I/O Properties

Use the FPGA I/O Property Node to access the following properties with this device.

Table 537. I/O Properties

Property	Description
Bit Timing	Specifies the baud rate as values for the Bit Timing Registers (BTR0 and BTR1).
Comm State	Describes the current communication state of the CAN controller.
Listen Only	Controls listen-only mode for passive monitoring/logging.

Property	Description
Log Bus Errors	Enables the logging of bus errors as frames that can be read using the *CAN Input* node.
Log Transceiver Faults	Enables the logging of transceiver faults as frames that can be read using the *CAN Input* node.
LS/FT Termination	Enables the setting of the low-speed/fault-tolerant transceiver termination to either 1 kiloohm (1.11 kΩ) or 5 kiloohm (4.99 kΩ).
Receive Error Counter	Provides access to the CAN controller Receive Error Counter.
Self Reception	Specifies whether to echo successfully transmitted CAN frames to be read using *CAN Input*.
Single Shot Transmit	Specifies whether to retry failed CAN frame transmissions.
SJA1000 Filter Code	Controls the SJA1000 Acceptance Code registers (ACR0 to ACR3).
SJA1000 Filter Mask	Controls the SJA1000 Acceptance Mask registers (AMR0 to AMR3).
SJA1000 Filter Mode	Controls the SJA1000 Acceptance Mode register.
Transceiver Mode	Sets the mode for the CAN transceiver and the associated mode in the SJA1000 CAN controller.
Transmit Error Counter	Provides access to the CAN controller Transmit Error Counter.

Module Properties

Use the FPGA I/O Property Node to access the following properties with this device.

Table 538. Module Properties

Property	Description
Module ID	Returns the module ID. Refer to *C Series Module IDs* for a list of modules and the associated IDs.
Serial	Returns the unique serial number of the CAN module.

Property	Description
Number	
Vendor ID	Returns the National Instruments vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the Single-Cycle Timed Loop.

NI 9853

The NI 9852 is a 2-port, low-speed/fault-tolerant CAN Module

FPGA I/O Node

You can use an FPGA I/O Node for either CAN Input or CAN Output.

Terminals in Software

You can select the following terminals for this device.

Table 539. CAN Terminals

Terminal	Description
CAN0	CAN Port 0
CAN1	CAN Port 1

I/O methods

This device supports the following I/O methods.

Table 540. I/O Methods

Method	Description
Abort Transmit	Abort a pending CAN frame transmission.
Reset	Reset the CAN port to the same state as when the FPGA VI started running.
Start	Start communication.

Method	Description
Stop	Stop communication.
Wait on Comm State Change	Wait for a change in the communication state of the CAN port (*Comm State* property).
Wait on Transceiver Wakeup	Wait for the *Transceiver Mode* property to change from Sleep to Normal mode due to a remote wakeup (bus activity) or local wakeup.
Wait on Transmit Complete	Wait for all frames written to CAN Output to complete transmission.

I/O Properties

Use the FPGA I/O Property Node to access the following properties with this device.

Table 541. I/O Properties

Property	Description
Bit Timing	Specifies the baud rate as values for the Bit Timing Registers (BTR0 and BTR1).
Comm State	Describes the current communication state of the CAN controller.
Listen Only	Controls listen-only mode for passive monitoring/logging.
Log Bus Errors	Enables the logging of bus errors as frames that can be read using the *CAN Input* node.
Log Transceiver Faults	Enables the logging of transceiver faults as frames that can be read using the *CAN Input* node.
Receive Error Counter	Provides access to the CAN controller Receive Error Counter.
Self Reception	Specifies whether to echo successfully transmitted CAN frames to be read using *CAN Input*.
Single Shot Transmit	Specifies whether to retry failed CAN frame transmissions.
SJA1000 Filter Code	Controls the SJA1000 Acceptance Code registers (ACR0 to ACR3).
SJA1000 Filter Mask	Controls the SJA1000 Acceptance Mask registers (AMR0 to AMR3).

Property	Description
SJA1000 Filter Mode	Controls the SJA1000 Acceptance Mode register.
Transceiver Mode	Sets the mode for the CAN transceiver and the associated mode in the SJA1000 CAN controller.
Transmit Error Counter	Provides access to the CAN controller Transmit Error Counter.

Module Properties

Use the FPGA I/O Property Node to access the following properties with this device.

Table 542. Module Properties

Property	Description
Module ID	Returns the module ID. Refer to *C Series Module IDs* for a list of modules and the associated IDs.
Serial Number	Returns the unique serial number of the CAN module.
Vendor ID	Returns the National Instruments vendor ID, 0x1093.

Single-Cycle Timed Loop

This device does not support the Single-Cycle Timed Loop.

CAN Module Configuration

Adding Targets and Devices

Configure the CAN module before creating an FPGA VI.

Before you can configure the CAN module, add the module to your CompactRIO configuration. Refer to **Add Targets and Devices** dialog box for information about how to add the CAN module to your CompactRIO configuration.

1. To add the CAN module to the Project Explorer window, select the CAN module

from the list of modules in the New C Series Module dialog box.
2. To open the Module Configuration dialog box, right-click the NI 9852 or NI 9853 module in the Project Explorer window and select **Properties** from the shortcut menu.

Module Configuration

Configuration properties are downloaded to the CAN module each time the FPGA VI runs.

Separate configuration properties are provided for each CAN port on the module. Configuration properties are labeled as **CAN0** tab and **CAN1** tab.

Module configuration properties apply to all CAN ports on the module.

Configuration properties are downloaded to the CAN module each time the FPGA VI runs. Configuration properties are saved as part of the LabVIEW project file and the FPGA VI image.

This module configuration often makes it possible for you to avoid setting properties in the VI, and focus on CAN Input/Output within the FPGA VI diagram.

Related concepts:

- Bit Timing
- Advanced CAN Port Properties
- CAN Input
- Error Terminals
- Start

CAN Port Properties

The **CAN0** and **CAN1** tabs provide configuration properties for their respective CAN port.

Baud Rate

This specifies the baud rate (bit rate) to use for CAN communication. Typical CAN baud

rates are listed as kilobits per second (kbps).

The default baud rate for the NI 9852 module is **125.0 kbps**. **125.0 kbps** is the typical baud rate used for low-speed transceivers.

The default baud rate for the NI 9853 module is **500.0 kbps**. **500.0 kbps** is the typical baud rate used for high-speed transceivers. If you prefer to specify the baud rate as bit timing register values, refer to *CAN Advanced Port Configuration*.

Auto Start

This **Boolean** property indicates whether to invoke the **Start** method automatically when the FPGA VI runs. The default is *TRUE* (enabled).

When **Auto Start** is *TRUE* (enabled), the **Start** method is invoked automatically when the FPGA VI runs, and the **Stop** method is invoked automatically when the FPGA VI stops running. This enables your FPGA VI diagram to begin using CAN Input and CAN Output nodes without first using an explicit **Start** method.

When **Auto Start** is *FALSE* (disabled), the **Start** method is not invoked automatically. You must use the **Start** method in your FPGA VI diagram to start communication. The behavior for **Stop** is the same as when **Auto Start** is *TRUE*, in that the **Stop** method is invoked automatically when the FPGA VI stops running.

Listen Only

This **Boolean** control indicates whether to enable the **Listen Only** feature for passive monitoring of the network. The default is *FALSE* (disabled).

When **Listen Only** is *FALSE* (disabled), you can transmit CAN messages normally using CAN Output. When CAN messages are received, those messages are acknowledged.

When **Listen Only** is *TRUE* (enabled), you cannot transmit CAN messages. When CAN messages are received, those messages are not acknowledged. The Philips SJA1000 CAN controller enters error passive state when **Listen Only** is enabled. Checking **Listen Only** enables passive monitoring of network traffic, which can be useful for debugging scenarios in which only one device exists on the network.

LS/FT Termination

The **LS/FT Termination** control applies to NI 9852 low-speed, fault-tolerant modules. This control enables you to set the LS/FT port termination to **1 kiloohm** or **5 kiloohm**.

Advanced

The **Advanced** button displays a dialog to configure advanced properties for the CAN port. Refer to *CAN Advanced Port Configuration* for more information.

CAN Module Properties

The **Module** tab provides configuration properties that apply to all CAN ports on the module.

Module Clock

Performance of the I/O, methods, and properties is determined by the speed of communication between LabVIEW FPGA and the module.

- On a CompactRIO Reconfigurable Embedded system, communication speed is limited to 20 MHz.

  Note To write CAN applications with consistent performance regardless of LabVIEW FPGA target, use the **Module Clock** property to select *10MHz*. This setting is compatible with the CompactRIO R Series Expansion system.

- On a CompactRIO R Series Expansion system, communication speed is limited to 10 MHz. This is the only setting available in the **Module Clock** property.

Advanced CAN Port Properties

Use the CAN Advanced Port Configuration dialog box to configure advanced properties for the CAN port. Open this dialog from the Module Configuration dialog box.

Related concepts:

- Bit Timing
- Module Configuration
- Error Terminals
- Arbitration
- CAN Input

Timeouts

Enter a short description of your concept here (optional).

Timeouts

The **Input Timeout (ms)** and **Output Timeout (ms)** properties specify timeout values to use with CAN nodes in the FPGA VI block diagram.

Each timeout is a signed 32-bit integer with a resolution of milliseconds, thus allowing a maximum value of approximately 25 days. Special values of *0* (do not wait) and *–1* (wait indefinitely) are supported.

The CAN Input function waits for a new CAN frame to be received, then returns that frame. The **Input Timeout (ms)** specifies how long to wait for a new frame to be received.

If you specify **Input Timeout (ms)** of 0, the CAN Input node will simply check to see if a new frame has arrived (non-blocking). If a new frame exists, CAN Input returns the frame with an error status of *FALSE* (success). If no new frame exists, CAN Input returns an error status of *TRUE* (error). Therefore, in order to poll for new frames using an input timeout of 0, you must enable **Error Terminals** for the CAN Input node. When an error is returned, you must invoke CAN Input again at a later time to ensure no data is lost.

The communication path from LabVIEW FPGA to the CAN port is implemented as a FIFO. This allows multiple frames to be sent to the CAN module for transmit, thus enabling VIs that generate full bus load.

When the output FIFO is full, the CAN Output node waits for an element to become available, then writes the frame to the FIFO. **Output Timeout (ms)** specifies how long to wait for a new element to become available, which occurs when a frame from a

previous CAN Output transmits successfully onto the network. If you specify **Output Timeout (ms)** of 0, the CAN Output node returns an error status of TRUE (error) if a new element is not available in the FIFO (non-blocking). When an error is returned, you must attempt CAN Output of the same frame again at a later time.

In addition to the CAN Output node, **Output Timeout (ms)** also applies to most methods and properties. For example, if **Output Timeout (ms)** is 0, the Abort Transmit method returns an error if the output FIFO is full. Refer to the *Arbitration* topic for more information on the output path from LabVIEW FPGA to the CAN port.

Bit Timing

When **Specify Baud Rate as Bit Timing Registers** is *FALSE* (unchecked), the baud rate for the CAN port is selected using the **Baud Rate** in the *Module Configuration* dialog box. The SJA1000 bit timing registers for the corresponding baud rate are grayed out.

When **Specify Baud Rate as Bit Timing Registers** is *TRUE* (checked), the **Baud Rate** in the Module Configuration dialog box is ignored. Two integer controls **BTR0** and **BTR1** set the baud rate for the CAN port.

BTR0 and **BTR1** values are specified as hexadecimal.



Note For information on CAN bit timing registers, refer to the *Philips SJA1000 standalone CAN controller* datasheet. This document is available to download from . This document is available to download from `www.nxp.com`.

Input Filter

The **Input Filter** selection controls the acceptance filter registers of the SJA1000 CAN controller.

The acceptance filter enables the SJA1000 to filter out certain CAN frames so that your FPGA VI can work with reduced overall traffic. The acceptance filter applies only to received frames. The acceptance filter does not apply to transmitted frames.

The **Input Filter** selection determines how you specify the following filter values in the

dialog box:

- **Receive All**—Receive all frames (all identifiers). This value is the default. Register values corresponding to reception of all identifiers are shown in **SJA1000 Filter Mode**, **SJA1000 Mask**, and **SJA1000 Code** (grayed).
- **Receive Selected**—Receive frames with selected identifiers. Based on a list of identifiers that you enter, the dialog will calculate values for the SJA1000 filter registers. The calculation attempts to filter out as many unwanted IDs as possible. Select the list of identifiers using the controls in the following table.

Table 543. Received Selected Identifier Controls

Identifier Controls	Description
ID Format	Selects whether your CAN network uses **Standard** identifiers (11-bit) or **Extended** identifiers (29-bit).
ID Display	Selects whether to use **Decimal** or **Hexadecimal** for the identifiers in the **Add** control and **Selected IDs** list.
Add From Database	Opens a file dialog from which you can select a CANdb file (`.DBC`) or an NI-CAN database file (`.NCD`). When you click the OK button in the file dialog, the identifiers for all messages in the database are added to the **Selected IDs**. The **Add From Database** button requires features of NI-CAN 2.3 or higher in order to import identifiers. If you do not have NI-CAN 2.3 or higher installed on your Windows system, the **Add From Database** button is grayed out.
Add	Enter a single identifier using the control to the right, then click this button to add the identifier to the **Selected IDs**.
Delete	Select a single identifier within the **Selected IDs** list, then click this button to delete that identifier from the list.
Delete All	Deletes all identifiers in **Selected IDs**.
Selected IDs	List of identifiers that you have selected using controls to the left. The dialog calculates the SJA1000 filters from this list, and the resulting register values are shown in **SJA1000 Filter Mode**, **SJA1000 Mask**, and **SJA1000 Code** (grayed). If you change the **Input Filter** from **Receive Selected** to **SJA1000 Format**, the register values in **SJA1000 Filter Mode**, **SJA1000 Mask**, and **SJA1000 Code** retain their calculated values.

Identifier Controls	Description
	This allows you to use **Receive Selected** to calculate filters for the identifiers, then change the registers directly for advanced filtering, such as checks for patterns in the data bytes.
Number of Unwanted IDs	Given the current filter calculation, this displays the number of identifiers that your FPGA VI will receive which are not listed in **Selected IDs**. Since the SJA1000 registers filter using bitwise masking, the filter efficiency is rarely 100%. If you use **ID Format** of **Extended**, this number can be large due to the 29 total bits used for identifiers.

- **SJA1000 Format**—Uses controls to set the SJA1000 registers directly. The following table lists the controls used with **SJA1000 Format**.

> **Note** For information on the SJA1000 acceptance filter registers (PeliCAN mode), refer to the *Philips SJA1000 standalone CAN controller* datasheet. This document is available to download from `www.nxp.com`.

Table 544. SJA1000 Format Controls

SJA1000 Format Control	Description
SJA1000 Filter Mode	Controls the Acceptance Filter Mode bit of the SJA1000 Mode register. Dual (0) specifies use of two filters, and Single (1) specifies use of a single filter.
SJA1000 Mask	Controls the SJA1000 Acceptance Mask 0–3 registers. The value is specified as hexadecimal. The least significant byte of the value is stored in register offset 3 (AMR3), the next least significant in offset 2 (AMR2), and so on.
SJA1000 Code	Controls the SJA1000 Acceptance Code 0–3 registers. The value is specified as hexadecimal. The least significant byte of the value is stored in register offset 3 (ACR3), the next least significant in offset 2 (ACR2), and so on.

CAN Input

Wait for a CAN frame to be received, then return that frame.

Complete the following steps to read from a CAN port using CAN Input.

1. Use the **Add Targets and Devices** dialog box to add the CAN module to your CompactRIO configuration. Right-click the CAN module in the **Project Explorer** window and select Properties to specify the configuration properties for the module.
2. Create an FPGA I/O item for CAN by right-clicking the FPGA target under **My Computer** in the **Project Explorer** window, selecting **New»FPGA I/O**, and then adding **CAN»NI 9852»CAN0** (or **CAN»NI 9853»CAN0**) and/or **CAN»NI 9852»CAN1** (or **CAN»NI 9853»CAN1**) to the FPGA I/O list.
3. Place an **I/O Node** from the FPGA I/O palette onto your FPGA VI diagram.
4. Right-click the I/O Node and select **Select FPGA I/O»CAN»NI 9852»CAN0** (or **Select FPGA I/O»CAN»NI 9853»CAN0**) or **Select FPGA I/O»CAN»NI 9852»CAN1** (or **Select FPGA I/O»CAN»NI 9853»CAN1**).
5. Right-click the I/O Node and select **Change to Read** to use the I/O node for CAN Input.
6. Right-click the I/O Node and choose **Properties** to select the Data Type.

Start communication on the CAN port prior to using CAN Input. Enable **Auto Start** in the Module Configuration, or invoke the Start method to start communication.

The **Input Timeout (ms)** of the CAN Advanced Port Configuration dialog box specifies how long to wait for a new frame to be received. If you specify **Input Timeout (ms)** of 0, the CAN Input node will simply check to see if a new frame has arrived (non-blocking).

For information on how CAN Input arbitrates with other CAN nodes, refer to *Arbitration*.



Note You can add more than one CAN input port to the FPGA I/O node, and also add analog and digital inputs to the same FPGA I/O node by right-clicking the FPGA I/O node and selecting **Add Element**. Right-click the created I/O name and choose **Select FPGA** I/O to specify the I/O name. If you use more than one input per node, LabVIEW executes them sequentially starting with the top input. If one input is blocked while waiting for the input, subsequent inputs are delayed. For example, if you read CAN0 and CAN1 with the same CAN Input node, that node waits for a frame to be received on CAN0, then it will wait for a frame to be received on CAN1. It is more efficient to access each CAN port using a distinct CAN Input node.

Related concepts:

- Module Configuration
- Advanced CAN Port Properties
- Arbitration
- Error Terminals

Properties Dialog

The **Data Type** selects the type for the CAN frame returned by the CAN Input node. The available values are **Cluster** (default) and **Array of 6 U32**.

The benefits of using the **Cluster** data type are:

- Simpler block diagram (easy to use)
- Fast and efficient, if you limit use to the block diagram only (not front panel)

The benefits of the **Array of 6 U32** data type, and handshaking integers to the host VI one at a time, are:

- Faster for transfer to/from host VI
- Fewer FPGA gates

Refer to the *Using Clusters and Arrays in LabVIEW FPGA Application Note* for more information on the use of clusters and arrays.

Node Inputs

-  **Error In**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.

Node Outputs

-  **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable. Unless you set **Input Timeout (ms)** to –1 to wait indefinitely, you must use the error terminals to determine whether the CAN Input successfully received a CAN frame.
-  **CAN*x***—If the **Data Type** is defined as **Cluster** (default), the CAN frame is represented as a cluster. Refer to the following table for the cluster elements.

Table 545. CAN*x* Cluster Elements

Data Type	Element	Description
U32	**Timestamp High**, **Timestamp Low**	Timestamp of when the frame was received. The timestamp is acquired at the end of the CAN frame. The high and low U32 represent a single U64 timestamp. The timestamp is large enough to avoid handling rollover. The timestamp is zero-based (relative). The time starts ticking at zero when CAN communication starts. The resolution is 100 ns.
U32	**Identifier**	Arbitration ID. If bit 29 (20000000 hexadecimal) is clear, this ID uses the standard format (11-bit). If bit 29 is set, this ID uses extended format (29-bit).
U8	**Type**	Type of frame. • **Data Frame**: 0 • **Remote Frame**: 1 • **Bus Error**: 6 • **Transceiver Fault**: 7
U8	**InfoA**	Reserved for future use.
U8	**InfoB**	Reserved for future use.
U8	**Data Length**	For data frames, this provides the number of bytes in **Data** (0–8). For remote frames, this indicates the number of bytes requested.
[U8]	**Data**	For data frames, this provides the data bytes. The array uses a fixed size of 8 bytes. For remote frames, the values contained in **Data** are not valid and should be ignored.

Data Type	Element	Description
		If the **Data Type** is defined as **Array of 6 U32**, the CAN frame is represented as an array of 6 U32 values, as shown in the following table.

Table 546. Array of 6 U32 Values

Most significant byte	**...**	**...**	**Least significant byte**
Timestamp (upper U32)			
Timestamp (lower U32)			
Identifier			
Type	**InfoA**	**InfoB**	**DataLength**
Data[0]	Data[1]	Data[2]	Data[3]
Data[4]	Data[5]	Data[6]	Data[7]

In the table above, the meaning of each element is the same as the **Cluster** data type.

CAN Output

Write a CAN frame to be transmitted.

Complete the following steps to write to a CAN port using CAN Output.

1. Use the **Add Targets and Devices** dialog box to add the CAN module to your CompactRIO configuration. Right-click the CAN module in the Project Explorer window and select Properties to specify the configuration properties for the module.
2. Create an FPGA I/O item for CAN by right-clicking the FPGA target under **My Computer** in the **Project Explorer** window, selecting **New»FPGA I/O**, and then adding **CAN»NI 9852»CAN0** (or **CAN»NI 9853»CAN0**) and/or **CAN»NI 9852»CAN1** (or **CAN»NI 9853»CAN1**) to the FPGA I/O list. Refer to New FPGA I/O dialog box (FPGA Module) for more information about creating and adding FPGA I/O to your CompactRIO configuration.
3. Place an **I/O Node** from the FPGA I/O palette onto your FPGA VI diagram.

4. Right-click the I/O Node and select **Select FPGA I/O»CAN»NI 9852»CAN0** (or **Select FPGA I/O»CAN»NI 9853»CAN0**) or **Select FPGA I/O»CAN»NI 9852»CAN1** (or **Select FPGA I/O»CAN»NI 9853»CAN1**).
5. Right-click the I/O Node and select **Change to Write** to use the I/O node for CAN Output.
6. Right-click the I/O Node and choose Properties to select the Data Type.

Start communication on the CAN port prior to using CAN Output. To start communication, enable Auto Start in the Module Configuration, or invoke the Start method.

The communication path from LabVIEW FPGA to the CAN port on the module is implemented as a FIFO. The CAN Output node waits for an available element in the FIFO, then writes the frame to the FIFO. If no other CAN frame exists in the FIFO, the CAN frame will begin to transmit immediately.

CAN Output does not wait for the requested transmit to complete on the network (acknowledgement). Call CAN Output followed by Wait on Transmit Complete (I/O Method) if you want to transmit CAN frames one at a time.

Write multiple frames to the output FIFO if you want to transmit CAN frames as fast as possible. The CAN module submits these frames to the SJA1000 controller as fast as possible, thus generating close to 100% busload.

When the output FIFO is full, the CAN Output node waits for an element to become available, then writes the frame to the FIFO. The **Output Timeout (ms)** of the CAN Advanced Port Configuration dialog box specifies how long to wait for a new element to become available, which occurs when a frame from a previous CAN Output transmits successfully onto the network. If you specify **Output Timeout (ms)** of 0, the CAN Output node returns an error status of TRUE (error) if a new element is not available (non-blocking). When an error is returned, attempt CAN Output of the same frame again at a later time.

In addition to the CAN Output node, the **Output Timeout (ms)** also applies to most methods and properties. For example, if **Output Timeout (ms)** is 0, the Abort Transmit method will return an error if the output FIFO is full. For more information on the output path from LabVIEW FPGA to the CAN port, refer to *Arbitration*.

Note You can add more than one CAN output port to the FPGA I/O node, and also add analog and digital outputs to the same FPGA I/O node by right-clicking the FPGA I/O node and selecting **Add Element**. Right-click the created I/O name and select **Select FPGA I/O** to specify the I/O name. If you use more than one output per node, LabVIEW will execute them sequentially starting with the top output. If one output is blocked while waiting for the output, subsequent outputs will be delayed.

Related concepts:

- Error Terminals
- Module Configuration
- Arbitration

Properties Dialog

Selects the **Data Type** for the CAN frame written to the CAN Input node. The available values are **Cluster** (default) and **Array of 6 U32**.

The benefits of using the **Cluster** data type are:

- Simpler block diagram (easy to use)
- Fast and efficient, if you limit use to the block diagram only (not front panel)

The benefits of the **Array of 6 U32** data type, and handshaking integers to the host VI one at a time, are:

- Faster for transfer to/from host VI
- Fewer FPGA gates

Refer to the *Using Clusters and Arrays in LabVIEW FPGA Application Note* for more information on the use of clusters and arrays.

Node Inputs

- **Error In**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.
- **CAN*x***—If the **Data Type** is defined as **Cluster** (default), the CAN frame is

represented as a cluster. Refer to the following table for the cluster elements.

Table 547. Canx Cluster Elements

Data Type	Element	Description
	**Timestamp High**	Ignored. The transmit request always occurs as soon as possible (not timed).
	**Timestamp Low**	Ignored. The transmit request always occurs as soon as possible (not timed).
	**Identifier**	Arbitration ID. If bit 29 (20000000 hexadecimal) is clear, this ID uses the standard format (11-bit). If bit 29 is set, this ID uses extended format (29-bit). This representation of the CAN identifier is the same as NI-CAN.
	**Type**	Type of frame. • **Data Frame**: 0 • **Remote Frame**: 1
	**InfoA**	Reserved for future use.
	**InfoB**	Reserved for future use.
	**Data Length**	For data frames, this provides the number of bytes in **Data** (0–8). For remote frames, this indicates the number of bytes requested.

Data Type	Element	Description
[U8]	**Data**	For data frames, this provides the data bytes. The array uses a fixed size of 8 bytes. If the **Data Type** is defined as **Array of 6 U32**, the CAN frame is represented as an array of 6 U32 values, as shown in the following table.

Table 548. Array of 6 U32 Values

Most significant byte	**...**	**...**	**Least significant byte**
Timestamp (upper U32)			
Timestamp (lower U32)			
Identifier			
Type	**InfoA**	**InfoB**	**DataLength**
Data[0]	Data[1]	Data[2]	Data[3]
Data[4]	Data[5]	Data[6]	Data[7]

In the table above, the meaning of each element is the same as the **Cluster** data type.

Node Outputs

- **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable. Unless you set the **Output Timeout (ms)** to -1 to wait indefinitely, you must use the error terminals to determine whether the CAN Output successfully submitted a CAN frame for transmit.

I/O Methods for CAN Modules

Use the Add Targets and Devices dialog box to add the CAN module to your CompactRIO configuration.

Complete the following steps to use an FPGA I/O Method Node for CAN.

1. Specify the configuration properties for the module by right-clicking the CAN module in the Project Explorer window and selecting `Properties`.
2. Create an FPGA I/O item for CAN by right-clicking the FPGA target under **My Computer** in the **Project Explorer** window, selecting **New»FPGA I/O**, and then adding **CAN»NI 9852»CAN0** (or **CAN»NI 9853»CAN0**) and/or **CAN»NI 9852»CAN1** (or **CAN»NI 9853»CAN1**) to the FPGA I/O list.
3. Place an I/O Method Node from the FPGA I/O palette onto your FPGA VI diagram.
4. Right-click the I/O Method Node, and select **Select Item»CAN»NI 9852»CAN0** (or **Select Item»CAN»NI 9853»CAN0**) or **Select Item»CAN»NI 9852»CAN1** (or **Select Item»CAN»NI 9853»CAN1**).
5. Right-click the I/O Method Node and choose **Select Method** to specify the desired method from the list of CAN methods.

For information on how the I/O Method Node arbitrates with other CAN nodes, refer to *Arbitration*.

Note There are no module-specific methods for NI 9852 or NI 9853. Adding **CAN»NI 9852** or **CAN»NI 9853** to the FPGA I/O node and choosing **Select Method** does not display any methods.

Related information:

- New FPGA I/O Dialog Box

Abort Transmit

Abort a pending CAN frame transmission.

Since the determinism of CAN depends on the prioritization of frames based on ID, the SJA1000 CAN controller allows you to abort a pending transmit to replace it with a higher priority ID.

When implementing protocols that require abort, ensure that only one transmit is pending. Call **Wait on Transmit Complete** before (or after) each call to **CAN Output**. If more than one transmit is pending at the time **Abort Transmit** is called, the **Transmit Success** Boolean could refer to any one of those pending frames.

Refer to the *I/O Methods* introduction for step by step instructions to place this method

in your LabVIEW diagram.

Node Inputs

- **Error In**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.
- **Timeout**—Specifies the time to wait for the pending transmission to succeed or abort. The resolution is in milliseconds. Since this method must wait for the pending transmission to indicate abort or success, use a timeout greater than 0. Using a **Timeout** of 10000 (ten seconds) is sufficient for most CAN baud rates (longer than a frame time). The **Timeout** value of *–1* (infinite wait) is not supported. Use the largest positive I32 value, which provides a maximum of approximately 25 days.

Node Outputs

- **Timed Out?**—Indicates a timeout. When **Timed Out?** is *TRUE*, the error cluster indicates success (not a timeout error). **Timed Out?** is TRUE only if you use a **Timeout** that is less than a frame-time for your specified CAN baud rate.
- **Transmit Success?**—Indicates whether the pending transmit was successfully aborted, or transmitted on the network before it could be aborted. This value is valid only if **Timed Out?** is *FALSE*. Returns *FALSE* if the pending transmit aborted successfully. Returns *TRUE* if the pending frame transmitted successfully on the network prior to the **Abort Transmit**.
- **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.

Reset

Reset the CAN port to the same state as when the FPGA VI started running.

The RESET method does the following:

- Resets the SJA1000.
- Restores all properties to their default values.

If you enable **Auto Start** in the tool, this method also re-starts communication.

When a CAN controller enters the *Bus Off* state (**Comm State** property), it no longer

communicates. Like most CAN controllers, the SJA1000 does not allow recovery from Bus Off to Error Active using the CAN controller register set. This behavior is normal within an embedded system (vehicle), but many test setups intentionally generate Bus Off, and therefore require a mechanism to recover (i.e. to run the test again). By performing a complete reset of the SJA1000, this method recovers from the Bus Off state.

This method also clears the output FIFO as well as any pending CAN node (including all Wait methods). The reset affects only the specified CAN port, and has no effect on the other CAN port of the module. For step by step instructions to place this method in your LabVIEW FPGA VI block diagram, refer to the *I/O Methods* introduction.

Node Inputs

- **Error In**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.

Node Outputs

- **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.

Related concepts:

- Comm State

Start

Start communication.

Cases for using this method, as opposed to enabling **Auto Start** in Module Configuration:

- Change configuration properties within the diagram (such as the **Listen Only** property)
- Auto-baud algorithms (must Stop/Start to set the baud rate for each attempt)
- Start CAN communication on a trigger (analog or digital input)

Node Inputs

- **Error In**—Optional. Not shown by default. Right-click the node and select

Error Terminals to enable.

Node Outputs

-  **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.

Related concepts:

- Error Terminals
- Module Configuration

Stop

Stop communication.

This method also clears the output FIFO as well as any pending CAN node (including all Wait methods).

Node Inputs

-  **Error In**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.

Node Outputs

-  **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.

Related concepts:

- Error Terminals

Wait on Comm State Change

Wait for a change in the communication state of the CAN port (`Comm State` property). This enables you to handle CAN communication state changes in a separate loop from your main CAN Input/Output loop.

If the **Comm State** property changes before you call Wait (relative to the previous Wait), the current Wait returns immediately. The **Comm State** is not considered to be changed when you first Start, and it does not change after Stop.

This method returns an error if invoked while communication is stopped.

Node Inputs

- **Error In**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.
- **Timeout**—Specifies the time to wait for the communication state to change. The resolution is in milliseconds. The special **Timeout** of 0 is used to poll for the changed state. If **Comm State** has changed from the previous call to Wait, the **Timed Out?** Boolean is *FALSE*. If **Comm State** has not changed, the **Timed Out?** Boolean is *TRUE*. The **Timeout** value of *–1* (infinite wait) is not supported, but you can use the largest positive I32 value, which provides a maximum of approximately 25 days.

Node Outputs

- **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.
- **Timed Out?**—Indicates a timeout. When Timed Out? is *TRUE*, the error cluster indicates success (not a timeout error).
- **Comm State**—Returns the new value of the `Comm State` property.

Wait on Transceiver Wakeup

Wait for the `Transceiver Mode` property to change from *Sleep* to *Normal* mode due to a remote wakeup (bus activity) or local wakeup. A local wakeup occurs if you use the I/O Property node to set `Transceiver Mode` property back to *Normal*.

If a wakeup occurs before you call Wait (relative to the previous Wait), the current Wait returns immediately. The wakeup is not considered to have occurred when you first Start, and it does not occur after Stop.

This method returns an error if invoked while communication is stopped.

Node Inputs

- **Error In**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.
- **Timeout**—Specifies the time to wait for the communication state to change. The resolution is in milliseconds. The special **Timeout** of 0 is used to poll for the

changed state. If **Comm State** has changed from the previous call to Wait, the **Timed Out?** Boolean is *FALSE*. If **Comm State** has not changed, the **Timed Out?** Boolean is *TRUE*. The **Timeout** value of *–1* (infinite wait) is not supported, but you can use the largest positive I32 value, which provides a maximum of approximately 25 days.

Node Outputs

- **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.
- **Timed Out?**—Indicates a timeout. When Timed Out? is *TRUE*, the error cluster indicates success (not a timeout error).

Related concepts:

- Transceiver Mode
- Error Terminals

Wait on Transmit Complete

Wait for all frames written to CAN Output to complete transmission. You typically use this to determine when all frames have been acknowledged.

If all transmits are complete before you call Wait (relative to the previous Wait), the current wait returns immediately. All transmits are considered complete after you first Start. This method returns an error if invoked while communication is stopped.

Node Inputs

- **Error In**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.
- **Timeout**—Specifies the time to wait for the communication state to change. The resolution is in milliseconds. The special **Timeout** of 0 is used to poll for the changed state. If **Comm State** has changed from the previous call to Wait, the **Timed Out?** Boolean is *FALSE*. If **Comm State** has not changed, the **Timed Out?** Boolean is *TRUE*. The **Timeout** value of *–1* (infinite wait) is not supported, but you can use the largest positive I32 value, which provides a maximum of approximately 25 days.

Node Outputs

- **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.
- **Timed Out?**—Indicates a timeout. When Timed Out? is *TRUE*, the error cluster indicates success (not a timeout error).

Related concepts:

- Transceiver Mode
- Error Terminals

I/O Properties for CAN Modules

Complete the following steps to use an FPGA I/O Property node for CAN:

1. Use the Add Targets and Devices dialog box to add the CAN module to your CompactRIO configuration. Right-click the CAN module in the Project Explorer window and select **Properties** to specify the configuration properties for the module.
2. Create an FPGA I/O item for CAN by right-clicking the FPGA target under **My Computer** in the **Project Explorer** window, selecting **New»FPGA I/O**, and then adding **CAN»NI 9852»CAN0** (or **CAN»NI 9853»CAN0**) and/or **CAN»NI 9852»CAN1** (or **CAN»NI 9853»CAN1**) to the FPGA I/O list.
3. Place an I/O Property Node from the FPGA I/O palette onto your FPGA VI diagram.
4. Right-click the I/O Property Node and select **Select Item»CAN»NI 9852»CAN0** (or **Select Item»CAN»NI 9853»CAN0**) or **Select Item»CAN»NI 9852»CAN1** (or **Select Item»CAN»NI 9853»CAN1**).
5. Right-click the I/O Property Node and choose **Select Property** to specify the appropriate property from the list of available CAN properties.

For information on how the I/O Property Node arbitrates with other CAN nodes, refer to Arbitration.

The description of each property includes the following table.

Table 549. I/O Property Table Structure

Data type	Permissions	Write while stopped?	Write while running?
<value>	<value>	<value>	<value>

- **Data type**—Shows the icon for the LabVIEW data type used for this property.
- **Permissions**—Read, Write, or Read/Write. If the property is Read/Write, you can right-click the property in the node and select **Change To Read** or **Change To Write** to select the direction. Otherwise, only one direction is supported.
- **Write while stopped?**—Yes or No to indicate whether you can write the property while CAN communication is stopped. You can always read a property with Read permissions.
- **Write while running?**—Yes or No to indicate whether you can write the property while CAN communication is running. You can always read a property with Read permissions.

Related concepts:

- LS/FT Termination
- Module ID
- Arbitration

Related information:

- New FPGA I/O Dialog Box

Bit Timing

Table 550. Bit Timing Property

Data type	Permissions	Write while stopped?	Write while running?
	Write	Yes	No

Each CAN baud rate corresponds to multiple values for two Bit Timing Registers (BTR). The bit fields in these registers are consistent across almost all CAN controllers. This property specifies the baud rate as values for the Bit Timing Registers. The least significant byte is BTR0, and the most significant byte is BTR1.

The default value for this property comes from Module Configuration. The **Baud Rate**

property is converted to commonly used BTR values, or you can specify BTR values directly in the CAN Advanced Port Configuration dialog.

For information on CAN Bit Timing Registers, refer to the Philips SJA1000 standalone CAN controller datasheet. This document is available for download on `www.nxp.com`.

Related concepts:

- Module Configuration
- Advanced CAN Port Properties

Comm State

Table 551. Comm State Property

Data type	Permissions	Read while stopped?	Read while running?	Write while stopped?	Write while running?
	Read	Yes	Yes	N/A	N/A

This property describes the current communication state of the CAN controller.

The values are:

- 0: Error Active
- 1: Error Passive
- 2: Bus Off

Use the Wait on Comm State Change method to detect changes in this property.

When you initially start communication, the CAN port will begin in the Error Active state as specified in the CAN standard (except when **Listen Only** is enabled). After bus errors cause a transition to Error Passive or Bus Off, that state will be retained even after you stop communication and restart. Use the Reset method to reset back to the *Error Active* state.

Listen Only

Table 552. Listen Only Property

Data type	Permissions	Read while stopped?	Read while running?	Write while stopped?	Write while running?
TF	Write	No	No	Yes	No

This property controls **Listen Only** mode for passive monitoring/logging.

- *FALSE* disables **Listen Only** mode. Received frames are acknowledged, and frames can be transmitted using **CAN Output**.
- *TRUE* enables **Listen Only** mode. The CAN port can only receive frames. The port does not transmit on the network. No acknowledgment is transmitted when a frame is received. When listen-only is enabled, the *Comm State* property begins in the *Error Passive* state.

Related concepts:

- I/O Properties for CAN Modules
- Module Configuration
- Comm State

Log Bus Errors

Table 553. Log Bus Errors Property

Data type	Permissions	Write while stopped?	Write while running?
TF	Write	Yes	Yes

This property enables the logging of bus errors as frames that can be read using the CAN Input node.

The bus error frame is logged when the Philips SJA1000 CAN controller detects a bus error and the Bus Error Interrupt occurs.

- *FALSE* indicates that the bus errors will not be logged and cannot be read using the CAN Input node (default).

- *TRUE* indicates that the bus errors will be logged as frames and can be read using the CAN Input node.

The bus error frame has the following format:

Table 554. Buss Error Frame Format

Arbitration ID	0
Data Length	4
Type	6
Data	**Bytes** • 0—Comm State (see below) • 1—Transmit Error Counter • 2—Receive Error Counter • 3—ECC Register • 4—X • 5—X • 6—X • 7—X

Note *X* means *Reserved* or *Don't Care*.

The first data byte (Comm State) indicates the current communication state of the CAN controller. The states are:

- 0—Error Active
- 1—Error Passive
- 2—Bus Off

Log Transceiver Faults

Table 555. Log Transceiver Faults Property

Data type	*Permissions*	*Write while stopped?*	*Write while running?*
TF	Write	Yes	Yes

This property enables the logging of transceiver faults as frames that can be read using the *CAN Input* node.

- *FALSE* indicates that the transceiver faults will not be logged as frames (default).
- *TRUE* indicates that logging is enabled and the transceiver faults will be logged as frames which can be monitored using the CAN Input node.

Note The TJA1041 transceivers on the NI 9853 high-speed module only detect faults during transmission.

The transceiver fault frame has the following format:

Table 556. Transceiver Fault Frame Format

Arbitration ID	0
Data Length	1
Type	7
Data	**Bytes** • 0—Transceiver fault. 0=fault cleared, 1=fault present • 1—X • 2—X • 3—X • 5—X • 6—X • 7—X

Note *X* means *Reserved* or *Don't Care*.

LS/FT Termination

Note This property applies only to the ports on the NI 9852 low-speed/fault-tolerant module.

Table 557. LS/FT Termination Property

Data type	Permissions	Write while stopped?	Write while running?
U32	Write	Yes	No

This property enables the setting of the low-speed/fault-tolerant transceiver termination on the NI-9852 module to either 1 kiloohm (1.11 kΩ) or 5 kiloohm (4.99 kΩ).

This termination affects both R_{RTH} and R_{RTL}. For more information on determining the required termination for your LS/FT node, refer to the ISO 11898-3 specification (CAN fault-tolerant transmission). You can also refer to the Connectors and Cables section of the *NI-CAN Hardware and Software Manual*.

- **1 kiloohm** indicates that the termination for the LS/FT CAN port will be 1 kΩ (default).
- **5 kiloohm** indicates that the termination for the LS/FT CAN port will be 5 kΩ.

Related concepts:

- I/O Properties for CAN Modules

Related information:

- New NI-CAN Hardware and Software Manual

Receive Error Counter

Table 558. Receive Error Counter Property

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
U8	Read	Yes	Yes	N/A	N/A

This property provides access to the CAN controller Receive Error Counter defined by the CAN standard.

Self Reception

Table 559. Self Reception Property

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
TF	Write	No	No	Yes	Yes

This property specifies whether to echo successfully transmitted CAN frames to be read using CAN Input.

Each reception occurs just as if the frame were received from another CAN device. For self reception to operate properly, another CAN node must receive and acknowledge each transmit. If a transmitted frame is not successfully acknowledged, it is not echoed for input.

- *FALSE* indicates transmits are not echoed (default).
- *TRUE* indicates transmits are echoed.

Single Shot Transmit

Table 560. Single Shot Transmit Property

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
TF	Write	No	No	Yes	Yes

This property specifies whether to retry failed CAN frame transmissions.

- *FALSE* indicates standard CAN behavior. Failed CAN frame transmissions are automatically retried (default).
- *TRUE* indicates single-shot. With single-shot enabled, if a CAN frame is not transmitted successfully, the CAN controller does not retry.

SJA1000 Filter Code

Table 561. SJA1000 Filter Code Property

Data type	*Permissions*	*Write while stopped?*	*Write while running?*
U32	Write	Yes	No

This property controls the SJA1000 Acceptance Code registers (ACR0 to ACR3). The least significant byte of the property is stored in register offset 3 (ACR3), the next least significant in offset 2 (ACR2), and so on.

- The default value comes from the CAN Advanced Port Configuration.
- The mask for the SJA1000 filter is specified using the *SJA1000 Filter Mask* property.
- The filter mode is specified using the *SJA1000 Filter Mode* property.

For information on the SJA1000 acceptance filter registers (PeliCAN mode), refer to the datasheet of the Philips SJA1000 standalone CAN controller. This document is available for download on `www.nxp.com`.

SJA1000 Filter Mask

Table 562. SJA1000 Filter Mask Property

Data type	*Permissions*	*Write while stopped?*	*Write while running?*
U32	Write	Yes	No

This property controls the SJA1000 Acceptance Mask registers (AMR0 to AMR3). The least significant byte of the property is stored in register offset 3 (AMR3), the next least significant in offset 2 (AMR2), and so on.

- The default value comes from the CAN Advanced Port Configuration.
- The code for the SJA1000 filter is specified using the *SJA1000 Filter Code* property.
- The filter mode is specified using the *SJA1000 Filter Mode* property.

For information on the SJA1000 acceptance filter registers (PeliCAN mode), refer to the datasheet of the Philips SJA1000 standalone CAN controller. This document is available for download on `www.nxp.com`.

SJA1000 Filter Mode

Table 563. SJA1000 Filter Mode Property

Data type	*Permissions*	*Write while stopped?*	*Write while running?*
	Write	Yes	No

This property controls the SJA1000 Acceptance Mode register. Dual (0) specifies use of two filters. Single (1) specifies use of a single filter.

- The default value comes from the CAN Advanced Port Configuration.
- The code and mask for the SJA1000 filter is specified using the *SJA1000 Filter Code* property.

For information on the SJA1000 acceptance filter registers (PeliCAN mode), refer to the datasheet of the Philips SJA1000 standalone CAN controller. This document is available for download on `www.nxp.com`.

Transceiver Mode

Table 564. Transceiver Mode Property

Data type	*Permissions*	*Write while stopped?*	*Write while running?*
	Read/Write	No	Yes

This property sets the mode for the CAN transceiver, as well as the associated mode in the SJA1000 CAN controller.

The values are:

- 0: Normal
- 1: Sleep

The Sleep mode places the SJA1000 and the CAN transceiver into a low-power state. This low-power state has no effect on LabVIEW FPGA hardware. The Sleep mode can be set only when the CAN controller is running (not when stopped).

A *local wakeup* occurs when you set **Transceiver Mode** from Sleep back to Normal.

A *remote wakeup* occurs when a remote node transmits a CAN frame (referred to as the wakeup frame). The wakeup frame wakes up the transceiver and CAN controller chip. The wakeup frame is not received or acknowledged by the SJA1000 CAN controller. When the wakeup frame ends, the CAN port enters Normal mode, and again receives and transmits CAN frames. If the node that transmitted the wakeup frame did not detect an acknowledgement (such as if other nodes were also waking), it will retry the transmission, and the retry will be received by the CAN port.

To determine when a remote wakeup occurs when the transceiver is in Sleep mode, invoke the Wait On Transceiver Wakeup method.

After communication starts, the SJA1000 must detect a *bus free condition* (11 bit times) before you can change the Transceiver Mode to Sleep. If you set the mode to Sleep before bus free, the SJA1000 automatically returns to Normal mode. After you start communication (run FPGA with **Auto Start** enabled or invoke Start method), wait at least 11 bit times before setting Transceiver Mode to Sleep.

Related concepts:

- Error Terminals
- I/O Properties for CAN Modules
- Wait on Transceiver Wakeup
- Start

Transmit Error Counter

Table 565. Transmit Error Counter Property

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
U8	Read	Yes	Yes	N/A	N/A

This property provides access to the CAN controller Transmit Error Counter defined by the CAN standard.

Module Properties

Module properties apply to the entire module.

To select a module property, right-click the I/O Property Node and select **FPGA I/O»NI 9852** or **FPGA I/O»NI 9853**.

Related concepts:

- Module ID
- Serial Number
- Vendor ID

Module ID

Table 566. Module ID Property

Data type	Permissions	Write while stopped?	Write while running?
	Read	N/A	N/A

This property returns the module identifier of the module.

- The module identifier for the NI 9852 CAN module is hexadecimal 71F3.
- The module identifier for the NI 9853 CAN module is hexadecimal 714F.

Related concepts:

- I/O Properties for CAN Modules

Serial Number

Table 567. Serial Number Property

Data type	Permissions	Write while stopped?	Write while running?
	Read	N/A	N/A

This property returns the serial number of the CAN module.

Vendor ID

Table 568. Vendor ID Property

Data type	Permissions	Write while stopped?	Write while running?
U16	Read	N/A	N/A

This property returns the vendor identifier of the module. The vendor identifier for National Instrumentsi s hexadecimal 1093.

Related concepts:

- I/O Properties for CAN Modules

Arbitration

Each CAN port uses dedicated resources in LabVIEW FPGA, as well as distinct communication paths between LabVIEW FPGA and the module. When you access two different CAN ports in your FPGA VI, the nodes in the LabVIEW FPGA VI block diagram execute independently.

For example, if you output a CAN frame to **CAN0** and simultaneously output a CAN frame to **CAN1**, the two CAN Output nodes execute independently. As another example, access to **CAN0** on a module in slot 1 occurs independently from access to **CAN0** on a module in slot 2.

