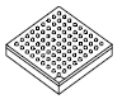
Freescale Semiconductor
Data Sheet: Advanced Information

Document Number: LS102MA
Rev. 1, **10/2014**

QorIQ LS102MA Data Sheet



LS102MA
LS101MA

The high-performance embedded low-power processor LS10xMA, addresses a wide variety of applications ranging from high-end VoIP enabled home gateways, Small-Medium Business (SMB) high performance security appliances, and Ethernet powered 802.11n Enterprise Access Points (EAP).

LS10xMA delivers scalability, superior packet handling capabilities, Quality of Service (QoS) hardware features, increased Virtual Private Network (VPN) and Secure Sockets Layer (SSL) throughput. In order to provide performance scalability and maximum flexibility, the processor includes single and dual ARM®11 core devices from 450 MHz to 650 MHz delivering up to 1600 DMIPS.

The chip leverages the energy efficient ARM core technology and Freescale's low-power design process to achieve the lowest power consumption in its class (<2W typical @ 650 MHz). Additionally, the companion board support package software provides a rich set of power management features to address the energy saving goals of service providers and product manufacturers.

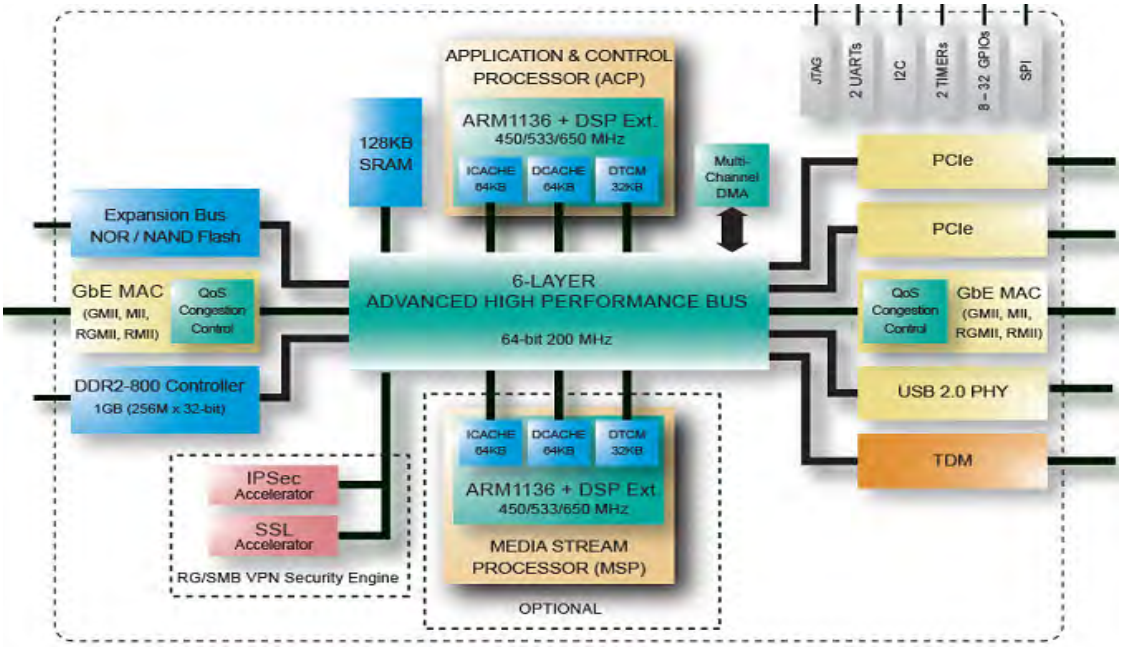
The I/O interfaces in conjunction with multi-layer bus architecture allows non-blocking concurrent transactions across all data interfaces, thus minimizing on-chip packet processing latency.

The OpenWRT Linux-based SDK is optimized for both single-core and dual-core operation. Figure 1-1 shows the block diagram of LS10xMA.

Table of Contents

This figure shows the major functional units within LS10xMA.

Figure 1-1 LS10xMA Block Diagram



1 Product Applications

The LS10xMA can be used as a general purpose processor with the following Data and Voice products applications:

- Intelligent and secure Triple-play wired and wireless home broadband gateways.
- Single or dual concurrent 802.11n EAP using Power-Over-Ethernet (POE).
- SMB router with IPSec VPN / SSL and firewall.
- Micro IP-PBX and Digital Enhanced Cordless Telecommunications (DECT) base stations.

1.1 Features by Device Summary

The LS10xMA devices differ in the number of complex voice channels they support in Data and Voice mode.

CAUTION: **The device is screened for the specified frequency. Freescale cannot guarantee the operation if any changes are made in software. Any changes made without prior approval from Freescale may cause unexpected results and void in the warranty.**

Table 1-1lists the LS10xMA features by device numbers.

Table 1-1 Features By Device Number

ARM 11™ Frequency	Number of Cores	Device Number	VoIP	Security	Ext Temp.
533 MHz	1	LS101MASN7DFA	N	N	N
		LS101MASE7DFA	Y	Y	N
		LS101MAXE7DFA	Y	Y	Y
650 MHz	1	LS101MASN7EHA	N	N	N
		LS101MASE7EHA	Y	Y	N
450 MHz	2	LS102MASN7BFA	N	N	N
		LS102MASE7BFA	Y	Y	N
650 MHz	2	LS102MASN7EHA	N	N	N
		LS102MASE7EHA	Y	Y	N

2 Technical Overview

This section gives an overview of LS10xMA device hardware interfaces, functional blocks, and software interfaces.

CAUTION: **Freescale provides schematic and layout review. However, Freescale HIGHLY RECOMMENDS customers to submit their designs to Freescale for a complete confidential review.** .

2.1 Hardware Interfaces and Functional Blocks

The LS10xMA device provides the following external interfaces and functional blocks:

- Ethernet Interfaces
- Expansion Bus (EXPBUS) provides interfaces to NOR flash, NAND flash, and any other peripheral device that uses a parallel configuration bus
- DDR2 SDRAM Interface—16-Bit/32-Bit DDR2 interface, running at frequencies up to 400 MHz (data rate 800 Mbps per data bit)
- Peripheral Component Interface Express (PCIe)
- Time-Division Multiplexing (TDM) Bus
- Universal Serial Bus (USB) 2.0 Interface—USB 2.0 Host controller with integrated PHY
- Inter-IC Bus (I2C) Master/Slave Interface
- Serial Peripheral Interface (SPI)
- General Purpose Input Output (GPIO) Interface
- Universal Asynchronous Receiver/Transmitter (UART) Serial Interface Port
- Joint Test Action Group (JTAG) Interface—JTAG Interface for Test and debugging
- Timer Interface
- Reference Clock and Reset Controller
- Security Co-Processor
- ARM 1136J-S Processors
- Internal SRAM (ARAM)
- AMBA Advanced High Performance Bus (AHB)
- AHB to APB Bus Bridge
- Memory DMA Engine (MDMA)
- Internal Boot ROM (IBR)
- Interrupt Controller

2.1.1 Ethernet Interfaces

The LS10xMA includes two ethernet ports: WAN (ETH0) and LAN (ETH2). Both Ethernet interfaces support MII, RMII, GMII, RGMII. There is a single MDIO interface for both Ethernet ports. These configurations conform to IEEE 802.3 and support full-duplex and half-duplex operation at 10/100/1000 Mbps. For additional information, see Section 6 Ethernet Interface.

2.1.2 Expansion Bus

The LS10xMA Expansion Bus (EXPBUS) provides address, data, and control lines for connection to system peripheral devices. The EXP Bus provides chip selects for system peripheral devices such as Flash memory, Boot ROM, and so on. Expansion Bus provides support up to 5 peripheral devices: NAND flash and 4 general-purpose device interfaces, that is NOR flash devices. NAND flash chip-select is controlled by software and driven on a GPIO pin. EXPBUS chip selects are configurable and can support up to 16 M of address space. By default, it is configured to 4 M of address space. The Expansion Bus is synchronous and can be driven with the AHB clock using a divide down of 3, 4, 5, 6, or 7, hence providing a maximum of 66 MHz. The expansion bus provides the following features:

- Provides support for multiplexed address-data mode with address latch enable.
- Provides support for Ready/Busy# acknowledge signal for terminating transactions. Detecting the de-assertion of Read/Busy# signal will over-ride the Chip-Select and Write Enable/ Read Enable to be de-asserted for the current transaction. Detection of Ready signal on rising/falling edge is configurable.

For more information, see Chapter 4, Expansion Bus Interface.