When two or more nodes in the LabVIEW FPGA VI block diagram access the same CAN port at the same time, arbitration can occur to determine which node executes first. This arbitration may cause delays or jitter in your application. For example, if you want to transmit a specific CAN frame when a digital input trigger is detected in LabVIEW FPGA, arbitration between the CAN Output node and other CAN nodes can cause jitter from the digital input pulse to the start of the CAN frame.

Node Arbitration

For the same CAN port, CAN nodes of the same type must arbitrate. There are four types of CAN nodes:

- CAN Input
- CAN Output
- I/O Method Node
- I/O Property Node

CAN nodes of different type do not arbitrate at the node level.

For most applications, CAN node arbitration does not occur, and when it does the jitter introduced is small. For example, you would not normally use two CAN Input nodes simultaneously on the same CAN port, because neither node would return the correct sequence of CAN frames.

The most common situation in which CAN node arbitration occurs is use of a Wait I/O Method simultaneously with another I/O Method. For example, if you invoke Wait on Transmit Complete with a **Timeout** of 10 seconds, then invoke Abort Transmit soon after, node-level arbitration causes the Abort Transmit to execute after the Wait on Transmit Complete (possibly 10 seconds after invocation). When using multiple I/O Methods, you can avoid this behavior by polling for the event using **Timeout** of 0.

Output Path Arbitration

For each CAN port, two communication paths exist between LabVIEW FPGA and the CAN module. The output path transfers information from LabVIEW FPGA to the CAN port, and the input path transfers information from the CAN port to LabVIEW FPGA.

When two or more nodes in the LabVIEW FPGA VI block diagram access the same CAN port, arbitration can occur for the underlying output path. This arbitration is different than the usual node arbitration in LabVIEW FPGA, which is limited to multiple instances of a node. For example, if you write a frame to **CAN0**, and read a property on **CAN0** at the same time, arbitration occurs for the output path, because the CAN Output Node and the I/O Property Node both send information from LabVIEW FPGA to the CAN port.

The following nodes use the output path to the CAN port:

- CAN Output
- I/O Method Node: Abort Transmit
- I/O Method Node: Reset

- I/O Method Node: Start
- I/O Method Node: Stop
- I/O Property Node: Read any property
- I/O Property Node: Write any property

When two or more of these nodes execute simultaneously for the same CAN port, arbitration occurs to force sequential execution, and some jitter may result. When arbitration occurs in the same FPGA clock cycle, CAN Output is prioritized first, then the I/O Method Node, then the I/O Property Node. When the nodes are the same, CAN node arbitration occurs.

The output path is implemented as a FIFO. This allows multiple frames to be sent to the CAN module for transmit, thus enabling tests that generate full bus load.

All nodes in the preceding list use the output FIFO with the exception of the Stop and Reset method. Since these methods stop CAN communication, they must execute regardless of pending transmit frames. Once the Stop or Reset method arbitrates successfully for the output path, it bypasses the FIFO and transfers the method to the CAN port for immediate execution.

When the output FIFO is full, a node in the preceding list (except Stop and Reset) must wait for an element to become available. This generally occurs when a frame from a previous CAN Output transmits successfully onto the network. The **Output Timeout (ms)** in the *CAN Advanced Port Configuration* dialog box specifies how long to wait for a new element to become available. Therefore, the **Output Timeout (ms)** applies to any CAN node in the preceding list, not just the CAN Output node.

If you specify **Output Timeout (ms)** of 0, the node simply checks to see if a new element is available (non-blocking). If an element exists, the node executes and error status of FALSE is returned (success). If no element exists, the node fails to execute, and error status of TRUE is returned (error). Therefore, in order to poll (such as writing frames without waiting), you must enable *Error Terminals* for the node. When an error is returned, you must attempt to execute the node again at a later time.

Input Path Arbitration

For the input path from the CAN port to LabVIEW FPGA, arbitration does not occur within LabVIEW FPGA. As soon as the CAN module has data for LabVIEW FPGA nodes,

the data transfers immediately.

When the CAN port attempts to transfer information for two or more nodes back to LabVIEW FPGA, arbitration for the input path occurs within the CAN module. For example, if a CAN frame is received at the same time that a property is read, the CAN module must decide which node's data to transfer first. Although this input path arbitration can cause jitter, that jitter is minimal compared to the effects you can see from CAN node arbitration or CAN output path arbitration.

Module Configuration

The configuration information in Module Configuration is transferred from LabVIEW FPGA to the CAN module when the FPGA VI runs. Although this transfer occurs automatically as soon as the VI begins to run, all CAN nodes must wait for this one-time configuration to complete. The time required for configuration can be up to 1 second.

If your FPGA VI does not execute a CAN node before the configuration time elapses, you will not notice any delay. If a CAN node is executed before the configuration time elapses, the CAN node will wait for configuration to complete, then execute normally. If you execute CAN nodes in a manner that does not depend on timing, this behavior is not a concern. However, if you require the first CAN Input or CAN Output in your FPGA VI to execute without delay, you can do this by adding one of the following to the beginning of your VI (before the main loop):

- 1 second delay
- CAN Property or Method node (for instance, a **Start** method node)

Summary

When an FPGA VI is limited to CAN Input and CAN Output for each CAN port, the jitter between node execution and CAN frame transfer is very small. This is due to the fact that each CAN port uses a distinct communication path for input data and output data.

When you mix methods or properties along with input/output, you may see increased jitter between node invocation and CAN frame transfer. This jitter is due to the arbitration. The jitter is typically only a few microseconds, so it will generally be well within the limits for your application.

Related concepts:

- Advanced CAN Port Properties
- CAN Input
- Error Terminals

Error Terminals

To detect errors, right-click the CAN node in the diagram and select **Error Terminals**. The resulting **Error In** and **Error Out** terminals provide the standard LabVIEW error cluster for error handling.

NI recommends using **Error Terminals** for CAN development. For some CAN nodes, the error terminals are essential. For example, you need to evaluate **Error Out** from a CAN Input node to determine if a CAN frame was received within the specified **Input Timeout (ms)**. The only cost of using error terminals is slight FPGA gate usage, not performance.

When an error occurs, use the LabVIEW **Explain Error Help** dialog box for a description of the error code. Right-click an error cluster and select **Explain Error** from the shortcut menu for more information about the error. You also can select **Explain Error** and enter the returned error code into the dialog box.

The error descriptions in the Explain Error dialog box are generic to all CompactRIO (C Series) modules. The following sections provide information specific to the CAN module.

Module Communication Error

Error Code: 65536

This error occurs when there is a catastrophic error with communication to the CAN module. Catastrophic errors can result when the following events occur:

- Unplug the CAN module.
- Invoke a CAN node while the CompactRIO Sleep line is asserted. This error does not occur for the Sleep feature of the CAN transceiver (**Transceiver Mode** property).
- Invoke a CAN node before the CAN module has reconfigured itself after the

CompactRIO Sleep line deasserts (approximately 1 second).
- An invalid data transfer between LabVIEW FPGA and the CAN module.

Incorrect Module Error

Error Code: 65537

This error occurs if the user plugs in another CompactRIO module into a slot configured for a CAN module.

Timeout Error

Error Code: 65538

This error occurs when:

- CAN Input does not receive a valid CAN frame within the specified **Input Timeout (ms)**.
- CAN Output does not detect an available element in the output FIFO within the specified **Output Timeout (ms)**.
- An I/O Method or I/O Property node does not detect an available element in the output FIFO within the specified **Output Timeout (ms)**. Refer to *Arbitration* for more information.

This error is not returned when a Wait or Abort Transmit method times out. Those methods indicate a timeout in the **Timed Out?** terminal.

Data Overflow Error

Error Code 65539

This error occurs from the CAN Input node. This indicates that you did not invoke CAN Input fast enough, and one or more CAN frames were lost. Since LabVIEW FPGA executes at a much faster rate than CAN frames are received, this can only happen when you introduce artificial delays in your FPGA VI. Invoke CAN Input at least once every 50 microseconds to avoid this error.

Not Run Mode Error

Error Code: 65540

This error occurs when communication is stopped and you use a CAN feature that requires communication to be started first. Refer to the preceding documentation to determine which features require running communication.

Start communication on the CAN port before using the feature (CAN Output, I/O Method, or I/O Property node). Start communication with one of the following methods:

- enable **Auto Start** in the Module Configuration.
- Invoke the Start method.

Run Mode Error

Error Code 65541

This error occurs when communication is started and you use a CAN feature that requires communication to be stopped. Refer to the preceding documentation to determine which feature cannot be used while communication is running.

Stop communication on the CAN port before using the feature (usually writing a property). To set initial properties, disable **Auto Start** in the *Module Configuration*, then set the desired properties, invoke the *Start* method, and proceed with your remaining application. To set properties in your main application loop, invoke the *Stop* method, set the relevant properties, then invoke the *Start* method.

Feature Not Supported Error

Error Code 65546

This error is returned by the CAN Property node when a property selected for reading or writing is not supported on the specified module.

Too Many Error Frames Warning

Error Code 65547

This warning occurs when a serious fault is detected on the node, resulting in a burst of error frames. This warning indicates that one or more error frames may not be reported to LabVIEW FPGA. All valid CAN data frames are reported.

Verify that cabling, baud rate, and termination of all nodes on the network are correct, and that proper bus power is supplied.

Related concepts:

- Advanced CAN Port Properties
- Transceiver Mode
- Arbitration
- CAN Output
- Module Configuration
- Start
- Stop

Converting Frames To and From Channels (Signals)

The CAN Input and CAN Output nodes in LabVIEW FPGA access CAN data as frames. Frames are the raw unit of transfer on the CAN bus.

For some applications, it is useful to access CAN data as *channels*, also known as *signals*.

- A CAN channel represents a field in the data of a frame with specific CAN identifier.

The field in the data is scaled to a floating-point value in physical units, such as Volts or Revolutions-per-minute. The specification for how CAN channels are converted to/from CAN frames is usually provided in a CAN database file, such as `Vector CANdb` files.

NI-CAN is the software for National Instruments PCI, PXI, and PCMCIA cards for CAN. The NI-CAN software provides features to enable conversion of CAN frames to/from

channels. This frame to channel conversion can be used with the CompactRIO CAN module, even if you do not use an NI CAN PCI, PXI, or PCMCIA card.

The NI-CAN software is supported on Windows as well as LabVIEW RT controllers. upported LabVIEW RT controllers include:

- Any PXI controller
- cRIO-9002 CompactRIO Real-Time Controller
- cRIO-9004 CompactRIO Real-Time Controller

Install NI-CAN to a LabVIEW RT controller using the software installation tool under **Remote Systems** in MAX.

Some examples of CompactRIO applications that use NI-CAN frame to channel conversion:

- **Logging**: The rugged enclosure and real-time capabilities of CompactRIO make it an ideal product for data logging in the field, such as drive testing of an automobile. For logging applications, the LabVIEW application on CompactRIO is simple: read CAN frames and store them in a file. When the CAN log file is later transferred from CompactRIO to a lab computer (usually running Windows), the application on the lab computer can read frames from the log file, and use NI-CAN to display the frames as CAN channel waveforms. In addition, if the LabVIEW application on CompactRIO stores a second log file with analog/digital samples, that data can be displayed on the lab computer as waveforms synchronized with the CAN channels.
- **Triggering/Filtering**: In a logging or other test application, you may need to evaluate CAN data to trigger some other task. For example, you may want to log a 5 second window of CAN/analog data when the "rpm" channel in CAN ID 5 first exceeds 4000 RPM. Although you could evaluate the raw data in CAN ID 5 to trigger the log, in some cases it is easier to simply evaluate the channel named "rpm" as defined in a CAN database. By installing NI-CAN on a CompactRIO Real-Time Controller such as the cRIO-9004, you can transfer CAN frames to NI-CAN, then read the "rpm" channel to detect when the value is greater than 4000.00.
- **Prototyping/Simulation**: For applications in which you must execute a control model within the CompactRIO Real-Time Controller, you typically wire CAN channels as inputs and outputs to the control model. In order to implement this, you can install NI-CAN on the CompactRIO Real-Time Controller. Your LabVIEW

FPGA VI reads and writes CAN frames, and transfers those CAN frames to/from LabVIEW RT as you would any other I/O. Your LabVIEW RT VI uses NI-CAN to convert the CAN frames to/from CAN channels, which are then wired into the control model.

Serial Interface Module Reference

Table 569. C Series Serial Interface Modules

Serial Module	Description
NI-9870	4-Port, RS232 C Series Serial Interface Module
NI-9871	4-Port, RS485/RS422 C Series Serial Interface Module

In this document, the NI-9870 and NI-9871 are collectively referred to as NI-987x.

Programming Overview

The LabVIEW FPGA application programming interface (API) is inherently different from other NI-Serial (for instance, VISA) and C-Series module APIs. Instead of I/O nodes, Read Byte and Write Byte methods on the FPGA I/O Method Node are used for data transfer.

Property nodes can dynamically change the module configuration. Additionally, most module configuration can be done directly from the Project Explorer Window before compiling and running a VI by right-clicking the appropriate module and selecting **Properties**. If using a non-standard baud rate, this configuration panel may also be used to calculate the appropriate *Baud Rate Divider* and *Baud Rate Prescaler* to write using property nodes.

All four ports on the module share one interface to the FPGA target which has a bandwidth of approximately 128 KB/s. This shared interface can be an important consideration when transferring large amounts of data at high rates. Because of the shared interface to the module, the module is only accessed by one LabVIEW FPGA node at a time. If there is more than one node present on the FPGA VI block diagram that will execute simultaneously, the LabVIEW FPGA Module will automatically arbitrate access to the module with a fair, round-robin arbitration scheme for port-specific property and method nodes. General module property nodes such as Serial #, Module ID# and Vendor ID# have a lower priority in the arbitration scheme.

In order to understand when this consideration of a shared interface should come into play, it is important to distinguish between port baud rate (baud) and data throughput

rate (bytes per second). The baud rate is the rate of electrical transitions (including overhead such as start, stop, and parity bits), whereas the data throughput rate is how fast actual data is passed to and from the application, or in this case, the FPGA.

The maximum data throughput rate for a given baud rate can be calculated by dividing the baud rate by the total number of bits in the serial message. For example, a common configuration might be 1 start bit, 8 data bits, and 1 stop bit, which adds to 10 total bits. A 9600 baud setup with this configuration could then support up to 960 (9600 divided by 10) data messages (bytes) per second. A serial port with a baud rate of 921,600 baud using this same configuration would average 92.2 KB/s in data throughput, and so on.

If the sum of the baud rates of all four ports is greater than 500,000 baud, this shared interface may begin to limit performance. Consider a configuration with all four ports operating at 460,800 baud. This setup has a maximum data throughput of 46.08 KB/s per port. The application can simultaneously read and write data at 46.08 KB/s to one port or 23.04 KB/s to two ports without any latency. However, attempting to read and write two ports at 46.08 KB/s could introduce some latency, depending on transfer sizes.

For reference, if all four ports have baud rates up to 115,200 baud, the module will function correctly for any data throughput rate. Most serial applications do not use the full bandwidth of a given baud rate transferring a large continuous block of data. Furthermore, NI 987x C-series modules feature individual 64-byte UART FIFO buffers for each port, which allow for small surges in transmit and receive data. Finally, if timeouts do occur at high baud rates, increasing the timeout value or decreasing the number of other module accesses (such as property nodes) may help reduce the timeouts.

C Series Serial Interface Modules

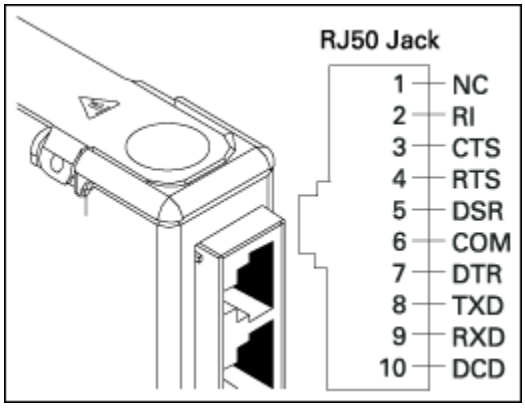
Table 570. C Series Serial Interface Modules

Serial Module	Description
NI-9870	4-Port, RS232 C Series Serial Interface Module
NI-9871	4-Port, RS485/RS422 C Series Serial Interface Module

In this document, the NI-9870 and NI-9871 are collectively referred to as NI-987x.

NI-987x Port Assignments

Figure 120. NI-9870/9871 Port Assignments



NI 987x (FPGA Interface)

FPGA I/O Node

The NI-987x devices do not support the FPGA I/O Node.

Terminals in Software

You can select the following terminals for the NI-987x devices.

Table 571. Terminals in Software

Terminal	Description
Port *x*	Port *x* of the module, where *x* is the number of the port. • The NI 9870 has ports 1 to 4. • The NI 9871 has ports 1 to 4.

Arbitration

The NI-987x devices do not support arbitration options. All port-specific property and method nodes have equal priority and will execute using fair, round-robin arbitration. Module-specific property nodes (Serial #, Module ID#, Vendor ID#) are given a lower priority in the arbitration scheme.

Although arbitration options are not supported, arbitration is used only when nodes are placed without dependence on each other. Using error clusters, sequence structures, and other forms of dependence is recommended when explicit program flow control is needed. Refer to `labview\examples\CompactRIO\Module Specific\NI 987x\Serial Loopback\NI-987x Serial Loopback.lvproj` for an example of how to control program flow in this way.

Use the FPGA I/O Method Node to configure the following methods with NI-987x devices. These devices do not support any module methods.

Table 572. I/O Methods

Method	Description
Clear Rx FIFO	Clears the 64-byte UART Receive FIFO (Rx) for the desired port. **Note** There are no wires into or out of this method in LabVIEW. This action is executed when the method is placed on the block diagram and the VI is run. To execute this method in a sequence, place the method within a Flat Sequence structure, or wire error handlers into and out of the FPGA I/O Method Node itself.
Read Byte	Reads a single byte from the 64-byte UART Receive (Rx) FIFO for the desired port. The method returns any *data errors*, such as parity or overrun. • **Data**: Returns the 8-bit unsigned value. • **Timeout (ms)**: Specifies the amount of time in milliseconds to keep retrying the read if no data is available. A value of `0` indicates a single try, and –1 indicates an infinite number of tries. **Caution** Setting –1 as a timeout value can make the program appear to hang while the node waits indefinitely to complete and pass on program control. • **Timed Out**: Returns TRUE if the method times out before receiving data. Refer to `labview\examples\CompactRIO\Module Specific\NI 987x\Serial Loopback\NI-987x Serial Loopback.lvproj` for an example that illustrates using DMA and interrupts in conjunction with LabVIEW FPGA loops and this FPGA I/O Method Node to read blocks of data.
Write	Writes a single byte to the 64-byte UART Transmit (Tx) FIFO for the desired port. If flow

Method	Description
Byte	control allows, the byte is transmitted immediately upon reaching the top of the FIFO. • **Data**: Specifies the 8-bit unsigned value to write. • **Timeout (ms)**: Specifies the amount of time in milliseconds to keep retrying the write if no data is available. A value of `0` indicates a single try, and –1 indicates an infinite number of tries. **Caution** Setting –1 as a timeout value while using flow control can make the program appear to hang while the node waits indefinitely to complete and pass on program control. • **Timed Out**: Returns TRUE if the method times out before data can be written to the Tx FIFO (the FIFO is full). Refer to `labview\examples\CompactRIO\Module Specific\NI 987x\Serial Loopback\NI-987x Serial Loopback.lvproj` for an example that illustrates using DMA and interrupts in conjunction with LabVIEW FPGA loops and this FPGA I/O Method Node to read blocks of data.

I/O Properties

Use the FPGA I/O Property Node to access the following properties with NI-987x devices.

Property	Description
Baud Rate (write only)	Sets the target baud rate for the port. If you set the baud rate, the appropriate Baud Rate Prescaler and Baud Rate Divider are set automatically, and are calculated according to this formula: *BaudRate = 3.6864 Mbps / (Prescaler * Divider)* • The maximum Baud Rate for the NI 9870 is 921.6 Kb/s. • The maximum Baud Rate for the NI 9871 is 3.6864 Mb/s.
Baud Rate Divider (read/write)	Specifies the divider for the baud rate as defined by the baud rate equation above. This property is useful for setting a custom baud rate programmatically. Use the C Series Module Properties dialog box as a calculator to find the appropriate divider for the desired custom baud rate. This value is an unsigned 16-bit value within the range 4 to 65,535.

Property	Description
	Note Any values outside of the valid range will be coerced into the range.
Baud Rate Prescaler (read/write)	Specifies the prescaler for the baud rate as defined by the baud rate equation above. This property is useful for setting a custom baud rate programmatically. Use the C Series Module Properties dialog box as a calculator to find the appropriate prescaler for the desired custom baud rate. This value is defined as a *type def*, either **1** or **4**. A value of 4 is normally used only for very low baud rates.
Break State (write only)	Sets the break state of the port to Unasserted (logic high) or Asserted (logic low). If you set the break state to **Asserted**, the property suspends byte transmission and places the transmission line in a break state until you set the break state to **Unasserted**.
Bytes at Port (read only)	Returns the number of bytes currently available in the UART Receive FIFO for the desired port. The number of bytes will be within the range 0–64.
CTS State (read only)	Returns the current state of the Clear To Send (CTS) input signal, either Unasserted (logic high) or Asserted (logic low).
Data Bits (read/write)	Specifies the number of data bits contained in each frame. Valid values are **5**, **6**, **7**, and **8**. If you select 5 data bits, you can select only 1 or 1.5 stop bits. If you select 6, 7, or 8 data bits, you can select only 1 or 2 stop bits.
DCD State (read only)	Returns the current state of the Data Carrier Detect (DCD) input signal, either Unasserted (logic high) or Asserted (logic low). The DCD signal is also known as the *Receive Line Signal Detect (RLSD)*. Modems often use the DCD signal to indicate the detection of a carrier, or modem, on the phone line.
DSR State (read only)	Returns the current state of the Data Set Ready (DSR) input signal, either Unasserted (logic high) or Asserted (logic low).
DTR State (write only)	Sets the Data Terminal Ready (DTR) output signal to Unasserted (logic high) or Asserted (logic low).
Flow Control (read/write)	Specifies the flow control method used for transmitting and receiving data. Valid values are **None**, **Software XON/XOFF**, **Hardware RTS/CTS**. DTR/DSR flow control is not available, but can be implemented using the DTR State and DSR State properties. Note that if Hardware (RTS/CTS) flow control is used, the RTS State write property cannot control the RTS state. The default is **None**.
Is Connected	Returns *TRUE* if the port is properly connected to a valid RS232 signal on another port or device.

Property	Description
(read only)	
Parity (read/ write)	Specifies the parity used with every frame that is transmitted or received. Valid values are **None** (0), **Odd** (1), **Even** (2), **Mark** (3), and **Space** (4). The default parity is **None**.
RI State (read only)	Returns the current state of the Ring Indicator (RI) input signal, either Unasserted (logic high) or Asserted (logic low). Modems often use the RI signal to indicate that the phone line is ringing.
RTS State (write only)	Sets the Request To Send (RTS) output signal to Unasserted (logic high) or Asserted (logic low). This property cannot be changed when you set the Flow Control property to **Hardware (RTS/CTS)** or the Transceiver Mode property to **2 Wire Auto**.
Stop Bits (read/ write)	Specifies the number of stop bits used to indicate the end of a frame. Valid values are **1**, **1.5**, and **2**. You can select **1.5** only if you set Data Bits to **5**, and you can select **2** only if you set Data Bits to **6**, **7**, or **8**. **Note** Because a setting of **1.5** stop bits is invalid with any Data Bits setting other than **5**, care must be taken to successfully transition to or from 1.5 stop bits. For example, if you have 8 data bits and 2 stop bits, and want to transition to 5 data bits and 1.5 stop bits, you will not be able to write either of the desired properties directly without getting an invalid property combinations error. Instead, move through a sequence of valid combinations to get to the desired settings. For this example, you would first set Stop Bits to **1**, then set Data Bits to **5**. At that point, Stop Bits may be reset to **1.5**.
XOFF Character (read/ write)	Specifies the value of the XOFF character used for Software (XON/XOFF) flow control. The character is 1 byte wide, and may be passed as either a hexadecimal (0–FF) or decimal value (0–255). **Note** This property will only work if Flow Control is defined as **Software XON/XOFF**.
XON Character (read/ write)	Specifies the value of the XON character used for Software (XON/XOFF) flow control. The character is 1 byte wide, and may be passed as either a hexadecimal (0–FF) or decimal value (0–255). **Note** This property will only work if Flow Control is defined as **Software XON/XOFF**.

Module Properties

Use the FPGA I/O Property Node to access the following properties with the NI-987x devices. These properties have lower priority than port-specific methods or properties. In arbitration, the port-specific methods and properties will be executed before these properties.

Table 573. Module Properties

Property	Description
Module ID	Returns the module ID. • The module ID for the NI 9870 is 0x7305. • The module ID for the NI 9871 is 0x7306.
Serial Number	Returns the unique serial number of the module.
Vendor ID	Returns the National Instruments vendor ID, 0x1093.

Single-Cycle Timed Loop

The NI-987x devices do not support the single-cycle Timed Loop.

Related concepts:

- Serial Communication

NI 987x (Scan Interface)

To use the NI 987x devices n Scan Interface mode, you must have NI-Serial RT with Scan Engine Support installed on the RT target. Use MAX to install NI-RIO on the RT target. If you use a Recommended Software Set, select **NI Scan Engine**, which includes NI-Serial 9870 and 9871 Scan Engine Support, on the Add-on Selection dialog box of the installer. If you do a custom install, select **NI-Serial 9870 and 9871 Scan Engine Support** as a sub-item of **NI-Serial RT** on the Features dialog box of the installer.

To put the NI 987x in Scan Interface mode, add it to your project directly under the chassis item. The serial ports of the NI 987x appear as selectable VISA I/O items that you can use in RT VIs. Use VISA resource name constants and controls to change

properties, read from, and write to the ports as if they were built into the RT target. The serial ports of the NI 9871 also appear in MAX alongside the built-in serial port(s) of the target under **Devices and Interfaces**.

Related concepts:

- Serial Communication

Connector Pinouts

The following connectors are used with C Series serial modules.

The cables (NI part number 182845-01) act as adapters between the two connector types. For more information about serial interface devices, visit `ni.com/serial`.

DB-9 Connector Pinout

Figure 121. DB-9 Male Connector Pinout

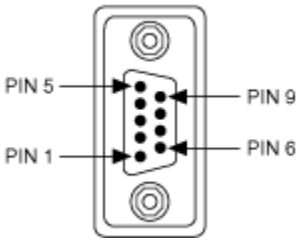


Table 574. Signal Descriptions

Pin	232 Signal DTE	485 Signal
1	DCD	GND
2	RXD	CTS+ (HSI+)
3	TXD	RTS+ (HSO+)
4	DTR	RXD+
5	GND	RXD–
6	DSR	CTS– (HSI–)
7	RTS	RTS– (HSO–)
8	CTS	TXD+
9	RI	TXD–

External Power Connector Pinout

Figure 122. External Power Connector Pinout

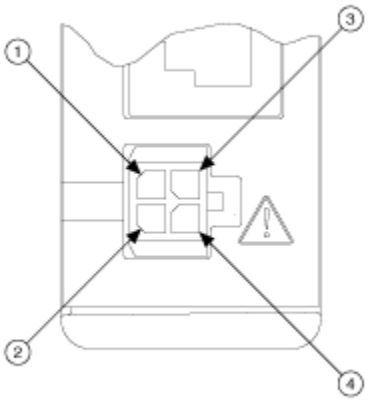


Table 575. Signal Descriptions

Pin	Signal
1	Power (8–28 V_{SUP})
2	COM
3	Power (8–28 V_{SUP})
4	COM

Note V_{SUP} pins and COM pins are hardwired together internally.

Modular Jack Connector (RJ50) Pinout

Figure 123. 10-Position Modular Jack Connector (RJ50) Pinout

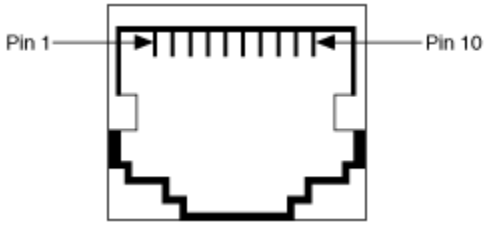


Table 576. Signal Descriptions

Pin	232 Signal	485 Signal
10	DCD	GND

Pin	232 Signal	485 Signal
9	RXD	CTS+ (HSI+)
8	TXD	RTS+ (HSO+)
7	DTR	RXD+
6	GND	RXD–
5	DSR	CTS– (HSI–)
4	RTS	RTS– (HSO–)
3	CTS	TXD+
2	RI	TXD–
1	No Connect	No Connect

Serial Communication

Communication Standards

The following table lists the features of the RS232, RS422, and RS485 standards.

Table 577. RS232, RS422, and RS485 Features

Feature	RS232	RS422	RS485
Type of transmission lines	Single-ended	Differential	Differential
Maximum number of drivers	1	1	32
Maximum number of receivers	1	10	32
Maximum cable length	50 ft	4,000 ft	4,000 ft
Maximum data rate	20 kB/s	10 MB/s	10 MB/s
Maximum CMV	± 25 V	±7 V	+12 to –7 V
Driver output	5 to 25 V	2 to 6 V	1.5 to 6 V
Driver load	<3 kΩ	100 Ω	60 Ω

RS232

As specified in the ANSI/EIA-232-D Standard, Interface Between Data Terminal Equipment and Data Circuit-Terminating Equipment Employing Serial Binary Data Interchange, RS232 standardizes serial communication between computers and between computer terminals and modems.

Many applications use the RS232 standard to interface peripherals to personal computers. RS232 uses transmission lines in which the state of each signal is represented by referencing the voltage level of a single line to ground.

RS232 was designed for serial communication up to distances of 50 ft and with data rates up to 20 kbytes/s. However, because of improvements in line drivers and cabling, it is possible to increase the actual performance of the bus past the limitations on speed and distance recommended in the specification.

RS422

As specified in the EIA/RS422-A Standard, Electrical Characteristics of Balanced Voltage Digital Interface Circuits, RS422 defines a serial interface much like RS232. However, RS422 uses balanced (or differential) transmission lines.

Balanced transmission lines use two transmission lines for each signal. The state of each signal is represented, not by a voltage level on one line referenced to ground as in RS232, but rather by the relative voltage of the two lines to each other.

For example, the TX signal is carried on two wires - wire A and wire B., where:

- A logical 1 is represented by the voltage on line A being greater than the voltage on line B.
- A logical 0 is represented by the voltage on line A being less than the voltage on line B.

Differential voltage transmission creates a signal that is more immune to noise as well as voltage loss due to transmission line effects. Use RS422 for longer distances and greater transmission speeds than RS232:

- Distances up to 4,000 ft

- Transmission speeds up to 10 MB/s

RS485

As specified in the EIA-485 Standard, Standard for Electrical Characteristics of Generators and Receivers for Use in Balanced Digital Multipoint Systems, RS485 expands on the RS422 standard by increasing the number of devices you can use from 10 to 32 and by working with half-duplex bus architectures.

Unlike the RS422 standard, RS485 addresses the issue of using multiple transmitters on the same line. RS485 defines the electrical characteristics necessary to ensure adequate signal voltages under maximum load, as well as short-circuit protection.

RS485 can also withstand multiple drivers driving conflicting signals at the same time.

RS485 Transceiver Control Modes

RS485 hardware supports four transceiver control modes:

- Four-wire mode.
- Two-wire mode: DTR controlled with echo. [17]
- Two-wire mode: DTR controlled.
- Two-wire mode: auto control.

Use hardware transceiver control to enable and disable transmitters and receivers so that they function on different bus topologies.

The following table lists the status of the transmitters and receivers under each transceiver control mode.

Table 578. RS485 Transceiver Control Modes

Mode	Transmitter State	Receiver State
Four-wire mode (default)	Always enabled	Always enabled
Two-wire mode: DTR controlled with echo	Enabled when DTR asserted	Always enabled
Two-wire mode: DTR controlled	Enabled when DTR asserted	Enabled when DTR unasserted

17. Data terminal ready (DTR) is a control signal indicating a device is ready to communicate.

Mode	Transmitter State	Receiver State
Two-wire mode: Auto Control	Enabled when transmitting data	Enabled when not transmitting data

Four-Wire Mode

Use the four-wire mode for most full-duplex systems. In the four-wire mode, the transmitter and receiver are always enabled. Four-wire mode is the default mode.

Two-Wire Mode: DTR Controlled with Echo

Use two-wire mode in half-duplex systems where you need to control the transmitter programmatically. In the DTR-with-echo mode, the transmitter is tri-stated when the DTR signal of the UART (Universal Asynchronous Receiver/Transmitter) is unasserted.

To transmit, the application first must enable the transmitter by asserting DTR. After the data is fully transmitted, your application unasserts DTR to disable the transmitter. Because the receiver is always enabled in this mode, you receive packets not only from other devices, but also your transmitter. Thus, your receiver echoes all data you transmit.

Two-Wire Mode: DTR Controlled

This mode is similar to the two-wire, DTR-with-echo mode.

Although this mode uses the same method as the DTR-with-echo mode to control the transmitter, the hardware automatically disables the receiver whenever the transmitter is enabled. You do not receive the packets sent from your transmitter.

Two-Wire Mode: Auto Control

In the two-wire auto control mode, serial hardware transparently enables the transmitter and receiver in a two-wire system.

Use this mode to remove the burden of transceiver control from your application. The hardware automatically enables the transmitter for each byte to be transmitted. The hardware also disables the receiver whenever the transmitter is enabled. You do not

receive the packets sent from your transmitter.

Handshaking lines (RTS/CTS) are disabled in the two-wire auto control mode mode.

Note When communicating with a two-wire device, National Instruments recommends using the two-wire Auto Control mode. This mode handles the transmitter/receiver enabling for a two-wire connection in your hardware. Two-wire Auto Control mode reduces the software overhead required to perform this operation in your application program. For more information about serial communication in two or four-wire modes, refer to *Serial Communication Issues*.

Bias Resistors

An RS485 transmission line enters an indeterminate state if no nodes are transmitting on it. This indeterminate state can cause the receivers to receive invalid data bits from noise picked up on the cable.

To prevent a line from receiving invalid data bits, force the transmission line into a known state. To force a known state, install two bias resistors at one node on the transmission line. This action creates a voltage divider that forces the voltage between the differential pair to be greater than 200 mV.

Note 200 mV is the threshold voltage for the receiver.

Duplex Architectures

Duplex refers to the means of bandwidth usage in a serial system.

The two common means of bi-directional serial communication are full duplex and half duplex.

Half-duplex communication involves a transmitter and a receiver connected to each end of the same wire or pair of wires. Because the same transmission line both sends and receives data, devices cannot send data in both directions at the same time. First, one device transmits over the wire(s) to the receiver of the second device. When the

first device finishes transmitting, both devices switch the connections from their transmitter to their receiver, or vice versa. The device that was receiving data can then transmit over the line.

In full-duplex communication, the devices use a separate wire (or pair of wires) for simultaneous transmission in each direction. The devices do not switch between transmitting and receiving.

In a differential serial bus, such as RS422 or RS485, a half-duplex system transmits and receives over the same twisted pair of wires. Half-duplex communication is often referred to as *two-wire* communication. Likewise, full-duplex communication is often referred to as *four-wire* communication because the full-duplex system uses a separate pair of wires for communication in each direction.

Full Duplex

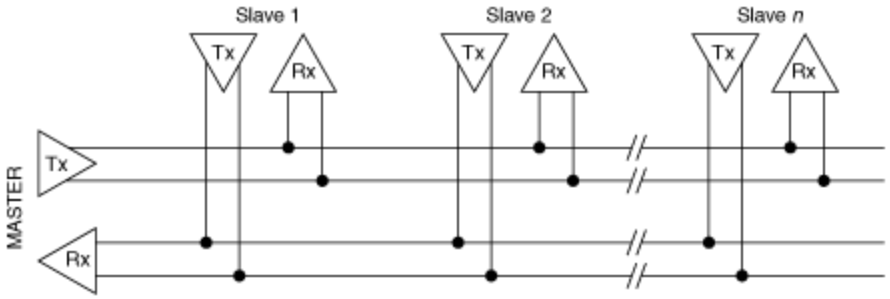
A typical full-duplex multidrop bus architecture involves a master-slave protocol. Only one device, the master, can control access to the bus. All other devices are slaves. Slave devices must wait for the master to give them access to the bus.

In a typical full-duplex system, one transmission line connects the bus master transmitter to all of the slave receivers. A second transmission line connects all of the slave transmitters to the bus master receiver.

Because each transmission line has two separate wires, a full-duplex system is often referred to as a four-wire system. The master uses four-wire mode, and the slaves use two-wire mode.

The following figure shows a typical full-duplex system.

Figure 124. Typical Full-Duplex System



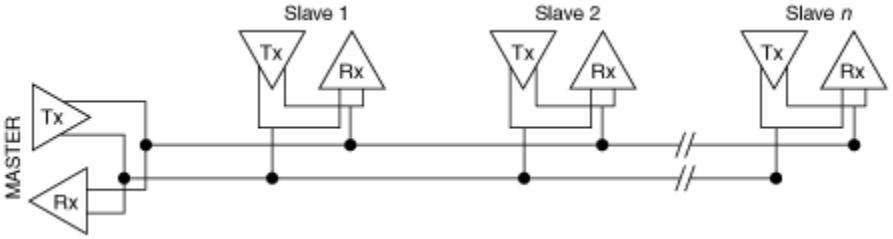
Half Duplex

A typical half-duplex multidrop bus architecture also involves a master-slave protocol. In a half-duplex system, all transmitters and receivers are connected to the same transmission line.

A half-duplex system is often referred to as a two-wire system.

The following figure shows a typical half-duplex system.

Figure 125. Typical Half-Duplex System



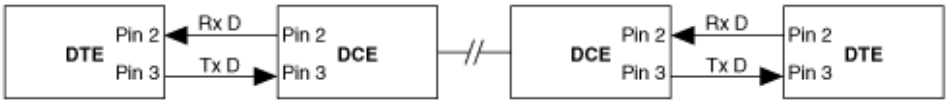
RS232 DTE versus DCE

In the RS232 specification, DTE (Data Terminal Equipment) and DCE (Data Communications Equipment) refer to the types of equipment on either end of a serial connection.

In general, DTE and DCE refer to computer equipment and modems, respectively. Because the RS232 specification mainly involves connecting a DTE directly to a DCE and vice versa, the pinouts are defined so that cabling is simple. A cable connected a computer to a modem by wiring pin 1 to pin 1, pin 2 to pin 2, and so on. This method is known as *straight-through* cabling.

The following figure shows straight-through cabling in a DTE-to-DCE interface.

Figure 126. Straight-Through Cabling in a DTE-to-DCE Interface



Straight-through cabling is still the standard method to connect a modem to your PC. However, because many applications use serial communication to connect two or

more DTEs without modems, the cabling becomes more complicated.

If two DTEs are wired together using a straight-through cable, one transmitter is connected to the other transmitter, and one receiver is connected to the other receiver. In this setup, no transmissions can occur. Cable the transmitter on one device to the receiver on the other device. This method is known as *null-modem* cabling, because it replaces the two modems that traditional RS232 applications would require between the two DTEs.

To communicate from one DTE serial port to another, use a null-modem cable. The following figure shows null-modem cabling in a DTE-to-DTE interface.

Figure 127. Null-Modem Cabling in a DTE-to-DTE Interface



RS485 Termination

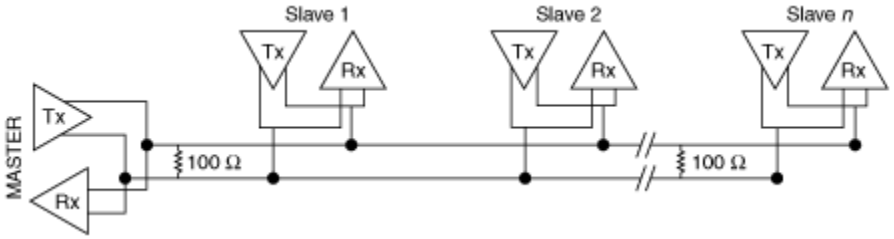
Each differential pair of wires is a transmission line. You must properly terminate the line to prevent reflections.

A common method of terminating a two-wire multidrop RS485 network is to install terminating resistors at each end of the multidrop network. If you daisy-chained multiple instruments together, you need a terminating resistor at only the first and last instruments. The terminating resistor should match the characteristic impedance of the transmission line (typically 100 Ω to 120 Ω).

You can order an optional DB-9 RS485 termination connector (NI part number 182844-01) that contains embedded terminating resistors for easy termination from `ni.com/serial`.

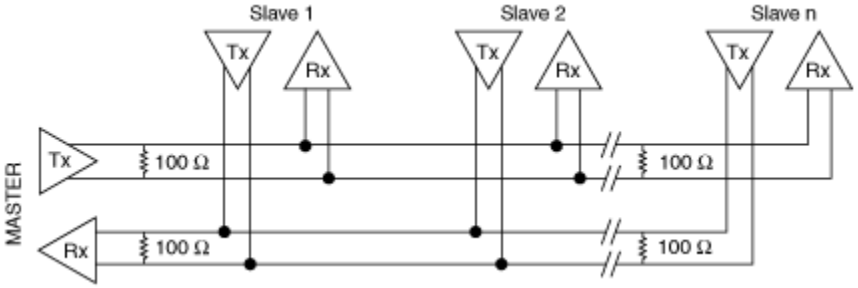
The following figure shows a multidrop network using terminating resistors.

Figure 128. Multidrop Network Using Terminating Resistors



The following figure shows a full-duplex network using terminating resistors.

Figure 129. Full-Duplex Network Using Terminating Resistors



Does the serial hardware have built-in support for protocols such as SDLC and ModBus?

Because the serial hardware uses UARTs for communication, protocols such as SDLC and ModBus are not supported in hardware. However, these protocols (or their derivatives) can be supported by software applications that handle the protocol translation.

How do I use 9 data bit framing?

While the C-Series serial modules directly support only 5, 6, 7, and 8 data bits, it may be possible to use the parity function to create a ninth data bit, depending on your application. To create and control a ninth data bit, enable parity and set parity mark or parity space for a ninth bit of either a 1 or 0, respectively. However, you must do this every time the ninth bit changes state, and thus it may not work for high-throughput applications.

What is the relationship between bits per second, throughput, and bus utilization?

The term *bits per second* (bps) refers to the bit rate of data transfer, with 1/bps being the length of time required to transmit one bit.

Throughput is the actual data transfer measured by the *amount of data transmitted ÷ time required to transmit*.

Bus utilization is the percentage of time that the serial bus is actively transmitting data. Because a wide range of factors such as flow control, processor latency, and device architecture can affect serial performance, bus utilization often is not 100 percent.

There is overhead associated with serial communication in the form of start bits, stop bits, and parity bits. This overhead reduces the throughput by 20–30 percent.The actual data throughput can be considerably less than the bits per second setting.

Motion Module Reference

Table 579. C Series Motion Modules

Motion Module	Description
NI-9501	Bipolar Chopper Stepper C Series Drive Motion Module
NI-9502	Brushless Servo C Series Motor Drive Module
NI-9503	PWM Stepper C Series Motor Drive Module
NI-9505	PWM Stepper C Series Motor Drive Module
NI-9512	Stepper, 1-Axis, Single Encoder C Series Motor Drive Interface Module
NI-9514	Servo, 1-Axis, Single Encoder C Series Motor Drive Interface Module
NI-9516	Servo, 1-Axis, Dual Encoder C Series Motor Drive Interface Module

Use the NI 9501/9502/9503/9505 motor drive modules in FPGA Interface Mode.

Use the NI-9512/9514/9516 drive interface modules in Scan Interface Mode.

C Series Drive Motion Modules

Table 580. C Series Drive Motion Modules

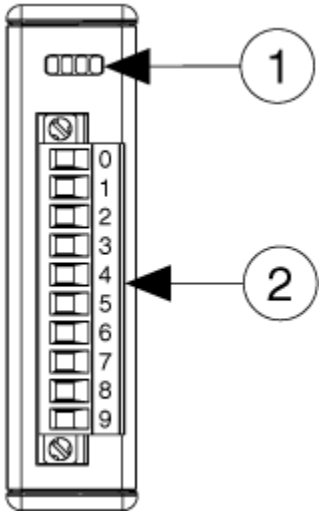
Motion Module	Description
NI-9501	Bipolar Chopper Stepper C Series Drive Motion Module
NI-9502	Brushless Servo C Series Motor Drive Module
NI-9503	PWM Stepper C Series Motor Drive Module
NI-9505	PWM Stepper C Series Motor Drive Module

Use the NI 9501/9502/9503/9505 motor drive modules in FPGA Interface Mode.

NI 950x LEDs and Connectors

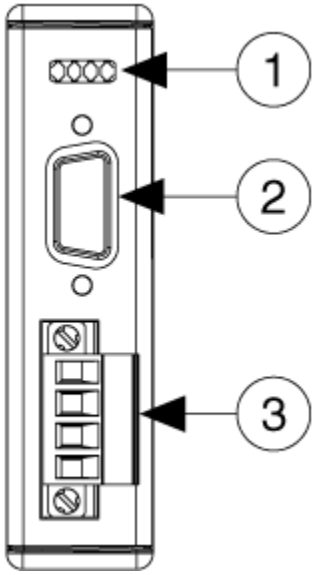
In this document, NI 950x refers to the NI 9501/9502/9503 C Series drive motion modules.

Figure 130. NI 9501/9502/9503 LEDs and Connectors



1. LEDs
2. 10-position Screw Terminal Connector

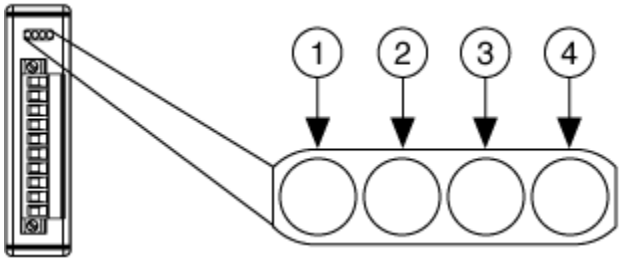
Figure 131. NI 9505 LEDs and Connectors



1. LEDs
2. 9-pin DSUB Connector

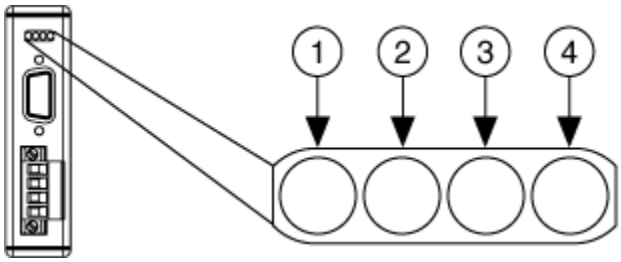
3. 4-position Screw Terminal Connector

Figure 132. NI 9501/9502/9503 LED Status Indicators



1. Enabled (Green)
2. Vsup (Motor Power) (Green)
3. User (Yellow)
4. Fault (Red)

Figure 133. NI 9505 LEDs



1. Power (Green)
2. V_{SUP} (Motor Power) (Green)
3. Disable (Yellow)
4. Fault (Red)

Figure 134. NI 9501/9502/9503 10-position Screw Terminal Assignments

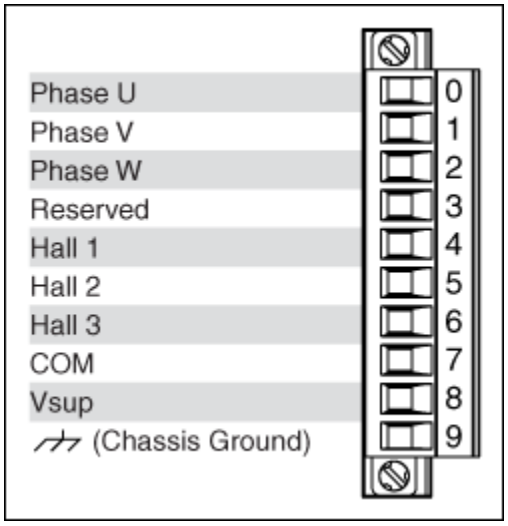


Figure 135. NI 9505 9-Pin DSUB Connector Pinout

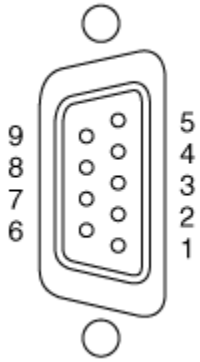


Table 581. NI 9505 9-Pin DSUB Connector Pin Assignments

Pin	Signal
1	Encoder Phase A+
2	Encoder Phase B+
3	Encoder Index+ (Phase Z+)
4	Emergency Stop (E-Stop)
5	+5 V (output)
6	Encoder Phase A–
7	Encoder Phase A–
8	Encoder Index– (Phase Z–)
9	Common (COM)–

Figure 136. NI 9505 4-position Screw Terminal

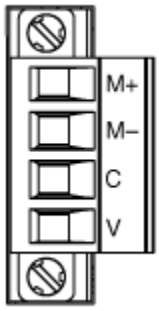


Table 582. NI 9505 4-position Screw Terminal Assignments

Terminal	Signal
M+	MOTOR+
M–	MOTOR–
C	COM (Motor DC power supply reference)
V	V_{SUP} (Motor DC power supply)

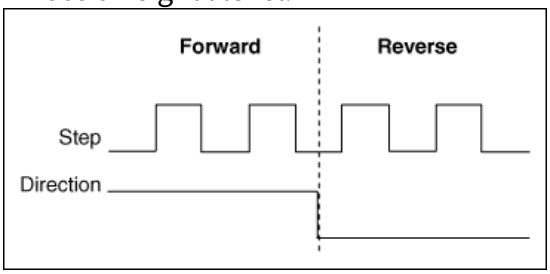
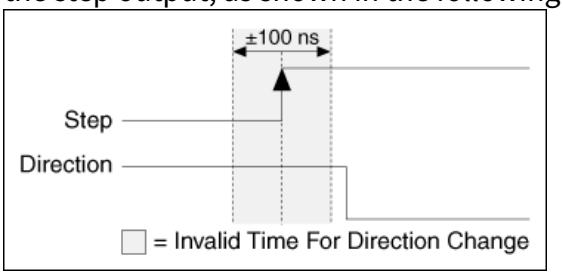
NI 9501 (FPGA Interface)

You can use an FPGA I/O Node, configured for reading or writing, with this device. All NI 9501 FPGA I/O operations execute in parallel when the I/O Node is configured with multiple I/O items.

You can select the following terminals for this device:

Table 583. Terminals in Software

Terminal	Type	Description
Step	Write	Sends step information to the motor as a series of digital pulses. Each positive edge on the Step output is considered a step. This output is available in a single-cycle Timed Loop. Steps cannot be generated at a rate faster than 5 MHz. The minimum pulse width of the **Step** output is 100 ns. **Note** You cannot use this output when the drive is disabled. Executing this node when the drive is disabled does not generate a step signal and returns error `–65631` (`NIMCMM_driveNotEnabled`).

Terminal	Type	Description
		For example programs, refer to `Velocity Control (open loop) - NI 9501` and `Setpoint Control (open loop) - NI 9501` in the `labview\examples\CompactRIO\Module Specific\NI 9501` directory.
Direction	Write	Specifies the direction of the motor. Set **Direction** to *TRUE* for clockwise motion. Set **Direction** to *FALSE* for counterclockwise motion. This output is available in a single-cycle Timed Loop. The minimum pulse width of the **Direction** output is 100 ns. The following figure shows the direction of motion when changing the **Direction** signal level:  To ensure that step pulses adjacent to a direction edge are applied in the correct direction, set the direction output at least 100 ns before or after a positive edge of the step output, as shown in the following figure:  **Note** You cannot use this output when the drive is disabled. Executing this node when the drive is disabled does not change the direction of motion and returns error `–65631` (`NIMCMM_driveNotEnabled`). Refer to the `Velocity Control (open loop) - NI 9501` and `Position Control (open loop) - NI 9501` examples in the `labview\examples\CompactRIO\Module Specific\NI 9501` directory for examples.
User LED	Write	Controls the User LED on the NI 9501 module. Set **User LED** to *TRUE* to turn the LED on. Set **User LED** to *FALSE* to turn the LED off. This terminal takes 5 µs to execute. Subsequent executions of this terminal or another property, method, or terminal in the same cycle wait until the previous request is complete before executing. For example, three subsequent calls to this terminal take 15 µs.

Arbitration

This device supports only the Arbitrate if Multiple Requestors Only option for arbitration. You cannot configure arbitration settings for this device.

Module Methods

Use the FPGA I/O Method Node to configure the following methods for this device. Refer to the `Velocity Control (open loop) - NI 9501` and `Position Control (open loop) - NI 9501` examples in the `labview\examples\CompactRIO\Module Specific\NI 9501` directory for examples.

Table 584. Module Methods

Method	Description
Clear Faults	Clears module faults. This method blocks until the command is sent to the module. Module faults do not clear if the fault condition is still present; always resolve the cause of the fault before executing this method. This method takes 5 µs to execute. Subsequent executions of this method or another property, method, or terminal in the same cycle wait until the previous request is complete before executing. For example, three subsequent calls to this method take 15 µs.
Disable Drive	Disables the drive. This method takes 5 µs to execute. Subsequent executions of this method or another property, method, or terminal in the same cycle wait until the previous request is complete before executing. For example, three subsequent calls to this method take 15 µs. The Disable Drive method takes priority over any other module method. If another method is called in the same cycle as the Disable Drive method, the Disable Drive method executes first.
Enable Drive	Enables the drive. This method takes 5 µs to execute. Subsequent executions of this method or another property, method, or terminal in the same cycle wait until the previous request is complete before executing. For example, three subsequent calls to this method take 15 µs. If the Enable Drive and Disable Drive methods are called in the same cycle, the Disable Drive method executes and the Enable Drive method is ignored. **Note** Executing this method when the drive is disabled due to a fault or other hardware issue returns error `–65632` (`NIMCMM_driveDisabled`). Use the **Get Status** method to determine the source of the fault, resolve the fault, clear the fault status using the **Clear Faults** method, then execute the **Enable Drive** method again.
Get	Obtains status information from the drive. This method takes 5 µs to execute. Subsequent

Method	Description
Status	executions of this method or another property, method, or terminal in the same cycle wait until the previous request is complete before executing. For example, three subsequent calls to this method take 15 µs. The following items are available: • **Drive Status**—Returns the drive status in an enum value. Valid values are the following: ◦ **Drive Disabled (Rated Phase Current set to Zero)**—The drive is disabled because the **Rated Phase Current** property is not initialized. You must set this property to the correct value before enabling the drive. ◦ **Drive Disabled (Disabled by fault)**—The drive is disabled because a drive fault has occurred. ◦ **Drive Disabled (Disabled by software)**—The drive is disabled by the Disable Drive method or because the **Enable Drive** method has not yet executed. ◦ **Drive Enabled**—The drive is enabled. • **Module Fault**—Returns *TRUE* if a module fault is present. Returns *FALSE* if no fault is present. • **Module Fault Code**—Returns the module *fault code*. Convert the fault code to a binary number to determine which fault bits are true. For example, a fault code of 4 is the binary number 100, indicating a fault in bit 2 of the fault bitmap (Module Short Circuit).

Module Properties

Use the FPGA I/O Property Node to access the following properties with this device. Refer to the `Velocity Control (open loop) - NI 9501` and `Position Control (open loop) - NI 9501` examples in the `labview\examples\CompactRIO\Module Specific\NI 9501` directory for examples using these properties.

Table 585. Module Properties

Property	Type	Description
Current Reduction Percentage	Read/ Write	Sets or returns the percentage by which to reduce the maximum current when the motor is stopped after the specified **Current Reduction Delay** has passed. The following options are available: • **0% (No Reduction)**—Disables the current reduction functionality. • **25% (default)**—Reduces the current by 25% of the maximum. • **50%**—Reduces the current by 50% of the maximum.

Property	Type	Description
Current Reduction Delay (ms)	Read/ Write	Sets or returns the time, in milliseconds, after the motor stops to wait before the current is reduced by the amount specified in the **Current Reduction Percentage** property. The default is 1,000 ms. Valid values are 0 ms to 2^{16} ms.
Microstepping Rate	Read/ Write	Sets or returns the current microstepping rate. The NI 9501 supports full, half, and microstep rates up to 256. Microstepping provides higher positional accuracy and smoother motion at the expense of reduced torque. **Caution** Updating this property while a move is in progress will increase or decrease the velocity of the axis. Update the **Microstepping Rate** when the drive is enabled. This allows you to perform a move at higher speeds using a low microstepping rate, then perform a move using a much higher microstepping rate to achieve high precision when reaching the final target position. This property takes 5 µs to execute. Subsequent executions of this property or another property, method, or terminal in the same cycle wait until the previous request is complete before executing. For example, three subsequent calls to this property take 15 µs. The following options are available: • **1 (Full Step)**—No microstepping. • **2 (Half Step)** • **4** • **8** • **16 (default)** • **32** • **64** • **128** • **256**
Module ID	Read	Returns the module ID, 0x751D.
Rated Phase Current	Read/ Write	Sets or returns the rated phase current of the stepper motor you are using in amps. The default value is 0 A. Valid values are greater than or equal to 0 A and less than or equal to the maximum value specified in the product specifications.

<table>
<tr><th>Property</th><th>Type</th><th>Description</th></tr>
<tr><td></td><td></td><td>Setting this property below 0 or above the maximum value returns error –65634 (NIMCMM_ratedPhaseCurrentOutOfRange). Setting this property to 0 returns error –65633 (NIMCMM_invalidRatedPhaseCurrent). Use the motor datasheet to determine this setting, based on the following table.
<table>
<tr><th>Connection Type</th><th>Rated Phase Current (Amps/Phase)</th></tr>
<tr><td>Bipolar 4-wire</td><td>same as datasheet Amps/Phase</td></tr>
<tr><td>6-wire series</td><td>datasheet Amps/Phase x 0.707</td></tr>
<tr><td>6-wire half-coil</td><td>same as datasheet Amps/Phase</td></tr>
<tr><td>8-wire series</td><td>datasheet Amps/Phase x 0.707</td></tr>
<tr><td>8-wire parallel</td><td>datasheet Amps/Phase x 1.414</td></tr>
<tr><td>8-wire half coil</td><td>same as datasheet Amps/Phase</td></tr>
</table>
This property takes 4.31 ms to complete execution. Subsequent executions of this property or another property, method, or terminal in the same cycle wait until the previous request is complete before executing. For example, three subsequent calls to this property take 12.93 ms.

Note Setting this property when the drive is enabled returns error –65630 (NIMCMM_onlyWhenDisabled). Disable the drive before updating this property.</td></tr>
<tr><td>Serial Number</td><td>Read</td><td>Returns the unique serial number of the module.</td></tr>
<tr><td>Vendor ID</td><td>Read</td><td>Returns the National Instruments vendor ID, 0x1093.</td></tr>
</table>

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop for the Step and Direction terminals of this device.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed

Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

NI 9501 Module Faults

NI 9501 modules can return the following faults as a decimal fault code. Convert the decimal fault code to a binary number to determine which fault bits are true. For example, a fault code of 2 is the binary number 10, indicating a fault in bit 1 of the fault bitmap (Module Overvoltage Fault).

Table 586.

Fault Bit	Fault Name	Description
0	Module Undervoltage Fault	The external power supply (V_{sup}) is not connected, does not meet the minimum power supply voltage specifications, or one of the module outputs has a short circuit. Verify that the output currents and power supply are within specifications and check the module connections.
1	Module Overvoltage Fault	The external power supply (V_{sup}) voltage exceeds the module power supply specifications. Verify that the power supply is within the module specifications and that the supply is properly connected to the module.
2–15	Reserved	—

NI 9502 (FPGA Interface)

The NI 9502 provides support for either Field Oriented Control commutation mode (default) or Trapezoidal commutation mode. The I/O nodes available for the module update according to the **Commutation Mode** selected in the C Series Module Properties dialog box.

Note Switching the **Commutation Mode** changes the available I/O nodes and properties for the NI 9502. If any I/O nodes from the unselected mode are present on the block diagram the nodes are broken and the VI does not compile.

FPGA I/O Node for Field Oriented Control Commutation Mode

You can use an FPGA I/O Node, configured for reading or writing, with this device.

Caution National Instruments strongly recommends sampling the NI 9502 current, Hall, and Vsup inputs in a single node to ensure consistent and jitter-free operation. Refer to *NI 9502 Timing* for more information about NI 9502 timing.

Note Refer to the NI SoftMotion Module Help, available by selecting **Help»LabVIEW Help** for examples demonstrating Field Oriented Control commutation mode.

You can select the following terminals for this device.

Table 587. Terminals in Software

Terminal	Type	Description
Phase U	Write	Controls the output of the PWM generator to the motor Phase U when the drive is enabled. A value of TRUE sets the output high, and a value of FALSE (default) sets the output low. The minimum high pulse width is 6 µs. The minimum low pulse width is200 ns.This terminal is available in a single-cycle Timed Loop. **Note** You cannot use this output when the drive is disabled. Executing this node when the drive is disabled does not set the output high and returns error –65631 (NIMCMM_driveNotEnabled).
Phase V	Write	Controls the output of the PWM generator to the motor Phase V when the drive is enabled. A value of TRUE sets the output high, and a value of FALSE (default) sets the output low. The minimum high pulse width is 6 µs. The minimum low pulse width is 200 ns. This terminal is available in a single-cycle Timed Loop. **Note** You cannot use this output when the drive is disabled. Executing this node when the drive is disabled does not set the output high and returns error –65631 (NIMCMM_driveNotEnabled).
Phase W	Write	Controls the output of the PWM generator to the motor Phase W when the drive is enabled. A value of TRUE sets the output high, and a value of FALSE (default) sets the output low. The minimum high pulse width is 6 µs. The minimum low pulse width is 200 ns. This terminal is available in a single-cycle Timed Loop.

Terminal	Type	Description
		Note You cannot use this output when the drive is disabled. Executing this node when the drive is disabled does not set the output high and returns error –65631 (NIMCMM_driveNotEnabled).
Phase U Current	Read	Samples the motor Phase U current and returns the current level in i16 format. To convert this LSB value to milliamps, multiply the LSB value by 4.028 mA/LSB. This terminal takes a minimum of 16 µs to return data. Refer to *NI 9502 Timing* for more information. This terminal is not available in a single-cycle Timed Loop. **Note** Phase V current can be calculated using the following formula: –(Phase U Current + Phase W Current).
Phase W Current	Read	Samples the motor Phase W current and returns the current level in i16 format. To convert this LSB value to milliamps, multiply the LSB value by 4.028 mA/LSB. This terminal takes a minimum of 16 µs to return data. Refer to *NI 9502 Timing* for more information. This terminal is not available in a single-cycle Timed Loop. **Note** Phase V current can be calculated using the following formula: –(Phase U Current + Phase W Current).
Vsup Voltage	Read	Samples the Vsup voltage and returns the voltage level in u16 format. To convert this LSB value to millivolts, multiply the LSB value by 8.8623 mV/LSB. This terminal takes a minimum of 16 µs to return data. Refer to *NI 9502 Timing* for more information. This terminal is not available in a single-cycle Timed Loop.
Hall 1	Read	Returns the state of Hall 1 as a digital input, where TRUE is high and FALSE is low. This terminal takes a minimum of 16 µs to return data. Refer to *NI 9502 Timing* for more information. This terminal is not available in a single-cycle Timed Loop.
Hall 2	Read	Returns the state of Hall 2 as a digital input, where TRUE is high and FALSE is low. This terminal takes a minimum of 16 µs to return data. Refer to *NI 9502 Timing* for more information. This terminal is not available in a single-cycle Timed Loop.
Hall 3	Read	Returns the state of Hall 3 as a digital input, where TRUE is high and FALSE is low. This terminal takes a minimum of 16 µs to return data. Refer to *NI 9502 Timing* for more information. This terminal is not available in a single-cycle Timed Loop.
User LED	Write	Controls the User LED on the NI 9502 module. Set the **User LED** terminal to TRUE to turn the LED on and FALSE to turn the LED off. This output is not available in a single-cycle Timed Loop.

FPGA I/O Node for Trapezoidal Commutation Mode

You can use an FPGA I/O Node, configured for reading or writing, with this device. Refer to *NI 9502 Timing* for more information about creating applications for the NI 9502.

Refer to the 9502 Trapezoidal Commutation example in the `<labview>\examples\CompactRIO\Module Specific\NI 9502` directory for an example using these terminals in Trapezoidal commutation mode.

You can select the following terminals for this device.

Table 588. Terminals in Software

Terminal	Type	Description
PWM	Write	Controls the output of the PWM generator to the active motor phase. The NI 9502 module outputs the PWM signal to the active phase based on the state of the Hall inputs. A value of TRUE sets the output high, and a value of FALSE (default) sets the output low. The minimum pulse width is 200 ns (high or low). This terminal is available in a single-cycle Timed Loop.  **Note** You cannot use this output when the drive is disabled. Executing this node when the drive is disabled does not generate a signal and returns error `-65631` (`NIMCMM_driveNotEnabled`).
Direction	Write	Controls the direction of the motor rotation. A value of TRUE indicates clockwise motor direction when facing the motor shaft. A value of FALSE (default) specifies counterclockwise motor direction when facing the motor shaft. The maximum update rate for this output is once per PWM cycle. This terminal is available in a single-cycle Timed Loop.  **Note** You cannot use this output when the drive is disabled. Executing this node when the drive is disabled does not generate a signal and returns error `-65631` (`NIMCMM_driveNotEnabled`).
Motor Current	Read	Monitors the current of the active phase and returns the current in i16 format. The NI 9502 determines which phase to return current for based on the Hall input state. To convert this LSB value to milliamps, multiply the LSB value by 4.028 mA/LSB. This terminal takes a minimum of 16 µs to return data. Refer to *NI 9502 Timing* for more information. This terminal is not available in a single-cycle Timed Loop.

Terminal	Type	Description
		Note The NI 9502 module measures the Phase U and Phase W current directly, but returns a calculated current value for Phase V using the following formula: –(Phase U Current + Phase W Current).
Vsup Voltage	Read	Monitors the Vsup voltage and returns the voltage level in u16 format. To convert this LSB value to millivolts, multiply the LSB value by 8.8623 mV/LSB. This terminal takes a minimum of 16 µs to return data. Refer to *NI 9502 Timing* for more information. This terminal is not available in a single-cycle Timed Loop.
Hall 1	Read	Returns the state of Hall 1 as a digital input, where TRUE is high and FALSE is low. This terminal takes a minimum of 16 µs to return data. Refer to *NI 9502 Timing* for more information. This terminal is not available in a single-cycle Timed Loop.
Hall 2	Read	Returns the state of Hall 2 as a digital input, where TRUE is high and FALSE is low. This terminal takes a minimum of 16 µs to return data. Refer to *NI 9502 Timing* for more information. This terminal is not available in a single-cycle Timed Loop.
Hall 3	Read	Returns the state of Hall 3 as a digital input, where TRUE is high and FALSE is low. This terminal takes a minimum of 16 µs to return data. Refer to *NI 9502 Timing* for more information. This terminal is not available in a single-cycle Timed Loop.
User LED	Write	Controls the User LED on the NI 9502 module. Set the **User LED** terminal to TRUE to turn the LED on and FALSE to turn the LED off. This output is not available in a single-cycle Timed Loop.

Arbitration

This device supports only the Arbitrate if Multiple Requestors Only option for arbitration. You cannot configure arbitration settings for this device.

Module Methods

Use the FPGA I/O Method Node to configure the following methods for this device.

Refer to the 9502 Trapezoidal Commutation example in the `<labview>\examples\CompactRIO\Module Specific\NI 9502` directory for an example using these methods in Trapezoidal commutation mode. Open trapezoidal commutation example

Note Refer to the NI SoftMotion Module Help, available by selecting **Help»LabVIEW Help** for examples demonstrating Field Oriented Control commutation mode.

Table 589. Module Methods

Method	Description
Clear Faults	Clears module *faults*. This method blocks until the command is sent to the module. Module faults do not clear if the fault condition is still present; always resolve the cause of the fault before executing this method.
Disable Drive	Disables the drive. The Disable Drive method takes priority over any other module method. If another method is called in the same cycle as the Disable Drive method, the Disable Drive method executes first.
Enable Drive	Enables the drive. If the Enable Drive and Disable Drive methods are called in the same cycle, the Disable Drive method executes and the Enable Drive method is ignored. **Note** Executing this method when the drive is disabled due to a fault returns error `-65632` (`NIMCMM_driveDisabled`). Use the **Get Status** method to determine the source of the fault, resolve the fault, clear the fault status using the **Clear Faults** method, then execute the **Enable Drive** method again.
Get Status	Returns the module status. Executing the Get Status method does not cause a communication transfer. Any communication with the module, such as reading the current, Hall, or Vsup inputs, or executing the Clear Faults, Disable Drive, or Enable Drive methods, updates the drive status and fault information. When there is no communication for 500 µs a transfer is forced to obtain the latest status from the module. The following items are available: • **Drive Status**—Returns the drive status in an enum value. The following values are available: ◦ **Drive Disabled (Disabled by fault)**—The drive is disabled because a drive fault has occurred. ◦ **Drive Disabled (Disabled by software)**—The drive is disabled by the Disable Drive method or because the **Enable Drive** method has not yet executed. ◦ **Drive Enabled**—The drive is enabled. • **Module Fault**—Returns TRUE if a module fault is present and FALSE if no fault is present. • **Module Fault Code**—Returns the module fault code as a decimal number in u16 format. Convert the fault code to a binary number to determine which fault bits are

Method	Description
	true. For example, a fault code of 8 is the binary number 0000 1000, indicating a fault in bit 3 of the fault bitmap (Overcurrent Fault). **Note** Refer to *NI 9502 Timing* for more information about the timing of this method.

Module Properties

Use the FPGA I/O Property Node to access the following properties with this device.

Refer to the 9502 Trapezoidal Commutation example in the `<labview>\examples\CompactRIO\Module Specific\NI 9502` directory for an example using these properties in Trapezoidal commutation mode.

Note Refer to the NI SoftMotion Module Help, available by selecting **Help»LabVIEW Help** for examples demonstrating Field Oriented Control commutation mode.

Table 590. Module Properties

Property	Type	Description
Hall Wiring (Trapezoidal commutation mode only)	Read/ Write	Configures the Hall input sequence for clockwise commutation by setting or returning the relationship between the Hall effect sensor inputs and the motor phases as an enum. This allows the NI 9502 module to send the PWM output to the correct motor phase. The default value is **1, 2, 3**. **Note** This property is not available in FOC commutation mode. If you use the NI 9502 with the FOC autophasing VI provided with the LabVIEW NI SoftMotion Module Premium, the VI determines the Hall relationship to use with FOC commutation mode.
Hall Filter	Read/ Write	Specifies the digital filter for the Hall effect sensor inputs in 25 ns clock tick increments. Signals that have a pulse width less than or equal to the digital filter period are filtered. Configure this property as a multiplier. For example, a setting of N specifies a 25 ns * N filter period. The default is 30 which represents a 750 ns filter. This property takes 16 µs to complete execution.

Property	Type	Description
		Refer to *NI 9502 Timing* for more information.
Module ID	Read	Returns the *module ID*, 0x751E.
Serial Number	Read	Returns the unique serial number of the module.
Vendor ID	Read	Returns the National Instruments vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop.

FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

NI 9502 Module Faults

NI 9502 modules can return the following faults as a decimal fault code. Convert the decimal fault code to a binary number to determine which fault bits are true. For example, a fault code of 2 is the binary number 0000 0010, indicating a fault in bit 1 of the fault bitmap (Module Overvoltage Fault).

Table 591. Module Faults

Fault Bit	Decimal (Binary)	Fault Name	Description
0	1 (0000 0001)	Module Undervoltage Fault	The external power supply (V_{sup}) is not connected, does not meet the minimum power supply voltage specifications, or one of the module outputs has a short circuit. Verify that the output currents and power supply are within specifications and check the module connections.
1	2 (0000 0010)	Module Overvoltage Fault	The external power supply (V_{sup}) voltage exceeds the module power supply specifications. Verify that the power supply is within the module specifications and that the supply is properly connected to the module.

Fault Bit	Decimal (Binary)	Fault Name	Description
2	4 (0000 0100)	Reserved	—
3	8 (0000 1000)	Overcurrent Fault	The commanded current output exceeds the NI 9502 maximum peak current of 8 A. **Note** The LabVIEW NI SoftMotion Module Premium provides a Current Limit Algorithm (I2T) VI that provides motor overload protection by coercing the current output when the requested current output exceeds user-specified system constraints. Your LabVIEW Platform DVD includes a free 30-day evaluation of the NI SoftMotion Module Premium that you can install to evaluate the Current Limit Algorithm VI and other FPGA Motor Control VIs.
4	16 (0001 0000)	Module Current Sampling Fault	A current, Hall, or Vsup input node was executed at the same time that a forced current sample was in progress. Refer to *NI 9502 Timing* for more information.
5–15	—	Reserved	—

NI 9502 Commutation Mode (FPGA Interface)

The NI 9502 provides support for either Trapezoidal commutation mode or Field Oriented Control commutation mode. The I/O nodes available for the module change depending on the **Commutation Mode** selected in the C Series Module Properties dialog box.

Trapezoidal Commutation

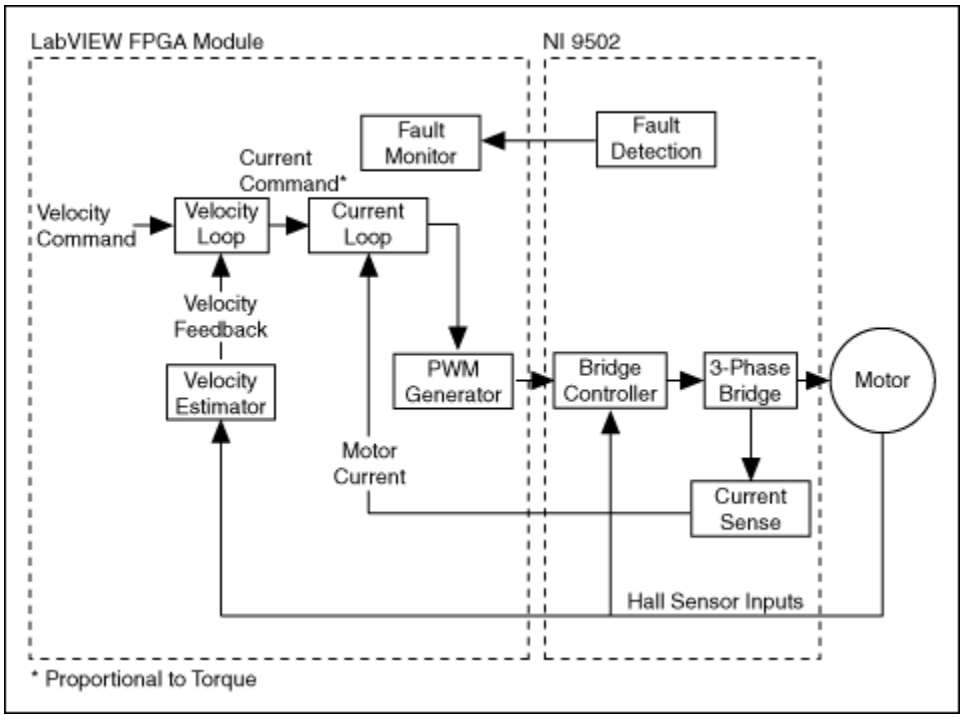
Trapezoidal commutation involves sending current through two of the three motor phases at any time. The phase transitions are controlled by the Hall effect sensor state changes. With trapezoidal commutation, one electrical revolution is divided into six steps of 60 degrees each.

The NI 9502 performs trapezoidal commutation based on the Hall effect sensor inputs.

For proper phase switching to occur, the Hall sensors from the motor must be connected to the correct Hall inputs on the NI 9502 module. The Hall sensor phase sequence is motor-dependent, so selecting the correct inputs requires manual testing. Refer to *Connecting Hall Effect Sensors for Trapezoidal Commutation* for a detailed walkthrough using an example VI for assistance in determining the proper Hall input connections to the NI 9502 module.

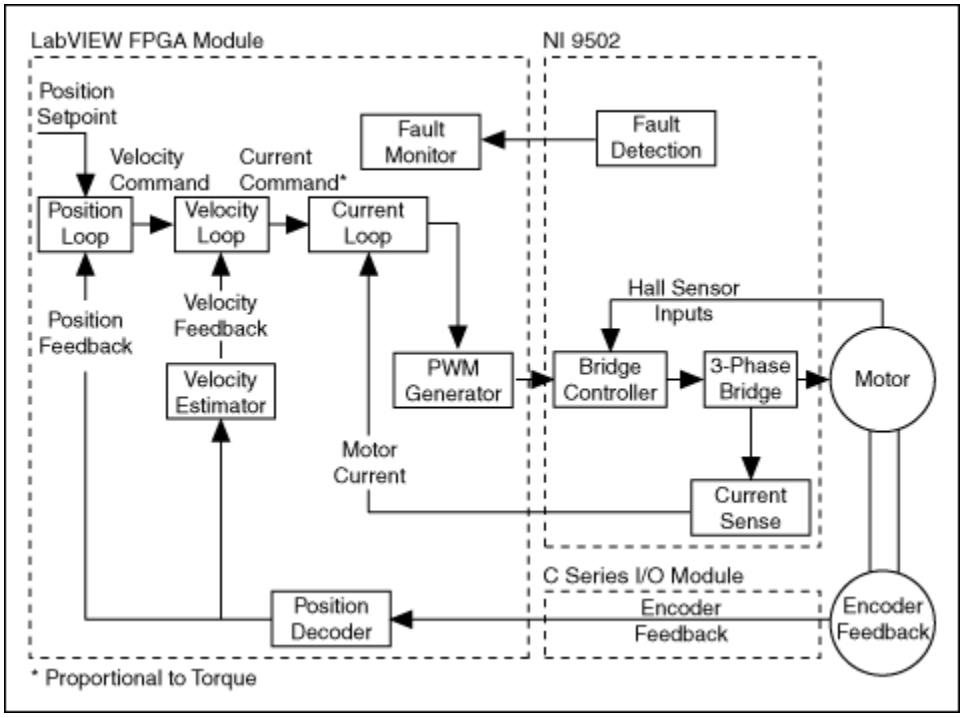
The following figure shows the NI 9502 module block diagram when it is used in Trapezoidal commutation mode.

Figure 137. NI 9502 Block Diagram showing Trapezoidal Commutation Mode



If your application requires position feedback you can also use a different I/O module for encoder feedback as shown in the following figure.

Figure 138. NI 9502 Block Diagram showing Encoder Feedback

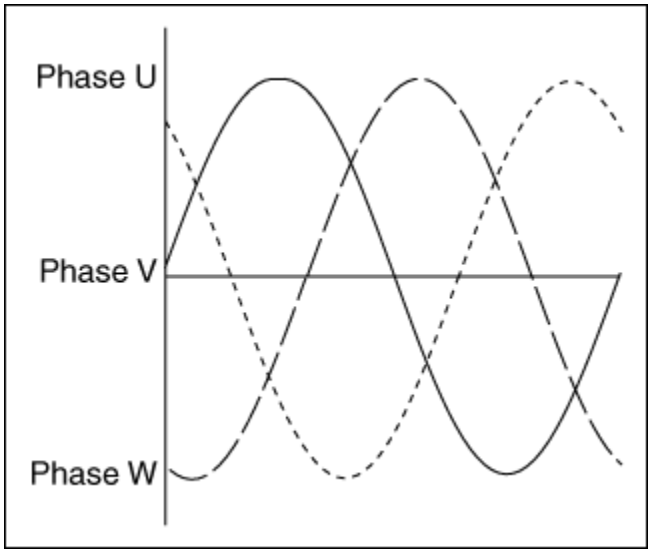


While trapezoidal commutation is simple to use, it does produce a torque ripple of up to 15% at slow speeds, which is not the case with *Field Oriented Control* commutation.

Field Oriented Control

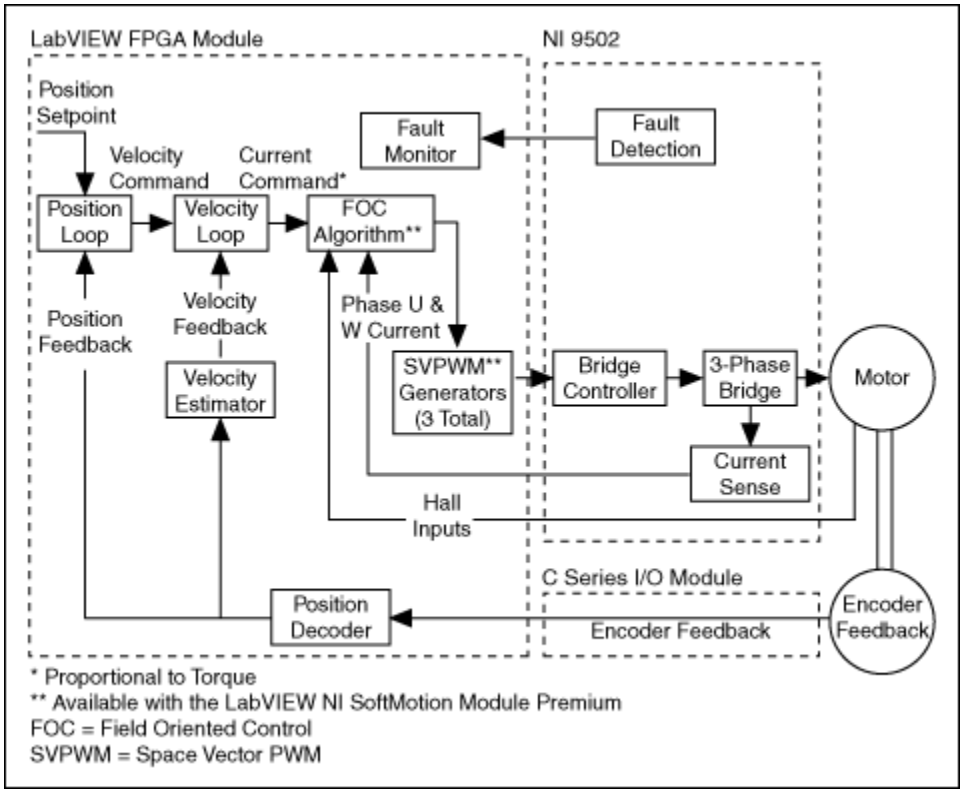
Field Oriented Control (FOC) commutation mode uses a more advanced algorithm that allows smooth motion at slow speeds and better efficiency at high speeds. The LabVIEW NI SoftMotion Module Premium provides VIs and example programs that make it easier to use the NI 9502 with FOC commutation mode. LabVIEW Platform includes a free 30-day evaluation of the NI SoftMotion Module Premium that you can install to evaluate using the NI 9502 in FOC mode. The following figure shows an example motor phasing diagram during FOC mode.

Figure 139. Motor Phasing Diagram during FOC Mode



The following figure shows the NI 9502 module block diagram when it is used in FOC commutation mode.

Figure 140. NI 9502 Block Diagram showing FOC Commutation Mode



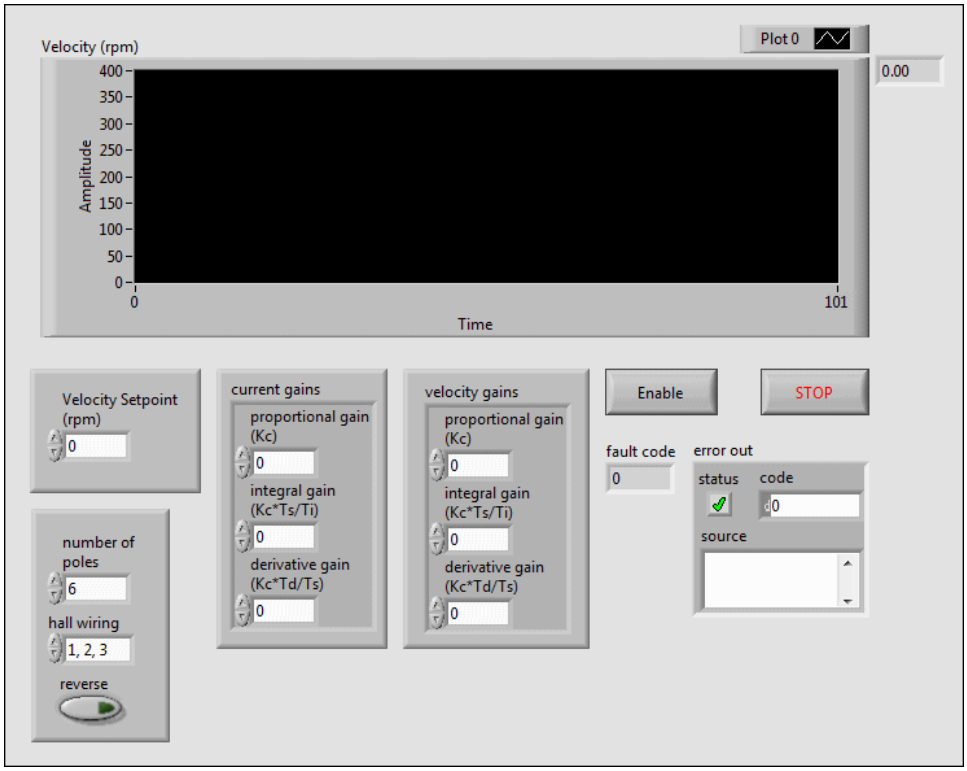
Connecting Hall Effect Sensors for Trapezoidal Commutation (FPGA Interface)

Use the Hall 1 input, Hall 2 input, and Hall 3 inputs to connect the motor Hall effect sensors. For proper phase switching to occur during Trapezoidal commutation mode, the Hall sensors from the motor must be connected to the correct Hall inputs on the NI 9502 module. The Hall sensor phase sequence is motor-dependent, so selecting the correct inputs may require manually testing the motor wiring.

The following tutorial walks you through the Hall sensor connections for the NI 9502 using the 9502 Trapezoidal Commutation example installed in the labview\examples\CompactRIO\Module Specific\NI 9502\Trapezoidal Commutation directory. It is recommended that you print out this topic so that you can fill in the table provided below.

1. Connect the motor and Hall sensor inputs to the drive, but do not connect the motor to a load.
2. Open the 9502 Trapezoidal Commutation example.
3. Configure the example for your FPGA target. Refer to *Moving Examples to Another FPGA Target* at `zone.ni.com` for information about how to do this.
4. Open the 9502 Trapezoidal Commutation desktop VI.
5. Ensure that the NI 9502 is in Trapezoidal Commutation mode by right-clicking the NI 9502 item in the **Project Explorer** window and selecting **Properties** from the shortcut menu.
6. Compile the FPGA VI.
7. Run the RT VI.
8. With the **hall wiring** property set to the default of **1, 2, 3**, enter a very small positive velocity (such as 20 RPM) in the **Velocity Setpoint (rpm)** control.
9. Click the **Enable** button to enable the drive and move the motor.
10. Record the highest velocity returned in the **Positive Velocity** column. The velocity is returned in the **Velocity (rpm)** graph on the front panel:

Figure 141. Velocity Graph on the Front Panel



Note If the motor moves erratically or not at all, place an X in both velocity columns, as this wiring scheme can be eliminated.

11. Click the **Enable** button again to disable the drive and stop the motor.
12. Enter the same velocity as a negative number in the **Velocity Setpoint** control.
13. Click the **Enable Drive** button to enable the drive and move the motor.
14. Record the highest velocity returned in the **Negative Velocity** column.
15. Click the **Enable Drive** button again to disable the drive and stop the motor.
16. Repeat steps 8 through 15 for the other five possible wiring combinations.
17. Stop the VI using the **Stop** button.

When the table is complete, look for the **hall wiring** value where the velocities in both directions were closest to the setpoint. This is the setting to use for your system. If these steps did not provide a conclusive result, repeat the test with a higher **Velocity Setpoint**. Once you have the correct **hall wiring** setting configure the PID gains as appropriate for your system and complete your application.

Table 592. Hall Wiring Property Results

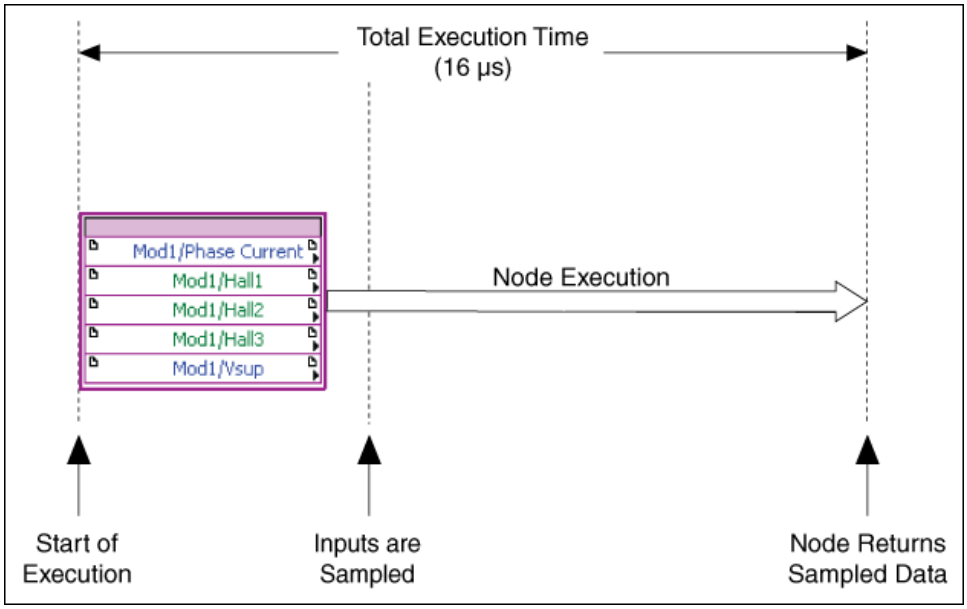
Hall Wiring Property Setting	Positive Velocity	Negative Velocity
1, 2, 3		
1, 3, 2		
2, 1, 3		
2, 3, 1		
3, 1, 2		
3, 2, 1		

NI 9502 Timing (FPGA Interface)

NI 9502 communication transfers can take up to 16 µs to complete. Sampling current, Hall, and Vsup inputs, or executing the Clear Faults, Enable Drive, or Disable Drive method result in a communication transfer to the module. When two or more of these input nodes or methods are called in parallel, the executions are serialized, which can lead to jitter in your application. National Instruments recommends the following guidelines for a jitter-free application at the maximum current loop rate:

- Configure a single FPGA I/O Node for sampling the NI 9502 current, Hall, and Vsup inputs simultaneously. This allows all inputs to be sampled in a single transfer. The following figure shows the complete data transfer time for each execution of the FPGA I/O Node.

Figure 142. Data Transfer Time of the FPGA I/O Node



- Avoid executing the NI 9502 Clear Faults method, Disable Drive method, and Enable Drive method when they are not needed. Each execution takes 16 µs to execute and no other requests can execute until each method completes execution. For example, do not place these methods in a loop that continually executes, but instead use a Case structure to allow execution only when needed. Running the module methods can introduce jitter into the current sampling loop.
- Executing the Get Status method does not cause a communication transfer. Any communication with the module, such as reading the current, Hall, or Vsup inputs, or executing the Clear Faults, Disable Drive, or Enable Drive methods, updates the drive status and fault information. When there is no communication for 500 µs, a transfer is forced to obtain the latest status from the module.
- Sample the current inputs at a rate faster than 500 µs. If current is sampled at a slower rate, the NI 9502 forces a current sample every 500 µs. The ensures that the motor phase currents do not exceed the NI 9502 maximum specified drive current of 8 A. If the current, Hall, or Vsup inputs nodes are executed at the same time that a forced sample is in progress the module returns a *Module Current Sampling Fault*.

The 9502 Trapezoidal Commutation example in the `<labview>\examples\CompactRIO\Module Specific\NI 9502` directory demonstrates recommended NI 9502 programming practices.

Note Refer to the NI SoftMotion Module Help, available by selecting

Help»LabVIEW Help for examples demonstrating Field Oriented Control commutation mode.

NI 9503 (FPGA Interface)

FPGA I/O Node

You can use an FPGA I/O Node, configured for reading or writing, with this device.

Caution National Instruments strongly recommends sampling the NI 9503 current and Vsup inputs in a single node to ensure consistent and jitter-free operation. Refer to *NI 9503 Timing* for more information.

You can select the following terminals for this device.

Table 593. Terminals in Software

Terminal	Type	Description
Phase A+	Write	Controls the Phase A+ output to the motor when the drive is enabled. A value of TRUE sets the output high, and a value of FALSE (default) sets the output low. The minimum high pulse width is 200 ns. The minimum low pulse width is 200 ns. The maximum high pulse width is 500 µs. This terminal is available in a single-cycle Timed Loop. **Note** You cannot use this output when the drive is disabled. Executing this node when the drive is disabled does not set the output and returns error –65631 (NIMCMM_driveNotEnabled).
Phase A-	Write	Controls the Phase A- output to the motor when the drive is enabled. A value of TRUE sets the output high, and a value of FALSE (default) sets the output low. The minimum high pulse width is 200 ns. The minimum low pulse width is 200 ns. The maximum high pulse width is 500 µs. This terminal is available in a single-cycle Timed Loop. **Note** You cannot use this output when the drive is disabled. Executing this node when the drive is disabled does not set the output and returns error –65631 (NIMCMM_driveNotEnabled).
Phase	Write	Controls the Phase B+ output to the motor when the drive is enabled. A value of

Terminal	Type	Description
B+		TRUE sets the output high, and a value of FALSE (default) sets the output low. The minimum high pulse width is 200 ns. The minimum low pulse width is 200 ns. The maximum high pulse width is 500 µs. This terminal is available in a single-cycle Timed Loop. **Note** You cannot use this output when the drive is disabled. Executing this node when the drive is disabled does not set the output and returns error −65631 (NIMCMM_driveNotEnabled).
Phase B-	Write	Controls the Phase B- output to the motor when the drive is enabled. A value of TRUE sets the output high, and a value of FALSE (default) sets the output low. The minimum high pulse width is 200 ns. The minimum low pulse width is 200 ns. The maximum high pulse width is 500 µs. This terminal is available in a single-cycle Timed Loop. **Note** You cannot use this output when the drive is disabled. Executing this node when the drive is disabled does not set the output and returns error −65631 (NIMCMM_driveNotEnabled).
Phase A Current	Read	Samples the motor Phase A current and returns the current level in i16 format. To convert this LSB value to milliamps, multiply the LSB value by 4.028 mA/LSB. This terminal takes a minimum of 18.5 µs to return data. Wait at least 1 µs after a phase change before reading the current. Refer to *NI 9503 Timing* for more information. This terminal is not available in a single-cycle Timed Loop.
Phase B Current	Read	Samples the motor Phase B current and returns the current level in i16 format. To convert this LSB value to milliamps, multiply the LSB value by 4.028 mA/LSB. This terminal takes a minimum of 18.5 µs to return data. Wait at least 1 µs after a phase change before reading the current. Refer to *NI 9503 Timing* for more information. This terminal is not available in a single-cycle Timed Loop.
Vsup Voltage	Read	Samples the Vsup voltage and returns the voltage level in u16 format. To convert this LSB value to millivolts, multiply the LSB value by 8.8623 mV/LSB. This terminal takes a minimum of 18.5 µs to return data. Refer to *NI 9503 Timing* for more information. This terminal is not available in a single-cycle Timed Loop.
User LED	Write	Controls the User LED on the NI 9503 module. Set the **User LED** terminal to TRUE to turn the LED on and FALSE to turn the LED off. This terminal is not available in a single-cycle Timed Loop.