2.1.3 DDR2 SDRAM Interface

The LS10xMA device provides address, data, and control lines for connection to DDR2 Synchronous Dynamic Random Access Memory (SDRAM). The DDR2 interface supports the following:

- 32-Bit or 16-Bit data bus interfacing to x8 and x16 memories (x4 not supported)
- Address bus supports up to 1 GB of memory with one chip select
- Supports up to SDRAM at the frequency of 125 MHz up to 400 MHz
- Support for DDR2 power saving modes: Power down mode and self refresh mode
- Supports burst lengths of 4 cycles
- Supports for 4 or 8 internal banks for concurrent operation
- 1.8V I/O (SSTL_18 compatible)

For more information, see Chapter 5, DDR2 SDRAM Interface.

2.1.4 Peripheral Component Interface Express (PCIe)

The LS10xMA device includes two single-lane PCIe interfaces conforming to the *PCI Express Base Specification*, *Revision 1.1*. Each PCIe interface can be independently configured to operate in Root Mode. PCIe interface share the same reference clock inputs, which must be provided by an external source. For more information, see Chapter 7, PCI Express Interface.

2.1.5 Time-Division Multiplexing (TDM) Bus

The TDM Bus interface provides one full-duplex, serial TDM bus providing a maximum of 24/32/64/128 timeslots (running at 1.5444 MHz, 2.048 MHz, 4.096 MHz, or 8.192 MHz, accordingly) for digital data transfer between devices such as Subscriber Line Interface Circuits (SLICs), and the LS10xMA. The TDM Interface timing parameters are programmable. For more information, see Chapter 10, TDM Bus Interface.

2.1.6 Universal Serial Bus (USB) 2.0 Interface

The USB 2.0 Host Interface supports High Speed (480 Mbps) mode, and is backward compatible with USB1.1. USB interface handles all of the USB 2.0 protocol and provides a simple Read/Write protocol to the application software. Some of the features of the USB Controller are:

- USB 2.0 compliance
- Low Speed (LS)—1.5 Mbps, Full Speed (FS)— 12 Mbps, and High Speed (HS)—480 Mbps
- External HUB support
- Integrated PHY
- Supports up to four end points, when configured in device mode

For more information, see Chapter 13, USB Interface.

2.1.7 Inter-IC Bus (I^2C) Master/Slave Interface

The LS10xMA device supports I^2C interface in master, slave, or in multi-master mode. Available Freescale drivers allow LS10xMA device to boot from a serial EEPROM through the I^2C interface. The I^2C Bus is a low-bandwidth, short distance protocol for on board communications. The devices are connected through two wires: Serial data (SDA) and Serial clock (SCL).

All devices must have a unique address to identify it on the bus. Slave devices have a predefined address, but the lower bits of the address can be assigned to allow for multiples of the same devices on the bus. The I^2C EEPROM can be used for booting by assigning 000 address for it.

The I^2C Bus supports multiple data speeds: standard (100 kbps), fast (400 kbps) and High Speed (3.4 Mbps) communications. Other features include:

- Built in collision detection
- Support for 7-Bit and 10-Bit addressing
- Operates as a master, a slave, or in multi-master mode

For more information, see Chapter 12, Inter-IC Interface.

2.1.8 Serial Peripheral Interface (SPI)

A multi-chain SPI provides a common interface mechanism for controlling peripheral devices. For example, SLICs from multiple vendors can be controlled on the SPI Bus. The LS10xMA device supports SPI and can be configured for booting. For more information, see Chapter 8, Serial Peripheral Interface.

2.1.9 General Purpose Input Output (GPIO) Interface

The LS10xMA device provides up to 31 GPIO signals. Of those 31, only 7 can be programmed to create interrupts with rising or falling or both edges of input signals. The interface signals are also level sensitive and can generate interrupt as long as the input is high. The GPIO block holds the configuration registers for the GPIO lines, the block also holds configuration registers for other blocks and input signals. For details, see Chapter 14, General Purpose Input Output.

NOTE: Freescale firmware utilities and example software drivers use GPIOs.To make optimal use of the GPIOs, review planned use of Freescale utilities and drivers with your Freescale representative.

2.1.10 Universal Asynchronous Receiver/Transmitter (UART) Serial Interface Port

The LS10xMA device provides two UART interfaces, with FIFOs. Only the Rx and Tx signals are provided, which can be connected to an RS232 driver for standard serial port operation. The UART interfaces provide a useful console port and can also be used as a software debug port. It is recommended to connect the UART pins to standard connectors such as RJ45, 9-pin DIN for easy access. For more information, see Section 11 Universal Asynchronous Receiver Transmitter.

2.1.11 Joint Test Action Group (JTAG) Interface

The LS10xMA device provides one JTAG interface, which can access both ARM® cores. The JTAG interface provides access to the ARM11 embedded In-Circuit Emulator (ICE) unit and supports IEEE 1149.1 Boundary Scan for manufacturing and test. For more information, see Chapter 17, Test and Debug Interface.

2.1.12 Timer Interface

The LS10xMA timer block includes six general purpose timers each of which can generate an interrupt when the upper limit is reached. Two of the general purpose timers can link to I/O pins and either drive an external output or allow external signals to increment the timer. This block also contains TDM frame sync timer and a watchdog timer.

2.1.13 Reference Clock and Reset Controller

A 24 MHz clock oscillator drives the reference clock interface. The single-clock requirement reduces EMI and system design complexity. For detailed requirements see Section 20.4 Reference Clock.

A Clock and Reset Controller block supports three independent PLLs. The Clock and Reset Controller supports alternative derivations of AHB clock from the reference clock. It also supports independent configuration of clock frequencies for the two ARMs and clock power-down of both ARMs. The Clock and Reset Controller has the following functions:

1. Generates two external GEM Reference clocks
2. Generates external TDM clock
3. Generates ARM and AHB clocks
4. Generates DDR clock – synchronous, pseudo-synchronous, or asynchronous to AHB
5. Independent clock power down for various blocks
6. Independent reset for various blocks
7. Watchdog reset with enable and disable mechanism
8. Global software reset

2.1.14 Security Co-Processor

The LS10xMA has an integrated hardware crypto engine, which can be used for IPSEC and SSL/TLS offloads, key generation acceleration, and random number generation.The security co-processor enables support of NIST-certified VPN technology and supports SSL sessions. As a co-processor, the main advantage this logic core offers over simple cryptographic accelerators is that it can perform crypto operation (cipher + digest) as well as processes IPSec Encapsulating Security Payload (ESP) and Authentication Header (AH) protocols for IPv4 and IPv6.

The co-processor's capabilities include:

- Performs AES128/192/256, DES, 3DES, cipher operation
- Performs MD5, SHA-1, SHA-256 digest operation
- Inserts or removes ESP/AH headers and trailers
- Updates IP header (protocol, length, checksum)
- Manages Security Association Database
- Performs TTL update and anti-replay checks
- Inbound and outbound operation fully processed in parallel
- Uses DMA for packets and contexts (SA)
- Hardware support for SSL
- Concurrent support for SSL sessions and IPSec tunnels
- True Random Number Generation (TRNG) capabilities
- Key acceleration hardware offload

The Security co-processor's DMA can be configured to handle up to 64 packets for ciphering and 64 more for deciphering without any processor intervention. It can also perform "raw cryptographic" processing through the AHB slave interface and an internal Security Resource Manager (SRM). This feature can be used to encrypt/decrypt any non IPSec block of data (AES or SHA1 for voice cipher) in parallel with any IPSec task.

The LS10xMA dual-processor architecture allows the IPSec core to be used by both the Media Stream Processor (MSP) and Application Control Processor (ACP). For example, the MSP can use the IPSec co-processor for a raw cryptographic task such as SRTP helper, or SSL services, while the ACP can use it for IPSec off-loading. For a list of features and complete details, see Chapter 9, Security Accelerator.

2.1.15 ARM 1136J-S Processors

The LS10xMA centers around two ARM1136J-S processors, symmetric in hardware architecture, supporting the ARM DSP extensions. Processor performance is improved by separate 64 KB instruction and data caches, as well as by 32 KB tightly-coupled data memory (DTCM). The DTCM is essentially a private SRAM attached to the ARM processor and mapped to a configurable 32 KByte contiguous address space. It is a software-managed, very fast (ARM speed) scratch pad memory.

The ARM 11's virtual memory system involves translations to physical addresses on each access supported by a hierarchy of Translation Lookaside Buffers (TLBs) and micro-TLBs. Both the physical address and an Application Space Identifier (ASID) index tag, cache entries. The use of both physical addresses and ASIDs, has demonstrated acceleration of context switching in Linux benchmarks.

Each ARM processor provides three 64-Bit AHB master (Instruction, data read and data write) ports. These AHB master ports operate at AMBA AHB interface clock frequency.