Arbitration

This device supports only the **Arbitrate if Multiple Requestors Only** option for arbitration. You cannot configure arbitration settings for this device.

Module Methods

Use the FPGA I/O Method Node to configure the following methods for this device.

Note Refer to the NI SoftMotion Module Help for examples. This is available by selecting `Help»LabVIEW Help`.

Table 594.

Method	Description
Clear Faults	Clears module faults. Module faults do not clear if the fault condition is still present; always resolve the cause of the fault before executing this method.
Disable Drive	Disables the drive.
Enable Drive	Enables the drive. **Note** Executing this method when the drive is disabled due to a fault returns error `–65632` (`NIMCMM_driveDisabled`). Use the **Get Status** method to determine the source of the fault, resolve the fault, clear the fault status using the **Clear Faults** method, then execute the **Enable Drive** method again.
Get Status	Returns the module status. The following items are available: • **Drive Status**—Returns the drive status in an enum value. The following values are available: ◦ **Drive Disabled (Disabled by fault)**—The drive is disabled because a drive fault has occurred. ◦ **Drive Disabled (Disabled by software)**—The drive is disabled by the Disable Drive method or because the **Enable Drive** method has not yet executed. ◦ **Drive Enabled**—The drive is enabled. • **Module Fault**—Returns TRUE if a module fault is present and FALSE if no fault is present. • **Module Fault Code**—Returns the module *fault code* as a decimal number in u16

Method	Description
	format. Convert the fault code to a binary number to determine which fault bits are true. For example, a fault code of 4 is the binary number 0000 0100, indicating a fault in bit 2 of the fault bitmap (Overcurrent Fault).

Module Properties

Use the FPGA I/O Property Node to access the following properties with this device.



Note Refer to the NI SoftMotion Module Help, available by selecting **Help»LabVIEW Help** for examples.

Table 595. Module Properties

Property	Type	Description
Module ID	Read	Returns the *module ID*, 0x751F.
Serial Number	Read	Returns the unique serial number of the module.
Vendor ID	Read	Returns the National Instruments vendor ID, 0x1093.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are multiples of 40 MHz, such as 40 MHz, 80 MHz, 120 MHz, and so on.

NI 9503 Module Faults

NI 9503 modules can return the following faults as a decimal fault code. Convert the decimal fault code to a binary number to determine which fault bits are true. For example, a fault code of 2 is the binary number 0000 0010, indicating a fault in bit 1 of the fault bitmap (*Module Overvoltage Fault*).

Table 596. NI 9503 Module Faults

Fault Bit	Decimal (Binary)	Fault Name	Description
0	1 (0000 0001)	Module Undervoltage Fault	The external power supply (V_{sup}) is not connected or does not meet the minimum power supply voltage specifications. Verify that the power supply is within the module specifications and check the module connections.
1	2 (0000 0010)	Module Overvoltage Fault	The external power supply (V_{sup}) voltage exceeds the module power supply specifications. Verify that the power supply is within the module specifications and check the module connections.
2	4 (0000 0100)	Overcurrent Fault	The current to the motor exceeds the NI 9503 maximum peak current of 8 A.
3–15	—	Reserved	—

NI 9503 Timing (FPGA Interface)

NI 9503 communication transfers can take up to 18.5 µs to complete. Sampling current or Vsup inputs, writing the User LED, or executing any of the methods, will result in a communication transfer to the module. If one of these nodes is called while a communication transfer is in progress, the node will not execute until the transfer is completed. National Instruments recommends the following guidelines for a jitter-free application at the maximum current loop rate:

- Configure a single FPGA I/O Node to sample the Phase A Current, Phase B Current, and Vsup at the same time. This allows all inputs to be sampled in a single transfer without jitter.
- Avoid unnecessary method node calls, because each execution requires a communication transfer and no other requests can execute until each method completes its transfer. For example, do not place these methods in a loop that continually executes, but instead use a Case structure to allow execution only when needed.

Note Refer to the NI SoftMotion Module Help, available by selecting **Help»LabVIEW Help** for examples.

NI 9505 (FPGA Interface)

You can use an FPGA I/O Node, configured for reading or writing, with this device. All NI 9505 FPGA I/O operations execute in parallel when the I/O Node is configured with multiple I/O items.

Note Wait at least one second before accessing the module immediately after inserting it.

Caution Do not perform property reads on the Module ID, Serial Number, or Vendor ID properties while accessing the other properties, methods, or terminals of the module.

You can select the following terminals for this device:

Table 597. Terminals in Software

Terminal	Type	Description
Current Sense	Read	Monitors the motor current and returns the current level in i16 format. To convert this LSB value to amps, multiply the LSB value returned by 0.0062 A/LSB. This input takes up to 20 µs to return data. Current value is positive for clockwise motor direction and negative for counterclockwise motor direction. This input is not available in a single-cycle Timed Loop. **Note** This terminal has a maximum delay of 205 ns + 6 system clock cycles and a minimum delay of 115 ns + 6 system clock cycles until the current is sampled. Refer to *NI 9505 Current Measurement Considerations* for detailed information about using this terminal for current control.
Encoder Phase A	Read	Encoder Phase A input. The encoder signals are raw digital input signals. These signals are used in the LabVIEW FPGA module for position and/or velocity feedback. This input is available in a single-cycle Timed Loop. **Note** To avoid returning invalid data, do not call this terminal unless V_{SUP} is present.

Terminal	Type	Description
		Refer to the Quadrature Decoder with Position & Velocity — NI 9505 example in the `labview\examples\CompactRIO\Module Specific\NI 9505` directory for an encoder example.
Encoder Phase B	Read	Encoder Phase B input. The encoder signals are raw digital input signals. These signals are used in the LabVIEW FPGA module for position and/or velocity feedback. This input is available in a single-cycle Timed Loop.  **Note** To avoid returning invalid data, do not call this terminal unless V_{SUP} is present. Refer to the Quadrature Decoder with Position & Velocity — NI 9505 example in the `labview\examples\CompactRIO\Module Specific\NI 9505` directory for an encoder example.
Encoder Index	Read	Encoder Index (Phase Z) input. The encoder signals are raw digital input signals. These signals are used in the LabVIEW FPGA module for position and/or velocity feedback. This input is available in a single-cycle Timed Loop.  **Note** To avoid returning invalid data, do not call this terminal unless V_{SUP} is present. Refer to the Quadrature Decoder with Position & Velocity — NI 9505 example in the `labview\examples\CompactRIO\Module Specific\NI 9505` directory for an encoder example.
Motor	Write	 **Caution** Do not violate the minimum pulse width of 2 µs (high or low) for this terminal. Doing so will result in unpredictable performance. Controls the power to the motor. A value of TRUE turns on the motor output. The default value of FALSE turns off this output. This output is available in a single-cycle Timed Loop. Refer to *NI 9505 Motor Output Pulse Width Considerations* for more information about using this signal.  **Note** If you add error terminals to the I/O node and an error comes into the node, the drive is disabled and the motor comes to a stop based on its own friction. Refer to the PWM Generation — NI 9505 example in the `labview\examples\CompactRIO\Module Specific\NI 9505` directory for an example using

Terminal	Type	Description
		PWM to vary power to the motor. **Note** The example does not account for the signal delays described in *NI 9505 Motor Output Pulse Width Considerations* and *NI 9505 Current Measurement Considerations*.
Drive Direction	Write	An enum value that selects the direction that the motor spins. This output is available in a single-cycle Timed Loop, however it may take up to 20 µs for this output to update. The following are valid values: • **Clockwise** (default)—Spin the motor clockwise. • **Counterclockwise**—Spin the motor counterclockwise. The maximum update rate for this output is once per PWM cycle.
E-Stop	Read	Displays the emergency stop status. A value of TRUE indicates that the emergency stop input is active. A value of FALSE indicates that the emergency stop input is inactive. The E-Stop input is disabled by default but you can still monitor the input status with this terminal. You can enable E-Stop using the **Enable E-Stop** property. This input is not available in a single-cycle Timed Loop. **Note** To avoid returning invalid data, do not call this terminal unless V_{SUP} is present.

Arbitration

This device supports only the Never Arbitrate option for arbitration. You cannot configure arbitration settings for this device.

Note Do not execute two instances of the Motor and Drive Direction terminals at the same time. Doing so will result in unexpected behavior.

Module Methods

Use the FPGA I/O Method Node to configure the following methods for this device:

Table 598. Module Methods

Method	Description
Disable Drive	Disables the drive. Executing this method takes up to 60 µs to complete and overrides any previous Enable Drive call.
Enable Drive	**Caution** The **Enable Drive** method is executed even if faults or an E-Stop condition are present. Do not continually execute this method in a loop without checking for faults or E-Stop as this will result in the drive never being disabled. Enables the drive. Executing this method takes up to 10 ms to complete. If this method is called while executing Disable Drive, the method is executed but the drive is not enabled.

Module Properties

Use the FPGA I/O Property Node to access the following properties with this device:

Table 599. Module Properties

Property	Type	Description
Drive Fault	Read	When TRUE, indicates the presence of a drive fault. A drive fault disables the drive. Causes for a drive fault are the following: **Caution** V_{SUP} greater than 40 V will result in damage to the module. • overvoltage • undervoltage • motor terminal (MOTOR±) short to V_{SUP} • motor terminal (MOTOR±) short to COM • sending commands to the motor before the drive is enabled **Note** Do not command motor movement until the drive is enabled with the **Enable Drive** method. Attempting to control the motor before it is enabled will result in a fault. • violating PWM minimum pulse width requirements. Refer to the Specifications section of the NI 9505 Operating Instructions for more information about PWM.

Property	Type	Description
Drive Status	Read	Returns the drive status in an enum value. Valid values are the following: • **Enabled**—The drive is enabled. • **Disabled**—The drive is disabled. • **Disabled by E-Stop**—The drive is disabled because the emergency stop signal is active and enabled. • **Disabled by Drive Fault**—The drive is disabled because a drive fault has occurred. • **Disabled by Temp. Fault**—The drive is disabled because the module temperature exceeded 115 °C. • **Unknown**—The drive status is unknown because of one of the following conditions: ◦ The module was removed while an application was running. ◦ An application is running and a module that is not an NI 9505 was inserted.
Enable E-Stop	Write	Enables the emergency stop input. Set the input to TRUE to cause an active emergency stop input to disable the drive. Set the input to FALSE (default) to ignore the emergency stop input.
Module ID	Read	Returns the *module ID*, 0x71ED. Do not read this property while performing timed I/O operations.
Over Temperature Fault	Read	When TRUE, indicates that the internal module temperature has exceeded 115 °C and the drive is disabled.
Serial Number	Read	Returns the unique serial number of the module. Do not read this property while performing timed I/O operations.
Vendor ID	Read	Returns the National Instruments vendor ID, 0x1093. Do not read this property while performing timed I/O operations.
V_{SUP} Present	Read	If TRUE, indicates that the user has properly connected the motor DC power supply to the NI 9505 screw terminals.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop for the Encoder Phase A, Encoder Phase B, Encoder Index, Motor, and Drive Direction terminals of this device. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Node Properties dialog box.

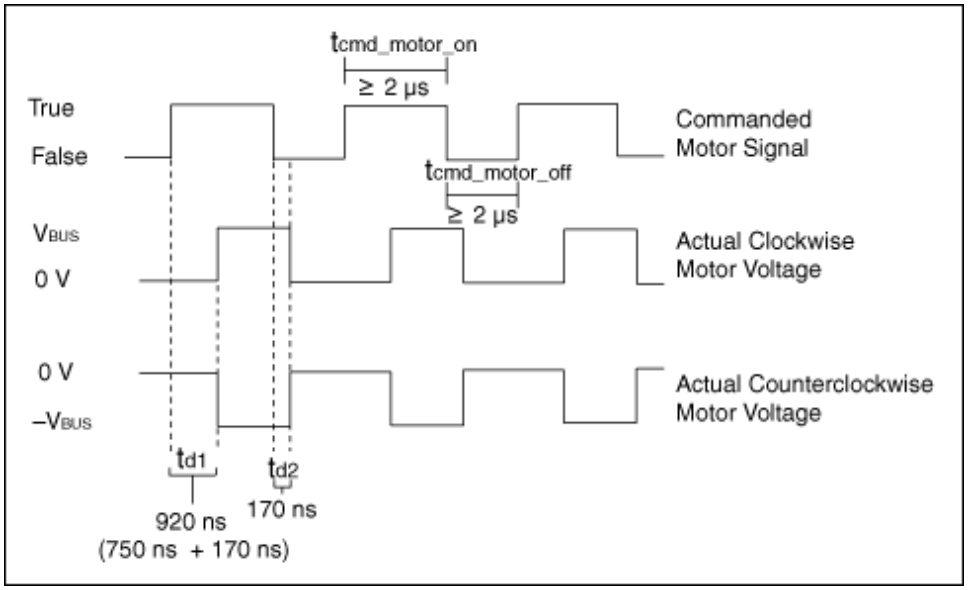
FPGA Target Clock Support

This device supports only top-level FPGA target clock rates and single-cycle Timed Loop clock rates that are 80 MHz or slower.

NI 9505 Motor Output Pulse Width Considerations (FPGA Interface)

The NI 9505 commanded Motor terminal output and actual Motor terminal output differ by a small amount. The following figure shows the commanded signal sent and the offset for both clockwise and counterclockwise motor voltage. Time offsets indicated are typical.

Figure 143. Commanded Sgnal sent and the Offset for Clockwise and Counterclockwise Motor Voltage



where t_{d1} = 750 ns + 170 ns is the delay between the active edge of the commanded motor signal to the active edge of the actual motor voltage

t_{d2} = 170 ns is the delay between the falling edge of the commanded motor signal to the falling edge of the actual motor voltage

$t_{cmd\_motor\_on}$ is the time that the Motor terminal of the LabVIEW FPGA I/O node is TRUE; pulse width cannot be between 0 µs and 2 µs

$t_{cmd\_motor\_off}$ is the time that the Motor terminal of the LabVIEW FPGA I/O node FALSE signal; pulse width cannot be between 0 µs and 2 µs

Note The PWM Generation — NI 9505 Example VI in the `labview\examples\CompactRIO\Module Specific\NI 9505` directory does not account for these signal delays.

Use the following equation to make sure the actual motor voltage matches your desired waveform:

$t_{cmd\_motor\_on\_calc} = t_{desired\_motor\_on} + 750\ ns$

Where:

Table 600.

where	$t_{cmd\_motor\_on\_calc}$	is an intermediate value used to determine the correct value of $t_{cmd\_motor\_on}$
	$t_{desired\_motor\_on}$	is the pulse width of the signal you require

Once you determine the value of $t_{cmd\_motor\_on\_calc}$ you need to also ensure that your pulse width is above the 2 µs minimum. Use the following table to help determine the value of $t_{cmd\_motor\_on}$ to use in your LabVIEW FPGA VI.

Note The minimum pulse width requirement refers to the commanded motor signal and not to the actual motor voltage.

$t_{cmd\_motor\_on\_calc}$	$t_{cmd\_motor\_on}$
0 µs	0 µs
0 to 2 µs	2 µs*
2 µs to ($T^{\dagger}$ – 2 µs)	$t_{cmd\_motor\_on\_calc}$
(T – 2 µs) to T	T – 2 µs*

Note The NI 9505 module cannot accept a pulse width less than 2 µs (high or low). Doing so will result in unpredictable performance.

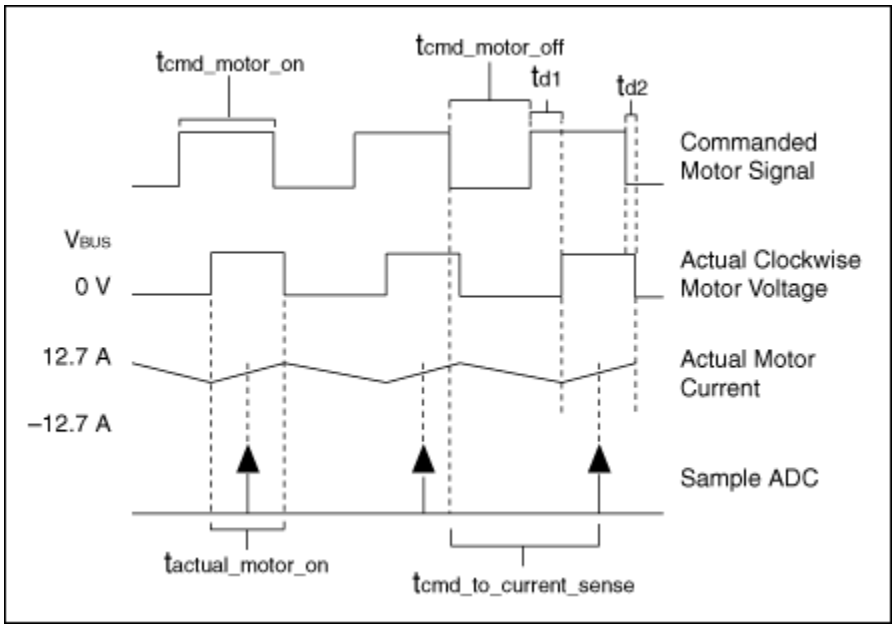
Note *T* is defined as the period of the commanded motor signal.

Use the $t_{cmd\_motor\_on}$ time to set the Motor terminal of the LabVIEW FPGA I/O node *TRUE* to compensate for signal delays. Refer to *NI 9505 Motor Output and Current Sense Example* for a sample motor signal calculation.

NI 9505 Current Measurement Considerations (FPGA Interface)

The current sense circuit uses a serial bus to transfer the 12-bit current measurement back to the FPGA. This data transfer can take up to 20 µs to complete. Due to this 20 µs delay, sampling the current multiple times during the motor output period is not practical. Since the current waveform through your inductive load is not the same over the entire period of the motor output signal, it is important to sample the current at the halfway point of the ON portion of the actual motor voltage to obtain an approximate average current. To ensure that current is sampled at the correct time, you must compensate for the delays in both the current sense interface and the *motor output interface*. The following figure demonstrates how to determine the correct time to sample the current.

Figure 144. Determine the Time to Sample the Current



Add the following logic into the FPGA VI for the Current Sense I/O node terminal to correctly compensate for the signal offsets and sample the current the appropriate

amount of clock ticks after the falling edge of the commanded motor signal. The formula is based on specified typical delays.

$$t_{cmd\_to\_current\_sense} = \left[t_{cmd\_motor\_off} + t_{d1} + \frac{t_{actual\_motor\_on}}{2} - t_{d\_current\_sense} \right]$$

where	$t_{cmd\_to\_current\_sense}$	is the time from the falling edge of the commanded motor signal until the Current Sense terminal of the LabVIEW FPGA I/O node is executed
	t_{d1} = 750 ns + 170 ns	is the delay between the active edge of the commanded motor signal to the active edge of the actual motor voltage
	$t_{actual\_motor\_on} = t_{cmd\_motor\_on}$ - 750 ns	
	$t_{d\_current\_sense} = \frac{6 \text{ clock ticks}}{f_{clk}} + 150 \text{ ns}$	is the delay from the Current Sense terminal of the LabVIEW FPGA I/O node call until the current is sampled
	f_{clk}	is the FPGA clock rate
	$t_{cmd\_motor\_off} = T - t_{cmd\_motor\_on}$	
	T	is the commanded motor period, typically 20 kHz

The equation further reduces to:

$$t_{cmd\_to\_current\_sense} = T + 395 \text{ ns} - \frac{t_{cmd\_motor\_on}}{2} - \frac{6}{f_{clk}}$$

To convert $t_{cmd\_to\_current\_sense}$ to the number of LabVIEW FPGA clock ticks, multiply by f_{clk}. Refer to *NI 9505 Motor Output and Current Sense Example* for a sample motor signal calculation.

NI 9505 Motor Output and Current Sense Example (FPGA Interface)

Use the following example as a guide to determine what values to use in your LabVIEW FPGA block diagram to compensate for the *Motor output pulse width considerations* and *current measurement considerations*.

1. Define your desired motor output waveform: $t_{desired\_motor\_on}$ is the desired on time of your motor output pulse $t_{desired\_motor\_off}$ is the desired off time of your

motor output pulse given the above T = commanded motor period = $t_{desired\_motor\_on} + t_{desired\_motor\_off}$

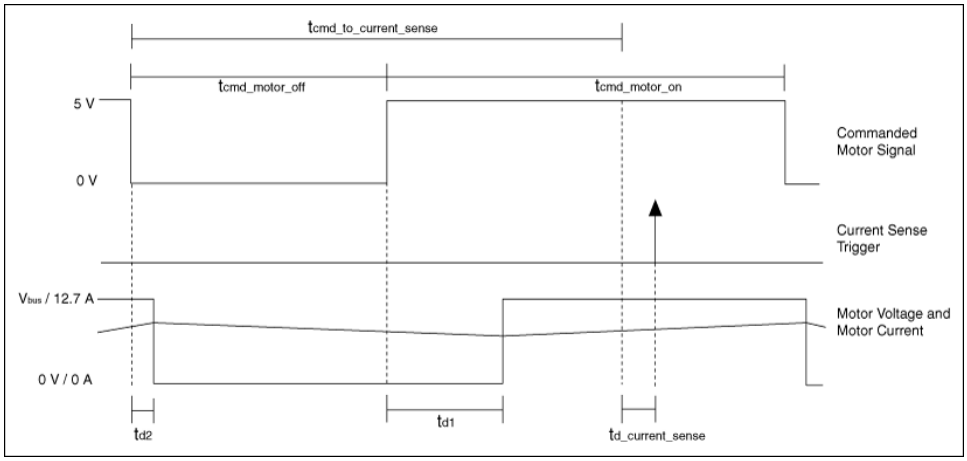
2. Compensate for the 750 ns delay described in *NI 9505 Motor Output Pulse Width Considerations*: $t_{cmd\_motor\_on\_calc} = t_{desired\_motor\_on} + 750$ ns
3. Determine the ON time ($t_{cmd\_motor\_on}$)for the Motor terminal on the LabVIEW FPGA I/O node. Refer to the table in *NI 9505 Motor Output Pulse Width Considerations* for information about determining this value. Once you calculate $t_{cmd\_motor\_on}$, convert this to clock ticks by multiplying $t_{cmd\_motor\_on}$ by f_{clk}. f_{clk} is the FPGA clock rate. Use this value to determine the amount of TRUE ticks per period (T) of the signal sent to the Motor terminal on the LabVIEW FPGA I/O node.
4. Calculate the delay needed to wait before sampling the current $t_{cmd\_to\_current\_sense}$. Refer to *NI 9505 Current Measurement Considerations* for more information.

   Figure 145.

$$t_{cmd\_to\_current\_sense} = T + 395\text{ ns} - \frac{t_{cmd\_motor\_on}}{2} - \frac{6}{f_{clk}}$$

   To convert $t_{cmd\_to\_current\_sense}$ to the number of LabVIEW FPGA clock ticks, multiply by f_{clk}. Use this value to determine the amount of clock ticks to wait after the last falling edge of the commanded motor signal and executing the Current Sense terminal on the LabVIEW FPGA I/O node.

Note The PWM Generation — NI 9505 Example VI in the `labview\examples\CompactRIO\Module Specific\NI 9505` directory does not account for these signal delays.



Calculating the Correct Values for the Motor and Current Sense Terminals

Given a 40 MHz top-level clock in the FPGA, the calculations to produce a 50 µs period PWM signal with a 25% duty cycle are as follows:

1. $t_{desired\_motor\_on}$ = 12.5 µs $t_{desired\_motor\_off}$ = 37.5 µs T = 50 µs
2. $t_{cmd\_motor\_on\_calc}$ = 12.5 µs + 750 ns = 13.25 µs
3. Because 13.25 µs is greater than 2 µs and less than (50 µs – 2 µs), $t_{cmd\_motor\_on}$ = 13.25 µs 13.25 µs * 40 MHz = **530 ticks** 530 ticks of the 2000 tick period sent to the Motor terminal of the LabVIEW FPGA I/O node are TRUE and the remaining 1470 ticks are FALSE.
4. Figure 146.

   $$t_{cmd\_to\_current\_sense} = 50\ \mu s + 395\ ns - \frac{13.25\ \mu s}{2} - \frac{6}{40\ MHz} = 43.62\ \mu s$$

   43.62 µs * 40 MHz = **1745 ticks** Therefore, we need a 1745 tick delay after the commanded motor signal falling edge (TRUE to FALSE) before calling the Current Sense terminal of the LabVIEW FPGA I/O node.

C Series Motor Drive Interface Modules

Table 601. C Series Motor Drive Interface Modules

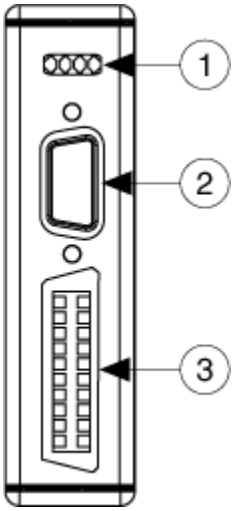
Motion Module	Description
NI-9512	Stepper, 1-Axis, Single Encoder C Series Motor Drive Interface Module
NI-9514	Servo, 1-Axis, Single Encoder C Series Motor Drive Interface Module
NI-9516	Servo, 1-Axis, Dual Encoder C Series Motor Drive Interface Module

Use the NI-9512/9514/9516 drive interface modules in Scan Interface Mode.

NI 951x LEDs and Connectors

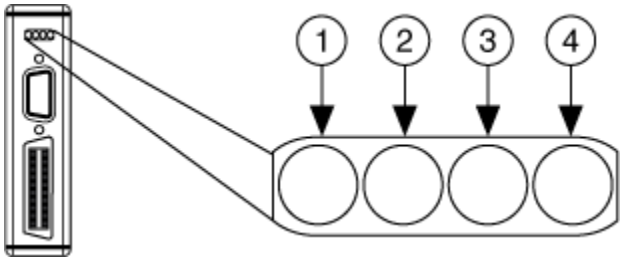
In this document, NI 951x refers to the NI-9512/9514/9516 C Series motor drive interface modules.

Figure 147. NI 9512/9514/9516 LEDs and Connectors



1. LEDs
2. 15-pin DSUB connector
3. MDR 20-pin connector

Figure 148. NI 9512/9514/9516 LED Status Indicators



1. Axis Status (Green)
2. Encoder Active (Green)
3. Limit Active (Yellow)
4. Axis Fault (Red)

Figure 149. NI 9512/9514/9516 15-Pin DSUB Connector Pinout

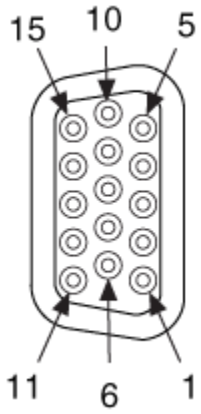


Table 602. NI 9512 15-Pin DSUB Connector Pin Assignments

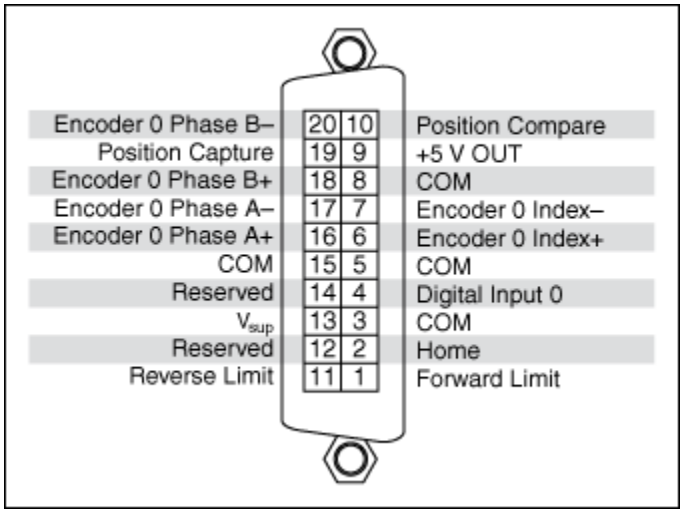
Pin	Signal
1	Reserved
2	Drive Enable
3	Digital Input 3
4	Digital Input 2
5	Digital Output 1
6	Reserved
7	COM
8	Digital Input 1
9	Direction (CCW)–
10	Step (CW)–
11	Digital Output 0
12	Vsup
13	Direction (CCW)+
14	COM
15	Step (CW)+

Table 603. NI 9514/9516 15-Pin DSUB Connector Pin Assignments

Pin	Signal
1	Drive Command COM

Pin	Signal
2	Drive Enable
3	Reserved
4	Reserved
5	Reserved
6	Drive Command
7	COM
8	Digital Input 1
9	Reserved
10	Reserved
11	Reserved
12	Vsup
13	Reserved
14	COM
15	Reserved

Figure 150. NI 9512/9514/9516 20-Pin MDR Connector Pinout



NI 951x Module Faults

NI 951x modules return the faults listed in the following table. To determine which

fault bits are true, convert the fault code to a binary number. For example, a fault code of 4 is the binary number 100, indicating a fault in bit 2 of the fault bitmap (Drive Enable Short Circuit).

Table 604. NI 951x Module Faults

Fault Bit	Fault Name	Description
0	Module Undervoltage Fault	The external power supply (V_{sup}) is not connected or does not meet the minimum power supply specifications. Verify that the power supply is within the module specifications and that the supply is properly connected to the module.
1	Module Overvoltage Fault	The external power supply (V_{sup}) exceeds the module voltage power supply specifications. Verify that the power supply is within the module specifications and that the supply is properly connected to the module.
2	Drive Enable Short Circuit	The Drive Enable output has a short circuit. Verify that the output current is within specifications and check your Drive Enable output connections.
3	Digital Output 0 Short Circuit	Digital Output 0 has a short circuit. Verify that the output current is within specifications and check your Digital Output connections.
4	Digital Output 1 Short Circuit	Digital Output 1 has a short circuit. Verify that the output current is within specifications and check your Digital Output connections.
5	Timing Synchronization Fault	The maximum amount of jitter and/or drift in the system has been exceeded. Verify that you do not have any processes running at a higher priority than the Scan method, and that your real-time target is not overloaded. If the problem persists, contact National Instruments technical support.
6	Watchdog Communication Fault	The communication between the module and the chassis has been interrupted. Verify that your system connections are correct, adjust your cabling to reduce noise on the system, verify that you are correctly incrementing the **Host Communication Watchdog** Write method input, or adjust the **Communication Watchdog Period**. If you do not require the watchdog functionality, you can also set the **Communication Watchdog Period** to 0 to disable it. Refer to Using the NI 951x Communication Watchdog for more information.
7	Processor Watchdog	A fault occurred in the module internal processor. Please contact National Instruments technical support.
8	Internal Power Supply Fault	The module has detected a failure in its internal power supply. Please contact National Instruments technical support.

Fault Bit	Fault Name	Description
9–15	Reserved	—

Downloading Firmware to an NI 951x Module

You can use the *C Series Module Properties* dialog box for the NI 9512, NI 9514, or NI 9516 to check the current firmware version on the NI 951x module and to download firmware to the module. Downloading the Firmware to the NI 951x Module

Before downloading firmware to the NI 951x module ensure that the chassis or the NI 951x module slot is configured for Scan Interface Mode and that the NI Scan Engine is in Configuration mode. Use the NI Distributed System Manager to view and configure the scan engine mode. Use the Get Scan Engine Mode VI to read the scan engine mode programmatically. Use the Set Scan Engine Mode VI to set the scan engine mode programmatically.



Note The NI 951x modules require high-speed interface slots when the module is used in Scan Interface programming mode. When the chassis is in Scan Interface programming mode, the high-speed interface slots are typically slots 1 through 4. When the chassis is in FPGA Interface programming mode, you can configure individual slots to use the high-speed interface.

Complete the following steps to download firmware to the module.

1. Right-click the NI 951x module in the **Project Explorer** window and select **Properties** from the shortcut menu to open the **C Series Module Properties** dialog box.
2. Click **Check Version**. The 951x Firmware Update Utility checks the firmware version on the module and displays the version number in **Firmware Version on Module**.
   - To upgrade firmware on the module, check that the version is greater than the version displayed in **Available Firmware Version**.
   - To downgrade firmware on the module, check that the version is less than the version displayed in **Available Firmware Version**.

3. Click **Download**. The firmware on the host is transferred to the module. This process may take several minutes.

> **Note**
>
> If the firmware update process is interrupted during transfer by removing the module or disconnecting power, click **Download** to restart the transfer. Firmware updates on NI 951x modules is supported only on legacy compactRIO controllers such as the cRIO-902x and cRIO-907x. It is not supported on Linux targets such as cRIO-906x and cRIO 903x.

NI 951x (Scan Interface)

Use the NI-9512, NI-9514, and NI-9516 C Series modules with the NI SoftMotion Module and an NI SoftMotion axis. To associate the module with an NI SoftMotion axis, right-click the target and select `New»NI SoftMotion Axis` from the shortcut menu.

For information about configuring and using this module in Scan Interface programming mode, refer to *Working with NI 951x C Series Modules* in the NI SoftMotion Module book of LabVIEW Help.

Single-Board RIO (sbRIO) Device Reference

Use Single-Board RIO (sbRIO) devices and RIO Mezzanine cards with NI CompactRIO Device Drivers in LabVIEW.

Table 605. Single-Board RIO Devices

sbRIO Device	Description
sbRIO-9603	Single-Board Reconfigurable I/O (DIO), 1 RS-232 Serial port, 1 CAN port, Artix-7 75T FPGA
sbRIO-9607	Single-Board Reconfigurable I/O (DIO), 1 RS-232 Serial port, 1 CAN port, Zynq-7020 FPGA
sbRIO-9608	Single-Board Reconfigurable I/O (DIO), 1 RS-232 Serial port, 1 CAN port, Artix-7 200T FPGA
sbRIO-9609	Single-Board Reconfigurable I/O (DIO), 1 RS-232 Serial port, 1 CAN port, Artix-7 200T FPGA
sbRIO-9627	Single-Board Reconfigurable I/O (MIO), 16 AI channels, 4 AO channels, 4 3.3 V DIO channels, 2 RS-232 Serial ports, 1 RS-485 Serial port, 1 CAN port, RIO mezzanine card connector, Zynq-7020 FPGA
sbRIO-9628	Single-Board Reconfigurable I/O (MIO and DIO), 16 AI channels, 4 AO channels, 4 5 V DIO channels, 2 RS-232 Serial ports, 1 RS-485 Serial port, 1 CAN port, RIO mezzanine card connector, Artix-7 100T FPGA
sbRIO-9629	Single-Board Reconfigurable I/O (MIO and DIO), 16 AI channels, 4 AO channels, 4 5 V DIO channels, 2 RS-232 Serial ports, 1 RS-485 Serial port, 1 CAN port, RIO mezzanine card connector, Artix-7 200T FPGA
sbRIO-9637	Single-Board Reconfigurable I/O (MIO and DIO), 16 AI channels, 4 AO channels, 28 3.3 V DIO channels, 2 RS-232 Serial ports, 1 RS-485 Serial port, 1 CAN port, Zynq-7020 FPGA
sbRIO-9638	Single-Board Reconfigurable I/O (MIO), 16 AI channels, 4 AO channels, 28 5 V DIO channels, 2 RS-232 Serial ports, 1 RS-485 Serial port, 1 CAN port, Artix-7 100T FPGA
sbRIO-9651 SOM	Embedded System on Module (SOM) with Real-Time Processor and Reconfigurable FPGA, 160 configurable DIO channels, 85,000 logic cells

Table 606. RIO Mezzanine Cards

RIO Mezzanine Card	Description
sbRIO-9683	RIO Mezzanine Card, General Purpose Inverter Controller
sbRIO-9684	RIO Mezzanine Card, 16-bit General Purpose Inverter Controller
sbRIO-9694	RIO Mezzanine Card, 96 channel digital I/O breakout board
sbRIO-9697	RIO Mezzanine Card, 2-slot C Series, 24 3.3 V DIO channels
sbRIO-9698	RIO Mezzanine Card, 1-Slot C Series, Secondary Ethernet Port, 24 3.3 V DIO channels
NI Digital I/O Mezzanine Card	96 DIO channels

Related concepts:

- Single-Board RIO System

Single-Board RIO Devices

Table 607. Single-Board RIO Devices

sbRIO Device	Description
sbRIO-9603	Single-Board Reconfigurable I/O (DIO), 1 RS-232 Serial port, 1 CAN port, Artix-7 75T FPGA
sbRIO-9607	Single-Board Reconfigurable I/O (DIO), 1 RS-232 Serial port, 1 CAN port, Zynq-7020 FPGA
sbRIO-9608	Single-Board Reconfigurable I/O (DIO), 1 RS-232 Serial port, 1 CAN port, Artix-7 200T FPGA
sbRIO-9609	Single-Board Reconfigurable I/O (DIO), 1 RS-232 Serial port, 1 CAN port, Artix-7 200T FPGA
sbRIO-9627	Single-Board Reconfigurable I/O (MIO), 16 AI channels, 4 AO channels, 4 3.3 V DIO channels, 2 RS-232 Serial ports, 1 RS-485 Serial port, 1 CAN port, RIO mezzanine card connector, Zynq-7020 FPGA
sbRIO-9628	Single-Board Reconfigurable I/O (MIO and DIO), 16 AI channels, 4 AO channels, 4 5 V DIO channels, 2 RS-232 Serial ports, 1 RS-485 Serial port, 1 CAN port, RIO mezzanine card connector, Artix-7 100T FPGA

sbRIO Device	Description
sbRIO-9629	Single-Board Reconfigurable I/O (MIO and DIO), 16 AI channels, 4 AO channels, 4 5 V DIO channels, 2 RS-232 Serial ports, 1 RS-485 Serial port, 1 CAN port, RIO mezzanine card connector, Artix-7 200T FPGA
sbRIO-9637	Single-Board Reconfigurable I/O (MIO and DIO), 16 AI channels, 4 AO channels, 28 3.3 V DIO channels, 2 RS-232 Serial ports, 1 RS-485 Serial port, 1 CAN port, Zynq-7020 FPGA
sbRIO-9638	Single-Board Reconfigurable I/O (MIO), 16 AI channels, 4 AO channels, 28 5 V DIO channels, 2 RS-232 Serial ports, 1 RS-485 Serial port, 1 CAN port, Artix-7 100T FPGA
sbRIO-9651 SOM	Embedded System on Module (SOM) with Real-Time Processor and Reconfigurable FPGA, 160 configurable DIO channels, 85,000 logic cells

sbRIO-9603/9608/9609/9628/9638/9629 CAN API

FPGA I/O Node

Use an FPGA I/O Node for CAN Input or CAN Output.

You can select the following terminals for this device.

Table 608. Terminals in Software

Terminal	Description
CAN0	CAN Port 0

I/O Methods

Use the following I/O methods on the sbRIO-9603/9608/9609/9628/9638/9629.

Table 609. I/O Methods

Method	Description
Abort Transmit	Abort a pending CAN frame transmission.
Reset	Reset the CAN port to the same state as when the FPGA VI started running.
Start	Start communication.

Method	Description
Stop	Stop communication.
Wait on Comm State Change	Wait for a change in the communication state of the CAN port (Comm State property).
Wait on Transceiver Wakeup	Wait for the Transceiver Mode property to change from Sleep to Normal mode due to a remote wakeup (bus activity) or local wakeup.
Wait on Transmit Complete	Wait for all frames written to CAN Output to complete transmission.

I/O Properties

Use the FPGA I/O Property Node to access the following properties with the sbRIO-9603/9608/9609/9628/9638/9629.

Table 610. I/O Properties

Property	Description	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
Baud Rate	Specifies the baud rate and default sample point for CAN bus.	No	No	Yes	No
Baud Rate Advanced	Specifies custom baud rate and sample point for CAN bus.	No	No	Yes	No
FD Baud Rate	Specifies the baud rate and default sample point for CAN bus with flexible data rate.	No	No	Yes	No
FD Baud Rate Advanced	Specifies custom baud rate and sample point for CAN bus with flexible data rate.	No	No	Yes	No
Comm State	Describes the current communication state of the CAN controller.	Yes	Yes	N/A	N/A
Listen Only	Controls listen-only mode for passive monitoring/logging.	No	No	Yes	No
I/O Mode	Sets the CAN operating mode to	No	No	Yes	No

Property	Description	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
	CAN(default), CAN FD or CAN FD + BRS.				
Input Timeout (ms)	Specifies the time to wait for the reading of CAN frame using CAN Input. The resolution is in milliseconds.	No	No	Yes	No
Output Timeout (ms)	Specifies the time to wait for the writing of CAN frame using CAN Output. The resolution is in milliseconds.	No	No	Yes	No
Log Bus Errors	Enables the logging of bus errors as frames that can be read using the CAN Input node.	No	No	Yes	Yes
Log Transceiver Faults	Enables the logging of transceiver faults as frames that can be read using the CAN Input node.	No	No	Yes	Yes
Self Reception	Specifies whether to echo successfully transmitted CAN frames to be read using CAN Input.	No	No	Yes	Yes
Single Shot Transmit	Specifies whether to retry failed CAN frame transmissions.	No	No	Yes	No
Receive Error Counter	Provides access to the CAN controller Receive Error Counter.	Yes	Yes	N/A	N/A
Transmit Error Counter	Provides access to the CAN controller Transmit Error Counter.	Yes	Yes	N/A	N/A
Transceiver Mode	Sets the mode for the CAN transceiver and the associated mode in the TCAN4550 CAN controller.	No	Yes	No	Yes

CAN Input

Wait for a CAN frame to be received, then return that frame.

Complete the following steps to read from a CAN port using CAN Input.

1. Use the **Add Targets and Devices** dialog box to add the CAN module to your CompactRIO configuration. Right-click the CAN module in the **Project Explorer** window and select Properties to specify the configuration properties for the module.
2. Create an FPGA I/O item for CAN by right-clicking the FPGA target under **My Computer** in the **Project Explorer** window, selecting **New»FPGA I/O**, and then adding **CAN0** to the FPGA I/O list.
3. Place an **I/O Node** from the FPGA I/O palette onto your FPGA VI diagram.
4. Right-click the I/O Node and select **Select FPGA I/O»CAN0**.
5. Right-click the I/O Node and select **Change to Read** to use the I/O node for CAN Input.
6. Right-click the I/O Node and choose **Properties** to select the Data Type.

Start communication on the CAN port prior to using CAN Input. Enable **Auto Start** in the Module Configuration, or invoke the *Start* method to start communication.

The **Input Timeout (ms)** of the CAN Advanced Port Configuration dialog box specifies how long to wait for a new frame to be received. If you specify **Input Timeout (ms)** of 0, the CAN Input node will simply check to see if a new frame has arrived (non-blocking).

CAN Input can store the result of two received packets. Receiving an additional packet without calling CAN Input will cause an overflow error. NI recommends placing CAN Input in a loop that is executed when the device receives a data packet.

For information on how CAN Input arbitrates with other CAN nodes, refer to Arbitration.



Note You can add more than one CAN input port to the FPGA I/O node, and also add analog and digital inputs to the same FPGA I/O node by right-clicking the FPGA I/O node and selecting **Add Element**. Right-click the created I/O name and choose **Select FPGA** I/O to specify the I/O name. If you use more than one input per node, LabVIEW executes them sequentially starting with the top input. If one input is blocked while waiting for the input, subsequent inputs are delayed. For example, if you read an Analog or Digital Input and CAN0 in the same input node, that node waits for a frame to be received on CAN0 and waits for Analog or Digital data to be available. Accessing the CAN0 port using a distinct CAN Input Node is more efficient.

Properties Dialog

The **Data Type** selects the type for the CAN frame returned by the CAN Input node. The available values are **Cluster** (default) and **Array of U32**.

The benefits of using the **Cluster** data type are:

- Simpler block diagram (easy to use)
- Fast and efficient, if you limit use to the block diagram only (not front panel)

The benefit of the **Array of U32** data type, and handshaking integers to the host VI one at a time, is that it is faster for transfer to/from host VI.

Refer to the *Using Clusters and Arrays in LabVIEW FPGA Application Note* for more information on the use of clusters and arrays.

Node Inputs

- **Error In**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.

Node Outputs

- **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable. Unless you set **Input Timeout (ms)** to –1 to wait indefinitely, you must use the error terminals to determine whether the CAN Input successfully received a CAN frame.
- **CAN*x***—If the **Data Type** is defined as **Cluster** (default), the CAN frame is represented as a cluster with the following elements:

Table 611. CAN*x* Cluster Elements

Data Type	Element	Description
U32	**Timestamp High**, **Timestamp Low**	Timestamp of when the frame was received. The timestamp is acquired at the end of the CAN frame. The high and low U32 represent a single U64 timestamp. The timestamp is large enough to avoid handling rollover. The timestamp is zero-based (relative). The time starts ticking at zero when CAN communication starts. The resolution is 100 ns.

Data Type	Element	Description
U32	**Identifier**	Arbitration ID. If bit 29 (20000000 hexadecimal) is clear, this ID uses the standard format (11-bit). If bit 29 is set, this ID uses extended format (29-bit).
U8	**Type**	Type of frame. • **Data Frame**: 0 • **Remote Frame**: 1 • **Bus Error**: 6 • **Transceiver Fault**: 7 • **CAN FD Data Frame**: 16 • **CAN FD+BRS Data Frame**:24
U8	**InfoA**	Reserved for future use.
U8	**InfoB**	Reserved for future use.
U8	**Data Length**	For data frames, this provides the number of bytes in **Data** (0–8 for CAN and 0–64 for CAN FD). For remote frames, this indicates the number of bytes requested.
[U8]	**Data**	For data frames, this provides the data bytes. The array uses a fixed size of 64 bytes. Values above **Data Length** are not valid and should be ignored. For remote frames, the values contained in **Data** are not valid and should be ignored. If the **Data Type** is defined as **Array of U32**, the CAN frame is represented as an array of U32 values, as shown in the following table.

Table 612. Array of U32 Values

Most significant byte	**...**	**...**	**Least significant byte**
Timestamp (upper U32)			
Timestamp (lower U32)			
Identifier			
Type	**InfoA**	**InfoB**	**Data Length**
Data[0]	Data[1]	Data[2]	Data[3]
Data[4]	Data[5]	Data[6]	Data[7]
Data[8]	Data[9]	Data[10]	Data[11]
Data[12]	Data[13]	Data[14]	Data[15]
Data[16]	Data[17]	Data[18]	Data[19]
Data[20]	Data[21]	Data[22]	Data[23]
Data[24]	Data[25]	Data[26]	Data[27]
Data[28]	Data[29]	Data[30]	Data[31]
Data[32]	Data[33]	Data[34]	Data[35]
Data[36]	Data[37]	Data[38]	Data[39]
Data[40]	Data[41]	Data[42]	Data[43]
Data[44]	Data[45]	Data[46]	Data[47]
Data[48]	Data[49]	Data[50]	Data[51]
Data[52]	Data[53]	Data[54]	Data[55]
Data[56]	Data[57]	Data[58]	Data[59]
Data[60]	Data[61]	Data[62]	Data[63]

In the table above, the meaning of each element is the same as the **Cluster** data type.

CAN Output

Write a CAN frame to be transmitted.

Complete the following steps to write to a CAN port using CAN Output.