2.1.16 Internal SRAM (ARAM)

The ARAM functions as an AHB slave. It provides 128 KB of dual-ported memory arranged as 64-Bit words and is accessible in 8-Bit, 16-Bit, 32-Bit, and 64-Bit widths. The ARAM provides fast internal storage for timing critical data structures. The ARAM controller is used for the following purposes:

- Control and manage the read and writes accesses through the 2 AHB ports to the SRAM
- Control and manage the semaphores mechanism for up to 16K ARAM configurable sections

Up to two masters can access from the AHB at the same time with different transactions at different memory addresses. Only when the access is done to the same address, one of the masters will be delayed until the other master is done. The decision of which master waits for the other to finish will be done according to a fixed or round robin priority, which is set through configuration.

2.1.17 AMBA Advanced High Performance Bus (AHB)

The AMBA Bus from ARM provides a central bus through which all AHB masters must access AHB slaves. This bus connects 14 master-capable and 10 slave-only devices. The AHB block has the following basic functionalities and features:

- Multi-master fixed system with an increased available bus bandwidth
- Multiple layers of arbitration. Controlled by host accessible registers
- 64-bit data bus for most Master/Slaves
- Re-mapping address regions

2.1.18 AHB to APB Bus Bridge

The AHB to APB Bus Bridge (AAB) connects the AHB to the APB. The AAB block converts AHB transfers into a suitable format for the slave devices or peripheral registers on the APB. The bus bridge always functions as a slave on the AHB bus and as a master on the APB bus.

2.1.19 Memory DMA Engine (MDMA)

A Memory DMA (MDMA) block is included to facilitate in inter-CPU data communication (virtual Ethernet interface) and in memory-to-memory data copies. All of these interfaces are connected to the CPU cores with appropriate control logic and DMA controllers. The MDMA block performs low-latency transfers of data blocks from or to ARAM and DDR memories. The MDMA block has the following basic functionalities:

- Performs data copy from one memory region to another
- Supports data management using an array-type list of descriptors
- Support for multiple virtual paths
- Independently set up and serviced by receive side and by transmit side (each possibly controlled by a different CPU)
- Congestion control assistance

2.1.20 Internal Boot ROM (IBR)

A 4096-word by 32-Bit Internal Boot Rom (IBR) provides startup initialization and BIST testing.

2.1.21 Interrupt Controller

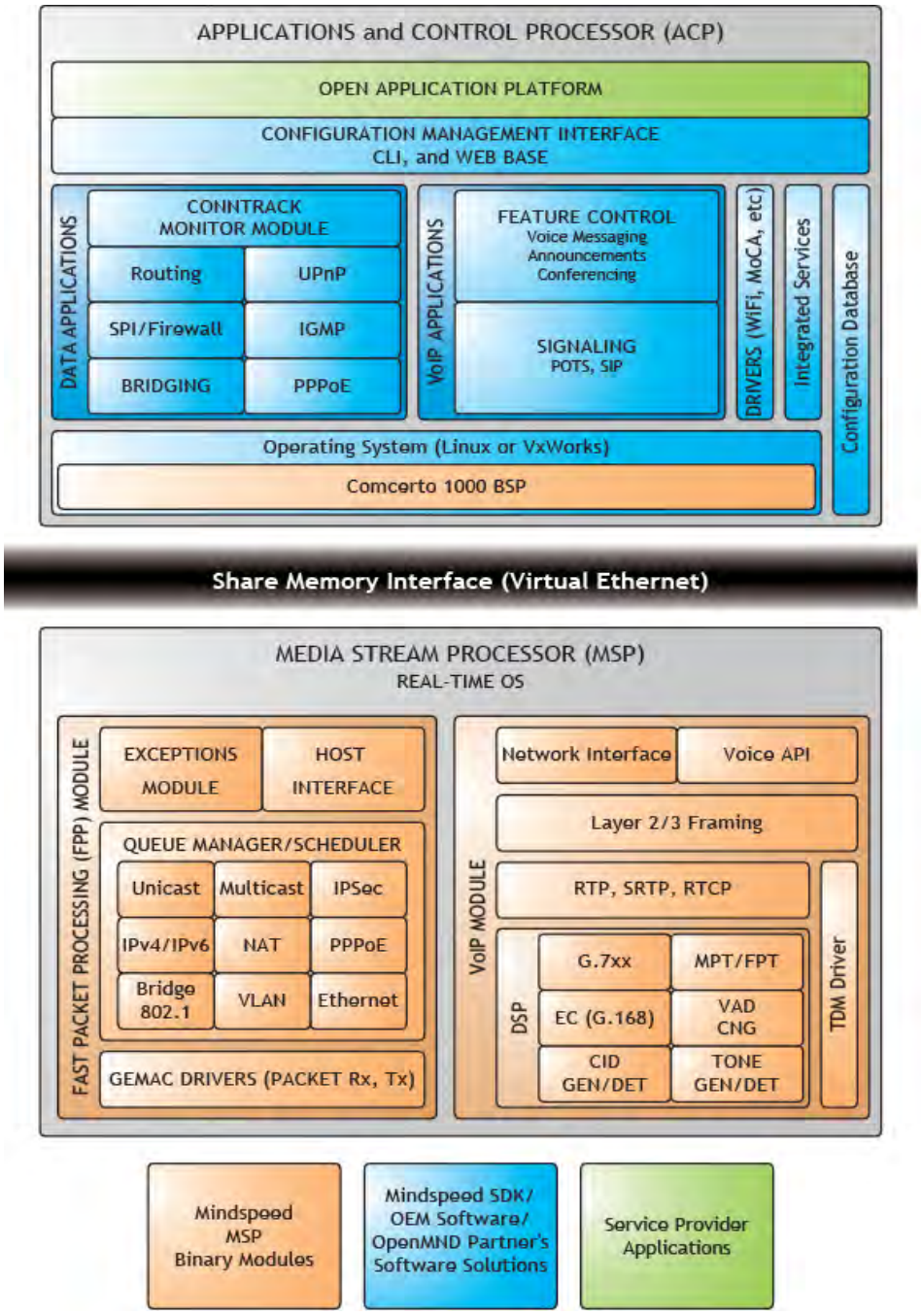
The Interrupt Controller block provides multiple status and mask registers to handle all device peripheral interrupts and generates IRQs and FIQs for the two ARM CPUs. It also contains inter-processor signaling features. The Interrupt Controller device requires multiple interrupt status registers and associated mask registers.

2.2 Software Interfaces

The LS10xMA user applications interface to the hard real time voice code by means of a shared memory interface within the device. This partitioning allows deterministic execution of the voice processing channels, at the same time providing the flexibility and performance to applications.

In the dual ARM11 architecture, one CPU runs the router software and the other CPU handles the Fast Packet Processing (FPP) firmware. The CPU that runs Linux is referred to as the ACP, and the CPU that runs FPP is referred to as the MSP. Figure 1-1 illustrates the dual core architecture and the overall partition between the ACP and MSP in LS10xMA devices. The VoIP module runs on the MSP, in parallel to the FPP module.

Figure 2-1 Software Architecture



2.2.1 Embedded Host Controller Firmware

The LS10xMA device is provided with a Board Support Package (BSP) for popular Operating Systems (OS) that provides the kernel operating environment for the host application. Applications are built and debugged on this device using standard third party ARM tool chains and debuggers.

2.2.2 Voice Channel Firmware

Voice channel firmware is provided in ARM AXF format for downloading to SDRAM, at boot time. This image contains code for FPP, signal processing, and encryption (if applicable).

2.2.3 Development Tools

The development tools include:

- JTAG-based ICE
- U-Boot commands and diagnostics
- Core dump procedures

3 Pinlist and Signal Summary

This section presents pin information, I/O pad details, and signal descriptions of the LS10xMA device. For a LS10xMA pin list as in '.txt' or '.xls' format, contact your Freescale sales representative. Figure 3-1illustrates the LS10xMA device top view.