1. Use the Add Targets and Devices dialog box to add the CAN module to your CompactRIO configuration. Right-click the CAN module in the Project Explorer window and select Properties to specify the configuration properties for the module.
2. Create an FPGA I/O item for CAN by right-clicking the FPGA target under **My Computer** in the **Project Explorer** window, selecting **New»FPGA I/O**, and then adding **CAN0** to the FPGA I/O list. Refer to New FPGA I/O dialog box (FPGA Module) for more information about creating and adding FPGA I/O to your CompactRIO configuration.
3. Place an I/O Node from the FPGA I/O palette onto your FPGA VI diagram.
4. Right-click the I/O Node and select **Select FPGA I/O»CAN0**.
5. Right-click the I/O Node and select **Change to Write** to use the I/O node for CAN Output.
6. Right-click the I/O Node and choose **Properties** to select the Data Type.

Start communication on the CAN port prior to using CAN Output. Enable **Auto Start** in the Module Configuration, or invoking the *Start* method to start communication.

The communication path from LabVIEW FPGA to the CAN port on the module is implemented as a FIFO. The CAN Output node waits for an available element in the FIFO, then writes the frame to the FIFO. If no other CAN frame exists in the FIFO, the CAN frame will begin to transmit immediately.

CAN Output does not wait for the requested transmit to complete on the network (acknowledgment). Call CAN Output followed by *Wait on Transmit Complete* (I/O Method) if you want to transmit CAN frames one at a time.

Write multiple frames to the output FIFO if you want to transmit CAN frames as fast as possible. The CAN module will submit these frames to the TCAN4550 controller as fast as possible.

When the output FIFO is full, the CAN Output node waits for an element to become available, then writes the frame to the FIFO. The **Output Timeout (ms)** of the CAN Advanced Port Configuration dialog box specifies how long to wait for a new element to become available, which occurs when a frame from a previous CAN Output transmits successfully onto the network. If you specify **Output Timeout (ms)** of 0, the CAN Output node returns an error status of TRUE (error) if a new element is not available (non-blocking). When an error is returned, you must attempt CAN Output of

the same frame again at a later time.

In addition to the CAN Output node, the **Output Timeout (ms)** also applies to most methods and properties. For more information on the output path from LabVIEW FPGA to the CAN port, refer to Arbitration.



Note You can add more than one CAN output port to the FPGA I/O node, and also add analog and digital outputs to the same FPGA I/O node by right-clicking the FPGA I/O node and selecting **Add Element**. Right-click the created I/O name and select **Select FPGA I/O** to specify the I/O name. If you use more than one output per node, LabVIEW will execute them sequentially starting with the top output. If one output is blocked while waiting for the output, subsequent outputs will be delayed.



Note The Transmit Packet Type may not exceed the mode set by the I/O Mode property.

Properties Dialog

Selects the **Data Type** for the CAN frame written to the CAN Input node. The available values are **Cluster** (default) and **Array of U32**.

The benefits of using the **Cluster** data type are:

- Simpler block diagram (easy to use)
- Fast and efficient, if you limit use to the block diagram only (not front panel)

The benefits of the **Array of U32** data type, and handshaking integers to the host VI one at a time, are:

- Faster for transfer to/from host VI
- Fewer FPGA gates

Refer to the *Using Clusters and Arrays in LabVIEW FPGA Application Note* for more information on the use of clusters and arrays.

Node Inputs

- **Error In**—Optional. Not shown by default. Right-click the node and select Error Terminals to enable.
- **CAN*x***—If the **Data Type** is defined as **Cluster** (default), the CAN frame is represented as a cluster. Refer to the following table for the cluster elements.

Table 613. CAN*x* Cluster Elements

Data Type	Element	Description
U32	**Timestamp High**,	Ignored. The transmit request always occurs as soon as possible (not timed).
U32	**Timestamp Low**	Ignored. The transmit request always occurs as soon as possible (not timed).
U32	**Identifier**	Arbitration ID. If bit 29 (20000000 hexadecimal) is clear, this ID uses the standard format (11-bit). If bit 29 is set, this ID uses extended format (29-bit).
U8	**Type**	Type of frame: • **Data Frame**: 0 • **Remote Frame**: 1 • **CAN FD Data Frame**: 16 • **CAN FD+BRS Data Frame**:24
U8	**InfoA**	Reserved for future use.
U8	**InfoB**	Reserved for future use.
U8	**Data Length**	For data frames, this provides the number of bytes in **Data** (0–8 for CAN and 0–64 for CAN FD). For remote frames, this

Data Type	Element	Description
		indicates the number of bytes requested.
	**Data**	For data frames, this provides the data bytes. The array uses a fixed size of 64 bytes. If the **Data Type** is defined as **Array of U32**, the CAN frame is represented as an array of U32 values, as shown in the following table.

Table 614. Array of U32 Values

Most significant byte	**...**	**...**	**Least significant byte**
Timestamp (upper U32)			
Timestamp (lower U32)			
Identifier			
Type	**InfoA**	**InfoB**	**Data Length**
Data[0]	Data[1]	Data[2]	Data[3]
Data[4]	Data[5]	Data[6]	Data[7]
Data[8]	Data[9]	Data[10]	Data[11]
Data[12]	Data[13]	Data[14]	Data[15]
Data[16]	Data[17]	Data[18]	Data[19]
Data[20]	Data[21]	Data[22]	Data[23]
Data[24]	Data[25]	Data[26]	Data[27]
Data[28]	Data[29]	Data[30]	Data[31]
Data[32]	Data[33]	Data[34]	Data[35]
Data[36]	Data[37]	Data[38]	Data[39]

Most significant byte	...	...	Least significant byte
Data[40]	Data[41]	Data[42]	Data[43]
Data[44]	Data[45]	Data[46]	Data[47]
Data[48]	Data[49]	Data[50]	Data[51]
Data[52]	Data[53]	Data[54]	Data[55]
Data[56]	Data[57]	Data[58]	Data[59]
Data[60]	Data[61]	Data[62]	Data[63]

In the table above, the meaning of each element is the same as the **Cluster** data type.

Node Outputs

- **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable. Unless you set the **Output Timeout (ms)** to -1 to wait indefinitely, you must use the error terminals to determine whether the CAN Output successfully submitted a CAN frame for transmit.

I/O Methods for sbRIO Devices

Reset

Reset the CAN port to the same state as when the FPGA VI started running.

The RESET method does the following:

- Resets the TCAN4550.
- Restores all properties to their default values.

If you enable **Auto Start** in the tool, this method also re-starts communication.

When a CAN controller enters the Bus Off state, it no longer communicates. Refer to the **Comm State** property for the Bus Off value. By performing a complete reset of the TCAN4550, this method recovers from the Bus Off state.

This method also clears the output FIFO as well as any pending CAN node, including all Wait methods. The reset affects only the specified CAN port. The reset has no effect on

the other CAN port of the module.

Node Inputs

- **Error In**—Optional. Not shown by default. To enable, right-click the node and select **Error Terminals**.

Node Outputs

- **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals**.

Related concepts:

- Comm State

Start

Start communication.

Cases for using this method, as opposed to enabling **Auto Start** in Module Configuration:

- Change configuration properties within the diagram (such as the **Listen Only** property)
- Auto-baud algorithms (must Stop/Start to set the baud rate for each attempt)
- Start CAN communication on a trigger (analog or digital input)

Node Inputs

- **Error In**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.

Node Outputs

- **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.

Stop

Stop communication.

This method also clears the output FIFO as well as any pending CAN node (including all Wait methods).

Node Inputs

- **Error In**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.

Node Outputs

- **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.

Abort Transmit

Abort a pending CAN frame transmission.

Since the determinism of CAN depends on the prioritization of frames based on ID, the TCAN4550 CAN controller allows you to abort a pending transmit to replace it with a higher priority ID.

When implementing protocols that require abort, ensure that only one transmit is pending. Call **Wait on Transmit Complete** before or after each call to **CAN Output**. If more than one transmit is pending at the time **Abort Transmit** is called, the **Transmit Success** Boolean could refer to any one of those pending frames.

Node Inputs

- **Error In**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.

Node Outputs

- **Transmit Success?**—Indicates whether the pending transmit was successfully aborted, or transmitted on the network before it could be aborted.
  - Returns *FALSE* when the pending transmit successfully aborted.
  - Returns *TRUE* when the pending frame transmitted successfully on the network prior to the **Abort Transmit**.
- **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.

Wait on Comm State Change

Wait for a change in the communication state of the CAN port (`Comm State` property). This enables you to handle CAN communication state changes in a separate loop from your main CAN Input/Output loop.

If the **Comm State** property changes before you call Wait (relative to the previous Wait), the current Wait returns immediately. The **Comm State** is not considered to be changed when you first Start, and it does not change after Stop.

This method returns an error if invoked while communication is stopped.

Node Inputs

- **Error In**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.
- **Timeout**—Specifies the time to wait for the communication state to change. The resolution is in milliseconds. The special **Timeout** of 0 is used to poll for the changed state. If **Comm State** has changed from the previous call to Wait, the **Timed Out?** Boolean is *FALSE*. If **Comm State** has not changed, the **Timed Out?** Boolean is *TRUE*. The **Timeout** value of *–1* (infinite wait) is not supported, but you can use the largest positive I32 value, which provides a maximum of approximately 25 days.

Node Outputs

- **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.
- **Timed Out?**—Indicates a timeout. When Timed Out? is *TRUE*, the error cluster indicates success (not a timeout error).
- **Comm State**—Returns the new value of the `Comm State` property.

Wait on Transceiver Wakeup

Wait for the `Transceiver Mode` property to change from *Sleep* to *Normal* mode due to a remote wakeup (bus activity) or local wakeup. A local wakeup occurs if you use the I/O Property node to set `Transceiver Mode` property back to *Normal*.

If a wakeup occurs before you call Wait (relative to the previous Wait), the current Wait

returns immediately. The wakeup is not considered to have occurred when you first Start, and it does not occur after Stop.

This method returns an error if invoked while communication is stopped.

Node Inputs

- **Error In**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.
- **Timeout**—Specifies the time to wait for the communication state to change. The resolution is in milliseconds. The special **Timeout** of 0 is used to poll for the changed state. If **Comm State** has changed from the previous call to Wait, the **Timed Out?** Boolean is *FALSE*. If **Comm State** has not changed, the **Timed Out?** Boolean is *TRUE*. The **Timeout** value of *–1* (infinite wait) is not supported, but you can use the largest positive I32 value, which provides a maximum of approximately 25 days.

Node Outputs

- **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.
- **Timed Out?**—Indicates a timeout. When Timed Out? is *TRUE*, the error cluster indicates success (not a timeout error).

Wait on Transmit Complete

Wait for all frames written to CAN Output to complete transmission. You typically use this to determine when all frames have been acknowledged.

If all transmits are complete before you call Wait (relative to the previous Wait), the current wait returns immediately. All transmits are considered complete after you first Start. This method returns an error if invoked while communication is stopped.

Node Inputs

- **Error In**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.
- **Timeout**—Specifies the time to wait for the communication state to change. The resolution is in milliseconds. The special **Timeout** of 0 is used to poll for the changed state. If **Comm State** has changed from the previous call to Wait, the

Timed Out? Boolean is *FALSE*. If **Comm State** has not changed, the **Timed Out?** Boolean is *TRUE*. The **Timeout** value of *–1* (infinite wait) is not supported, but you can use the largest positive I32 value, which provides a maximum of approximately 25 days.

Node Outputs

- **Error Out**—Optional. Not shown by default. Right-click the node and select **Error Terminals** to enable.
- **Timed Out?**—Indicates a timeout. When Timed Out? is *TRUE*, the error cluster indicates success (not a timeout error).

I/O Properties

Complete the following steps to use an FPGA I/O Property node for CAN:

1. Use the Add Targets and Devices dialog box to add the CAN module to your CompactRIO configuration. Right-click the CAN module in the Project Explorer window and select `Properties` to specify the configuration properties for the module.
2. Create an FPGA I/O item for CAN by right-clicking the FPGA target under **My Computer** in the **Project Explorer** window, selecting **New»FPGA I/O** to the FPGA I/O list.
3. Place an I/O Property Node from the FPGA I/O palette onto your FPGA VI diagram.
4. Right-click the I/O Property Node and select **Select Item»CAN»NI 9852»CAN0**.
5. Right-click the I/O Property Node and choose **Select Property** to specify the appropriate property from the list of available CAN properties.

For information on how the I/O Property Node arbitrates with other CAN nodes, refer to *Arbitration*.

The description of each property includes the following table.

Table 615. I/O Property Table Structure

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
<value>	<value>	<value>	<value>	<value>	<value>

- **Data type**—Shows the icon for the LabVIEW data type used for this property.
- **Permissions**—Read, Write, or Read/Write. If the property is Read/Write, you can right-click the property in the node and select **Change To Read** or **Change To Write** to select the direction. Otherwise, only one direction is supported.
- **Read While Stopped?**—Yes or No to indicate whether you can read the property while CAN communication is stopped.
- **Read While Running?**—Yes or No to indicate whether you can read the property while CAN communication is running.
- **Write while stopped?**—Yes or No to indicate whether you can write the property while CAN communication is stopped.
- **Write while running?**—Yes or No to indicate whether you can write the property while CAN communication is running.

Related concepts:

- Arbitration

Baud Rate

Table 616.

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
U16	Write	No	No	Yes	No

Each CAN baud rate selection configures the correspond Baud Rate for the transceiver when in classical CAN mode or when transmitting data before the BRS bit when in CAN FD + BRS mode. This property must always be set even when I/O Mode is set to CAN FD + BRS as the CAN FD + BRS protocol always has a portion of the frame that transfer in classical CAN mode.

The supported values in kb/s are 5.00, 6.150, 10.00, 12.5, 16.00, 20.00, 25.00, 31.25, 33.33, 40.00, 50.00, 62.50, 80.00, 83.33, 100.0, 125.0, 160.0, 200.0, 250.0, 400.0, 500.0, 800.0, and 1000. These bit fields are selected to create the desired baud rate with a default sample point of 87.5 percent.

This property converts the baud rate as selected into corresponds Synchronization Jump Width (SJW), Nominal Bit Rate Prescaler (BRP), Nominal Time Segment Before Sample Point (TSEG1), and Nominal Time Segment After Sample Point (TSEG2).

Baud Rate Advanced

Table 617. Baud Rate Advanced Property

Field	Data Type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
Tq	U16	Write	No	No	Yes	No
SJW	U8	Write	No	No	Yes	No
TSEG1	U8	Write	No	No	Yes	No
TSEG2	U8	Write	No	No	Yes	No

This property allows you to configure the custom CAN baud rate by setting the nominal bit timing and prescaler register of the CAN controller. You can set the following fields.

Table 618. Baud Rate Advanced Property Field Values

Field	Description	Valid Values
Time quantum (Tq)	Programs the baud rate prescaler	25 to 12,800 in increments of 0x19 (25 decimal)Note that the actual hardware interpretation of this value is one more than the programmed value
Synchronization Jump Width (SJW)	The maximum number of time quanta by which a bit sampling period can be extended or shortened as a result of re-synchronization.	0 to 127
Time Segment 1 (TSEG1)	The time segment before the sample point.	1 to 0xFF (1 to 255 decimal)
Time Segment 2 (TSEG2)	The time segment after the sample point.	0 to 0x7F (0 to 127 decimal)

The time quantum is calculated by the following formula.

$$Time\ quantum = (BRP + 1) * 25$$

Where

- *BRP* = nominal bit rate prescaler value

Use the custom sample point calculator on the FPGA I/O Properties page under `CAN0»Baud Rate»<Custom>` to determine the values to enter in this field to obtain the desired sample point.

`Invalid Parameter` or `Error 65545` is returned if you insert an invalid input.

FD Baud Rate

Table 619. FD Baud Rate Property

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
U16	Write	No	No	Yes	No

This property sets the baud rate for the transceiver when in FD + BRS CAN mode when transmitting data after the BRS bit. FD Baud Rate or FD Baud Rate Advanced property only need to be set when I/O mode is set to CAN FD + BRS.

The supported values in kb/s are 200, 250, 400, 500, 800, 1000, 1250, 1600, 2000, 2500, 4000 and 5000. These bit fields are selected to create the desired baud rate with a default sample point of 87.5%

This property converts the baud rate as selected into corresponds Time Quantum (TQ), Synchronization Jump Width (SJW), Data Time Segment Before Sample Point (TSEG1), Data Time Segment After Sample Point (TSEG2), Transmitter Delay Compensation Enabler(TDC) and Transmitter Delay Compensation Filter Window Length (TDCF).

FD Baud Rate Advanced

Table 620. FD Baud Rate Advanced Property, Output Data Type

Field	Data Type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
Tq	U16	Write	No	No	Yes	No
SJW	U8	Write	No	No	Yes	No
TSEG1	U8	Write	No	No	Yes	No
TSEG2	U8	Write	No	No	Yes	No
TDC	TF	Write	No	No	Yes	No

Field	Data Type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
TDCF	U8	Write	No	No	Yes	No
TDCO	U8	Write	No	No	Yes	No

This property allows you to configure the custom CAN FD baud rate by setting the nominal bit timing and prescaler register of the CAN controller. The table below lists the available fields.

Table 621. FD Baud Rate Advanced Property Field Values

Field	Description	Valid Values
Transmitter Delay Compensation (TDC)	Enables or disables this feature.	FALSE = disabledTRUE = enabled
Transmitter Delay Compensation Offset (TDCO)	Defines the distance between the delay from transmit to receive point and secondary sample point.	0 to 127
Transmitter Delay Compensation Filter Window Length (TDCF)	Defines the minimum value for the secondary sample point position. Enabled when TDCF is greater than TDCO.	0 to 127
Time quantum (Tq)	Programs the baud rate prescaler	25 to 800 in increments of 25 ns
Time Segment 1 (TSEG1)	The time segment before the sample point.	0 to 31
Time Segment 2 (TSEG2)	The time segment after the sample point.	0 to 15
Synchronization Jump Width (SJW)	The maximum number of time quanta by which a bit sampling period can be extended or shortened as a result of re-synchronization.	0 to 15Note that the actual hardware interpretation of this value is one more than the programmed value

The time quantum is calculated by the following formula.

$$Time\ quantum = (BRP + 1) * 25$$

Where

- *BRP* = nominal bit rate prescaler value

Use the custom sample point calculator on the FPGA I/O Properties page under `CAN0»Baud Rate»<Custom>` to determine the values to enter in this field to obtain the desired sample point.

`Invalid Parameter` or `Error 65545` is returned if you insert an invalid input.

Comm State

Table 622. Comm State Property

Data type	Permissions	Read while stopped?	Read while running?	Write while stopped?	Write while running?
	Read	Yes	Yes	N/A	N/A

This property describes the current communication state of the CAN controller.

The values are:

- 0: Error Active
- 1: Error Passive
- 2: Bus Off

Use the Wait on Comm State Change method to detect changes in this property.

When you initially start communication, the CAN port will begin in the Error Active state as specified in the CAN standard (except when **Listen Only** is enabled). After bus errors cause a transition to Error Passive or Bus Off, that state will be retained even after you stop communication and restart. Use the Reset method to reset back to the *Error Active* state.

Listen Only

Table 623. Listen Only Property

Data type	Permissions	Read while stopped?	Read while running?	Write while stopped?	Write while running?
	Write	No	No	Yes	No

This property controls **Listen Only** mode for passive monitoring/logging.

- *FALSE* disables **Listen Only** mode. Received frames are acknowledged, and frames can be transmitted using **CAN Output**.
- *TRUE* enables **Listen Only** mode. The CAN port can only receive frames. The port does not transmit on the network. No acknowledgment is transmitted when a frame is received. When listen-only is enabled, the *Comm State* property begins in the *Error Passive* state.

I/O Mode

Table 624. I/O Mode Property

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
U8	Write	No	No	Yes	No

This property configures the CAN operating mode. The following table lists property values.

Table 625. I/O Mode Property Values

Property	Description
CAN(default)	The default CAN 2.0 A/B standard I/O mode as defined in ISO 11898-1:2003. A fixed baud rate is used for transfer, and the payload length is limited to 8 bytes.
CAN FD	The CAN FD mode as specified in the CAN with Flexible Data-Rate specification, version 1.0. Payload lengths up to 64 are allowed, but they are transmitted at a single fixed baud rate.
CAN FD + BRS	The CAN FD as specified in the CAN with Flexible Data-Rate specification, version 1.0, with the optional Baud Rate Switching enabled. The same payload lengths as CAN FD mode are allowed; additionally, the data portion of the CAN frame is transferred at a different (higher) baud rate.

Input Timeout

Table 626. Input Timeout Property

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
U32	Write	No	No	Yes	No

This property specifies the time to wait for the reading of CAN frame using CAN Input. The resolution is in milliseconds.

The timeout value of -1 (infinite wait) is supported.

Output Timeout (ms)

Table 627. Output Timeout (ms) Property

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
U32	Write	No	No	Yes	No

This property specifies the time to wait for the writing of CAN frame using CAN Output. The resolution is in milliseconds.

Since this property set the wait for the pending transmission, use a timeout greater than zero. Using a timeout of 10,000 (ten seconds) is sufficient for most CAN baud rates (longer than a frame time).

The timeout value of -1 (infinite wait) is not supported, however you can use the largest positive I32 value, which provides a maximum of approximately 25 days.

Log Bus Errors

Table 628. Log Bus Errors Property

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
TF	Write	No	No	Yes	Yes

This property enables the logging of bus errors as frames that can be read using the CAN Input node.

The bus error frame is logged when the Texas Instruments TCAN4550 CAN controller detects a bus error.

- *FALSE* indicates that the bus errors will not be logged and cannot be read using the CAN Input node (default).
- *TRUE* indicates that the bus errors will be logged as frames and can be read using the CAN Input node.

The bus error frame has the following format:

Table 629. Buss Error Frame Format

Arbitration ID	0
Data Length	5
Type	6
Data	**Bytes** • 0—Comm State (see below) • 1—Transmit Error Counter • 2—Receive Error Counter • 3—Detected Bus Error • 4—X • 5—X • 6—X • 7—X

Note *X* means Reserved or *Don't Care*.

The first data byte (Comm State) indicates the current communication state of the CAN controller. The states are:

- 0: Error Active
- 1: Error Passive
- 2: Bus Off

The fourth data byte (Detected Bus Error) indicates the error detected during a frame transmission. The states are:

- 0: None
- 1: Stuff
- 2: Form
- 3: Ack
- 4: Bit 1
- 5: Bit 0
- 6: CRC

Log Transceiver Faults

Table 630. Log Transceiver Faults Property

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
TF	Write	No	No	Yes	Yes

This property enables the logging of transceiver faults as frames that can be read using the CAN Input node.

- *FALSE* indicates that the transceiver faults will not be logged as frames (default).
- *TRUE* indicates that logging is enabled and the transceiver faults will be logged as frames which can be monitored using the CAN Input node.

The transceiver fault frame has the following format:

Table 631. Transceiver Fault Frame Format

Arbitration ID	0
Data Length	1
Type	7
Data	**Bytes** • 0—Transceiver fault. 0=fault cleared, 1=fault present • 1—X • 2—X

	• 3—X • 5—X • 6—X • 7—X

Note *X* means *Reserved* or *Don't Care*.

Self Reception

Table 632. Self Reception Property

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
TF	Write	No	No	Yes	Yes

This property specifies whether to echo successfully transmitted CAN frames to be read using CAN Input.

Each reception occurs just as if the frame were received from another CAN device. For self reception to operate properly, another CAN node must receive and acknowledge each transmit. If a transmitted frame is not successfully acknowledged, it is not echoed for input.

- *FALSE* indicates transmits are not echoed (default).
- *TRUE* indicates transmits are echoed.

Single Shot Transmit

Table 633. Single Shot Transmit Property

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
TF	Write	No	No	Yes	Yes

This property specifies whether to retry failed CAN frame transmissions.

- *FALSE* indicates standard CAN behavior. Failed CAN frame transmissions are

automatically retried (default).
- *TRUE* indicates single-shot. With single-shot enabled, if a CAN frame is not transmitted successfully, the CAN controller does not retry.

Receive Error Counter

Table 634. Receive Error Counter Property

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
U8	Read	Yes	Yes	N/A	N/A

This property provides access to the CAN controller Receive Error Counter defined by the CAN standard.

Transmit Error Counter

Table 635. Transmit Error Counter Property

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
U8	Read	Yes	Yes	N/A	N/A

This property provides access to the CAN controller Transmit Error Counter defined by the CAN standard.

Transceiver Mode

Table 636. Transceiver Mode Property

Data type	Permissions	Read While Stopped?	Read While Running?	Write While Stopped?	Write While Running?
U8	Read/Write	No	Yes	No	Yes

This property sets the mode for the CAN transceiver, as well as the associated mode in the TCAN4550 CAN controller.

The values are:

- 0: Normal
- 1: Sleep

The Sleep mode places the TCAN4550 and the CAN transceiver into a low-power state. This low-power state has no effect on LabVIEW FPGA hardware. The Sleep mode can be set only when the CAN controller is running (not when stopped).

A *local wakeup* occurs when you set Transceiver Mode from Sleep back to Normal.

A *remote wakeup* occurs when a remote node transmits a CAN frame (referred to as the wakeup frame). The wakeup frame wakes up the transceiver and CAN controller chip. The wakeup frame is not received or acknowledged by the TCAN4550 CAN controller. When the wakeup frame ends, the CAN port enters Normal mode, and again receives and transmits CAN frames. If the node that transmitted the wakeup frame did not detect an acknowledgment (such as if other nodes were also waking), it retries the transmission, and the retry is received by the CAN port.

To determine when a remote wakeup occurs when the transceiver is in Sleep mode, invoke the Wait On Transceiver Wakeup method.

sbRIO-9603

sbRIO-9603 Pinouts

Figure 151. Connector J3, CAN Port

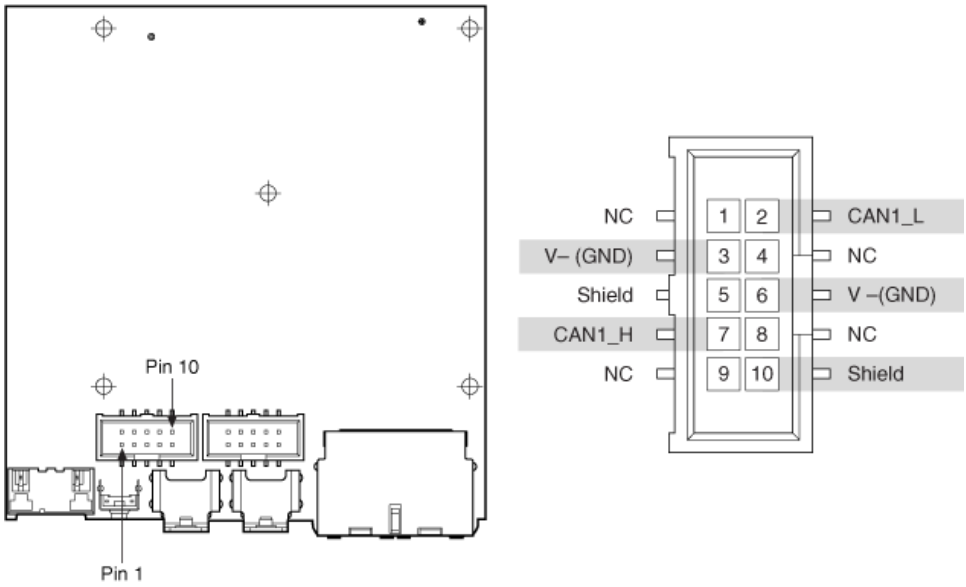
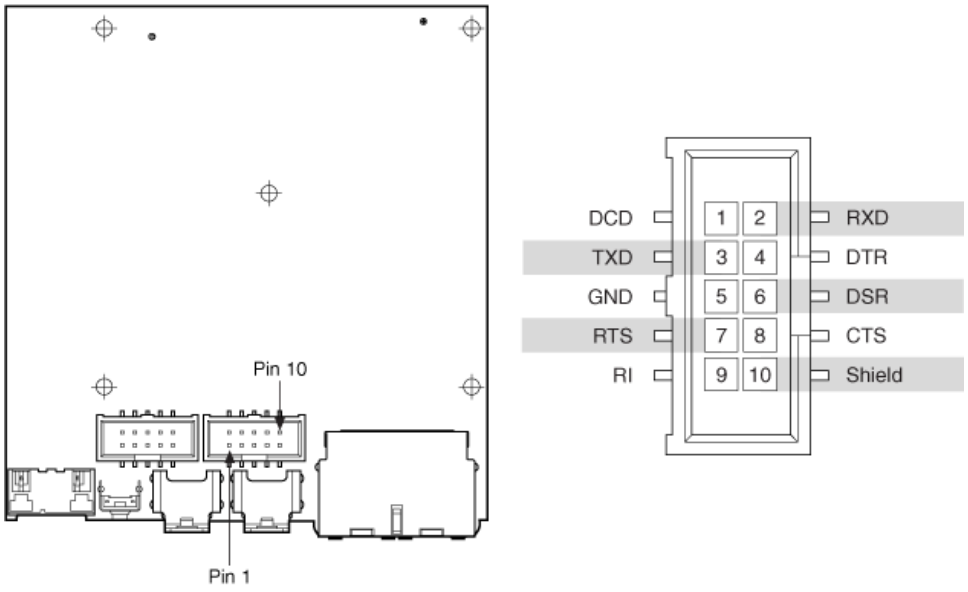


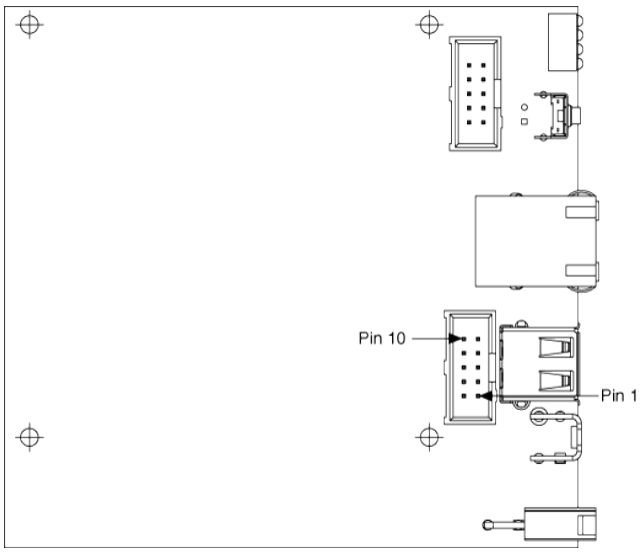
Figure 152. Connector J4, Serial Port



sbRIO-9607

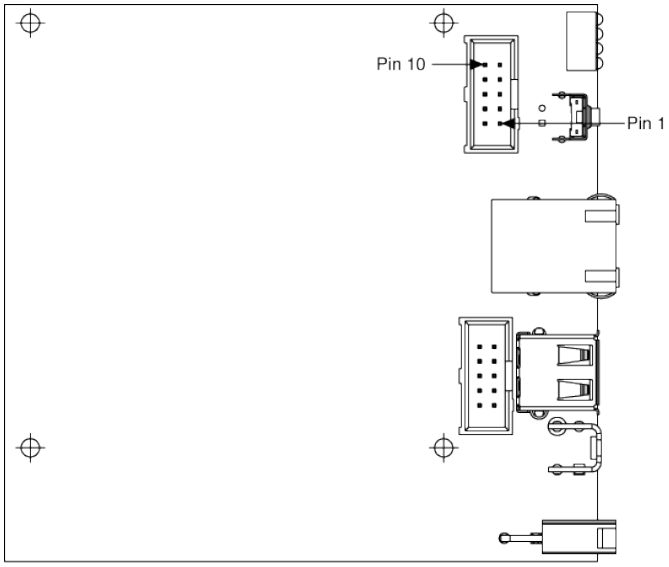
sbRIO-9607 Pinouts

Figure 153. Connector W500, CAN Port



NC	1	2	CAN1_L
V– (GND)	3	4	NC
SHIELD	5	6	V– (GND)
CAN1_H	7	8	NC
NC	9	10	SHIELD

Figure 154. Connector W501, Serial Port

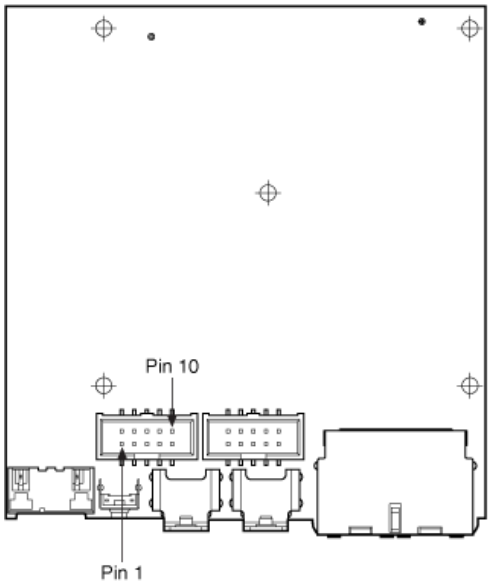


DCD	1	2	RXD
TXD	3	4	DTR
GND	5	6	DSR
RTS	7	8	CTS
RI	9	10	SHIELD

sbRIO-9608

sbRIO-9608 Pinouts

Figure 155. Connector J3, CAN Port



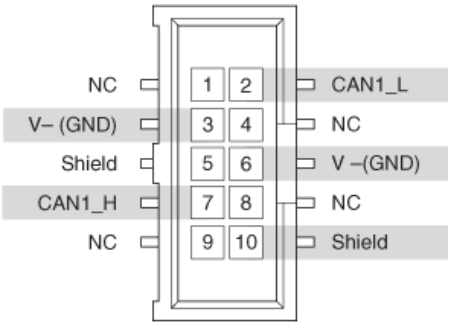
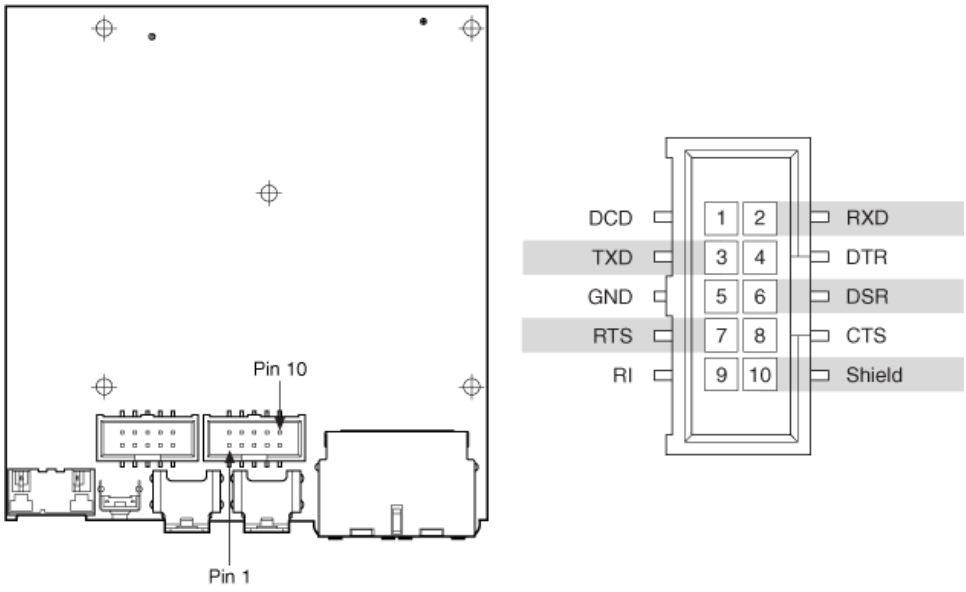


Figure 156. Connector J4, Serial Port



sbRIO-9609

sbRIO-9609 Pinouts

Figure 157. Connector J3, CAN Port

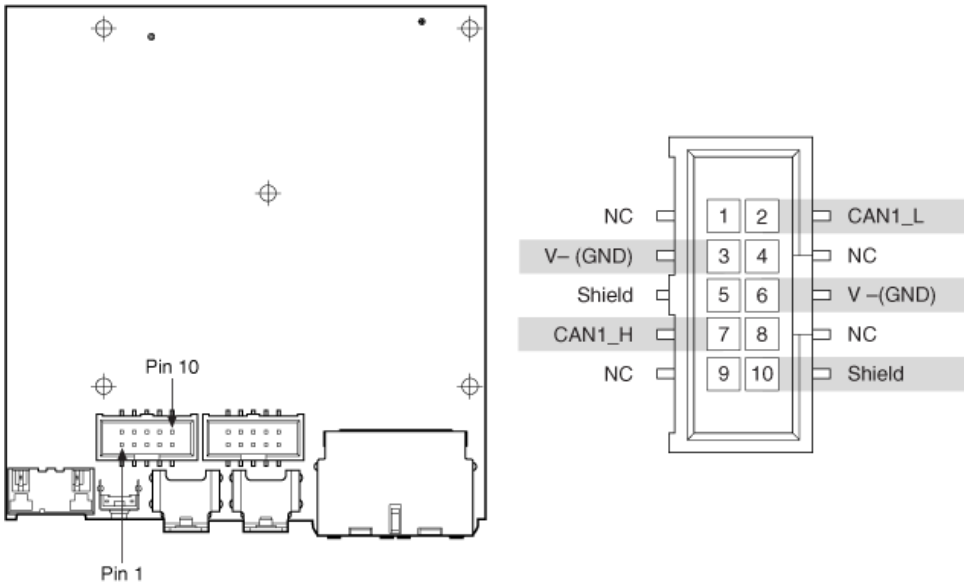
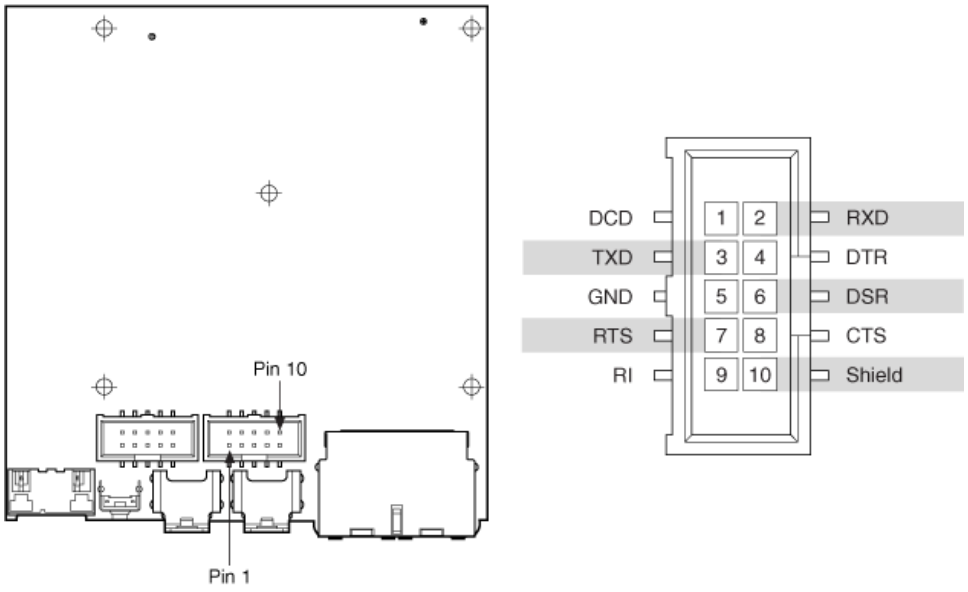


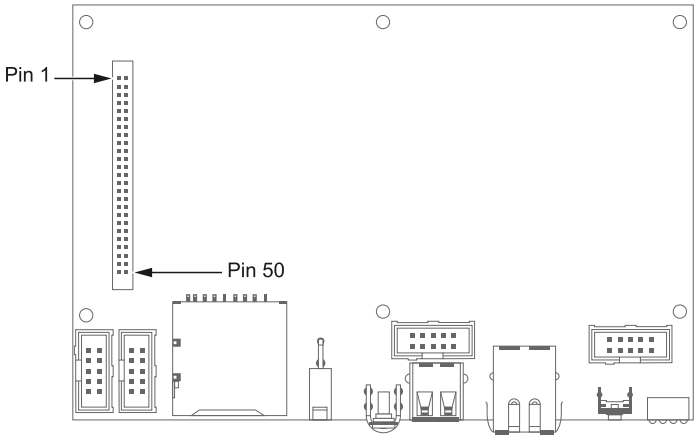
Figure 158. Connector J4, Serial Port



sbRIO-9627

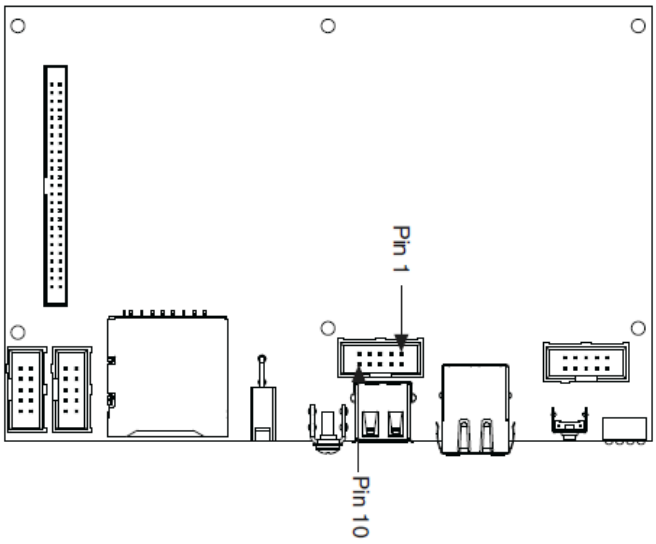
sbRIO-9627 Pinouts

Figure 159. Connector J5, MIO



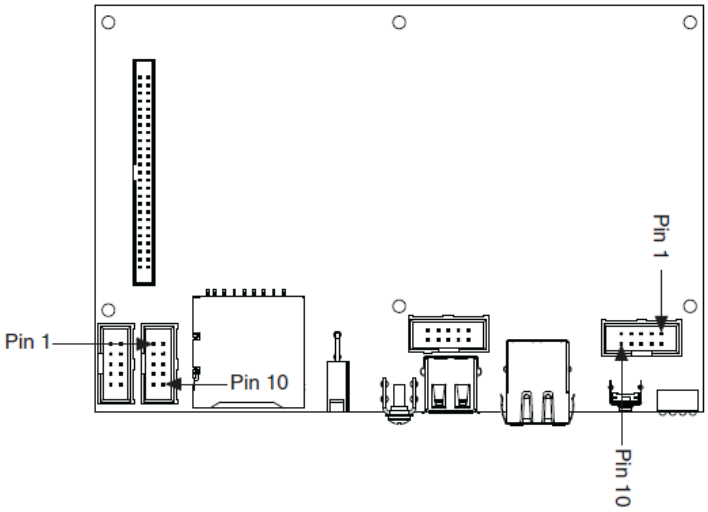
GND	1	2	AI0
AI8	3	4	GND
AI9	5	6	AI1
GND	7	8	AI2
AI10	9	10	GND
AI11	11	12	AI3
GND	13	14	AI4
AI12	15	16	GND
AI13	17	18	AI5
GND	19	20	AI6
AI14	21	22	GND
AI15	23	24	AI7
GND	25	26	GND
GND	27	28	AO0
GND	29	30	AO1
GND	31	32	AO2
GND	33	34	AO3
GND	35	36	NC
GND	37	38	NC
GND	39	40	NC
GND	41	42	NC
GND	43	44	DIO0
GND	45	46	DIO1
GND	47	48	DIO2
GND	49	50	DIO3

Figure 160. Connector W1, CAN Port



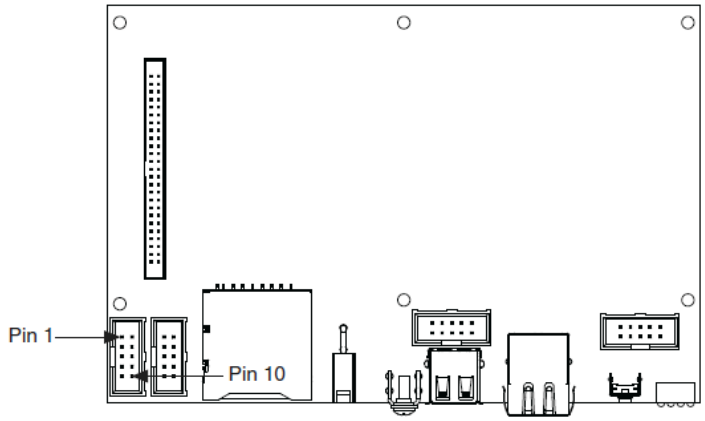
NC	1	2	CAN1_L
V– (GND)	3	4	NC
SHIELD	5	6	V– (GND)
CAN1_H	7	8	NC
NC	9	10	SHIELD

Figure 161. Connectors W2 and W4, RS-232 Serial Ports



DCD	1	2	RXD
TXD	3	4	DTR
GND	5	6	DSR
RTS	7	8	CTS
RI	9	10	SHIELD

Figure 162. Connector W3, RS-485 Serial Port



GND	1	2	NC
NC	3	4	RXD+
RXD–	5	6	NC
NC	7	8	TXD+
TXD–	9	10	SHIELD

sbRIO-9627 (FPGA Interface)

Use an FPGA I/O Node configured for reading and writing with this device.

Use the FPGA I/O Node to access the following terminals for this device.

Table 637. Terminals in Software

Terminal	Description
Connector0/AI*x*	Analog input channel *x*, where *x* is the number of the channel. The sbRIO-9627 has AI channels 0 to 15.
Connector0/AO*x*	Analog output channel *x*, where *x* is the number of the channel. The sbRIO-9627 has AO channels 0 to 3.
Connector0/DIO*x*	Digital input/output channel *x*, where *x* is the number of the channel. The sbRIO-9627 has channels 0 to 3. Use the FPGA I/O Node, the Set Output Data method or Set Output Enable method to access this channel.

Arbitration

Configure the arbitration settings for the DIO channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 638. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Set Output Enable	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.

Module Methods

This device does not support any module methods.

Module Properties

Use the FPGA I/O Property Node to access the following properties for this device.

Table 639. Module Properties

Property	Description
Terminal Mode	Sets the terminal mode for a channel as RSE (referenced single-ended) or DIFF (differential). This property overwrites the value you configure in the **FPGA I/O Properties** dialog box.
Voltage Range	Sets the input range for a channel as ±10 V, ±5 V, ±2 V, or ±1 V. This property overwrites the value you configure in the **FPGA I/O Properties** dialog box.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop for digital I/O only. To configure the number of output synchronizing registers or input synchronizing registers for the channels on this device, use the Advanced Code Generation page of the FPGA I/O Properties dialog box.

sbRIO-9628

sbRIO-9628 Pinouts

Figure 163. Connector J2, MIO

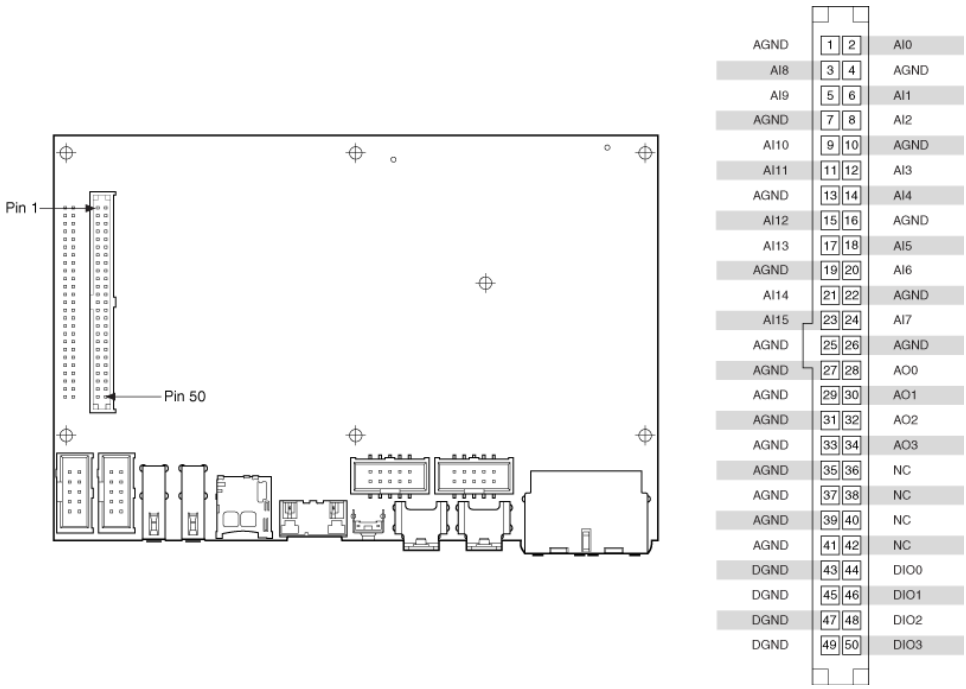


Figure 164. Connector J3, CAN Port

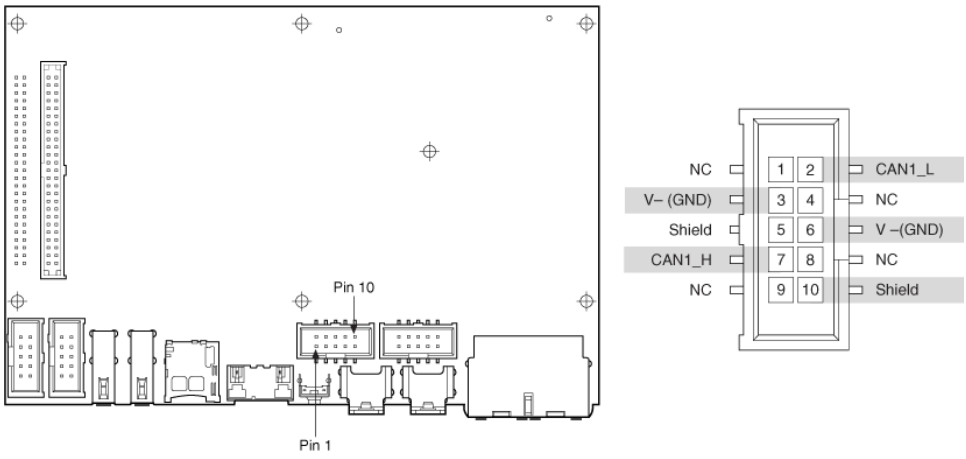


Figure 165. Connectors J4 and J6, RS-232 Serial Ports

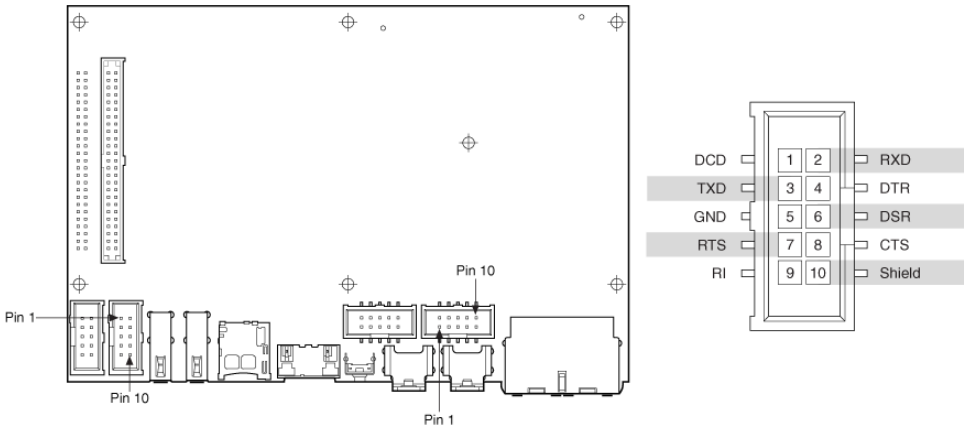
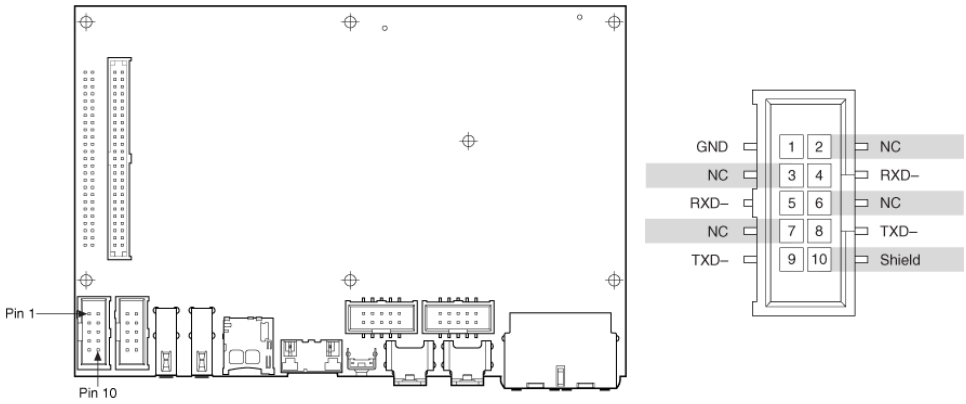


Figure 166. Connector J5, RS-485 Serial Port



sbRIO-9628 (FPGA Interface)

Use an FPGA I/O Node configured for reading and writing with this device.

Use the FPGA I/O Node to access the following terminals for this device.

Table 640. Terminals in Software

Terminal	Description
Conn0_AI/AI*x*	Analog input channel *x*, where *x* is the number of the channel. The sbRIO-9628 has AI channels 0 to 15.
Conn0_AO/AO*x*	Analog output channel *x*, where *x* is the number of the channel. The sbRIO-9628 has AO channels 0 to 3.
Conn0_DIO0-3/DIO*x*	Digital input/output channel *x*, where *x* is the channel number. The sbRIO-9628 has DIO channels 0 to 3.
Conn0_DIO0-3/DIO3:0	Digital port consisting of channels 0 through 3. Channel 3 signifies the MSB and channel 0 signifies the LSB.

Arbitration

You cannot configure arbitration settings for analog input and analog output channels of this device. Analog input and analog output channels on this device only support the Arbitrate if Multiple Requestors Only option for arbitration.

Configure the arbitration settings for digital output channels on this device in the Advanced Code Generation page on the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

Digital input channels on this device only support the Never Arbitrate option for arbitration. You cannot configure arbitration settings for digital input channels on this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 641. I/O Methods

Method	I/O Type	Description
Set Output Data	DIO	Refer to the FPGA I/O Method Node topic for a description of this method.
Set Output Enable	DIO	Sets the line direction of the digital channel or the digital port. Refer to the FPGA I/O Method Node topic for a description of this method.

Module Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 642. Module Methods

Method	I/O Type	Description
Check Status	DIO	Returns a Boolean value that indicates whether the DIO module is ready for I/O operations.

I/O Properties

Use the FPGA I/O Property Node to access the following properties for this device.

Table 643. I/O Properties

Property	I/O Type	Description
Terminal Mode	AI	Sets the terminal mode for a channel as RSE (referenced single-ended) or DIFF (differential). This property overwrites the value you configure in the **Module Properties** dialog box. You cannot configure channels 8 through 15 to DIFF mode.
Voltage Range	AI	Sets the input range for a channel as ±10 V, ±5 V, ±2 V, or ±1 V. This property overwrites the value you configure in the **Module Properties** dialog box.
LSB Weight	AO	Returns the LSB weight in nV/LSB for the channel. Use this value to convert AO data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
Offset	AO	Returns the calibration offset in nV for the channel. Use this value to convert AO data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following properties for this device.

Table 644. Module Properties

Property	I/O Type	Description
LSB Weight (±10 V range)	AI	Returns the LSB weight in pV/LSB for the ±10 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
LSB Weight (±5 V range)	AI	Returns the LSB weight in pV/LSB for the ±5 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
LSB Weight (±2 V range)	AI	Returns the LSB weight in pV/LSB for the ±2 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
LSB Weight (±1 V range)	AI	Returns the LSB weight in pV/LSB for the ±1 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.

Property	I/O Type	Description
Offset (±10 V range)	AI	Returns the calibration offset in nV for the ±10 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
Offset (±5 V range)	AI	Returns the calibration offset in nV for the ±5 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
Offset (±2 V range)	AI	Returns the calibration offset in nV for the ±2 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
Offset (±1 V range)	AI	Returns the calibration offset in nV for the ±1 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop for digital I/O only. To configure the number of output synchronizing registers or input synchronizing registers for the channels on this device, use the Advanced Code Generation page of the FPGA I/O Properties dialog box

The Set Output Enable method node is not supported in the single-cycle Timed Loop. When writing to a DIO I/O node in a single-cycle Timed Loop, the I/O node will not change the DIO line directions. To set the line directions to output when writing to the DIO channels in a single-cycle Timed Loop, either configure the default line directions to output in the **Module Properties** dialog box, or use the Set Output Enable method node outside a single-cycle Timed Loop.

This device supports the Number of Synchronizing Registers for Output Data synchronizing register option when used in SCTL output. This option supports the same functionality as the Number of Synchronizing Registers for Output Data option described in the Advanced Code Generation FPGA I/O Properties Page (FPGA Module) topic, with the exception that you can use this option only in a single-cycle Timed Loop. Implement either 0 or 1 synchronizing registers inside the single-cycle Timed Loop. Note that if you configure 0 synchronizing registers outside of the single-cycle Timed Loop, the FPGA VI implements 1 synchronizing register by default.

When the device is within a single-cycle Timed Loop, it must be ready to perform digital I/O before a loop containing digital I/O starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready. The module also might ignore or delay digital output operations if it is not ready.

Converting Analog Input Values for the sbRIO-9628/9629/9638 (FPGA Interface)

To configure the FPGA I/O Node to return calibrated, fixed-point data for the onboard analog input module, set the **Calibration Mode** to Calibrated in the Module Properties dialog box for the module.

When **Calibration Mode** is set to **Raw**, the FPGA I/O Node returns uncalibrated, binary values. Convert the binary values to calibrated volts by applying the calibration constants. The conversion of binary data to calibrated data can be done in the host VI.

Using an Equation to Convert and Calibrate Values

Use the following equation in the host VI to convert binary analog input values to calibrated volts:

$$Volts = (Binary\ Value \times LSB\ Weight - Offset)$$

where

- *Binary Value* is the signed value returned by the FPGA I/O Node.
- *LSB Weight* is the value returned by the LSB Weight property.
- *Offset* is the value returned by the Offset property.
- The *LSB Weight* property is returned in units of pV/LSB. The *Offset* property is returned in units of nV.

To convert to calibrated volts, use the FPGA I/O Property Node to read the LSB Weight and Offset properties for the desired input range. If you do not want to read the LSB Weight and Offset values from the module, you can convert to uncalibrated engineering units by using the following typical values for *Offset* and *LSB Weight*:

$$Offset = 0\ V$$

The typical *LSB Weight* depends on the input range. The following table lists the LSB weight value for each input range.

Table 645. Typical LSB Weight

Input Range	Typical LSB Weight
±10 V	320 µV/LSB
±5 V	160 µV/LSB
±2 V	64 µV/LSB
±1 V	32 µV/LSB

Converting Analog Output Values for the sbRIO-9628/9629/9638 (FPGA Interface)

To configure the FPGA I/O Node to accept calibrated, fixed-point data in units of volts for the onboard analog output module, set the **Calibration Mode** to Calibrated in the Module Properties dialog box for the module.

When **Calibration Mode** is set to **Raw**, the FPGA I/O Node accepts only binary values. Convert output voltage values to binary values using the calibration constants before writing them to the FPGA I/O node. The conversion of voltage values to binary values can be done in the host VI.

Using an Equation to Convert Voltage to Binary

Use the following equation in the host VI to convert the analog output values to binary values:

```
Binary Value = (Voltage Value - Offset) ÷ LSB Weight
```

where

- *Binary Value* is the value you write to the FPGA I/O Node.
- Voltage Value is the voltage in V that you want the channel to output.
- Offset is the value returned by the Offset property.
- LSB Weight is the value returned by the LSB Weight property.

The LSB Weight property is returned in units of nV/LSB. The Offset property is returned in units of nV.
NI recommends using calibrated values for analog output. To convert calibrated voltage values, use the FPGA I/O Property Node to read theLSB Weight and Offset properties. If you do not want to read the LSB Weight and Offset values from the module, you can convert uncalibrated voltage values by using the following typical values for Offset and LSB Weight:

- `Offset = 0 V`
- `LSB Weight = 320 µV/LSB`

sbRIO-9629

sbRIO-9629 Pinouts

Figure 167. Connector J2, MIO

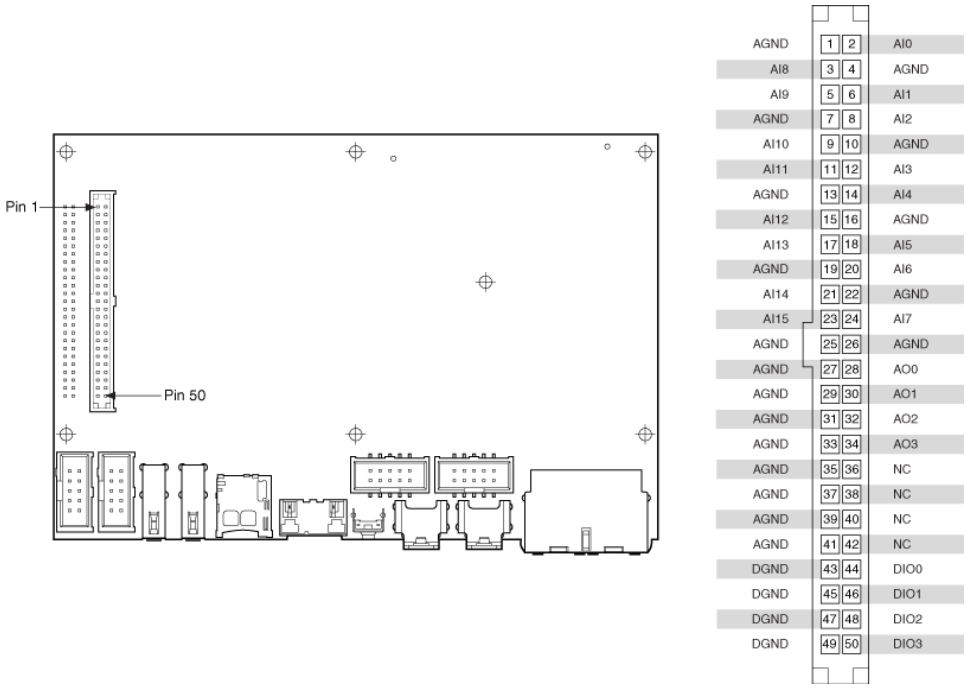


Figure 168. Connector J3, CAN Port

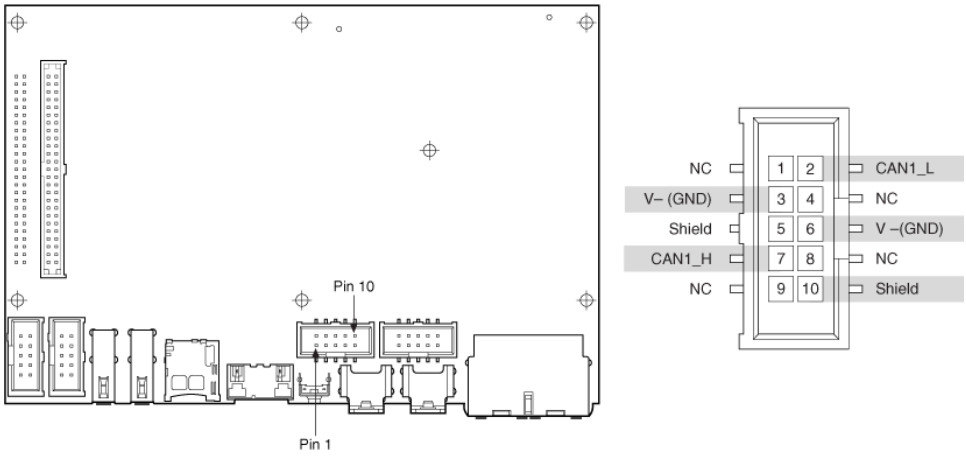


Figure 169. Connectors J4 and J6, RS-232 Serial Ports

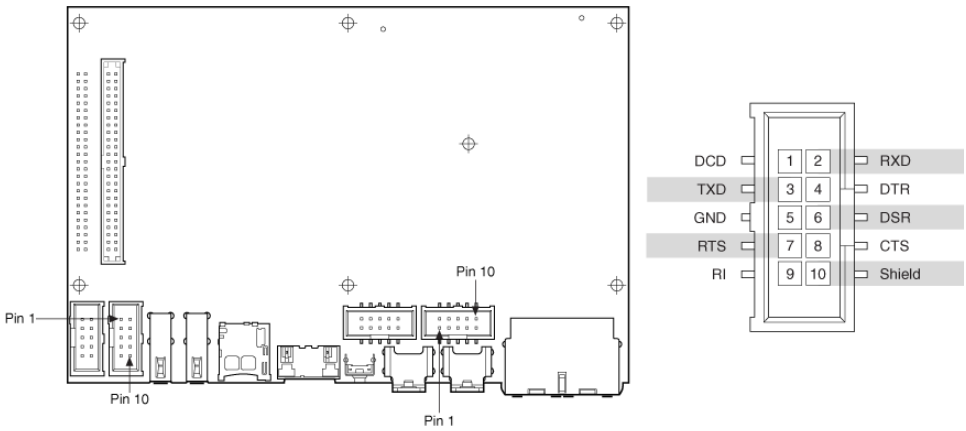
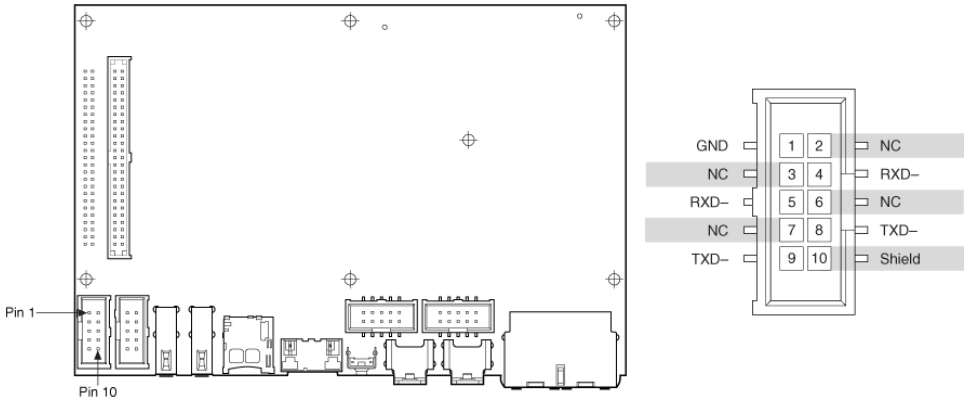


Figure 170. Connector J5, RS-485 Serial Port



sbRIO-9629 (FPGA Interface)

Use an FPGA I/O Node configured for reading and writing with this device.

Use the FPGA I/O Node to access the following terminals for this device.

Table 646. Terminals in Software

Terminal	Description
Conn0_AI/AI*x*	Analog input channel *x*, where *x* is the number of the channel. The sbRIO-9629 has AI channels 0 to 15.
Conn0_AO/AO*x*	Analog output channel *x*, where *x* is the number of the channel. The sbRIO-9629 has AO channels 0 to 3.
Conn0_DIO0-3/DIO*x*	Digital input/output channel *x*, where *x* is the channel number. The sbRIO-9629 has DIO channels 0 to 3.
Conn0_DIO0-3/DIO3:0	Digital port consisting of channels 0 through 3. Channel 3 signifies the MSB and channel 0 signifies the LSB.

Arbitration

You cannot configure arbitration settings for analog input and analog output channels of this device. Analog input and analog output channels on this device only support the Arbitrate if Multiple Requestors Only option for arbitration.

Configure the arbitration settings for digital output channels on this device in the Advanced Code Generation page on the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

Digital input channels on this device only support the Never Arbitrate option for arbitration. You cannot configure arbitration settings for digital input channels on this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 647. I/O Methods

Method	I/O Type	Description
Set Output Data	DIO	Refer to the FPGA I/O Method Node topic for a description of this method.
Set	DIO	Sets the line direction of the digital channel or the digital port. Refer to the FPGA

Method	I/O Type	Description
Output Enable		I/O Method Node topic for a description of this method.

Module Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 648. Module Methods

Method	I/O Type	Description
Check Status	DIO	Returns a Boolean value that indicates whether the DIO module is ready for I/O operations.

I/O Properties

Use the FPGA I/O Property Node to access the following properties for this device.

Table 649. I/O Properties

Property	I/O Type	Description
Terminal Mode	AI	Sets the terminal mode for a channel as RSE (referenced single-ended) or DIFF (differential). This property overwrites the value you configure in the **Module Properties** dialog box. You cannot configure channels 8 through 15 to DIFF mode.
Voltage Range	AI	Sets the input range for a channel as ±10 V, ±5 V, ±2 V, or ±1 V. This property overwrites the value you configure in the **Module Properties** dialog box.
LSB Weight	AO	Returns the LSB weight in nV/LSB for the channel. Use this value to convert AO data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
Offset	AO	Returns the calibration offset in nV for the channel. Use this value to convert AO data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following properties for this device.

Table 650. Module Properties

Property	I/O Type	Description
LSB Weight (±10 V range)	AI	Returns the LSB weight in pV/LSB for the ±10 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
LSB Weight (±5 V range)	AI	Returns the LSB weight in pV/LSB for the ±5 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
LSB Weight (±2 V range)	AI	Returns the LSB weight in pV/LSB for the ±2 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
LSB Weight (±1 V range)	AI	Returns the LSB weight in pV/LSB for the ±1 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
Offset (±10 V range)	AI	Returns the calibration offset in nV for the ±10 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
Offset (±5 V range)	AI	Returns the calibration offset in nV for the ±5 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
Offset (±2 V range)	AI	Returns the calibration offset in nV for the ±2 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
Offset (±1 V range)	AI	Returns the calibration offset in nV for the ±1 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop for digital I/O only. To configure the number of output synchronizing registers or input synchronizing registers for the channels on this device, use the Advanced Code Generation page of the FPGA I/O Properties dialog box

The Set Output Enable method node is not supported in the single-cycle Timed Loop. When writing to a DIO I/O node in a single-cycle Timed Loop, the I/O node will not

change the DIO line directions. To set the line directions to output when writing to the DIO channels in a single-cycle Timed Loop, either configure the default line directions to output in the **Module Properties** dialog box, or use the Set Output Enable method node outside a single-cycle Timed Loop.

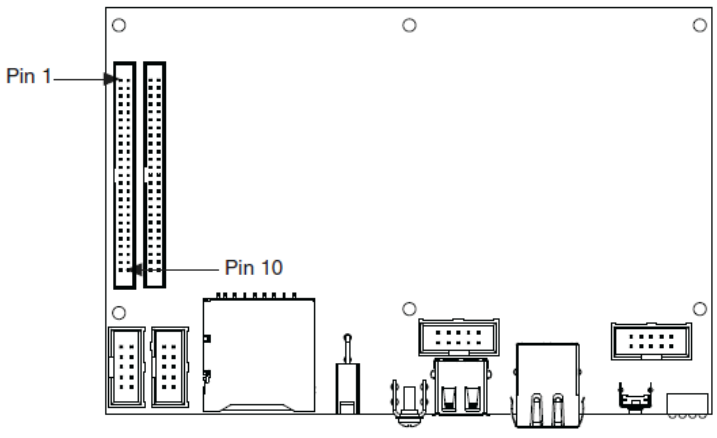
This device supports the Number of Synchronizing Registers for Output Data synchronizing register option when used in SCTL output. This option supports the same functionality as the Number of Synchronizing Registers for Output Data option described in the Advanced Code Generation FPGA I/O Properties Page (FPGA Module) topic, with the exception that you can use this option only in a single-cycle Timed Loop. Implement either 0 or 1 synchronizing registers inside the single-cycle Timed Loop. Note that if you configure 0 synchronizing registers outside of the single-cycle Timed Loop, the FPGA VI implements 1 synchronizing register by default.

When the device is within a single-cycle Timed Loop, it must be ready to perform digital I/O before a loop containing digital I/O starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input operations return invalid data if the module is not ready. The module also might ignore or delay digital output operations if it is not ready.

sbRIO-9637

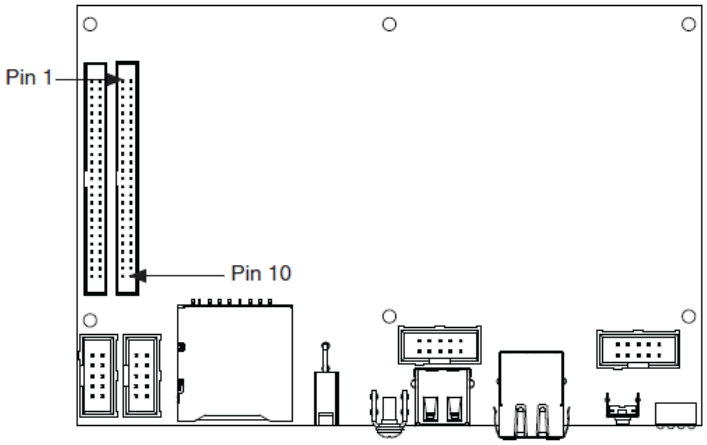
sbRIO-9637 Pinouts

Figure 171. Connector J4, DIO



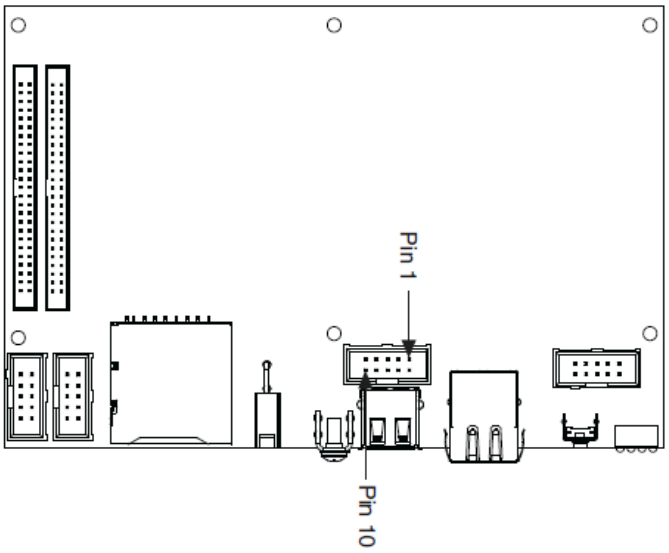
GND	1	2	DIO4
GND	3	4	DIO5
GND	5	6	DIO6
GND	7	8	DIO7
GND	9	10	DIO8
GND	11	12	DIO9
GND	13	14	DIO10
GND	15	16	DIO11
GND	17	18	DIO12
GND	19	20	DIO13
GND	21	22	DIO14
GND	23	24	DIO15
GND	25	26	DIO16
GND	27	28	DIO17
GND	29	30	DIO18
GND	31	32	DIO19
GND	33	34	DIO20
GND	35	36	DIO21
GND	37	38	DIO22
GND	39	40	DIO23
GND	41	42	DIO24
GND	43	44	DIO25
GND	45	46	DIO26
GND	47	48	DIO27
5 V	49	50	5 V

Figure 172. Connector J5, MIO



GND	1	2	AI0
AI8	3	4	GND
AI9	5	6	AI1
GND	7	8	AI2
AI10	9	10	GND
AI11	11	12	AI13
GND	13	14	AI14
AI12	15	16	GND
AI13	17	18	AI5
GND	19	20	AI6
AI14	21	22	GND
AI15	23	24	AI7
GND	25	26	GND
GND	27	28	AO0
GND	29	30	AO1
GND	31	32	AO2
GND	33	34	AO3
GND	35	36	NC
GND	37	38	NC
GND	39	40	NC
GND	41	42	NC
GND	43	44	DIO0
GND	45	46	DIO1
GND	47	48	DIO2
GND	49	50	DIO3

Figure 173. Connector W1, CAN Port



NC	1	2	CAN1_L
V– (GND)	3	4	NC
SHIELD	5	6	V– (GND)
CAN1_H	7	8	NC
NC	9	10	SHIELD

Figure 174. Connectors W2 and W4, RS-232 Serial Ports

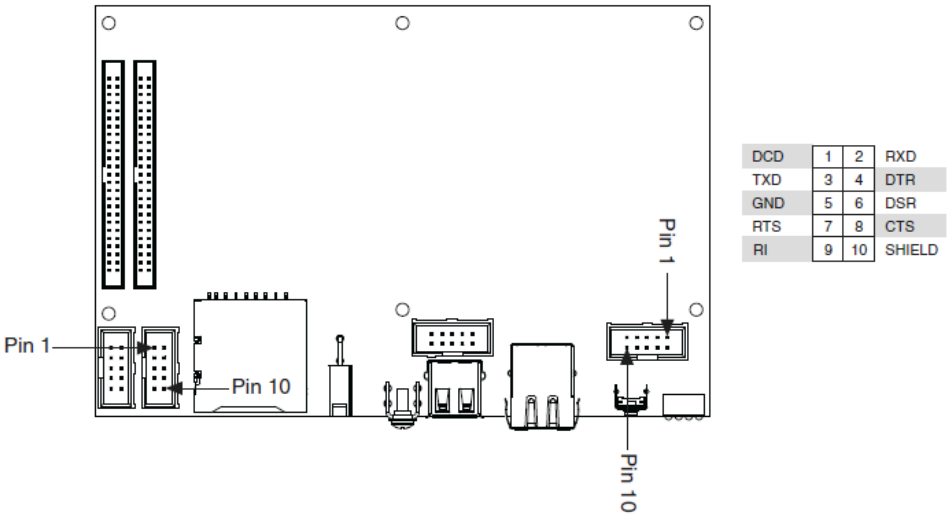
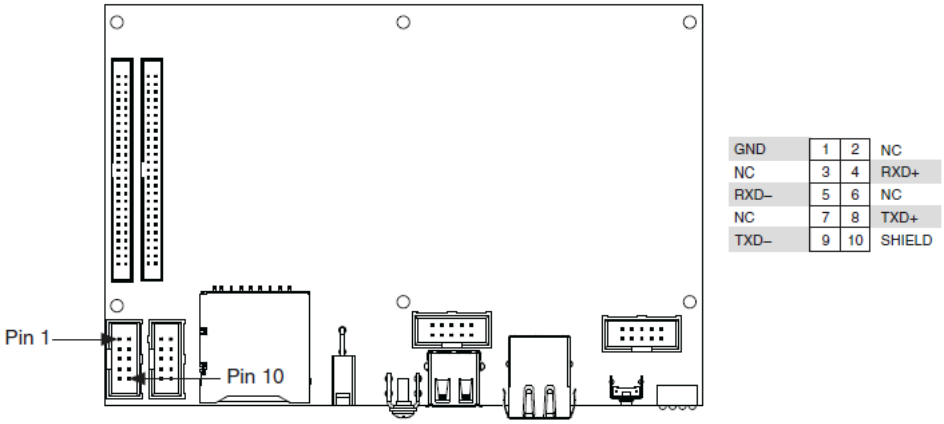


Figure 175. Connector W3, RS-485 Serial Port



sbRIO-9637 (FPGA Interface)

Use an FPGA I/O Node configured for reading and writing with this device.

Use the FPGA I/O Node to access the following terminals for this device.

Table 651. Terminals in Software

Terminal	Description
Connector0/AI*x*	Analog input channel *x*, where *x* is the number of the channel. The sbRIO-9637 has AI channels 0 to 15.
Connector0/AO*x*	Analog output channel *x*, where *x* is the number of the channel. The sbRIO-9637 has AO channels 0 to 3.

Terminal	Description
Connector*x*/DIO*y*	Digital input/output channel *y* on connector *x*, where *y* is the channel number and *x* is the connector number. The sbRIO-9637 has channels 0 to 27. Use the FPGA I/O Node, the Set Output Data method, or the Set Output Enable method to access this channel.

Arbitration

Configure the arbitration settings for the DIO channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 652. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Set Output Enable	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.

Module Methods

This device does not support any module methods.

Module Properties

Use the FPGA I/O Property Node to access the following properties for this device.

Table 653. Module Properties

Property	Description
Terminal Mode	Sets the terminal mode for a channel as RSE (referenced single-ended) or DIFF (differential). This property overwrites the value you configure in the **FPGA I/O Properties** dialog box.

Property	Description
Voltage Range	Sets the input range for a channel as ±10 V, ±5 V, ±2 V, or ±1 V. This property overwrites the value you configure in the **FPGA I/O Properties** dialog box.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop for digital I/O only. To configure the number of output synchronizing registers or input synchronizing registers for the channels on this device, use the Advanced Code Generation page of the FPGA I/O Properties dialog box.

sbRIO-9638

sbRIO-9638 Pinouts

Figure 176. Connector J1, DIO

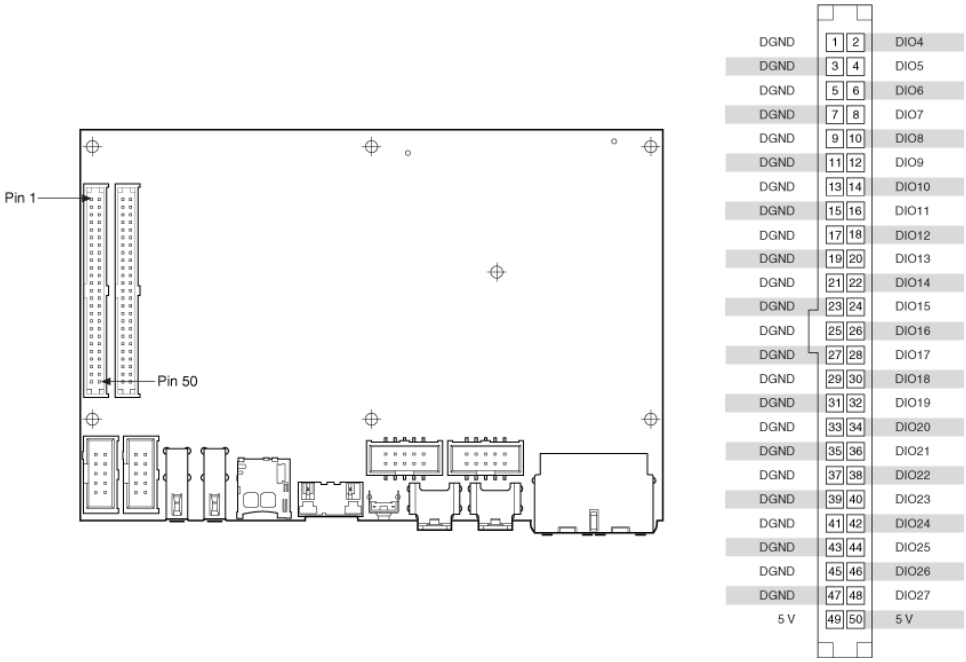


Figure 177. Connector J2, MIO

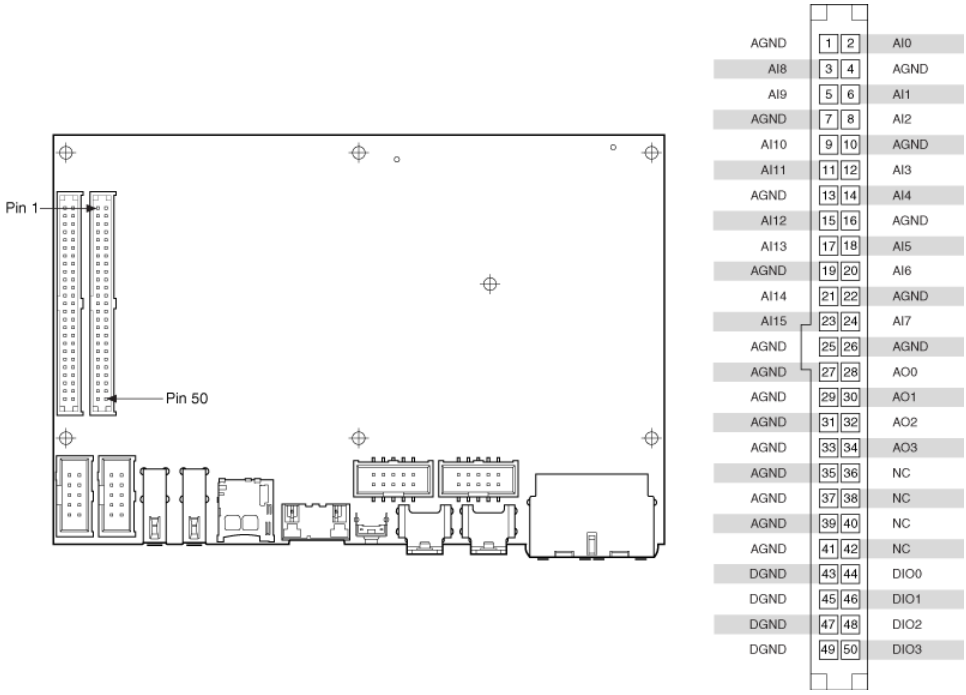


Figure 178. Connector J3, CAN Port

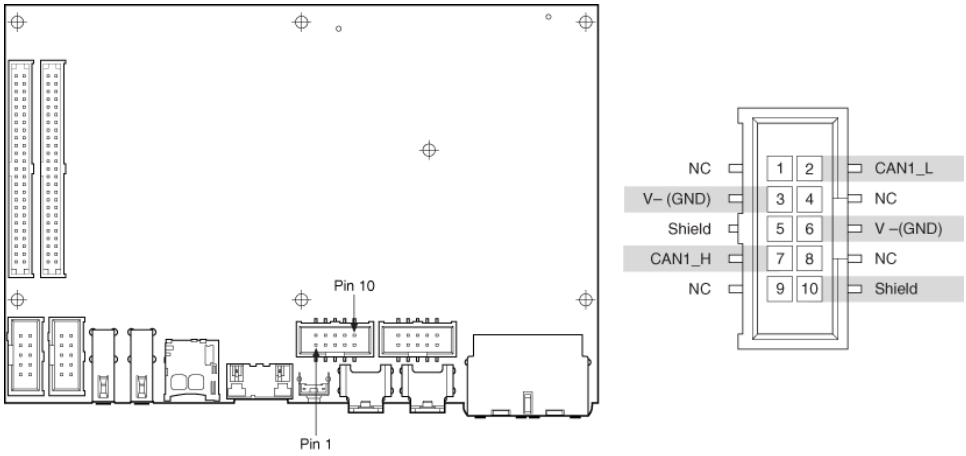


Figure 179. Connectors J4 and J6, RS-232 Serial Ports

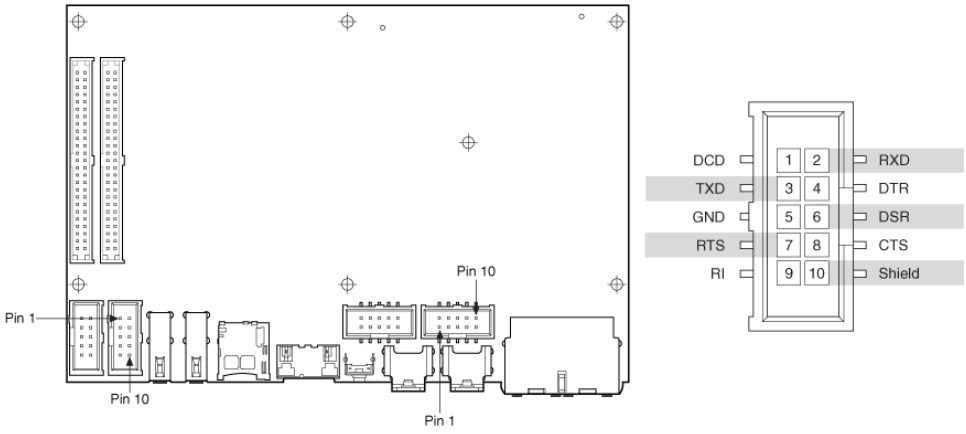
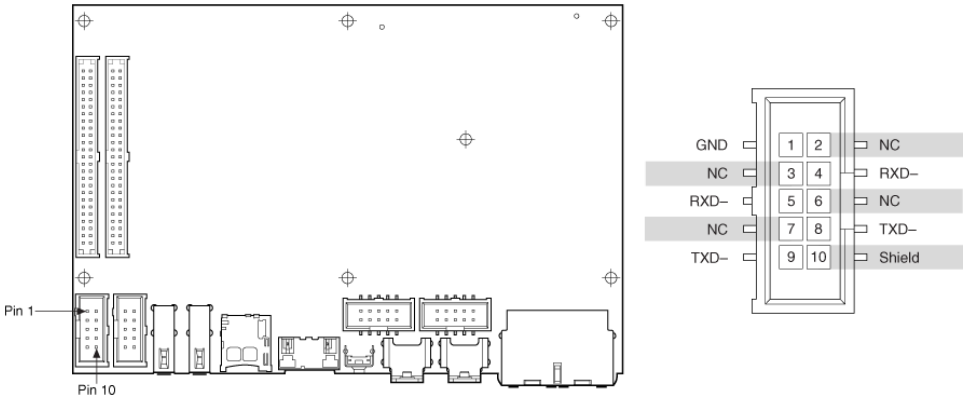


Figure 180. Connector J5, RS-485 Serial Port



sbRIO-9638 (FPGA Interface)

Use an FPGA I/O Node configured for reading and writing with this device.

Use the FPGA I/O Node to access the following terminals for this device.

Table 654. Terminals in Software

Terminal	Description
Conn0_AI/AI*x*	Analog input channel *x*, where *x* is the number of the channel. The sbRIO-9638 has AI channels 0 to 15.
Conn0_AO/AO*x*	Analog output channel *x*, where *x* is the number of the channel. The sbRIO-9638 has AO channels 0 to 3.
Conn*w*_DIO*x*-*y*/DIO*z*	Digital input/output channel *z* on connector *w*, where *z* is the channel number, *w* is the connector number, and *x* and *y* specify the module that contains the channel. The sbRIO-9638 has DIO channels 0 to 27 organized into 4 separate modules.
Conn*w*_DIO*x*-*y*/DIO*y*:*x*	Digital port consisting of channels *x* and *y* on connector *w*. Channel *y* signifies the MSB and channel *x* signifies the LSB.

Arbitration

You cannot configure arbitration settings for analog input and analog output channels of this device. Analog input and analog output channels on this device only support the Arbitrate if Multiple Requestors Only option for arbitration.

Configure the arbitration settings for digital output channels on this device in the

Advanced Code Generation page on the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

Digital input channels on this device only support the Never Arbitrate option for arbitration. You cannot configure arbitration settings for digital input channels on this device.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 655. I/O Methods

Method	I/O Type	Description
Set Output Data	DIO	Refer to the FPGA I/O Method Node topic for a description of this method.
Set Output Enable	DIO	Sets the line direction of the digital channel or the digital port. Refer to the FPGA I/O Method Node topic for a description of this method.

Module Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 656. Module Methods

Method	I/O Type	Description
Check Status	DIO	Returns a Boolean value that indicates whether the DIO module is ready for I/O operations.

I/O Properties

Use the FPGA I/O Property Node to access the following properties for this device.

Table 657. I/O Properties

Property	I/O Type	Description
Terminal Mode	AI	Sets the terminal mode for a channel as RSE (referenced single-ended) or DIFF (differential). This property overwrites the value you configure in the **Module Properties** dialog box. You cannot configure channels 8 through 15 to DIFF mode.
Voltage Range	AI	Sets the input range for a channel as ±10 V, ±5 V, ±2 V, or ±1 V. This property overwrites the value you configure in the **Module Properties** dialog box.
LSB Weight	AO	Returns the LSB weight in nV/LSB for the channel. Use this value to convert AO data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
Offset	AO	Returns the calibration offset in nV for the channel. Use this value to convert AO data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.

Module Properties

Use the FPGA I/O Property Node to access the following properties for this device.

Table 658. Module Properties

Property	I/O Type	Description
LSB Weight (±10 V range)	AI	Returns the LSB weight in pV/LSB for the ±10 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
LSB Weight (±5 V range)	AI	Returns the LSB weight in pV/LSB for the ±5 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
LSB Weight (±2 V range)	AI	Returns the LSB weight in pV/LSB for the ±2 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
LSB Weight (±1 V range)	AI	Returns the LSB weight in pV/LSB for the ±1 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
Offset (±10 V range)	AI	Returns the calibration offset in nV for the ±10 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
Offset (±5 V	AI	Returns the calibration offset in nV for the ±5 V range. Use this value to convert

Property	I/O Type	Description
range)		AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
Offset (±2 V range)	AI	Returns the calibration offset in nV for the ±2 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.
Offset (±1 V range)	AI	Returns the calibration offset in nV for the ±1 V range. Use this value to convert AI data if you set the **Calibration Mode** to **Raw** in the **Module Properties** dialog box.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop for digital I/O only. To configure the number of output synchronizing registers or input synchronizing registers for the channels on this device, use the Advanced Code Generation page of the FPGA I/O Properties dialog box

The Set Output Enable method node is not supported in the single-cycle Timed Loop. When writing to a DIO I/O node in a single-cycle Timed Loop, the I/O node will not change the DIO line directions. To set the line directions to output when writing to the DIO channels in a single-cycle Timed Loop, either configure the default line directions to output in the **Module Properties** dialog box, or use the Set Output Enable method node outside a single-cycle Timed Loop.

This device supports the Number of Synchronizing Registers for Output Data synchronizing register option when used in SCTL output. This option supports the same functionality as the Number of Synchronizing Registers for Output Data option described in the Advanced Code Generation FPGA I/O Properties Page (FPGA Module) topic, with the exception that you can use this option only in a single-cycle Timed Loop. Implement either 0 or 1 synchronizing registers inside the single-cycle Timed Loop. Note that if you configure 0 synchronizing registers outside of the single-cycle Timed Loop, the FPGA VI implements 1 synchronizing register by default.

When the device is within a single-cycle Timed Loop, it must be ready to perform digital I/O before a loop containing digital I/O starts. Poll the **Ready** output of the Check Status method to determine whether the module is ready. Digital input

operations return invalid data if the module is not ready. The module also might ignore or delay digital output operations if it is not ready.

Converting Analog Input Values for the sbRIO-9628/9629/9638 (FPGA Interface)

To configure the FPGA I/O Node to return calibrated, fixed-point data for the onboard analog input module, set the **Calibration Mode** to Calibrated in the Module Properties dialog box for the module.

When **Calibration Mode** is set to **Raw**, the FPGA I/O Node returns uncalibrated, binary values. Convert the binary values to calibrated volts by applying the calibration constants. The conversion of binary data to calibrated data can be done in the host VI.

Using an Equation to Convert and Calibrate Values

Use the following equation in the host VI to convert binary analog input values to calibrated volts:

$$Volts = (Binary\ Value \times LSB\ Weight - Offset)$$

where

- *Binary Value* is the signed value returned by the FPGA I/O Node.
- *LSB Weight* is the value returned by the LSB Weight property.
- *Offset* is the value returned by the Offset property.
- The *LSB Weight* property is returned in units of pV/LSB. The *Offset* property is returned in units of nV.

To convert to calibrated volts, use the FPGA I/O Property Node to read the LSB Weight and Offset properties for the desired input range. If you do not want to read the LSB Weight and Offset values from the module, you can convert to uncalibrated engineering units by using the following typical values for *Offset* and *LSB Weight*:

$$Offset = 0\ V$$

The typical *LSB Weight* depends on the input range. The following table lists the LSB weight value for each input range.