Figure 3-1 Device Top View

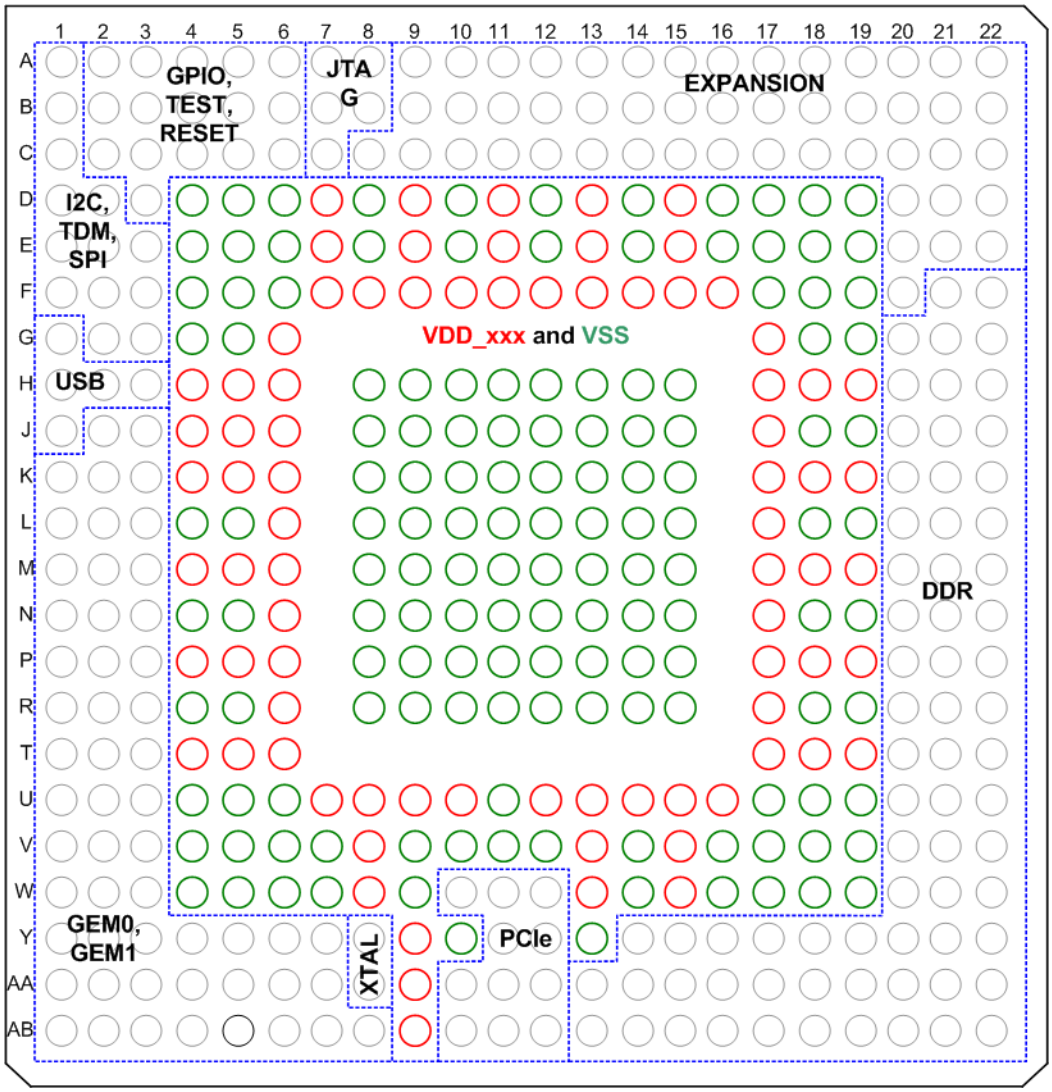


Table 3-1 lists the Pin details of LS10xMA device. For description about I/O pad types see Table 3-2, I/O Type Descriptions.

3.1 Pin List

Table 3-1 Landing List

Location	Signal Symbol	Dir	I/O Type
A1	I2C_SCL /GPIO_18	Both (B)	Ihu/Ots8
A2	GPIO_1	B	Ihu/Ots16
A3	GPIO_0	B	Ihu/Ots16
A4	GPIO_3	B	Ihu/Ots16
A5	UART1_RX / GPIO_22	B	Ihu/Ots16
A6	UART0_TX	Output (O)	Ots8
A7	JTAG_TRST_N	Input (I)	Ihu
A8	JTAG_TDO	O	Ots8
A9	EXP_A_2 /TM_ETH0_MODE_0	B	Id/Ots8
A10	EXP_A_5 /TM_ETH1_MODE_1	B	Id/Ots8
A11	EXP_A_17 /GPIO_10	B	Ihd/Ots8
A12	EXP_WE_N	O	Ots8
A13	EXP_A_8 /TM_TDM_CLKSRC_ EN	B	Id/Ots8
A14	EXP_A_11 /TM_GPBT_OP1	B	Id/Ots8
A15	EXP_A_14 /GPIO_14	B	Ihd/Ots8
A16	EXP_NAND_RE_N /EXP_A_22	B	Ihu/Ots8
A17	EXP_DQ_7	B	I/Ots8
A18	EXP_DQ_13	B	I/Ots8
A19	EXP_DQ_12	B	I/Ots8
A20	EXP_DQ_3	B	I/Ots8
A21	EXP_DQ_9	B	I/Ots8
A22	EXP_DQ_8	B	I/Ots8
B1	I2C_SDA /GPIO_19	B	Ihu/Ots8
B2	TIM_EVNT1 /GPIO_16	B	Ihu/Ots8
B3	TM_EXT_RESET /GPIO_17	B	Ihd/Ots8
B4	GPIO_2	B	Ihu/Ots16
B5	EXP_IRQ / GPIO_7	B	Ihu/Ots16
B6	UART0_RX	I	Ihu
B7	JTAG_TCK	I	Ihu
B8	JTAG_TMS	I	Ihu
B9	EXP_A_3 /TM_ETH0_MODE_1	B	Id/Ots8
B10	EXP_A_6 /TM_BOOT_OP1	B	Id/Ots8
B11	EXP_NAND_CS /EXP_A_18 / GPIO_29	B	Ihu/Ots8

Table 3-1 Landing List (continued)

Location	Signal Symbol	Dir	I/O Type
B12	EXP_NAND_CLE /EXP_A_20 /GPIO_31	B	Ihu/Ots8
B13	EXP_A_9 /TM_EXP_NAND_SEL	B	Id/Ots8
B14	EXP_A_12 /TM_PCIE_EXT_ REFCLK	B	Id/Ots8
B15	EXP_A_15 /GPIO_8	B	Ihd/Ots8
B16	EXP_A_16 /GPIO_9	B	Ihd/Ots8
B17	EXP_DQ_14	B	I/Ots8
B18	EXP_DQ_5	B	I/Ots8
B19	EXP_DQ_4	B	I/Ots8
B20	EXP_DQ_10	B	I/Ots8
B21	EXP_DQ_1	B	I/Ots8
B22	EXP_DQ_0	B	I/Ots8
C1	TDM_DR	I	Ihd
C2	TIM_EVNT0 /GPIO_15	B	Ihu/Ots8
C3	RESET_N	I	Ih
C4	TM_BSCANMODE	I	Id
C5	GPIO_5 /EXP_CS3_N	B	Ihu/Ots16
C6	UART1_TX /GPIO_23	B	Ihu/Ots16
C7	JTAG_TDI	I	Ihu
C8	EXP_A_1 /TM_PCIE1_MODE	B	Id/Ots8
C9	EXP_A_4 / TM_ETH1_MODE_0	B	Id/Ots8
C10	EXP_A_7 /TM_BOOT_OP_0	B	Id/Ots8
C11	EXP_NAND_RDY/EXP_A_21 / GPIO_6	B	Ihu/Ots8
C12	EXP_NAND_ALE /EXP_A_19 / GPIO_30	B	Ihu/Ots8
C13	EXP_A_10 /TM_GPBT_OP0	B	Id/Ots8
C14	EXP_A_13 /GPIO_11	B	Ihd/Ots8
C15	EXP_NAND_WE_N /EXP_A_23	B	Ihu/Ots8
C16	EXP_DQ_15	B	I/Ots8
C17	EXP_DQ_6	B	I/Ots8
C18	EXP_DQ_11	B	I/Ots8
C19	EXP_DQ_2	B	I/Ots8
C20	EXP_DM_1	O	Ots8
C21	EXP_DM_0	O	Ots8
C22	EXP_CLK	O	Ot8

Table 3-1 Landing List (continued)

Location	Signal Symbol	Dir	I/O Type
D1	TDM_FS	B	Ih/Ot8
D2	TDM_DX	O*	Id/Ots8
D3	TM_TESTMODE_N	I	Iu
D4	VSS	—	GND
D5	VSS	—	GND
D6	VSS	—	GND
D7	VDDO_EXP	—	PWR
D8	VSS	—	GND
D9	VDDO_EXP	—	PWR
D10	VSS	—	GND
D11	VDDO_EXP	—	PWR
D12	VSS	—	GND
D13	VDDO_EXP	—	PWR
D14	VSS	—	GND
D15	VDDO_EXP	—	PWR
D16	VSS	—	GND
D17	VSS	—	GND
D18	VSS	—	GND
D19	VSS	—	GND
D20	EXP_RE_N	O	Ots8
D21	EXP_RDY/BSY_N /GPIO_21	B	Ihu/Ots8
D22	EXP_ALE / GPIO_20	B	Ihd/Ots8
E1	SPI_SCLK /GPIO_24	B	Ihu/Ots8
E2	SPI_RXD /GPIO_12	B	Ihd/Ots8
E3	TDM_CK	B	Ih/Ot16
E4	VSS	—	GND
E5	VSS	—	GND
E6	VSS	—	GND
E7	VDDO_EXP	—	PWR
E8	VSS	—	GND
***- Though it has bidirectional pad, it is used as output only.**			
E9	VDDO_EXP	—	PWR
E10	VSS	—	GND
E11	VDDO_EXP	—	PWR
E12	VSS	—	GND