Table 659. Typical LSB Weight

Input Range	Typical LSB Weight
±10 V	320 µV/LSB
±5 V	160 µV/LSB
±2 V	64 µV/LSB
±1 V	32 µV/LSB

Converting Analog Output Values for the sbRIO-9628/9629/9638 (FPGA Interface)

To configure the FPGA I/O Node to accept calibrated, fixed-point data in units of volts for the onboard analog output module, set the **Calibration Mode** to Calibrated in the Module Properties dialog box for the module.

When **Calibration Mode** is set to **Raw**, the FPGA I/O Node accepts only binary values. Convert output voltage values to binary values using the calibration constants before writing them to the FPGA I/O node. The conversion of voltage values to binary values can be done in the host VI.

Using an Equation to Convert Voltage to Binary

Use the following equation in the host VI to convert the analog output values to binary values:

```
Binary Value = (Voltage Value - Offset) ÷ LSB Weight
```

where

- *Binary Value* is the value you write to the FPGA I/O Node.
- Voltage Value is the voltage in V that you want the channel to output.
- Offset is the value returned by the Offset property.
- LSB Weight is the value returned by the LSB Weight property.

The LSB Weight property is returned in units of nV/LSB. The Offset property is returned in units of nV.

NI recommends using calibrated values for analog output. To convert calibrated voltage values, use the FPGA I/O Property Node to read theLSB Weight and Offset properties. If you do not want to read the LSB Weight and Offset values from the module, you can convert uncalibrated voltage values by using the following typical values for Offset and LSB Weight:

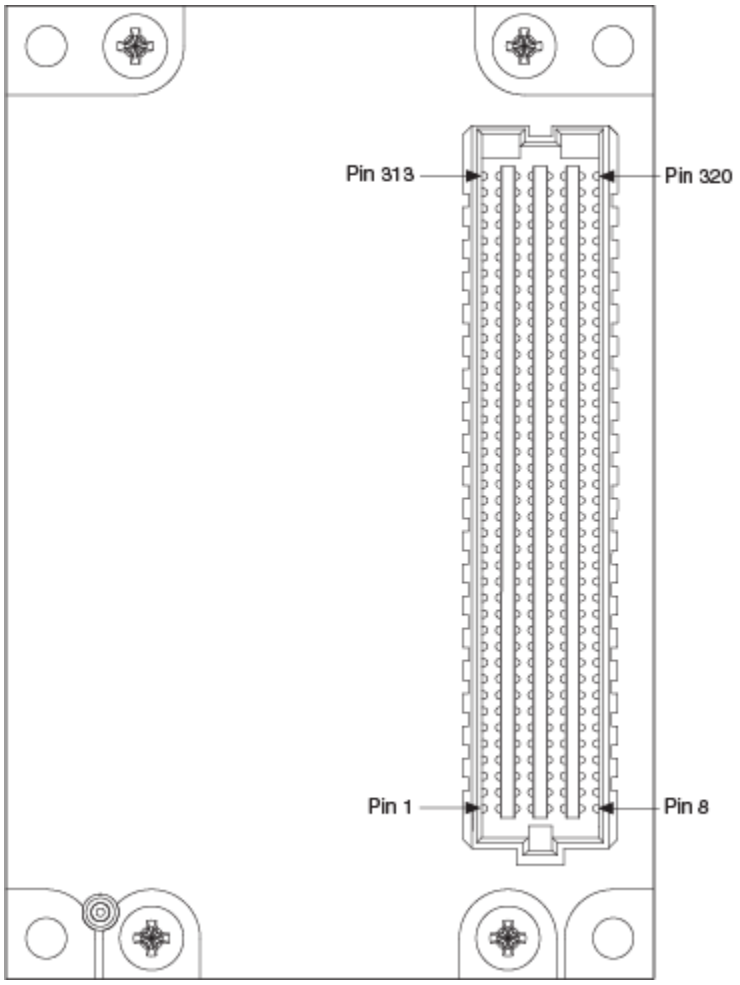
- `Offset = 0 V`
- `LSB Weight = 320 µV/LSB`

sbRIO-9651

Embedded System on Module (SOM) with Real-Time Processor and Reconfigurable FPGA, 160 configurable DIO channels, 85,000 logic cells

sbRIO-9651 Pinout

Figure 181. J1 Connector



Note Connector J1 pin assignments are available in the *sbRIO-9651 Specifications*.

Related concepts:

- Getting Started with the sbRIO-9651 in LabVIEW

Related information:

- sbRIO-9651 Specifications

Getting Started with the sbRIO-9651 in LabVIEW

The sbRIO-9651 System on Module (SOM) requires a mated carrier board in order to function as a complete LabVIEW FPGA and LabVIEW Real-Time target. The sbRIO-9651 reserves some connector pins for specific functionality, such as primary Ethernet and USB ports. The sbRIO-9651 also provides several banks of additional pins that you can configure for purposes specific to your application.

The sbRIO-9651 SOM Development Kit includes a reference carrier board. This board demonstrates how to implement additional I/O functionality, such as Serial, CAN, and secondary Ethernet.

To use the sbRIO-9651 as a LabVIEW FPGA and LabVIEW Real-Time target in a LabVIEW project, first create a socketed component-level IP (CLIP) that defines the I/O configuration to use in your application.

Use this tutorial to create a CLIP for the sbRIO-9651 and add it to a LabVIEW project.

Before You Begin:

- Before you write any software, use the *sbRIO-9651 System on Module Carrier Board Design Guide* to plan the layout of your carrier board and define the purpose of each pin on the connector. Gather the following information to create a CLIP for the sbRIO-9651:
  - Processor peripherals to enable
  - Pins to use
  - I/O standards and drive strength to use for the pins

- Required I/O voltage level
- LabVIEW FPGA I/O Nodes the application requires
- Timing constraints or clock resources

After gathering this information, proceed to *Creating a Component-Level IP for the sbRIO-9651*.

Related concepts:

- sbRIO-9651

Related tasks:

- Creating a Component-Level IP for the sbRIO-9651

Related information:

- sbRIO-9651 SOM Carrier Board Design Guide

Creating a Component-Level IP for the sbRIO-9651

Use the sbRIO CLIP Generator to generate a unique socketed CLIP that you can import into a LabVIEW project. The sbRIO CLIP Generator completes the following tasks for you: Generates processor peripheralsConfigures the raw FPGA I/O pinsGenerates a LabVIEW FPGA I/O Node interface for the socketed CLIP

Complete the following steps to launch the sbRIO CLIP Generator and create a new CLIP.

1. Create a new LabVIEW project.
2. Add an FPGA target to the project.
3. In the Add Targets and Devices dialog box, select **Real-Time Single-Board RIO»sbRIO-9651**.
4. In the Project Explorer window, right-click the **Chassis (sbRIO-9651)** item and select **New»FPGA Target**.
5. Right-click the **FPGA Target** item and select **Launch sbRIO CLIP Generator**



Note You can also launch the sbRIO CLIP Generator outside of LabVIEW:

- (Windows 8.*x*) Click the **NI Launcher** tile on the Start screen and select **CompactRIO»Single-Board RIO»sbRIO CLIP Generator**.
- (Windows 7 or earlier) Select **Start»All Programs»NI»CompactRIO»Single-Board RIO»sbRIO CLIP Generator**.

6. Perform the following tasks in the sbRIO CLIP Generator:
   - Name the CLIP
   - Configure the peripherals to enable
   - Create a LabVIEW FPGA I/O Node interface for the CLIP
   - Configure clock resources for the CLIP



Tip For help using the sbRIO CLIP Generator, click **Help** in the sbRIO CLIP Generator to launch the *NI Single-Board RIO CLIP Generator Help*. Additional information is also available in the `<NI>\CompactRIO\sbRIO\CLIP Generator` directory.



Note To make changes to an existing CLIP after it has been created, manually edit the generated CLIP files. You cannot use the sbRIO CLIP Generator to edit an existing CLIP. However, you can load the configuration for a previously saved CLIP so you can use it as the basis for a new CLIP.

After creating the CLIP, add it to a LabVIEW project.

Related concepts:

- Getting Started with the sbRIO-9651 in LabVIEW

Related tasks:

- Implementing FPGA VIs for the sbRIO-9651

Adding a CLIP for a Single-Board RIO Device to a LabVIEW Project

Before you begin, use the sbRIO CLIP Generator to generate a unique socketed CLIP.

Complete the following steps to add the CLIP you created using the sbRIO CLIP Generator for a Single-Board RIO target to a LabVIEW project.

1. Create a new project.
2. Add an FPGA target to the project. In the Add Targets and Devices dialog box, select **Real-Time Single-Board RIO** and then select your device.
3. In the Project Explorer window, right-click the **Chassis** item and select **New»FPGA Target**.
4. Right-click the **RMC Socket** or **sbRIO-9651 Socket** item and select **Properties** to launch the Component-Level IP Properties or **RMC Socket Properties** dialog box.
5. On the General page, use the **RMC or sbRIO-9651 Socketed Component Level IP Declaration** drop-down list to select the CLIP you created.
6. On the Clock Selections page, configure any clocks that you specified on the Clock Settings page of the sbRIO CLIP Generator.
7. Click **OK** to close the Component-Level IP Properties dialog box.

Proceed to *Implementing FPGA VIs for the sbRIO-9651*.
Related tasks:

- Implementing FPGA VIs for the sbRIO-9651

Implementing FPGA VIs for the sbRIO-9651

Complete the following tasks before performing this procedure.

- Use the sbRIO CLIP Generator to *generate a unique socketed CLIP*.
- dd the CLIP to a LabVIEW project.
- Connect the sbRIO-9651 to your computer with a USB cable or through a network.
- Configure the sbRIO-9651 with Measurement & Automation Explorer (MAX).

Complete the following steps to implement a LabVIEW FPGA VI for the sbRIO-9651.

1. Create and compile your FPGA VI.
2. If you enabled any peripherals in your CLIP, such as secondary ethernet, CAN, or serial, download your FPGA VI to the flash of the sbRIO-9651 to load on boot in order to ensure that the driver for the peripheral can load properly at boot time.

You should now be able to run and deploy the system just like any other sbRIO device.

Related tasks:

- Creating a Component-Level IP for the sbRIO-9651
- Adding a CLIP for a Single-Board RIO Device to a LabVIEW Project

RIO Mezzanine Cards

Table 660. RIO Mezzanine Cards

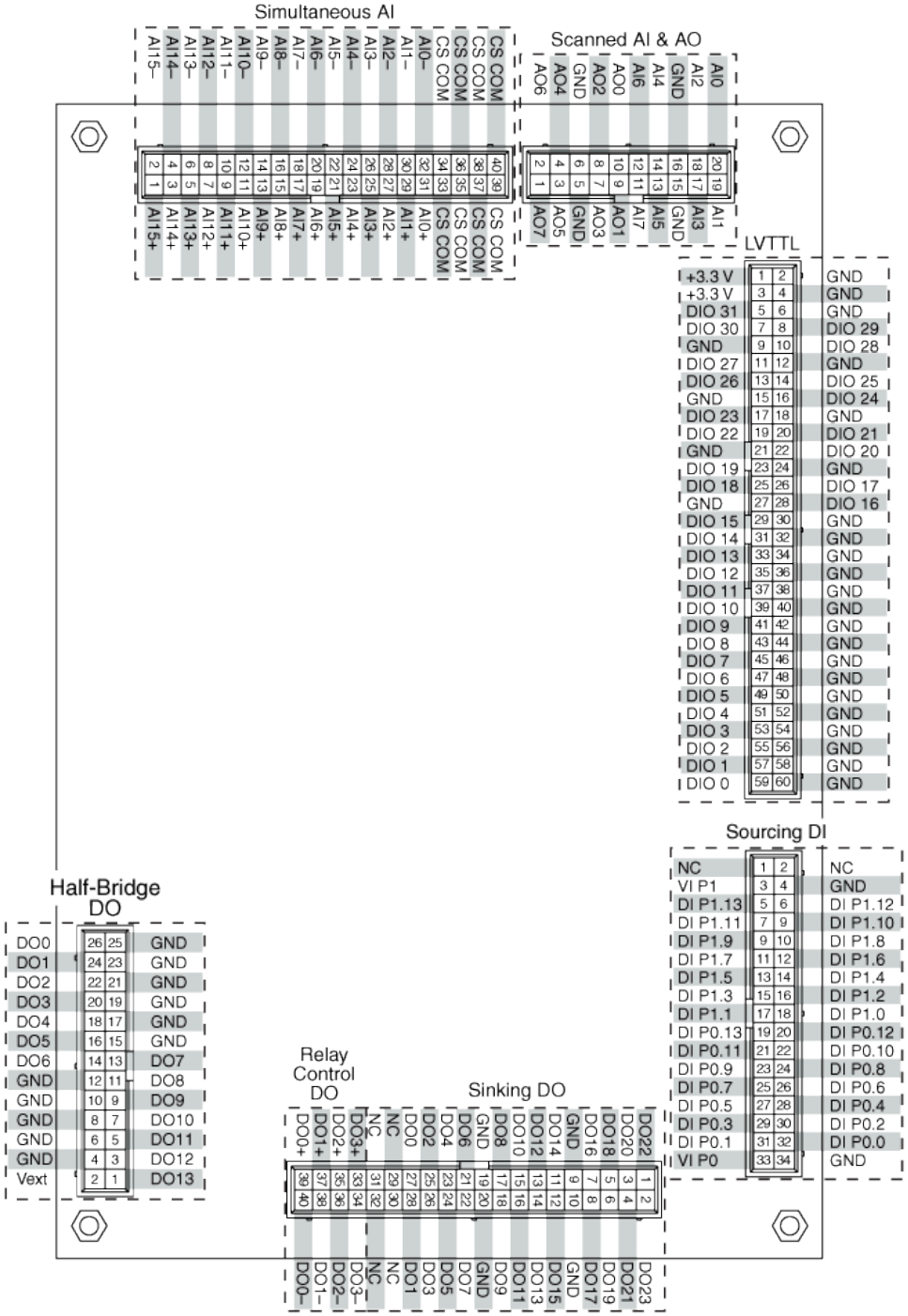
RIO Mezzanine Card	Description
sbRIO-9683	RIO Mezzanine Card, General Purpose Inverter Controller
sbRIO-9684	RIO Mezzanine Card, 16-bit General Purpose Inverter Controller
sbRIO-9694	RIO Mezzanine Card, 96 channel digital I/O breakout board
sbRIO-9697	RIO Mezzanine Card, 2-slot C Series, 24 3.3 V DIO channels
sbRIO-9698	RIO Mezzanine Card, 1-Slot C Series, Secondary Ethernet Port, 24 3.3 V DIO channels
NI Digital I/O Mezzanine Card	96 DIO channels

sbRIO-9683

Caution Use the sbRIO-9683 with only sbRIO-9605/9606/9607 devices. The sbRIO-9683 is not electrically or mechanically compatible with other NI sbRIO devices.

sbRIO-9683 Pinout

Figure 182. sbRIO-9683 Pinout



sbRIO-9683 (FPGA Interface)

Use an FPGA I/O Node configured for reading and writing with this device.

Use the FPGA I/O Node to access the following terminals for this device.

Table 661. Terminals in Software

Terminal	Description
AO*x*	Analog output channel where *x* is the number of the channel. The sbRIO-9683 has AO channels 0 to 7 that you can add to a project.
Half-Bridge DO*x*	Half-bridge digital output channel where *x* is the number of the channel. The sbRIO-9683 has Half-bridge DO channels 0 to 13 that you can add to a project.
Half-Bridge DO13:0	Digital port consisting of Half-bridge DO channels 0 through 13. Channel 13 is returned in the MSB Channel 0 is returned in the LSB.
Half-Bridge DO7:0	Digital port consisting of Half-bridge DO channels 0 through 7. Channel 7 is returned in the MSB. Channel 0 is returned in the LSB.
Half-Bridge DO13:8	Digital port consisting of Half-bridge DO channels 8 through 13. Channel 13 is returned in the MSB. Channel 8 is returned in the LSB.
LVTTL DIO*x*	LVTTL digital input/output channel where *x* is the number of the channel. The sbRIO-9683 has LVTTL DIO channels 0 to 31 that you can add to a project.
LVTTL DIO7:0	Digital port consisting of LVTTL DIO channels 0 through 7. Channel 7 is returned in the MSB. Channel 0 is returned in the LSB.
LVTTL DIO15:8	Digital port consisting of LVTTL DIO channels 8 through 15. Channel 15 is returned in the MSB. Channel 8 is returned in the LSB.
LVTTL DIO23:16	Digital port consisting of LVTTL DIO channels 16 through 23. Channel 23 is returned in the MSB, and channel 16 is returned in the LSB.
LVTTL DIO31:24	Digital port consisting of LVTTL DIO channels 24 through 31. Channel 31 is returned in the MSB. Channel 24 is returned in the LSB.
Relay Control DO*x*	Relay control digital output channel where *x* is the number of the channel. The sbRIO-9683 has relay control DO channels 0 to 3 that you can add to a project.
Relay Control DO3:0	Digital port consisting of relay control DO channels 0 through 3. Channel 3 is returned in the MSB. Channel 0 is returned in the LSB.
Scanned AI*x*	Scanned analog input channel where *x* is the number of the channel. The sbRIO-9683 has scanned AI channels 0 to 7 that you can add to a project.
Simultaneous AI*x*	Simultaneous analog input channel where *x* is the number of the channel. The sbRIO-9683 has simultaneous AI channels 0 to 15 that you can add to a project.
Sinking DO*x*	Sinking digital output channel where *x* is the number of the channel. The sbRIO-9683 has sinking DO channels 0 to 23 that you can add to a project.

Terminal	Description
Sinking DO23:0	Digital port consisting of sinking DO channels 0 through 23. Channel 23 is returned in the MSB. Channel 0 is returned in the LSB.
Sinking DO7:0	Digital port consisting of sinking DO channels 0 through 7. Channel 7 is returned in the MSB. Channel 0 is returned in the LSB.
Sinking DO15:8	Digital port consisting of sinking DO channels 8 through 15. Channel 15 is returned in the MSB. Channel 8 is returned in the LSB.
Sinking DO23:16	Digital port consisting of sinking DO channels 16 through 23. Channel 23 is returned in the MSB. Channel 16 is returned in the LSB.
Sourcing P0 DI*x*	Sourcing digital input channel where *x* is the number of the channel. The sbRIO-9683 has sourcing P0 DI channels 0 to 13 that you can add to a project.
Sourcing P0 DI13:0	Digital port consisting of sourcing DI P0 channels 0 through 13. Channel 13 is returned in the MSB. Channel 0 is returned in the LSB.
Sourcing P0 DI7:0	Digital port consisting of sourcing DI P0 channels 0 through 7. Channel 7 is returned in the MSB. Channel 0 is returned in the LSB.
Sourcing P0 DI13:8	Digital port consisting of sourcing DI P0channels 8 through 13. Channel 13 is returned in the MSB. Channel 8 is returned in the LSB.
Sourcing P1 DI*x*	Sourcing digital input channel where *x* is the number of the channel. The sbRIO-9683 has sourcing P1 DI channels 0 to 13 that you can add to a project.
Sourcing P1 DI13:0	Digital port consisting of sourcing DI P1 channels 0 through 13. Channel 13 is returned in the MSB. Channel 0 is returned in the LSB.
Sourcing P1 DI7:0	Digital port consisting of sourcing DI P1 channels 0 through 7. Channel 7 is returned in the MSB. Channel 0 is returned in the LSB.
Sourcing P1 DI13:8	Digital port consisting of sourcing DI P1 channels 8 through 13. Channel 13 is returned in the MSB. Channel 8 is returned in the LSB.

Arbitration

You can configure the arbitration settings for the LVTTL DIO, half-bridge DO, relay control DO, and sinking DO channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate for the LVTTL DIO, half-bridge DO, and relay control DO channels. The default arbitration setting is Arbitrate if Multiple Requestors Only for the sinking DO channels.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for LVTTL DIO channels of this device.

Table 662. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Set Output Enable	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.

Method	Description
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.

RMC Methods

Use the FPGA I/O Method Node to access the following RMC methods for this device.

Table 663. RMC Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the half-bridge DO on the device is ready.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 664. I/O Properties

Property	Description
Voltage Range	Returns the input range for a simultaneous analog input bank.

RMC Properties

Use the FPGA I/O Property Node to access the following RMC properties for this device.

Table 665. RMC Properties

Property	Description
Voltage Range Simultaneous AIx:x	Sets the input range for a simultaneous analog input bank as either ±5 V or ±10 V where *x*:x is the bank. This method overwrites the value that is configured in the RIO Mezzanine Card Properties dialog box.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the high-speed DO channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box.

Avoiding Timing Uncertainty with the NI 9683 (FPGA Interface)

Follow these guidelines to avoid introducing timing uncertainty into scanned AI Node and AO Node method calls.

- Use a single I/O Node to access scanned AI and AO operations to ensure proper sequencing.
- Do not perform the following operations concurrently:
  - Scanned AI Node call
  - AO Node call

Note Simultaneous AI on the NI 9683 does not affect scanned AI and AO and is not affected by scanned AI and AO.

Scaling NI 9683 Analog Input Values (FPGA Interface)

To configure the FPGA I/O Node to return scaled, fixed-point data for the device, set the **Scaling Mode** to **Scaled** in the **RIO Mezzanine Card Properties** dialog box. IWhen **Scaling Mode** is set to Raw, the FPGA I/O Node returns unscaled, binary values.

Convert into volts using the following equations. Convert these values in the host VI.

Scanned AI

Use the following equation to convert binary values into volts for the scanned AI on the NI 9683.

$$Volts = Binary\ Value \times 5\ V \div 4096$$

Where

- *Binary Value* is the signed or unsigned value returned by the FPGA I/O Node

Simultaneous AI

Use the following equations to convert binary values into volts for the simultaneous AI on the NI 9683.

±10 V Range	$Volts = Binary\ Value \times 20\ V \div 4096$ Where • *Binary Value* is the signed or unsigned value returned by the FPGA I/O Node
±5 V Range	$Volts = Binary\ Value \times 10\ V \div 4096$ Where • *Binary Value* is the signed or unsigned value returned by the FPGA I/O Node

Scaling NI 9683 Analog Output Values (FPGA Interface)

To configure the FPGA I/O Node to accept fixed-point data in volts when writing to the device, set the **Scaling Mode** to Scaled in the **RIO Mezzanine Card Properties** dialog box. When **Scaling Mode** is set to Raw, the FPGA I/O Node accepts binary values only. Convert output voltage values to binary values using the following equation. Convert these values in the host VI.

$$Binary\ Value = Volts \times 4096 \div 5\ V$$

Where

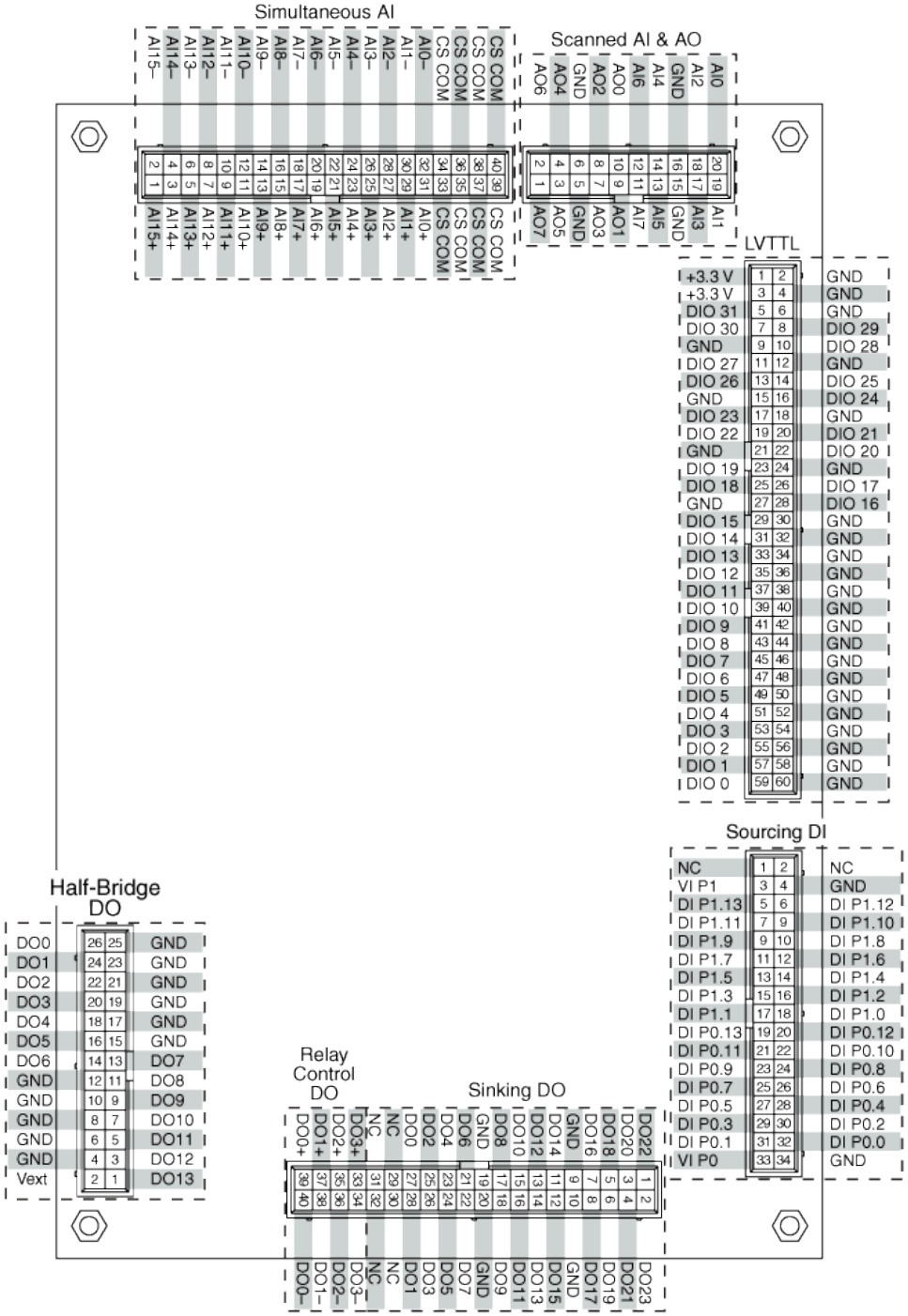
- *Binary Value* is the value you write to the FPGA I/O Node

sbRIO-9684

> **Caution** Use the sbRIO-9684 with only NI sbRIO-9603/9605/9606/9607 devices. The sbRIO-9684 is not electrically or mechanically compatible with other NI sbRIO devices.

sbRIO-9684 Pinout

Figure 183. sbRIO-9684 Pinout



sbRIO-9684 (FPGA Interface)

Use an FPGA I/O Node configured for reading and writing with this device.

Use the FPGA I/O Node to access the following terminals for this device.

Table 666. Terminals in Software

Terminal	Description
AO*x*	Analog output channel where *x* is the number of the channel. The sbRIO-9684 has AO channels 0 to 7 that you can add to a project.
Half-Bridge DO*x*	Half-bridge digital output channel where *x* is the number of the channel. The sbRIO-9684 has Half-bridge DO channels 0 to 13 that you can add to a project.
Half-Bridge DO13:0	Digital port consisting of Half-bridge DO channels 0 through 13. Channel 13 is returned in the MSB, and channel 0 is returned in the LSB.
Half-Bridge DO7:0	Digital port consisting of Half-bridge DO channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
Half-Bridge DO13:8	Digital port consisting of Half-bridge DO channels 8 through 13. Channel 13 is returned in the MSB, and channel 8 is returned in the LSB.
LVTTL DIO*x*	LVTTL digital input/output channel where *x* is the number of the channel. The sbRIO-9684 has LVTTL DIO channels 0 to 31 that you can add to a project.
LVTTL DIO7:0	Digital port consisting of LVTTL DIO channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
LVTTL DIO15:8	Digital port consisting of LVTTL DIO channels 8 through 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
LVTTL DIO23:16	Digital port consisting of LVTTL DIO channels 16 through 23. Channel 23 is returned in the MSB, and channel 16 is returned in the LSB.
LVTTL DIO31:24	Digital port consisting of LVTTL DIO channels 24 through 31. Channel 31 is returned in the MSB, and channel 24 is returned in the LSB.
Relay Control DO*x*	Relay control digital output channel where *x* is the number of the channel. The sbRIO-9684 has relay control DO channels 0 to 3 that you can add to a project.
Relay Control DO3:0	Digital port consisting of relay control DO channels 0 through 3. Channel 3 is returned in the MSB, and channel 0 is returned in the LSB.
Scanned AI*x*	Scanned analog input channel where *x* is the number of the channel. The sbRIO-9684 has scanned AI channels 0 to 7 that you can *add to a project*.
Simultaneous AI*x*	Simultaneous analog input channel where *x* is the number of the channel. The sbRIO-9684 has simultaneous AI channels 0 to 15 that you can add to a project.
Sinking DO*x*	Sinking digital output channel where *x* is the number of the channel. The sbRIO-9684 has sinking DO channels 0 to 23 that you can add to a project.

Terminal	Description
Sinking DO23:0	Digital port consisting of sinking DO channels 0 through 23. Channel 23 is returned in the MSB, and channel 0 is returned in the LSB.
Sinking DO7:0	Digital port consisting of sinking DO channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
Sinking DO15:8	Digital port consisting of sinking DO channels 8 through 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
Sinking DO23:16	Digital port consisting of sinking DO channels 16 through 23. Channel 23 is returned in the MSB, and channel 16 is returned in the LSB.
Sourcing P0 DI*x*	Sourcing digital input channel where *x* is the number of the channel. The sbRIO-9684 has sourcing P0 DI channels 0 to 13 that you can add to a project.
Sourcing P0 DI13:0	Digital port consisting of sourcing DI P0 channels 0 through 13. Channel 13 is returned in the MSB, and channel 0 is returned in the LSB.
Sourcing P0 DI7:0	Digital port consisting of sourcing DI P0 channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
Sourcing P0 DI13:8	Digital port consisting of sourcing DI P0channels 8 through 13. Channel 13 is returned in the MSB, and channel 8 is returned in the LSB.
Sourcing P1 DI*x*	Sourcing digital input channel where *x* is the number of the channel. The sbRIO-9684 has sourcing P1 DI channels 0 to 13 that you can add to a project.
Sourcing P1 DI13:0	Digital port consisting of sourcing DI P1 channels 0 through 13. Channel 13 is returned in the MSB, and channel 0 is returned in the LSB.
Sourcing P1 DI7:0	Digital port consisting of sourcing DI P1 channels 0 through 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
Sourcing P1 DI13:8	Digital port consisting of sourcing DI P1 channels 8 through 13. Channel 13 is returned in the MSB, and channel 8 is returned in the LSB.

Arbitration

You can configure the arbitration settings for the LVTTL DIO, half-bridge DO, relay control DO, and sinking DO channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate for the LVTTL DIO, half-bridge DO, and relay control DO channels. The default arbitration setting is Arbitrate if Multiple Requestors Only for the sinking DO channels.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for LVTTL DIO channels of this device.

Table 667. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Set Output Enable	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Wait on Any Edge	Pauses the execution of the I/O Method Node until the next falling or rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Any Edge method waits for the next falling or rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Falling Edge	Pauses the execution of the I/O Method Node until the next falling edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Falling Edge method waits for the next falling edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.
Wait on High Level	Pauses the execution of the I/O Method Node until the digital signal is high. The **Timeout** input specifies in FPGA clock ticks how long the Wait on High Level method waits for the next high level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.

Method	Description
Wait on Low Level	Pauses the execution of the I/O Method Node until the digital signal is low. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Low Level method waits for the next low level. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.
Wait on Rising Edge	Pauses the execution of the I/O Method Node until the next rising edge of the digital signal. The **Timeout** input specifies in FPGA clock ticks how long the Wait on Rising Edge method waits for the next rising edge. • A value of 0 causes the method to time out immediately. • A negative value causes the method to wait indefinitely. • A positive value causes the method to wait for that number of clock ticks before timing out.

RMC Methods

Use the FPGA I/O Method Node to access the following RMC methods for this device.

Table 668. RMC Methods

Method	Description
Check Status	Returns a Boolean value that indicates whether the half-bridge DO on the device is ready.

I/O Properties

Use the FPGA I/O Property Node to access the following I/O properties for this device.

Table 669. I/O Properties

Property	Description
LSB Weight (±10V range)	Returns the LSB weight in V/LSB for the ±10V range.

Property	Description
LSB Weight (±5V range)	Returns the LSB weight in V/LSB for the ±5V range.
Offset (±10V range)	Returns the calibration offset in V for the ±10V range.
Offset (±5V range)	Returns the calibration offset in V for the ±5V range.
Voltage Range	Returns the input range for a simultaneous analog input bank.

RMC Properties

Use the FPGA I/O Property Node to access the following RMC properties for this device.

Table 670. RMC Properties

Property	Description
Voltage Range Simultaneous AIx:x	Sets the input range for a simultaneous analog input bank as either ±5 V or ±10 V where *x*:x is the bank. This method overwrites the value that is configured in the RIO Mezzanine Card Properties dialog box.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the high-speed DO channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box.

Avoiding Timing Uncertainty with the sbRIO-9684 (FPGA Interface)

Follow these guidelines to avoid introducing timing uncertainty into scanned AI Node and AO Node method calls.

- Use a single I/O Node to access scanned AI and AO operations to ensure proper sequencing.
- Do not perform the following operations concurrently:

Table 671.

	– Scanned AI Node call
	– AO Node call

Note Simultaneous AI on the sbRIO-9684 does not affect scanned AI and AO and is not affected by scanned AI and AO.

Scaling sbRIO-9684 Analog Input Values (FPGA Interface)

Set the **Scaling Mode** to **Scaled** in the **RIO Mezzanine Card Properties** dialog box for the NI 9684 if you want the FPGA I/O Node to return scaled, fixed-point data for the device. If you set the **Scaling Mode** to **Raw**, the FPGA I/O Node returns unscaled, binary values for the NI 9684 that you can convert into volts using the following equations. You must convert these values in the host VI Scanned AI

Use the following equation to convert binary values into volts for the scanned AI on the sbRIO-9684.

Volts = Binary Value × 5 V ÷ 4096

Table 672.

where	Binary Value is the signed or unsigned value returned by the FPGA I/O Node

Simultaneous AI

Use the following equations to convert binary values into volts for the simultaneous AI on the sbRIO-9684 for both ±10V and ±5V ranges.

Volts = (Binary Value × LSB Weight + Offset)*

Table 673.

where	Binary Value is the signed value returned by the FPGA I/O Node
	LSB Weight is the value returned by the LSB Weight property
	Offset is the value returned by the Offset property

If you do not want to read the LSB Weight and Offset values from the module, you can convert to uncalibrated engineering units by using the following values for Offset and LSB Weight:

Offset = 0

$$\text{LSB Weight} = \text{Typical Input Span} \div 2^{\text{ADC Resolution}}$$

Table 674.

where	Typical Input Span is 20.832V for ±10V range and 10.416V for ±5V range
	ADC Resolution is the ADC resolution value in the *hardware documentation for the module*

*LSB Weight and Offset are formatted and cannot be used as is. Refer to the Binary to Nominal VI in the `LabVIEW\examples\CompactRIO\RMC Specific\NI 9684\NI 9684 High Speed Acquisition\NI 9684 High Speed Acquisition\NI 9684 High Speed Acquisition.lvproj` for an example on how to convert the LSB Weight and Offset to the correct format.

Scaling sbRIO-9684 Analog Output Values (FPGA Interface)

Set the **Scaling Mode** to **Scaled** in the **RIO Mezzanine Card Properties** dialog box for the NI 9684 if you want the FPGA I/O Node to accept fixed-point data in volts when writing to the device. If you set the **Scaling Mode** to **Raw**, the FPGA I/O Node accepts only binary values. You can convert output voltage values to binary values using the following equation. You must convert these values in the host VI.

$$\text{Binary Value} = \text{Volts} \times 4096 \div 5\ \text{V}$$

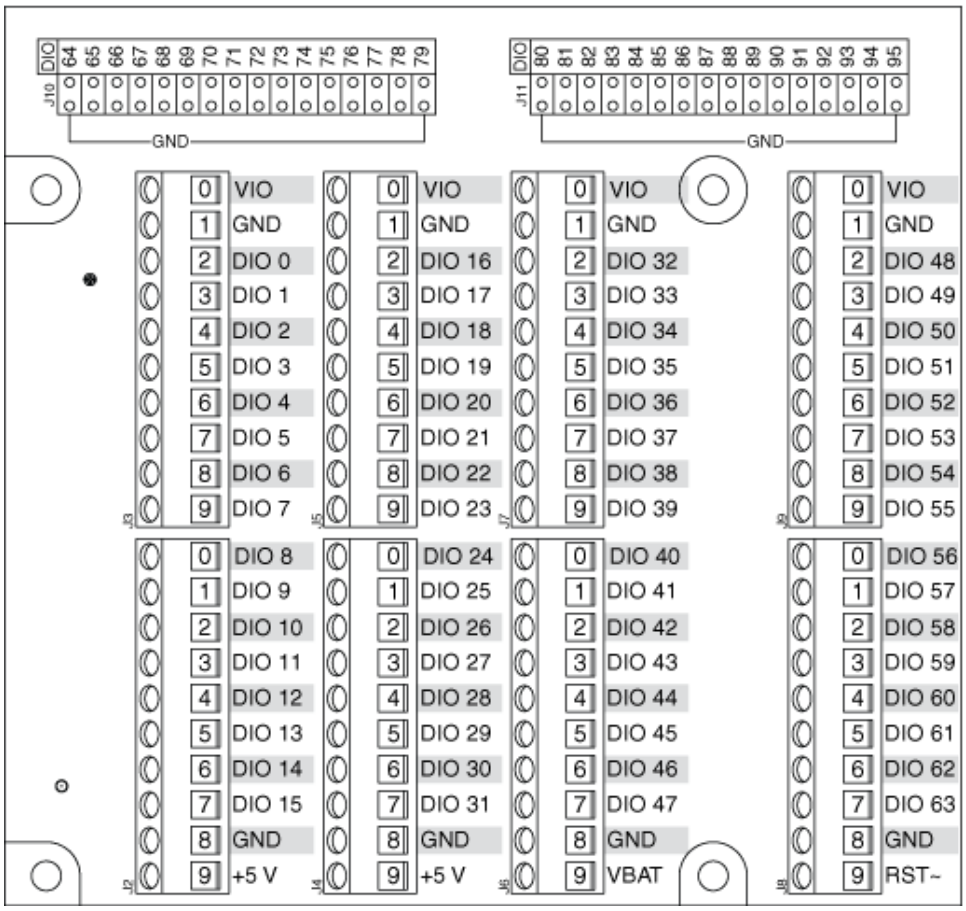
Table 675.

where	Binary Value is the value you write to the FPGA I/O Node

sbRIO-9694

sbRIO-9694 Pinout

Figure 184. sbRIO-9694 Pinout



sbRIO-9694 (FPGA Interface)

Use an FPGA I/O Node configured for reading and writing with this device.

Table 676. Terminals in Software

Terminal	Description
DIO*x*	Digital input/output channel where *x* is the number of the channel. The sbRIO-9694 has DIO channels 0 to 95 that you can add to a project.
DIO15:0	Digital port consisting of DIO channels 0 through 15. Channel 15 is returned in the MSB, and channel 0 is returned in the LSB.
DIO31:16	Digital port consisting of DIO channels 16 through 31. Channel 31 is returned in the MSB, and channel 16 is returned in the LSB.

Terminal	Description
DIO47:32	Digital port consisting of DIO channels 32 through 47. Channel 47 is returned in the MSB, and channel 32 is returned in the LSB.
DIO63:48	Digital port consisting of DIO channels 48 through 63. Channel 63 is returned in the MSB, and channel 48 is returned in the LSB.
DIO79:64	Digital port consisting of DIO channels 64 through 79. Channel 79 is returned in the MSB, and channel 64 is returned in the LSB.
DIO95:80	Digital port consisting of DIO channels 80 through 95. Channel 95 is returned in the MSB, and channel 80 is returned in the LSB.

Arbitration

You can configure the arbitration settings for the DO channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for LVTTL DIO channels of this device.

Table 677. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Set Output Enable	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.

Module Methods

This device does not support any module methods.

Properties

This device does not support any properties.

Single-Cycle Timed Loop

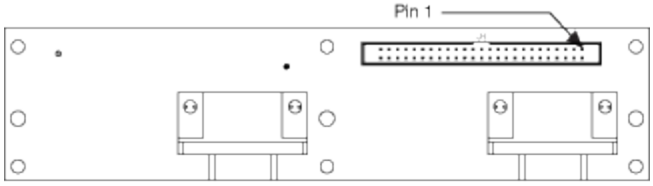
This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box.

sbRIO-9697

Note Use the sbRIO-9697 with only sbRIO-9607/9627 devices.

sbRIO-9697 Pinout

Figure 185. sbRIO-9697 Pinout



J1, DIO

D GND	1	2	DIO0
D GND	3	4	DIO1
D GND	5	6	DIO2
D GND	7	8	DIO3
D GND	9	10	DIO4
D GND	11	12	DIO5
D GND	13	14	DIO6
D GND	15	16	DIO7
D GND	17	18	DIO8
D GND	19	20	DIO9
D GND	21	22	DIO10
D GND	23	24	DIO11
D GND	25	26	DIO12
D GND	27	28	DIO13
D GND	29	30	DIO14
D GND	31	32	DIO15
D GND	33	34	DIO16
D GND	35	36	DIO17
D GND	37	38	DIO18
D GND	39	40	DIO19
D GND	41	42	DIO20
D GND	43	44	DIO21
D GND	45	46	DIO22
D GND	47	48	DIO23
+5V	49	50	+5V

sbRIO-9697 (FPGA Interface)

Use an FPGA I/O Node configured for reading and writing with this device.

Table 678. Terminals in Software

Terminal	Description
DIOx	Digital input/output channel where x is the number of the channel. The sbRIO-9697 has DIO channels 0 to 23 that you can add to a project.
DIO15:0	Digital port consisting of DIO channels 0 through 15. Channel 15 is returned in the MSB, and channel 0 is returned in the LSB.
DIO23:16	Digital port consisting of DIO channels 16 through 23. Channel 23 is returned in the MSB, and channel 16 is returned in the LSB.

Arbitration

You can configure the arbitration settings for the DO channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for LVTTL DIO channels of this device.

Table 679. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Set Output Enable	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.

Module Methods

This device does not support any module methods.

Properties

This device does not support any properties.

Single-Cycle Timed Loop

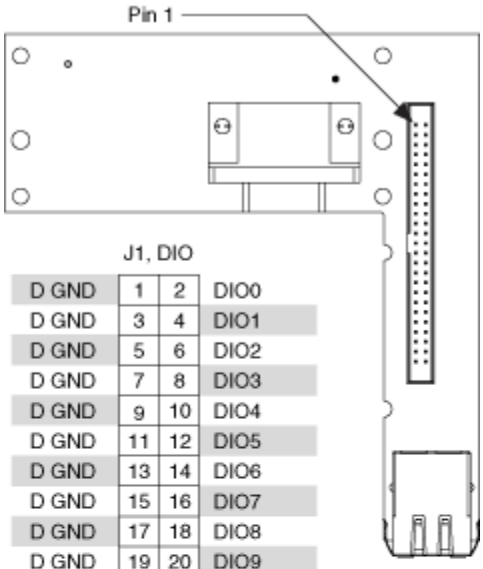
This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box.

sbRIO-9698

Note Use the sbRIO-9698 only with sbRIO-9607/9627 devices.

sbRIO-9698 Pinout

Figure 186. sbRIO-9698 Pinout



J1, DIO

D GND	1	2	DIO0
D GND	3	4	DIO1
D GND	5	6	DIO2
D GND	7	8	DIO3
D GND	9	10	DIO4
D GND	11	12	DIO5
D GND	13	14	DIO6
D GND	15	16	DIO7
D GND	17	18	DIO8
D GND	19	20	DIO9
D GND	21	22	DIO10
D GND	23	24	DIO11
D GND	25	26	DIO12
D GND	27	28	DIO13
D GND	29	30	DIO14
D GND	31	32	DIO15
D GND	33	34	DIO16
D GND	35	36	DIO17
D GND	37	38	DIO18
D GND	39	40	DIO19
D GND	41	42	DIO20
D GND	43	44	DIO21
D GND	45	46	DIO22
D GND	47	48	DIO23
+5V	49	50	+5V

sbRIO-9698 (FPGA Interface)

Use an FPGA I/O Node configured for reading and writing with this device.

Table 680. Terminals in Software

Terminal	Description
DIOx	Digital input/output channel where x is the number of the channel. The sbRIO-9698 has DIO channels 0 to 23 that you can add to a project.
DIO15:0	Digital port consisting of DIO channels 0 through 15. Channel 15 is returned in the MSB, and channel 0 is returned in the LSB.
DIO23:16	Digital port consisting of DIO channels 16 through 23. Channel 23 is returned in the MSB, and channel 16 is returned in the LSB.

Arbitration

You can configure the arbitration settings for the DO channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for LVTTL DIO channels of this device.

Table 681. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Set Output Enable	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.

Module Methods

This device does not support any module methods.

Properties

This device does not support any properties.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box.

NI Digital I/O Mezzanine Card (FPGA Interface)

Use an FPGA I/O Node configured for reading and writing with this device.

Use the FPGA I/O Node to access the following terminals.

Table 682. Terminals in Software

Terminal	Description
DIO*x*	Digital input/output channel where x is the number of the channel. The digital I/O mezzanine card exposes DIO channels 0 to 95 that you can add to a project.
DIO15:0	Digital port consisting of DIO channels 0 through 15. Channel 15 is returned in the MSB, and channel 0 is returned in the LSB.
DIO31:16	Digital port consisting of DIO channels 16 through 31. Channel 31 is returned in the MSB, and channel 16 is returned in the LSB.
DIO47:32	Digital port consisting of DIO channels 32 through 47. Channel 47 is returned in the MSB, and channel 32 is returned in the LSB.
DIO63:48	Digital port consisting of DIO channels 48 through 63. Channel 63 is returned in the MSB, and channel 48 is returned in the LSB.
DIO79:64	Digital port consisting of DIO channels 64 through 79. Channel 79 is returned in the MSB, and channel 64 is returned in the LSB.
DIO95:80	Digital port consisting of DIO channels 80 through 95. Channel 95 is returned in the MSB, and channel 80 is returned in the LSB.

Arbitration

You can configure the arbitration settings for the DO channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for LVTTL DIO channels of this device.

Table 683. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Set Output Enable	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.

Module Methods

This device does not support any module methods.

Properties

This device does not support any properties.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop. Configure the number of output synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. Configure the number of input synchronizing registers for the channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box.

Related concepts:

- Reading from C Series Channels
- Writing to C Series Channels

- Configuring a Project

NI Single-Board RIO CLIP Generator

The NI Single-Board RIO (sbRIO) Component-Level IP (CLIP) Generator application generates a unique socketed CLIP.

The sbRIO CLIP Generator completes the following tasks:

- Generates processor peripherals
- Configures the raw FPGA I/O pins
- Generates a LabVIEW FPGA I/O Node interface for the socketed CLIP

Import the CLIP into a LabVIEW project for the following targets:

- NI sbRIO-9603
- NI sbRIO-9607
- NI sbRIO-9608
- NI sbRIO-9609
- NI sbRIO-9627
- NI sbRIO-9628
- NI sbRIO-9629
- NI sbRIO-9651 System on Module (SOM)

Launch the sbRIO CLIP Generator using these methds:

- From the LabVIEW Project Explorer window, right-click `Chassis (<target>)»FPGA Target` and select Launch sbRIO CLIP Generator.
- Windows 8.*x*: Click the **NI Launcher** tile on the Start screen and select `CompactRIO»Single-Board RIO»sbRIO CLIP Generator`.
- Windows 7 or earlier: Select `Start»All Programs»National Instruments»CompactRIO»Single-Board RIO»sbRIO CLIP Generator`.