Table 3-1 Landing List (continued)

Location	Signal Symbol	Dir	I/O Type
E13	VDDO_EXP	—	PWR
E14	VSS	—	GND
E15	VDDO_EXP	—	PWR
E16	VSS	—	GND
E17	VSS	—	GND
E18	VSS	—	GND
E19	VSS	—	GND
E20	EXP_CS2_N	O	Ots8
E21	EXP_CS1_N	O	Ots8
E22	EXP_CS0_N	O	Ots8
F1	SPI_SS1_N /GPIO_26	B	Ihu/Ots8
F2	SPI_SS2_N /GPIO_27	B	Ihu/Ots8
F3	SPI_TXD /GPIO_25	B	Ihu/Ots8
F4	VSS	—	GND
F5	VSS	—	GND
F6	VSS	—	GND
F7	VDDO_EXP	—	PWR
F8	VDD	—	PWR
F9	VDDO_EXP	—	PWR
F10	VDD	—	PWR
F11	VDDO_EXP	—	PWR
F12	VDD	—	PWR
F13	VDDO_EXP	—	PWR
F14	VDD	—	PWR
F15	VDDO_EXP	—	PWR
F16	VDDO_EXP	—	PWR
F17	VSS	—	GND
F18	VSS	—	GND
F19	VSS	—	GND
F20	EXP_A_0 /TM_PCIE0_MODE	B	Id/Ots8
F21	DDR_DATA_5	B	MEMIO
F22	DDR_DATA_2	B	MEMIO
G1	USB_VBUS_STAT	I	I
G2	SPI_SS0_N /GPIO_13	B	Ihu/Ots8
G3	GPIO_28	B	Ihu/Ots8
G4	VSS	—	GND

Table 3-1 Landing List (continued)

Location	Signal Symbol	Dir	I/O Type
G5	VSS	—	GND
G6	VDD	—	PWR
G7	—	—	—
G8	—	—	—
G9	—	—	—
G10	—	—	—
G11	—	—	—
G12	—	—	—
G13	—	—	—
G14	—	—	—
G15	—	—	—
G16	—	—	—
G17	VDD	—	PWR
G18	VSS	—	GND
G19	VSS	—	GND
G20	DDR_DATA_0	B	MEMIO
G21	DDR_DATA_7	B	MEMIO
G22	DDR_DQS0_N	B	MEMIODIF
H1	USB_DM	B	USBDATA
H2	Do Not Connect	B	USBANALOG
H3	USB_REF	—	USBANALOG
H4	VDDO_EXP	—	PWR
H5	VDDO_EXP	—	PWR
H6	VDDO_EXP	—	PWR
H7	—	—	—
H8	VSS	—	GND
H9	VSS	—	GND
H10	VSS	—	GND
H11	VSS	—	GND
H12	VSS	—	GND
H13	VSS	—	GND
H14	VSS	—	GND
H15	VSS	—	GND
H16	—	—	—
H17	VDDO_DDR2	—	PWR
H18	VDDO_DDR2	—	PWR

Table 3-1 Landing List (continued)

Location	Signal Symbol	Dir	I/O Type
H19	VDDO_DDR2	—	PWR
H20	DDR_DQS_0	B	MEMIODIF
H21	DDR_DM_0	O	MEMIO
H22	DDR_DATA_6	B	MEMIO
J1	USB_DP	B	USBDATA
J2	GMII0_RXD4	I	I
J3	GMII0_RXD0 /RGMII0_RXD0 /MII0_RXD0 /RMII0_RXD0	I	I
J4	VDDA25_USB	—	PWR
J5	VDDA25_USB	—	PWR
J6	VDDA25_USB	—	PWR
J7	—	—	—
J8	VSS	—	GND
J9	VSS	—	GND
J10	VSS	—	GND
J11	VSS	—	GND
J12	VSS	—	GND
J13	VSS	—	GND
J14	VSS	—	GND
J15	VSS	—	GND
J16	—	—	—
J17	VDD	—	PWR
J18	VSS	—	GND
J19	VSS	—	GND
J20	DDR_DATA_1	B	MEMIO
J21	DDR_DATA_3	B	MEMIO
J22	DDR_DATA_4	B	MEMIO
K1	GMII0_RXD2/RGMII0_RXD2/ MII0_RXD2/RMII0_PHY_DPX	I	I
K2	GMII0_RXD5	I	I
K3	GMII0_RXD1 /RGMII0_RXD1/ MII0_RXD1/RMII0_RXD1	I	I
K4	VDDA33_USB	—	PWR
K5	VDDA33_USB	—	PWR
K6	VDDA33_USB	—	PWR
K7	—	—	—
K8	VSS	—	GND
K9	VSS	—	GND

Table 3-1 Landing List (continued)

Location	Signal Symbol	Dir	I/O Type
K10	VSS	—	GND
K11	VSS	—	GND
K12	VSS	—	GND
K13	VSS	—	GND
K14	VSS	—	GND
K15	VSS	—	GND
K16	—	—	—
K17	VDDO_DDR2	—	PWR
K18	VDD_REF	—	PWR
K19	VDD_REF	—	PWR
K20	DDR_DATA_13	B	MEMIO
K21	DDR_DATA_10	B	MEMIO
K22	DDR_DATA_8	B	MEMIO
L1	GMII0_RXD7	I	I
L2	GMII0_RXD3 /RGMII0_RXD3 /MII0_RXD3 /RMII0_PHY_SPEED	I	I
L3	GMII0_RXD6	I	I
L4	VSS	—	GND
L5	VSS	—	GND
L6	VDD	—	PWR
L7	—	—	—
L8	VSS	—	GND
L9	VSS	—	GND
L10	VSS	—	GND
L11	VSS	—	GND
L12	VSS	—	GND
L13	VSS	—	GND
L14	VSS	—	GND
L15	VSS	—	GND
L16	—	—	—
L17	VDD		PWR
L18	VSS	—	GND
L19	VSS	—	GND
L20	DDR_DATA_15	B	MEMIO
L21	DDR_DQS_1_N	B	MEMIODIF
L22	DDR_DQS_1	B	MEMIODIF

Table 3-1 Landing List (continued)

Location	Signal Symbol	Dir	I/O Type
M1	GMII0_RX_DV / RGMII0_RX_CTL / MII0_RX_DV / RMII0_CRS_DV	I	I
M2	GMII0_RX_ER / MII0_RX_ER	I	I
M3	GMII0_RX_CLK / RGMII0_RXC / MII0_RX_CLK / RMII0_CLK	I	I
M4	VDDO_ETH0	—	PWR
M5	VDDO_ETH0	—	PWR
M6	VDDO_ETH0	—	PWR
M7	—	—	—
M8	VSS	—	GND
M9	VSS	—	GND
M10	VSS	—	GND
M11	VSS	—	GND
M12	VSS	—	GND
M13	VSS	—	GND
M14	VSS	—	GND
M15	VSS	—	GND
M16	—	—	—
M17	VDDO_DDR2	—	PWR
M18	VDDO_DDR2	—	PWR
M19	VDDO_DDR2	—	PWR
M20	DDR_DM_1	O	MEMIO
M21	DDR_DATA_14	B	MEMIO
M22	DDR_DATA_9	B	MEMIO
N1	GMII0_CRS / MII0_CRS	I	I
N2	GMII0_REFCLK/RGMII0_REFCLK/ MII0_REFCLK/ RMII0_REFCLK	O	Ot16
N3	GMII0_COL/MII0_COL	I	I
N4	VSS	—	GND
N5	VSS	—	GND
N6	VDD	—	PWR
N7	—	—	—

Table 3-1 Landing List (continued)

Location	Signal Symbol	Dir	I/O Type
N8	VSS	—	GND
N9	VSS	—	GND
N10	VSS	—	GND
N11	VSS	—	GND
N12	VSS	—	GND
N13	VSS	—	GND
N14	VSS	—	GND
N15	VSS	—	GND
N16	—	—	—
N17	VDD	—	PWR
N18	VSS	—	GND
N19	VSS	—	GND
N20	DDR_DATA_11	B	MEMIO
N21	DDR_DATA_12	B	MEMIO
N22	DDR_CLK_N	O	MEMIODIF
P1	GMII0_TX__ER / RGMII0_TX_CTL /MII0_TX_ER	O	Ot16
P2	GMII0_TX_CLK / MII0_TX_CLK	I	I
P3	GMII0_TX_EN /RGMII0_TXC / MII0_TX_EN /RMII0_TX_EN	O	Ot16
P4	VDDO_ETH0	—	PWR
P5	VDDO_ETH0	—	PWR
P6	VDDO_ETH0	—	PWR
P7	—	—	—
P8	VSS	—	GND
P9	VSS	—	GND
P10	VSS	—	GND
P11	VSS	—	GND
P12	VSS	—	GND
P13	VSS	—	GND
P14	VSS	—	GND
P15	VSS	—	GND
P16	—	—	—
P17	VDDO_DDR2	—	PWR
P18	VDDO_DDR2	—	PWR
P19	VDDO_DDR2	—	PWR

Table 3-1 Landing List (continued)

Location	Signal Symbol	Dir	I/O Type
P20	DDR_RAS_N	O	MEMIO
P21	DDR_ODT	O	MEMIO
P22	DDR_CLK	O	MEMIODIF
R1	GMII0_TXD4	O	Ot16
R2	GMII0_TXD0 /RGMII0_TXD0 / MII0_TXD0 /RMII0_TXD0	O	Ot16
R3	GMII0_TXD5	O	Ot16
R4	VSS	—	GND
R5	VSS	—	GND
R6	VDD	—	PWR
R7	—	—	—
R8	VSS	—	GND
R9	VSS	—	GND
R10	VSS	—	GND
R11	VSS	—	GND
R12	VSS	—	GND
R13	VSS	—	GND
R14	VSS	—	GND
R15	VSS	—	GND
R16	—	—	—
R17	VDD	—	PWR
R18	VSS	—	GND
R19	VSS	—	GND
R20	DDR_CAS_N	O	MEMIO
R21	DDR_CLKE	O	MEMIOPL
R22	DDR_WE_N	O	MEMIO
T1	GMII0_TXD1/RGMII0_TXD1/ MII0_TXD1 /RMII0_TXD1	O	Ot16
T2	GMII0_TXD6	O	Ot16
T3	GMII0_TXD2/RGMII0_TXD2/ MII0_TXD2/RMII0_RX_ER	B	I/Ot16
T4	VDDO_ETH1	—	PWR
T5	VDDO_ETH1	—	PWR
T6	VDDO_ETH1	—	PWR
T7	—	—	—
T8	—	—	—
T9	—	—	—

Table 3-1 Landing List (continued)

Location	Signal Symbol	Dir	I/O Type
T10	—	—	—
T11	—	—	—
T12	—	—	—
T13	—	—	—
T14	—	—	—
T15	—	—	—
T16	—	—	—
T17	VDDO_DDR2	—	PWR
T18	VDDO_DDR2	—	PWR
T19	VDDO_DDR2	—	PWR
T20	DDR_A_10	O	MEMIO
T21	DDR_BA_0	O	MEMIO
T22	DDR_CS0_N	O	MEMIO
U1	GMII0_TXD7	O	Ot16
U2	GMII0_TXD3/RGMII0_TXD3/ MII0_TXD3/RMII0_PHY_LINK	B	I/Ot16
U3	GMII0_GTX_CLK	O	Ot16
U4	VSS	—	GND
U5	VSS	—	GND
U6	VSS	—	GND
U7	VDD	—	PWR
U8	VDDO_ETH1	—	VDD
U9	VDD	—	PWR
U10	VDDO_PCIE_2.5	—	PWR
U11	VSS	—	GND
U12	VDD_PCIE_1	—	PWR
U13	VDDO_DDR2	—	PWR
U14	VDD	—	PWR
U15	VDDO_DDR2	—	PWR
U16	VDD	—	PWR
U17	VSS	—	GND
U18	VSS	—	GND
U19	VSS	—	GND
U20	DDR_A_6	O	MEMIO
U21	DDR_A_3	O	MEMIO
U22	DDR_A_2	O	MEMIO

Table 3-1 Landing List (continued)

Location	Signal Symbol	Dir	I/O Type
V1	GEM_MDC	O	Ots8
V2	GEM_MDIO	B	I/Ots8
V3	GMII1_GTX_CLK	O	Ot16
V4	VSS	—	GND
V5	VSS	—	GND
V6	VSS	—	GND
V7	VSS	—	GND
V8	VDDO_ETH1	—	PWR
V9	VSS	—	GND
V10	VSS	—	GND
V11	VSS	—	GND
V12	VSS	—	GND
V13	VDDO_DDR2	—	PWR
V14	VSS	—	GND
V15	VDD_REF	—	—
V16	VSS	—	GND
V17	VSS	—	GND
V18	VSS	—	GND
V19	VSS	—	GND
V20	DDR_A_7	O	MEMIO
V21	DDR_A_11	O	MEMIO
V22	DDR_A_13	O	MEMIO
W1	GMII1_TXD3/RGMII1_TXD3/ MII1_TXD3/RMII1_PHY_LINK	B	I/Ot16
W2	GMII1_TXD7	O	Ot16
W3	GMII1_TX_EN/RGMII1_TXC/ MII1_TX_EN/RMII1_TX_EN	O	Ot16
W4	VSS	—	GND
W5	VSS	—	GND
W6	VSS	—	GND
W7	VSS	—	GND
W8	VDDO_ETH1	—	PWR
W9	VSS	—	GND
W10	PCIE_RESREF	—	PCIE
W11	PCIE1_HSI_P	I	PCIE
W12	PCIE0_HSI_P	I	PCIE
W13	VDDO_DDR2	—	PWR

Table 3-1 Landing List (continued)

Location	Signal Symbol	Dir	I/O Type
W14	VSS	—	GND
W15	VDD_REF	—	PWR
W16	VSS	—	GND
W17	VSS	—	GND
W18	VSS	—	GND
W19	VSS	—	GND
W20	DDR_A_8	O	MEMIO
W21	DDR_A_12	O	MEMIO
W22	DDR_A_9	O	MEMIO
Y1	GMII1_TXD2/RGMII1_TXD2 / MII1_TXD2/RMII1_RX_ER	B	I/Ot16
Y2	GMII1_TXD5	O	Ot16
Y3	GMII1_TX_CLK/MII1_TX_CLK	I	I
Y4	GMII1_REFCLK/RGMII1_REFCLK/ MII1_REFCLK/ RMII1_REFCLK	O	Ot16
Y5	GMII1_RX_ER / MII1_RX_ER	I	I
Y6	GMII1_RXD3 / RGMII1_RXD3 / MII1_RXD3 / RMII1_PHY_SPEED	I	I
Y7	GMII1_RXD4	I	I
Y8	TM_PLL_XO	O	OSCL
Y9	VDD_PLL_2	—	PWR
Y10	VSS	—	GND
Y11	PCIE1_HSI_M	I	PCIE
Y12	PCIE0_HSI_M	I	PCIE
Y13	VSS	—	GND
Y14	DDR_DM_3	O	MEMEIO
Y15	DDR_DATA_31	B	MEMEIO
Y16	DDR_DATA_29	B	MEMEIO
Y17	DDR_DATA_17	B	MEMEIO
Y18	DDR_DQS_2	B	MEMIODIF
Y19	DDR_A_14	O	MEMIO
Y20	DDR_A_0	O	MEMIO
Y21	DDR_A_5	O	MEMIO
Y22	DDR_A_4	O	MEMIO

Table 3-1 Landing List (continued)