Related tasks:

- Creating a Component-Level IP for an NI sbRIO Target

Creating a Component-Level IP for an NI sbRIO Target

Complete the following procedures to create a new component-level IP (CLIP) for the NI sbRIO Target.

1. Creating a New CLIP
2. Adding Peripherals to a CLIP
3. Configuring I/O Pins on the Target
4. Creating a LabVIEW I/O Node Interface for a CLIP
5. Configuring Clock Resources for a CLIP
6. Finishing the CLIP Design

Related tasks:

- Adding Peripherals to a CLIP

Creating a New CLIP

Use the Create CLIP page of the sbRIO CLIP Generator to create a new CLIP.

1. On the Create CLIP page, configure the following options:
   2. **CLIP Name**—The name of the CLIP as it appears in LabVIEW and the name of the VHDL entity. The sbRIO CLIP Generator creates a folder with this name in the `Documents\LabVIEW Data\CompactRIO\CLIPs\<target>` directory. The folder contains generated files that also use this name, such as `<clipname>.vhd`, `<clipname>.xml`, and `<clipname>.xdc`.

      Note To distribute CLIP files after creating them, install the CLIP files to the `<National Instruments>\Shared\CompactRIO\CLIPs\<target>` directory. CLIP files installed to this directory are available for selection in the Component-Level IP Properties dialog box for the NI sbRIO target to a LabVIEW project.

   3. **Create New CLIP**—Specifies the options to use for configuring the state of the sbRIO CLIP Generator. The following options are available:
      - Create a new CLIP with no existing configuration.
      - Load the configuration for a previously saved CLIP to use as the basis for a new CLIP.

Note To change an existing CLIP after it has been created, manually edit the generated CLIP files. You cannot use the sbRIO CLIP Generator to edit an existing CLIP.

4. **Target Type**—The NI sbRIO device in your application.
5. **CLIP Description**—The CLIP description appears on the General page of the Component-Level IP Properties dialog box in the LabVIEW project.

6. Click **Next** to continue to the Peripherals page.
   Use the Peripherals page to add peripherals to the CLIP.

Adding Peripherals to a CLIP

A peripheral is user-configurable capability of the NI sbRIO device that requires pins, FPGA space, or both pins and FPGA space in order to function. A peripheral can be a Serial, CAN, or secondary Ethernet port.

Complete the following steps to select the processor peripherals to implement in the CLIP design.

To

1. Open the Peripherals page of the sbRIO CLIP Generator .
2. Enable the **Enabled** column of the **Additional Peripherals** pane for each peripheral you want to add to the CLIP.
3. Configure the enabled peripheral with the **Configuration** pane. Refer to the following notes for information about peripherals specific to different NI sbRIO targets.
4. To configure additional peripherals, repeat steps 1 and 2.
5. Click **Next** to continue to the Pin Configuration page.
   Use the Pin Configuration to configure the pins that each peripheral uses.

Note The following peripheral guidelines apply to all targets:

- You might not need to configure all the lines for a selected peripheral to function, depending on the needs of your application. For example, if

you enable the **Serial2** port, you might not need to configure any RTS or Modem lines for your application.
- If you do not select a pin, the sbRIO CLIP Generator does not automatically assign a pin for you. An unassigned pin may open a pin configuration warning.
- Peripherals cannot reserve the same pins. If a peripheral uses a specific DIO pin and another peripheral attempts to request it, the Enabled column for the second peripheral becomes disabled and a warning icon appears.
- Serial ports may be notated differently depending on the environment in which you are working. Refer to *Serial Port Naming Conventions* for additional information.

- **sbRIO-9607/27 SD Card** —For information about specific SD card support, refer to the *RMC Connector Design Guide*.
- **sbRIO-9651 SOM ETH2 peripheral**—Enable the **Eth2** peripheral to allow communication over a secondary Ethernet network interface controller. Enabling this peripheral reserves a subset of *Bank 2* pins and sets the voltage level for those pins. However, you cannot modify which specific pins the secondary Ethernet peripheral uses. You can select the pins to use for the green and yellow Ethernet speed LED signals.



Note Peripherals that require different bank voltages cannot be used together on the same bank. For example, the **Eth2** peripheral, which requires 2.5 V, cannot be on the same bank as the **Serial2** peripheral, which requires 3.3 V.



Caution If your CLIP design uses the primary RS-232 port for console access, National Instruments recommends that you define a pin for the Data Terminal Ready (DTR) and Data Carrier Detect (DCD) signals. These signals automatically log a user out when the serial cable is disconnected. Failure to implement these signals for console access might cause a security vulnerability.

Related concepts:

- Serial Port Naming Conventions

Related tasks:

- Configuring I/O Pins on the Target
- Creating a Component-Level IP for an NI sbRIO Target

Related information:

- RMC Connector Design Guide

Serial Port Naming Conventions

The following table describes how the serial ports on an sbRIO device may be notated differently depending on the working environment.

Note Environments can use use zero-indexed ports or one-indexed ports.

Table 684. sbRIO Serial Port Naming Conventions using the sbRIO-9651 SOM Development Kit Reference Carrier Board

Hardware Silkscreen	Linux TerminalFile System (sysfs)	VISA Names in MAX
Serial1	ttys0	ASRL1
Serial2	ttys1	ASRL2
Serial3	ttys2	ASRL3
Serial4	ttys3	ASRL4
Serial5	ttys4	ASRL5
Serial6	ttys5	ASRL6

Note In the table above, the rows for Serial 3, Serial 4, and Serial 6 indicate ports that are not implemented by default in hardware.

Related tasks:

- Adding Peripherals to a CLIP

Configuring I/O Pins on the Target

Click the **Import from CSV** or **Export to CSV** buttons on the bottom left of this page to import or export pin configuration data in a CSV file.

Note

- The options on the Pin Configuration page of the of the sbRIO CLIP Generator correspond to Xilinx I/O standards and attributes.
- Complete the following steps to use the Pin Configuration page of the sbRIO CLIP Generator to configure the I/O pins on the NI sbRIO target.

1. Select a pin in the CLIP Pins table.
2. Use the **Pin Settings** pane to configure the following options for the pin:
   3. **I/O Standard**—The I/O standard of the I/O buffer for the pin.
   4. **Direction**—The direction of the I/O buffer for the pin.

      Note The direction you select is independent of the direction of a LabVIEW port or clock. For example, if you select **Input** for the pin direction you can still use the pin in a **Read/Write** LabVIEW FPGA I/O node, but writing to the pin will have no effect.

   5. **Drive Strength**—The output drive strength, in mA, for the pin.
   6. **Slew**—The output slew rate, which is the rate of transition between output levels, for the pin.
   7. **Pull**—The pull behavior for the pin. Select one of the following options:
      - **None**—Does not affect the input pin.
      - **Up**—Drives the input pin to its high value when the pin is not being driven.
      - **Down**—Drives the input pin to its low value when the pin is not being driven.
      - **Keeper**—Preserves the last value of the pin when the pin is not being driven.

Note

- Refer to Xilinx documentation for more information about these

and other I/O standards and attributes.
- Some pins that a peripheral uses may allow you to specify a pin setting value of **Custom**. For such pins to function correctly, you must *modify the generated CLIP files* after you finish using the sbRIO CLIP Generator to supply the required custom value.

8. Use the **Bank Voltage Levels** pane to specify the voltage level supplied to each bank of pins.

Note
- Not all configurations of pin settings and bank voltage levels are valid. You must specify a valid configuration to enable the **Next** button and continue in the sbRIO CLIP Generator. If you specify an invalid configuration, the **Next** button is disabled and the CLIP Pins table displays a warning icon. Hover over the warning icon to display a tooltip that includes information about the error to resolve.
- If you configure a pin to use a differential I/O standard, the pin and its positive or negative pair function as a single input or output. For example, if you configure the DIO_23 pin to use the LVDS25 I/O standard, the DIO_23_N pin becomes disabled in the CLIP Pins table and all of its settings match those of the DIO_23 pin. This change occurs because the value of the DIO_23 pin now represents the voltage differential between the DIO_23 and DIO_23_N pins, and therefore the DIO_23_N pin no longer has a meaningful value.

Caution Although the sbRIO CLIP Generator provides error-checking to ensure that your pin settings are valid, nothing prevents you from supplying a different voltage to the physical bank than the voltage specified in software.

9. Click **Next** to continue to the LabVIEW Interface page.

Related concepts:

- Finishing the CLIP Design

Related tasks:

- Adding Peripherals to a CLIP
- Creating a LabVIEW I/O Node Interface for a CLIP

Creating a LabVIEW I/O Node Interface for a CLIP

Complete the following steps to use the LabVIEW Interface page of the sbRIO CLIP Generator to create the LabVIEW FPGA I/O Node interface for the NI sbRIO target.



Note Click the **Import from CSV** or **Export to CSV** buttons on the bottom left of this page to import or export LabVIEW I/O node data in a CSV file.

1. From the **Available Pins** control, select one or more pins.
2. From the **I/O Direction** control, select one of the following options:
   3. **Read-only**—Creates one read-only I/O item in the LabVIEW project.
   4. **Read/Write**—Creates one read-only I/O item and two writable I/O items in the LabVIEW project. The first of these writable I/O items enables the output on the individual pins associated with the I/O item, and the second determines the value being output on the pins. You can read external data on the pins if the output is not enabled.
5. From the **LabVIEW Data Type** control, select the LabVIEW data type to use for the FPGA I/O Node.

   Note If you select fewer pins than the width of the data type (for example, you select 7 pins but specify a U8 data type), the created I/O item uses only the least significant bits. If you select more pins than the width of the data type, the sbRIO CLIP Generator creates multiple I/O items of the same data type.
6. Click **Add**. The **LabVIEW I/O Nodes** table lists the created I/O node.
7. (Optional) Use the **LabVIEW Name** column in the LabVIEW I/O Nodes table to rename the I/O item.
   This name appears in the LabVIEW project and on FPGA I/O Nodes on the LabVIEW block diagram. The name must meet the following requirements:
   - The I/O item name is unique. The name can not be a duplicate of another I/O item name.

    - The I/O item name starts with a letter.
    - The I/O item name contains only letters, numbers, and underscores.
8. (Optional) Expand the information in the Pin Details column to review the details for a created FPGA I/O Node.
   FPGA I/O Node details appear in the node's `<clipname>.vhd` file.
9. Repeat the steps 1 to 6 steps to create additional FPGA I/O Nodes for the CLIP.
   You can drag and drop created FPGA I/O Nodes in the LabVIEW I/O Nodes table.
10. Click **Next** to continue to the Clock Settings page.
    Use the Clock Settings page to configure clock resources for the CLIP.

Related tasks:

- Configuring I/O Pins on the Target
- Configuring Clock Resources for a CLIP

Related information:

- LabVIEW FPGA I/O Method Node

Configuring Clock Resources for a CLIP

Complete the following steps to use the Clock Settings page of the sbRIO CLIP Generator to do the following:

- Specify the pins to connect to LabVIEW FPGA clock lines
- Reserve additional clock resources in the CLIP for the NI sbRIO target.

The sbRIO CLIP Generator will create the VHDL logic required to connect the pins to LabVIEW FPGA clock lines.

1. From the **Available Pins** control, select a pin.
   The **Clock Capability** column specifies one of the following configurations for each pin:
    - **None**—The pin does not provide a direct connection to the clock distribution network on the FPGA.
    - **Single-region**—The pin provides a direct connection to the clock distribution network on a specific region, or bank of pins, on the FPGA.
    - **Multi-region**—The pin provides a direct connection to the clock distribution

network on the entire FPGA.

2. From the **Clock Direction** control, select one of the following options:
   3. **To CLIP**—Exports a clock signal from LabVIEW FPGA to the CLIP.
   4. **From CLIP**—Imports a clock signal from the CLIP to LabVIEW FPGA.



Note For best performance, National Instruments recommends that you configure From CLIP clocks to use pins with single- or multi-region clock capability, depending on how widely your CLIP design will use this clock in the FPGA. Select **Multi-region** if you are not sure which region to select. You can still configure a clock line for a pin that is not clock-capable, but the sbRIO CLIP Generator displays a warning icon for the clock to indicate that performance from this pin might diminish.

5. Click **Add**.
   The **LabVIEW Clocks** table lists the created clock line.
6. If you specified a From CLIP clock, configure the following options in the **Clock Parameters** section. This is required for the LabVIEW FPGA Module to handle the clock signal correctly:
   7. **Frequency in MHz**—The minimum and maximum frequencies the clock signal supports. If the clock signal has a fixed frequency, specify the same value for the minimum and maximum.
   8. **Duty Cycle**—The minimum and maximum percentage of time the clock signal is high.
   9. **Accuracy in PPM**—The amount, in parts per million (PPM), that the clock signal may deviate from its specified frequency under normal operating conditions.
   10. **Jitter in Picoseconds**—The acceptable absolute difference, in picoseconds, between any one clock period and the average clock period. Jitter may also be called *frequency stability*.
   11. **Supports Derived Clocks**—When you enable this option, the sbRIO CLIP Generator adds a **Source Clock Ready** signal and **Derived Clocks Valid** signal to the generated VHDL file for the CLIP.
12. In the **Additional Clock Resources Reserve** section, configure the following options if your CLIP design will include any clock management resources from the FPGA.
   13. **MMCMs**—The number of mixed-mode clock manager (MMCM) resources to reserve.
   14. **BUFGs**—The number of global clock buffer (BUFG) resources to reserve.

15. Repeat the preceding steps to create additional clock lines for the CLIP.
    You can drag and drop created clock lines in the LabVIEW Clocks table.
16. Click **Next** to continue to the CLIP Summary page.
    Use the CLIP Summary page to finish the CLIP design.

Related concepts:

- Finishing the CLIP Design

Related tasks:

- Creating a LabVIEW I/O Node Interface for a CLIP

Finishing the CLIP Design

Complete the following steps to finish using the sbRIO CLIP Generator.

1. Review the information on the CLIP Summary page, including any warnings listed in the **Configuration Warnings** control.
2. Make any changes and resolve warnings that are critical to your application, and click **Finish**.
3. Review the generated CLIP files in the `Documents\LabVIEW Data\CompactRIO\CLIPs\<target>\<clipname>` directory.

> **Note** If you want to make changes to an existing CLIP after it has been created, you must manually edit the generated CLIP files. You cannot use the sbRIO CLIP Generator to edit an existing CLIP.

Optional Tasks to Extend the Generated CLIP Files

(Optional) To support additional CLIP design considerations, complete the following steps to manually edit the generated CLIP files.

1. To add any custom IP or make modifications to your CLIP design, edit the `<clipname>.vhd` file.
2. To reference a file that the sbRIO CLIP Generator did not generate, complete the following steps to edit the `<clipname>.xml` file:

a. Search for the `<ImplementationList>` tag.
b. Within the `<ImplementationList>` tag, add the following code:

```
<Path Name="relative path to file">
   <SimulationFileList>
     <SimulationModelType>Same as synthesis</SimulationModelType>
     </SimulationFileList>
</Path>
```

3. To add timing constraints to your CLIP design, edit the `<clipname>.xdc` file.

After you finish using the sbRIO CLIP Generator, you can complete the following tasks:

- Add the CLIP you created for the NI sbRIO target to a LabVIEW project.
- Implement a LabVIEW FPGA VI that interacts with the CLIP I/O for the NI sbRIO target.

To view information about completing these tasks, open the installed LabVIEW Help and navigate to `NI CompactRIO Device Drivers»Devices»Single-Board RIO`.

Related tasks:

- Configuring Clock Resources for a CLIP

myRIO Module Reference

Use myRIO modules with NI CompactRIO Device Drivers in LabVIEW.

Tip For information about creating applications for myRIO modules, refer to *Getting Started with the myRIO Toolkit*.

Table 685. myRIO Devices

myRIO Module	Description
myRIO-1900	Portable, reconfigurable I/O module. 10 AI channels, 6 AO channels, 40 DIO channels, Xilinx Z-7010 FPGA
myRIO-1950	Portable, reconfigurable I/O module. 10 AI channels, 6 AO channels, 40 DIO channels, Xilinx Z-7010 FPGA
NI ELVIS	Portable, reconfigurable I/O control module. 8 AI channels, 4 AO channels, 32 DIO channels, Xilinx Z-7010 FPGA
NI ELVIS III	Reconfigurable I/O module. 16 AI channels, 4 AO channels, 40 DIO channels, Xilinx Z-7020 FPGA

Note ELVIS refers to *NI Engineering Laboratory Virtual Instrumentation Suite*.

Related information:

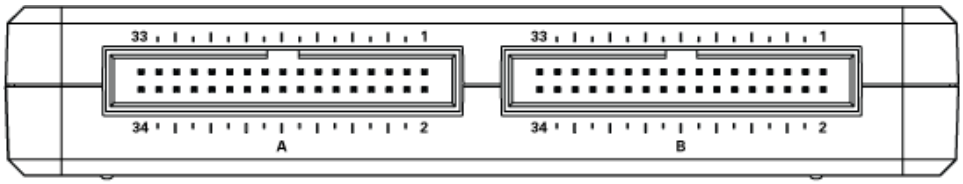
- Getting Started with the myRIO Toolkit
- myRIO-1900 Getting Started Guide and Specifications
- myRIO-1950 Getting Started Guide and Specifications

myRIO-1900

The *NI myRIO-1900 User Guide and Specifications* ships with the LabVIEW myRIO Toolkit.

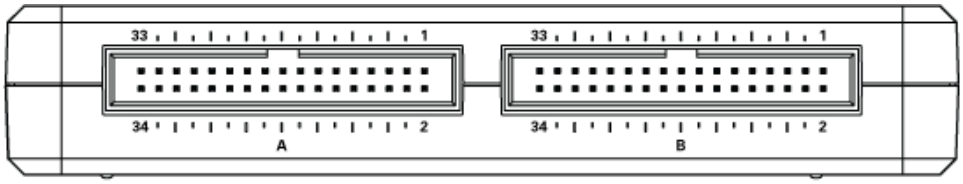
myRIO-1900 Pinouts

Figure 187. Connector A Pinout



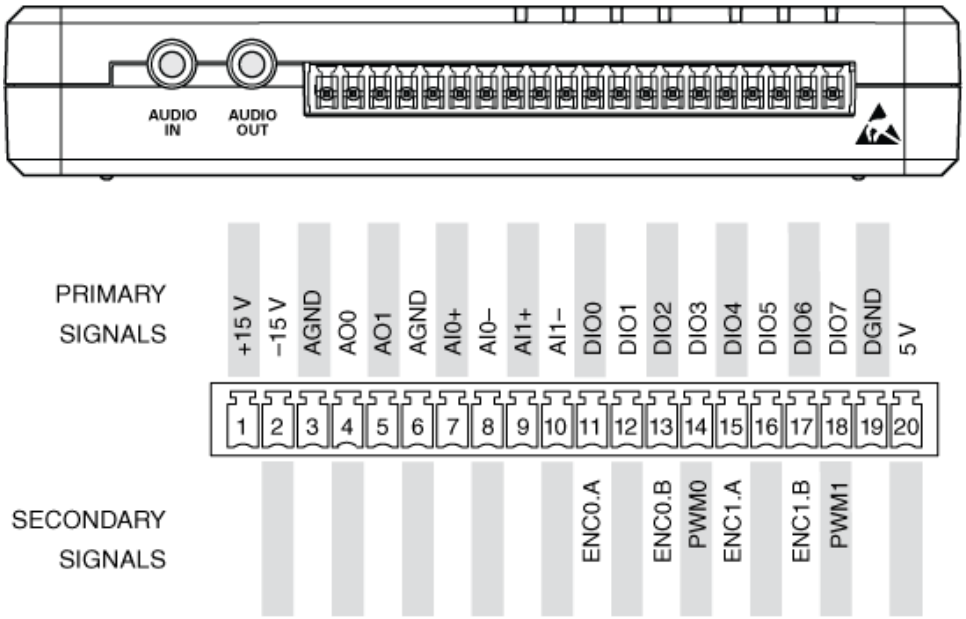
	+3.3 V	DIO10 / PWM2	DIO9 / PWM1	DIO8 / PWM0	DIO7 / SPI.MOSI	DIO6 / SPI.MISO	DIO5 / SPI.CLK	DIO4	DIO3	DIO2	DIO1	DIO0	AI3	AI2	AI1	AI0	+5
PRIMARY/SECONDARY SIGNALS	33	31	29	27	25	23	21	19	17	15	13	11	9	7	5	3	1
	34	32	30	28	26	24	22	20	18	16	14	12	10	8	6	4	2
	DIO15 / I2C.SDA	DIO14 / I2C.SCL	GND	GND	DIO13	GND	DIO12 / ENC.B	GND	DIO11 / ENC.A	GND	UART.TX	GND	UART.RX	GND	GND	AO1	AO0

Figure 188. Connector B Pinout



	+3.3 V	DIO10 / PWM2	DIO9 / PWM1	DIO8 / PWM0	DIO7 / SPI.MOSI	DIO6 / SPI.MISO	DIO5 / SPI.CLK	DIO4	DIO3	DIO2	DIO1	DIO0	AI3	AI2	AI1	AI0	+5
PRIMARY/SECONDARY SIGNALS	33	31	29	27	25	23	21	19	17	15	13	11	9	7	5	3	1
	34	32	30	28	26	24	22	20	18	16	14	12	10	8	6	4	2
	DIO15 / I2C.SDA	DIO14 / I2C.SCL	GND	GND	DIO13	GND	DIO12 / ENC.B	GND	DIO11 / ENC.A	GND	UART.TX	GND	UART.RX	GND	GND	AO1	AO0

Figure 189. Connector C Pinout



Related concepts:

- Programming Interfaces
- Configuring a Project with Connected Hardware
- Configuring a Project with Offline Hardware

myRIO-1900 (FPGA Interface)

Use an FPGA I/O Node configured for reading and writing with this device.

Use the FPGA I/O Node to access the following terminals for this device.

Table 686. Terminals in Software

Terminal	Description
Audio*x*	AudioIn/Left, AudioIn/Right, AudioOut/Left, AudioOut/Right
ConnectorA/AI*x*	Analog input channel *x*, where *x* is the number of the channel. Connector A has AI channels 0 to 3.
ConnectorA/AO*x*	Analog output channel *x*, where *x* is the number of the channel. Connector A has AO channels 0 to 1.
ConnectorA/DIO*y*	Digital input/output channel *y* on Connector A, where *y* is the number of the channel. Connector A has channels 0 to 15. Use the FPGA I/O Node, the Set Data Output method, or the Set Data Enable method to access this channel.

Terminal	Description
ConnectorA/DIO7:0	Digital port consisting of channels 0 to 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
ConnectorA/DIO15:8	Digital port consisting of channels 8 to 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
ConnectorB/AI*x*	Analog input channel *x*, where *x* is the number of the channel. Connector B has AI channels 0 to 3.
ConnectorB/AO*x*	Analog output channel *x*, where *x* is the number of the channel. Connector B has AO channels 0 to 1.
ConnectorB/DIO*y*	Digital input/output channel *y* on Connector B, where *y* is the number of the channel. Connector B has channels 0 to 15. Use the FPGA I/O Node, the Set Data Output method, or the Set Data Enable method to access this channel.
ConnectorB/DIO7:0	Digital port consisting of channels 0 to 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
ConnectorB/DIO15:8	Digital port consisting of channels 8 to 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
ConnectorC/AI*x*	Analog input channel *x*, where *x* is the number of the channel. Connector C has AI channels 0 to 1.
ConnectorC/AO*x*	Analog output channel *x*, where *x* is the number of the channel. Connector C has AO channels 0 to 1.
ConnectorC/DIO*y*	Digital input/output channel *y* on Connector C, where *y* is the number of the channel. Connector C has channels 0 to 7. Use the FPGA I/O Node, the Set Data Output method, or the Set Data Enable method to access this channel.
Onboard I/O	AccelerationX, AccelerationY, and AccelerationZ. Use the FPGA I/O Node to access these channels.
Onboard I/O	Button0, LED0, LED1, LED2, and LED3. Use the FPGA I/O Node to access these channels.

Arbitration

Configure the arbitration settings for the DIO channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 687. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Set Output Enable	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.

Module Methods

This device does not support any module methods.

Properties

This device does not support any properties.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop for digital I/O only. To configure the number of output synchronizing registers or input synchronizing registers for the channels on this device, use the Advanced Code Generation page of the FPGA I/O Properties dialog box.

myRIO-1950

The *NI myRIO-1950 User Guide and Specifications* ship with the LabVIEW myRIO Toolkit.

myRIO-1950 Pinouts

Figure 190. Connector A Pinout

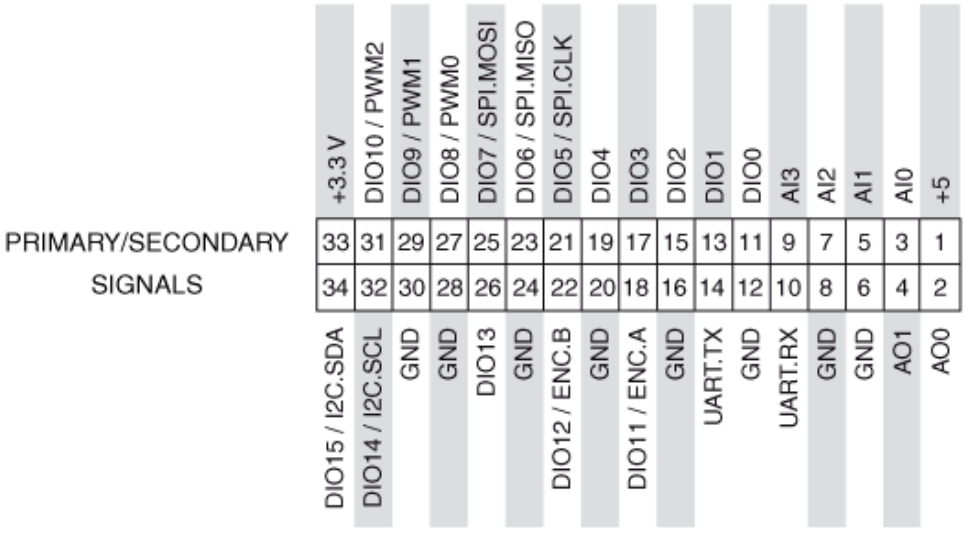


Figure 191. Connector B Pinout

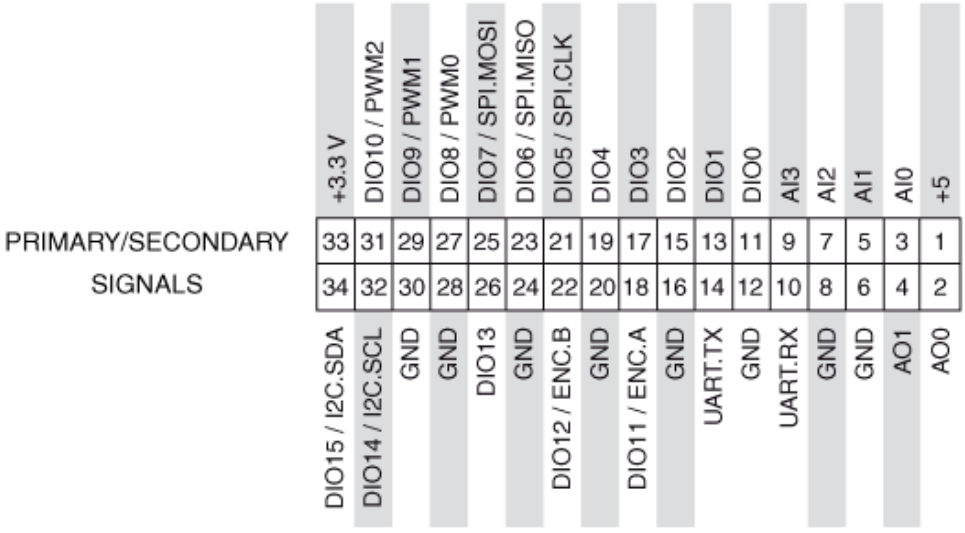
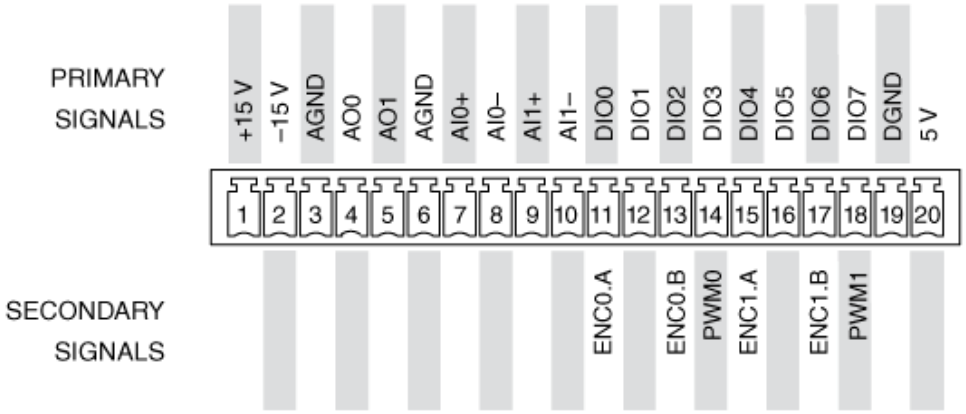


Figure 192. Connector C Pinout



Related concepts:

- Programming Interfaces

myRIO-1950 (FPGA Interface)

Use an FPGA I/O Node configured for reading and writing with this device.

Use the FPGA I/O Node to access the following terminals for this device.

Table 688. Terminals in Software

Terminal	Description
ConnectorA/AI*x*	Analog input channel *x*, where *x* is the number of the channel. Connector A has AI channels 0 to 3.
ConnectorA/AO*x*	Analog output channel *x*, where *x* is the number of the channel. Connector A has AO channels 0 to 1.
ConnectorA/DIO*y*	Digital input/output channel *y* on Connector A, where *y* is the number of the channel. Connector A has channels 0 to 15. Use the FPGA I/O Node, the Set Data Output method, or the Set Data Enable method to access this channel.
ConnectorA/DIO7:0	Digital port consisting of channels 0 to 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
ConnectorA/DIO15:8	Digital port consisting of channels 8 to 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
ConnectorB/AI*x*	Analog input channel *x*, where *x* is the number of the channel. Connector B has AI channels 0 to 3.
ConnectorB/	Analog output channel *x*, where *x* is the number of the channel. Connector B has AO

Terminal	Description
AO*x*	channels 0 to 1.
ConnectorB/DIO*y*	Digital input/output channel *y* on Connector B, where *y* is the number of the channel. Connector B has channels 0 to 15. Use the FPGA I/O Node, the Set Data Output method, or the Set Data Enable method to access this channel.
ConnectorB/DIO7:0	Digital port consisting of channels 0 to 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
ConnectorB/DIO15:8	Digital port consisting of channels 8 to 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
Onboard I/O	AccelerationX, AccelerationY, and AccelerationZ. Use the FPGA I/O Node to access these channels.
Onboard I/O	Button0, LED0, LED1, LED2, and LED3. Use the FPGA I/O Node to access these channels.

Arbitration

Configure the arbitration settings for the DIO channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 689. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Set Output Enable	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.

Module Methods

This device does not support any module methods.

Properties

This device does not support any properties.

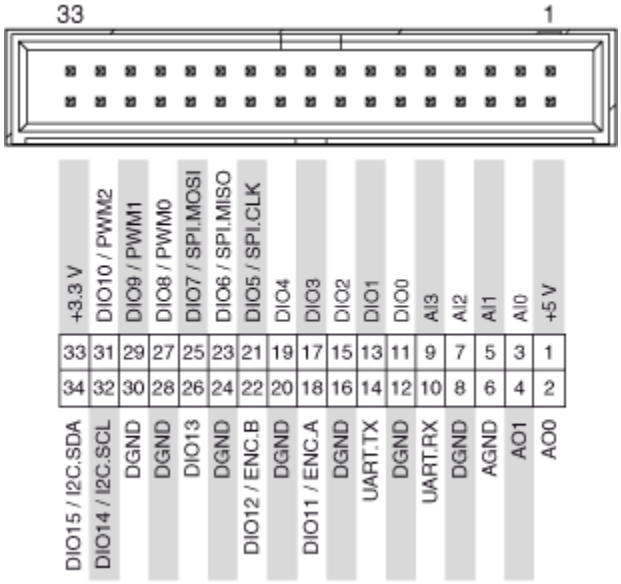
Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop for digital I/O only. To configure the number of output synchronizing registers or input synchronizing registers for the channels on this device, use the Advanced Code Generation page of the FPGA I/O Properties dialog box.

NI ELVIS RIO Control Module

NI ELVIS RIO Control Module Pinout

Figure 193. NI ELVIS RIO Control Module Pinout



Related concepts:

- NI ELVIS RIO Control Module (FPGA Interface)
- Configuring a Project with Connected Hardware
- Configuring a Project with Offline Hardware

NI ELVIS RIO Control Module (FPGA Interface)

Use an FPGA I/O Node configured for reading and writing with this device.

Use the FPGA I/O Node to access the following terminals for this device.

Table 690. Terminals in Software

Terminal	Description
ConnectorA/AI*x*	Analog input channel *x*, where *x* is the number of the channel. Connector A has AI channels 0 to 3.
ConnectorA/AO*x*	Analog output channel *x*, where *x* is the number of the channel. Connector A has AO channels 0 to 1.
ConnectorA/DIO*y*	Digital input/output channel *y* on Connector A, where *y* is the number of the channel. Connector A has channels 0 to 15. Use the FPGA I/O Node, the Set Data Output method, or the Set Data Enable method to access this channel.
ConnectorA/DIO7:0	Digital port consisting of channels 0 to 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
ConnectorA/DIO15:8	Digital port consisting of channels 0 to 7. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
ConnectorB/AI*x*	Analog input channel *x*, where *x* is the number of the channel. Connector B has AI channels 0 to 3.
ConnectorB/AO*x*	Analog output channel *x*, where *x* is the number of the channel. Connector B has AO channels 0 to 1.
ConnectorB/DIO*y*	Digital input/output channel *y* on Connector B, where *y* is the number of the channel. Connector B has channels 0 to 15. Use the FPGA I/O Node, the Set Data Output method, or the Set Data Enable method to access this channel.
ConnectorB/DIO7:0	Digital port consisting of channels 0 to 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
ConnectorB/DIO15:8	Digital port consisting of channels 0 to 7. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
Onboard I/O	Button0, LED0, LED1, LED2, and LED3. Use the FPGA I/O Node to access these channels.

Arbitration

Configure the arbitration settings for the DIO channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 691. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Set Output Enable	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.

Module Methods

This device does not support any module methods.

Properties

This device does not support any properties.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop for digital I/O only. To configure the number of output synchronizing registers or input synchronizing registers for the channels on this device, use the Advanced Code Generation page of the FPGA I/O Properties dialog box.

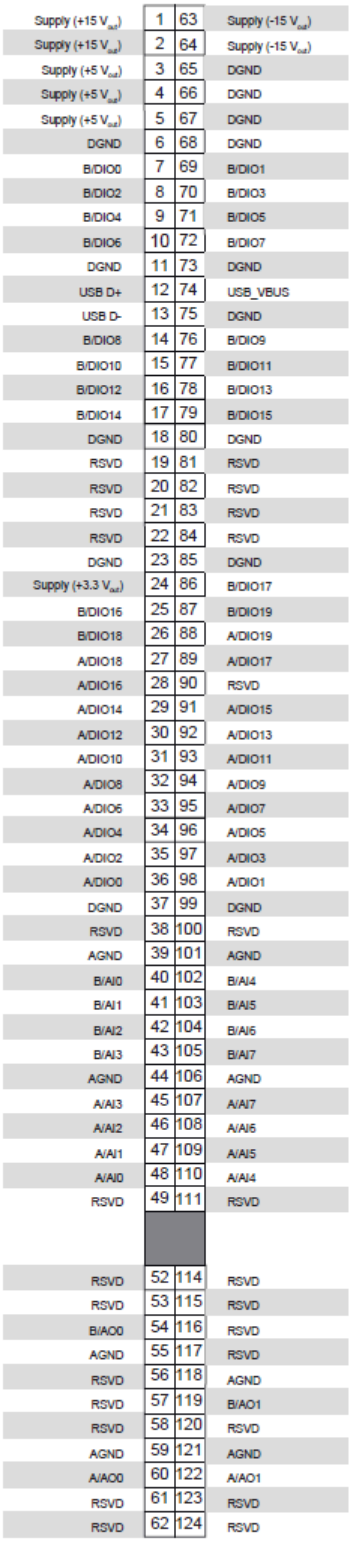
Related concepts:

- Reading from C Series Channels
- Writing to C Series Channels

NI ELVIS III

NI ELVIS III Pinout

Figure 194. NI ELVIS III Pinout



Related concepts:

- Configuring a Project with Connected Hardware
- Configuring a Project with Offline Hardware

NI ELVIS III (FPGA Interface)

Use an FPGA I/O Node configured for reading and writing with this device.

Use the FPGA I/O Node to access the following terminals for this device.

Table 692. Terminals in Software

Terminal	Description
ConnectorA/AI*x*	Analog input channel *x*, where *x* is the number of the channel. Connector A has AI channels 0 to 7.
ConnectorA/AO*x*	Analog output channel *x*, where *x* is the number of the channel. Connector A has AO channels 0 to 1.
ConnectorA/DIO*y*	Digital input/output channel *y* on Connector A, where *y* is the number of the channel. Connector A has channels 0 to 19. DIO 16 and DIO 17 are shared with the UART and Console Out application. When the UART or Console Out application is enabled, DIO 16 functions as the UART Rx line, and DIO 17 fun*x*ctions as the UART Tx line. Use the FPGA I/O Node, the Set Data Output method, or the Set Data Enable method to access this channel.
ConnectorA/DIO7:0	Digital port consisting of channels 0 to 7. Channel 7 is returned in the MSB, and channel 0 is returned in the LSB.
ConnectorA/DIO15:8	Digital port consisting of channels 8 to 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
ConnectorB/AI*x*	Analog input channel , where *x* is the number of the channel. Connector B has AI channels 0 to 7.
ConnectorB/AO*x*	Analog output channel *x*, where *x* is the number of the channel. Connector B has AO channels 0 to 1.
ConnectorB/DIO*y*	Digital input/output channel *y* on Connector B, where *y* is the number of the channel. Connector B has channels 0 to 19. DIO 16 and DIO 17 are shared with the UART and Console Out application. When the UART or Console Out application is enabled, DIO 16 functions as the UART Rx line, and DIO 17 functions as the UART Tx line. Use the FPGA I/O Node, the Set Data Output method, or the Set Data Enable method to access this channel.
ConnectorB/	Digital port consisting of channels 0 to 7. Channel 7 is returned in the MSB, and

Terminal	Description
DIO7:0	channel 0 is returned in the LSB.
ConnectorB/DIO15:8	Digital port consisting of channels 8 to 15. Channel 15 is returned in the MSB, and channel 8 is returned in the LSB.
Onboard I/O	Button0, LED0, LED1, LED2, LED3, Inst Trig Out and Inst Trig In. Button0 returns the value of the onboard button status. LED0:3 controls the four onboard LEDs. Inst Trig Out sends the trigger signal to the instrumentation board. Inst Trig In receives the trigger signal from the instrumentation board. Use the FPGA I/O Node to access these channels.

Note When the application board is turned off, the digital outputs go into tristate and the analog outputs go to 0 V. When the power is turned on, the digital outputs and the analog outputs go to the value that was last set. For the digital outputs, this happen immediately. For the analog outputs, there is a delay for the values to go from 0 V to the last written value. If an analog output operation is invoked while this is happening, it gets held off and may potentially affect the loop timing.

Arbitration

Configure the arbitration settings for the DIO channels of this device in the Advanced Code Generation page of the FPGA I/O Properties dialog box. The default arbitration setting is Never Arbitrate.

I/O Methods

Use the FPGA I/O Method Node to access the following I/O methods for this device.

Table 693. I/O Methods

Method	Description
Set Output Data	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.
Set Output Enable	Refer to the FPGA I/O Method Node topic in the *LabVIEW FPGA Module Programming Reference Manual* for a description of this method.

Module Methods

Table 694. Module Methods

Method	Description
IO Sample	Acquires a single sample from the module. The channel number, terminal mode, and voltage range are configurable at run time.

Note Do not configure the VI to run an IO Sample method and an AI IO Node in parallel when both are targeted to the same connector. Doing so results in an incorrect reading from the IO Sample method. The AI IO Node still gives the correct reading, but will not be able to meet the specified data rate. The same thing applies to having two IO Sample methods run in parallel when both are targeted to the same connector.

Properties

Table 695. Module Properties

Property	Description
Conversion Time (ticks)	Sets the convert to convert period when scanning through a channel List. This property allows users to set a minimum convert to convert period of 1 us and a maximum of 1 ms. This property is in the unit of ticks, with a tick rate of 40MHz or a tick resolution of 25 ns. This property does not affect the conversion time of the IO Sample method.
UART Enable	Enables or disables the UART usage through DIO 16 and DIO 17 for both Connector A and Connector B. UART is disabled by default.
Console Out Enable	Reads the Console Out settings. This property is applicable for Connector A only.
Terminal Mode	Sets the terminal mode for a channel as RSE (referenced single-ended) or DIFF (differential). This property overwrites the value you configure in the **FPGA I/O Properties** dialog box.
Voltage Range	Sets the input range for a channel as ±10 V, ±5 V, ±2 V, or ±1 V. This property overwrites the value you configure in the **FPGA I/O Properties** dialog box.

Single-Cycle Timed Loop

This device supports the single-cycle Timed Loop for digital I/O only. To configure the

number of output synchronizing registers or input synchronizing registers for the channels on this device, use the Advanced Code Generation page of the FPGA I/O Properties dialog box.

Related concepts:

- Reading from C Series Channels
- Writing to C Series Channels
- IO Sample Method (FPGA Interface)

IO Sample Method (FPGA Interface)

The IO Sample Method module method acquires a single sample from the module. The channel number, terminal mode, and voltage range are configurable at run time. To use this module method, select it in an FPGA I/O Method Node that is configured for the appropriate device and/or channel.



Note Do not configure the VI to run an IO Sample method and an AI IO Node in parallel when both are targeted to the same connector. This will cause incorrect reading from the IO Sample method. The AI IO Node still gives the correct reading, but will not be able to meet the specified data rate. The same thing applies to having two IO Sample methods run in parallel when both are targeted to the same connector.

- **Voltage Range [i+1]**—Contains the voltage range setting that gets loaded into the module conversion pipeline. This setting affects the data to be sampled one iteration into the future. Refer to *Conversion Timing* for more information.
- **Terminal Mode [i+1]**—Contains the terminal mode setting that gets loaded into the module conversion pipeline. This setting affects the data to be sampled one iteration into the future. Refer to *Conversion Timing* topic for more information.
- **Channel Number [i+1]**—Contains the channel number that gets loaded into the module conversion pipeline. This setting affects the data to be sampled one iteration into the future. Refer to *Conversion Timing* topic for more information.
- **Data [i]**—Returns the data from the current sample.
- **error in**—Describes error conditions that occur before this VI or function runs. The default is `no error`. If an error occurred before this VI or function runs, the VI or function passes the **error in** value to **error out**. This VI or function runs normally

only if no error occurred before this VI or function runs. If an error occurs while this VI or function runs, it runs normally and sets its own error status in **error out**. Use **error in** and **error out** to check errors and to specify execution order by wiring **error out** from one node to **error in** of the next node.

status	Returns *TRUE* (X) if an error occurred before this VI or function ran. Returns *FALSE* (checkmark) to indicate a warning or that no error occurred before this VI or function ran. The default is *FALSE*.
code	The error code number identifying an error. The default is `0`. If status is *TRUE*, code is a nonzero *error code*. If status is *FALSE*, code is `0` or a warning code.
source	Always contains an empty string. Strings are not supported in LabVIEW FPGA.

- **error out**—Contains error information. If **error in** indicates that an error occurred before this VI or function ran, **error out** contains the same error information. Otherwise, it describes the error status that this VI or function produces. Right-click the **error out** indicator on the front panel and select **Explain Error** from the shortcut menu for more information about the error.

status	Returns *TRUE* (X) if an error occurred before this VI or function ran. Returns *FALSE* (checkmark) to indicate a warning or that no error has occurred.
code	The error code number identifying an error. The default is `0`. The default is `0`. If status is *TRUE*, code is a nonzero *error code*.

	If status is *FALSE*, code is 0 or a warning code.
abc **source**	Always contains an empty string. Strings are not supported in LabVIEW FPGA.

Using This Method

The IO Sample method provides an efficient and flexible interface to the module. Use this method to acquire a single sample from any of the channels on the module, at any range and with any available input mode.

When this method executes, the module performs a single conversion on the next channel present in the conversion pipeline on the module. The data from this conversion is returned via the **Data [i]** method output. While the conversion data is read from the module, the new configuration information specified by the three method inputs (**Voltage Range [i+1]**, **Terminal Mode [i+1]**, and **Channel Number [i+1]**) is loaded into the configuration pipeline on the module. The pipeline is one sample deep. The configuration information specified on one execution of the IO Sample method determines which channel will be sampled by the IO Sample method one iteration into the future. Refer to the *Conversion Timing* topic for more details.

Related concepts:

- Conversion Timing for the NI ELVIS III (FPGA Interface)

Conversion Timing for the NI ELVIS III (FPGA Interface)

The NI ELVIS III implements a one-element deep pipeline to access the AI channels per connector. This pipeline results in maximum sample rate, but may not result in the best resolution. To achieve better resolution, you will need to add your own loop delay and slow down the sampling rate. When using the FPGA I/O Node to sample channels, the pipeline is automatically managed by the FPGA I/O Node, and the channels within the FPGA I/O Node are sampled in numerical order regardless of the order they appear in the node.

If the first channel request in the FPGA I/O Node does not match the first channel

request stored in the module pipeline, there will be a delay before the first sample occurs. This delay is caused by the input settling time, which is required for the input value to reach a steady level before starting conversion.

If the next channel request in the FPGA I/O Node matches the previous channel request, no delay occurs because that channel has already settled. If a change occurs in the Voltage Range setting and Terminal Mode setting,the FPGA I/O Node waits for the settling delay.

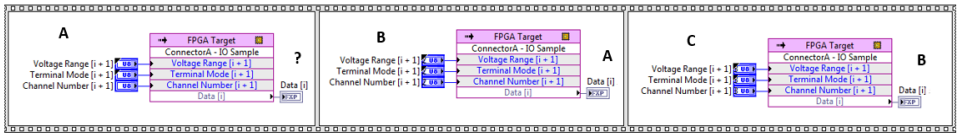
To reduce this settling delay, use the Conversion Time Property Node. This property node allows a range from 40 ticks to 40,000 ticks. One tick corresponds to 25 ns. The convert to convert period is 75 ticks by default.

- Set Conversion Time to 40 ticks to scan through the channel list at 1 MS/s at a lower resolution.
- Set Conversion Time to 40,000 ticks to scan through the channel list at 1 kS/s at a higher resolution.

When using the IO Sample method, take steps to manage this pipeline in the VIs. To read one channel,make the request one cycle before the sample time.

The following diagram illustrates how the pipeline works within a sequence structure. A typical application would use a loop structure to iterate through a scan list, which is a predefined list of channels and settings, continuously. The sequence structure shows how data moves into the ELVIS III pipeline and then is converted.

Figure 195. NI Elvis III Pipeline Sequence Structure, Typical



Related concepts:

- NI ELVIS III (FPGA Interface)
- IO Sample Method (FPGA Interface)

© 2026 National Instruments Corporation.