Location	Signal Symbol	Dir	I/O Type
AA1	GMII1_TXD6	O	Ot16
AA2	GMII1_TXD0 / RGMII1_TXD0 / MII1_TXD0 / RMII1_TXD0	O	Ot16
AA3	GMII1_COL / MII1_COL	I	I
AA4	GMII1_CRS / MII1_CRS	I	I
AA5	GMII1_RX_CLK/RGMII1_RXC /MII1_RX_CLK/RMII1_CLK	I	I
AA6	GMII1_RXD6	I	I
AA7	GMII1_RXD1/RGMII1_RXD1 / MII1_RXD1/RMII1_RXD1	I	I
AA8	TM_PLL_XI	I	OSCL
AA9	VDD_PLL_1	—	PWR
AA10	PCIE_REFCLK_M	I	PCIE
AA11	PCIE1_HSO_P	O	PCIE
AA12	PCIE0_HSO_P	O	CIE
AA13	DDR_DATA_27	B	MEMIO
AA14	DDR_DATA_30	B	MEMIO
AA15	DDR_DQS_3_N	B	MEMIODIF
AA16	DDR_DATA_26	B	MEMIO
AA17	DDR_DATA_19	B	MEMIO
AA18	DDR_DM_2	O	MEMIO
AA19	DDR_DATA_23	B	MEMIO
AA20	DDR_DATA_18	B	MEMIO
AA21	DDR_BA_1	O	MEMIO
AA22	DDR_A_1	O	MEMIO
AB1	GMII1_TXD1/RGMII1_TXD1/ MII1_TXD1/RMII1_TXD1	O	Ot16
AB2	GMII1_TXD4	O	Ot16
AB3	GMII1_TX__ER/RGMII1_TX_CTL/ MII1_TX_ER	O	Ot16
AB4	GMII1_RX_DV /RGMII1_RX_CTL / MII1_RX_DV / RMII1_CRS_DV	I	I
AB5	GMII1_RXD7	I	I

Table 3-1 Landing List (continued)

Location	Signal Symbol	Dir	I/O Type
AB6	GMII1_RXD2 / RGMII1_RXD2 / MII1_RXD2 / RMII1_PHY_DPX	I	I
AB7	GMII1_RXD5	I	I
AB8	GMII1_RXD0 /RGMII1_RXD0/ MII1_RXD0 /RMII1_RXD0	I	I
AB9	VDD_PLL_0	—	PWR
AB10	PCIE_REFCLK_P	I	PCIE
AB11	PCIE1_HSO_M	O	PCIE
AB12	PCIE0_HSO_M	O	PCIE
AB13	DDR_DATA_28	B	MEMIO
AB14	DDR_DATA_25	B	MEMIO
AB15	DDR_DQS_3	B	MEMIODIF
AB16	DDR_DATA_24	B	MEMIO
AB17	DDR_DATA_20	B	MEMIO
AB18	DDR_DATA_22	B	MEMIO
AB19	DDR_DQS_2_N	B	MEMIODIF
AB20	DDR_DATA_16	B	MEMIO
AB21	DDR_DATA_21	B	MEMIO
AB22	DDR_BA_2	O	MEMIO

3.2 Design Notes

The LS10xMA device requires four DC power levels. Analog power and ground connections to PCIe, USB and PLL pins require separate treatment. For details on power sequencing, refer to Section 19.1 Power Supply Sequencing.

No Connect pins must remain unconnected. Pull Low pins must be connected to ground.

Unused Interface Termination Recommendations

In case the following interfaces/signals are unused, leave them unconnected.

- I^2C, SPI, UART, Timer, GPIO, and JTAG
- EXP. Unused signals can be left unconnected.
- DDR 16 bit: The most significant bits, such as DDR_DM[3:2], DDR_DQ[31:16], DDR_DQS[3:2], and DDR_DQS#[3:2] should be left open.
- TDM—If TDM port is unused then all signals could be left unconnected, and block should not be taken out of reset (software configuration).
- GEM—If GEM port is unused then all signals could be left unconnected, and block should not be taken out of reset (software configuration).
- Miscellaneous

  - TM_EXT_RESET, GEM0_REFCLK, GEM1_REFCLK. Unused signals can be left unconnected.
- PCIe and USB
  - Keep all the power connection and power to spec levels. Use software to put unused lanes to power down or sleep mode to save power.
  - Float all unused I/O pins, except PCIE_RESREF or USB_REF pins that still need to be connected to external resistors. But the Resistor value and precision can differ from the exact spec if it can lower the cost. If the resistor is shared with any used lane or channel, then it still needs to be exactly as specified in value and precision.

NOTE: The Freescale LS10xMA Device is a multi-core System-on-Chip (SOC) device operating at very high frequencies. Appropriate power supply and layout considerations are required to guarantee optimum performance.

Freescale provides schematic and layout review. Freescale **HIGHLY RECOMMENDS** customers submit their designs to Freescale for a complete and confidential review.

3.3 Input/Output PAD Types

NOTE: Freescale provides IBIS files for LS10xMA devices. Contact the FAE or sales representative for more details.

Table 3-2 gives the I/O pad descriptions.

Table 3-2 I/O Type Descriptions

I/O Type	Description
USBDATA	USB 2.0 compliant analog data
USBANALOG	USB 2.0 compliant analog
PCIE	PCI Express compliant
I	Digital input
Id/Ots8	Digital input, pull down/ Digital output, 3-state, Slew rate, 8mA
Ihd/Ots8	Digital input with hysteresis, pull down/ Digital output, 3-state, Slew rate, 8mA
Ihu/Ots8	Digital input with hysteresis, pull up / Digital output, 3-state, Slew rate, 8mA
Ihu/Ots16	Digital input with hysteresis, pull up / Digital output, 3-state, Slew rate, 16mA
I/Ot8	Digital input / Digital output, 3-state, 8mA
Ih/Ot8	Digital input with hysteresis / Digital output, 3-state, 8mA
I/Ot16	Digital input / Digital output, 3-State, 16mA
Ih/Ot16	Digital input with hysteresis / Digital output, 3-state, 16mA
I/Ots8	Digital input / Digital output, 3-state, Slew rate, 8mA
Id	Digital input, pull down
Ihd	Digital input with hysteresis, pull down
Iu	Digital input, pull up

Table 3-2 I/O Type Descriptions (continued)

I/O Type	Description
Ihu	Digital input with hysteresis, pull up
Ih	Digital input with hysteresis
Ot8	Digital output, 3-state, 8mA
Ot16	Digital output, 3-state, 16mA
Ots8	Digital output, 3-state, Slew control, 8mA
OSCL	Oscillator I/O
MEMIOPL	STTL 1.8 I/O with 75 kΩ pull-down; 6.7 mA/ 13.4 mA
MEMIO	STTL 1.8 I/O 6.7 mA/ 13.4 mA
MEMIODIF	STTL 1.8 I/O differential 6.7 mA/ 13.4 mA

3.4 Signal Summary

Section 3.4.1 to Section 3.4.13 provide the signal summary details.

3.4.1 Expansion Bus

See Section 4 Expansion Bus Interface for more details.

Table 3-3 Expansion Bus Interface Signals (Hardware Controlled)

Signal Name	Dir	Size	Signal Description
EXP_CLK	O	1	Expansion Bus Clock
EXP_CSx	O	4	Expansion Bus Chip Select
EXP_NAND_CS	O	1	NAND Flash Chip Enable
EXP_WE# / EXP_RW#	O	1	Write Enable / Read-Write
EXP_RE# /EXP_STRB#	O	1	Read Enable / Strobe
EXP_NAND_RE#	O	1	Read Enable dedicated for NAND Flash
EXP_NAND_WE#	I	1	Write Enable dedicated for NAND Flash
EXP_DM[1:0]	O	2	Data Byte Mask
EXP_RDY_BSY#	I	1	Peripheral Device Ready / Busy Status
EXP_A[23:0]	O	24	Expansion Address Bus
EXP_ALE	O	1	Address Latch Enable
EXP_DQ[15:0]	B	16	Data Bus

Table 3-4 Expansion Bus Interface Signals (Software Controlled)

Signal Name	Dir	Size	Signal Description
EXP_NAND_CS#	O	1	NAND Chip Select
EXP_NAND_CLE	O	1	NAND Command Latch Enable
EXP_NAND_ALE	O	1	Address Latch Enable
EXP_NAND_RDY/BSY#	I	1	Ready/Busy NAND Device Status

3.4.2 DDR2 SDRAM

See Section 5 DDR2 SDRAM Interface for more details.

Table 3-5 DDR2 Signals

Signal Name	Dir	Size	Signal Description
DDR_CLK, DDR_CLK#	O	2	Differential Clock Outputs
DDR_CLKE	O	1	Clock Enable Active High
DDR_CS#	O	1	Chip Select
DDR_RAS#	O	1	Memory Row Address Strobe
DDR_CAS#	O	1	Memory Column Address Strobe
DDR_WE#	O	1	Memory Write Enable
DDR_DM[1]	O	4	Data Byte Mask
DDR_BA	O	3	Bank Select Address
DDR_A	O	15	Address Bus
DDR_DQ[1]	B	32	Data Bus
DDR_DQS, DDR_DQS#[1]	B	8	Differential Data Strobe
DDR_ODT	O	1	On-Die Termination.

Note:
1. In 16- Bit mode DDR_DM[3:2], DDR_DQ[31:16], DDR_DQS[3:2], and DDR_DQS#[3:2] are not used.

3.4.3 WAN (Ethernet0) and LAN (Ethernet2)

See Section 6 Ethernet Interface for more details.

3.4.3.1 MII Mode

Table 3-6 MII Signals

Signal Name	Dir	Size	Signal Description
MII_REFCLK	O	1	Reference Clock Output
MII_TX_CLK	I	1	Transmit Clock

Table 3-6 MII Signals (continued)

Signal Name	Dir	Size	Signal Description
MII_TXD[3:0]	O	4	MII Transmit Data
MII_TX_EN	O	1	MII Transmit Enable
MII_TX_ER	O	1	MII Transmit Error
MII_RX_CLK	I	1	MII Receive Clock
MII_RXD[3:0]	I	4	MII Receive Data
MII_RX_DV	I	1	MII Receive Data Valid
MII_RX_ER	I	1	MII Received Error Detected
MII_CRS	I	1	MII Carrier Sense
MII_COL	I	1	MII Collision Detect

3.4.3.2 RMII Mode

Table 3-7 RMII Signals

Signal Name	Dir	Size	Signal Description
RMII_CLK	I	1	Reference Clock Input
RMII_REFCLK	O	1	Reference Clock Output
RMII_TX_EN	O	1	Transmit Enable
RMII_TXD[1:0]	O	2	Transmit Data
RMII_CRS_DV	I	1	Carrier Sense / Receive Data Valid
RMII_RX_ER	I	1	Receive Error Detected
RMII_RXD[1:0]	I	2	Receive data
RMII_PHY_LINK	I	1	Link indication
RMII_PHY_DPX	I	1	Duplex indication
RMII_PHY_SPEED	I	1	Speed indication

3.4.3.3 GMII Mode

Table 3-8 GMII Signals

Signal Name	Dir	Size	Signal Description
GMII_REFCLK	O	1	Reference Clock Output
GMII_GTX_CLK	O	1	GMII Gigabit Transmit Clock
GMII_TX_CLK	I	1	GMII Transmit Clock
GMII_TXD[7:0]	O	8	GMII Transmit Data
GMII_TX_EN	O	1	GMII Transmit Enable

Table 3-8 GMII Signals (continued)

Signal Name	Dir	Size	Signal Description
GMII_TX_ER	O	1	GMII Transmit Error
GMII_RX_CLK	I	1	GMII Receive Clock
GMII_RXD[7:0]	I	8	GMII Receive Data.
GMII_RX_DV	I	1	GMII Receive Data Valid
GMII_RX_ER	I	1	GMII Received Error Detected
GMII_CRS	I	1	GMII Carrier Sense
GMII_COL	I	1	GMII Collision Detect

3.4.3.4 RGMII Mode

Table 3-9 RGMII Signals

Signal Name	Dir	Size	Signal Description
RGMII_REFCLK	O	1	RGMII Reference Clock Output
RGMII_TXC	O	1	RGMII Transmit Clock
RGMII_TXD[3:0]	O	4	RGMII Transmit Data
RGMII_RXC	I	1	RGMII Receive Clock
RGMII_RX_CTL	I	1	RGMII Receive Control
RGMII_RXD[3:0]	I	4	RGMII Receive Data
RGMII_TX_CTL	O	1	RGMII Transmit Control

3.4.3.5 Management Interface

One MDIO port support both the WAN and LAN Ethernet ports.

Table 3-10 Management Interface Signals

Signal Name	Dir	Size	Signal Description
GEM_MDC	O	1	Serial Management Interface Clock
GEM_MDIO	B	1	Serial Management Interface Data

3.4.4 PCIe

Table 3-11 PCI Express Signal

Symbol	Dir.	Size	Name/Function
Lane 1			
PCIE0_HSO_P	O	1	Lane 1 serial output, differential high
PCIE0_HSO_M	O	1	Lane 1 serial output, differential low

Table 3-11 PCI Express Signal (continued)

Symbol	Dir.	Size	Name/Function
PCIE0_HSI_P	I	1	Lane 1 serial input, differential high
PCIE0_HSI_M	I	1	Lane 1 serial input, differential low
Lane 2			
PCIE1_HSO_P	O	1	Lane 2 serial output, differential high
PCIE1_HSO_M	O	1	Lane 2 serial output, differential low
PCIE1_HSI_P	I	1	Lane 2 serial input, differential high
PCIE1_HSI_M	I	1	Lane 2 serial input, differential low
Clock Module			
PCIE_RESREF	B	1	Reference resistor connection. Connect to ground with a 191R 1% resistor.
PCIE_REFCLK_P	I	1	Reference clock input, differential high
PCIE_REFCLK_M	I	1	Reference clock output differential low

See Chapter 7, PCI Express Interface for more details.

3.4.5 TDM Bus

See Chapter 10, TDM Bus Interface for more details.

Table 3-12 TDM Signals

Signal Name	Dir.	Description
TDM_CK	B	Data clock input from network, or from device if it is the source of the clock
TDM_FS	B	Frame synchronization input from the network or from device
TDM_DX	O	Data transmit output from the device
TDM_DR	I	Data receive input from the network

3.4.6 SPI

See Chapter 8, Serial Peripheral Interface for more details.

Table 3-13 SPI Signals

Signal Name	Dir.	Size	Description
SPI_SCLK	O	1	Serial Bit-rate Clock
SPI_TXD	O	1	Transmit Data Signal
SPI_RXD	I	1	Receive Data Signal
SPI_SS#	O	3	Slave Select Output

3.4.7 UART

See Chapter 11, Universal Asynchronous Receiver Transmitter for more details.

Table 3-14 LS10xMA UART Interface Signals

Signal Name	Dir.	Size	Signal Description
UART[1:0]_RX	I	2	UART serial input
UART[1:0]_TX	O	2	UART serial output

3.4.8 I^2C

See Chapter 12, Inter-IC Interface for more details.

Table 3-15 LS10xMA I^2C Interface Signals

Signal Name	Dir	Size	Signal Description
I2C_SCL	B	1	I^2C Input/Output Clock Line
I2C_SDA	B	1	I^2C Input/Output Data Line

3.4.9 USB

The LS10xMA device supports one USB 2.0 port. See Chapter 13, USB Interface for more details.

Table 3-16 USB Interface Signals

Signal Name	Dir	Size	Signal Description
USB_DP	B	1	Positive channel connected to the serial USB cable. 3.3V analog signal.
USB_DM	B	1	Negative channel connected to the serial USB cable. 3.3V analog signal.
USB_VBUS	—	—	5V, 500 mA power supply. Required by the USB specification. **NOTE:** Not connected to the LS10xMA device, but necessary to support external interface.
USB_GND	—	—	Ground for USB. NOTE: Not connected to the LS10xMA device, but necessary to support external interface
USB_REF	B	1	External resistor connection for current reference.
USB_GPANAIO	B	1	Analog Test Pin
USB_VBUS_STAT	I	1	VBUS Status is an indicator of the internal USB VBUS. When the LS10xMA USB controller is in host mode this signal goes low to indicate a power fault condition (such as over-current) on the sourced VBUS. When the LS10xMA USB controller is in device mode this signal goes low to indicate a device mode error. **NOTE:** This I/O is +5 VDC tolerant.

3.4.10 GPIO

See Chapter 14, General Purpose Input Output for more details.

Table 3-17 GPIO Interface Signals

Signal Name	Dir	Size	Signal Description
GPIO[7:5]	B	3	General purpose 05 through 07 I/Os. These pins can generate interrupts.
GPIO[3:0]	B	4	General purpose 00 through 03 I/Os. These pins can generate interrupts.
GPIO08 – GPIO26	B	19	General purpose I/Os 08 through 26. These pins cannot generate interrupts. Note that at the block level a total of 31 GPIO pins are supported, but not all these pins are output on the die.

3.4.11 Timers/Counters

Table 3-18 Timer Interface Signals

Signal Name	Dir	Size	Signal Description
TIM_EVNT[1:0]	B	2	Timer Event

3.4.12 JTAG

For more on this topic, see Chapter 17, Test and Debug Interface. There are two JTAG ports. JTAG0 supports debug of ARM0 and boundary scan. JTAG1 supports debug of ARM1.

Table 3-19 JTAG Signals

Signal Name	I/O	Size	Signal Name/Description
JTAG_TRST#	I	1	JTAG Test Reset
JTAG _TCK	I	1	JTAG Test Clock
JTAG_TMS	I	1	JTAG Test Mode Select
JTAG_TDI	I	1	JTAG Test Input Data
JTAG_TDO	O	1	JTAG Test Output Data

3.4.13 Test and Miscellaneous

Table 3-20 Test and Miscellaneous

Symbol	Dir.	Size	Name/Function
RESET_N	I	1	Device reset
TM_PLL_XI	I	1	Reference clock in
TM_PLL_XO	O	1	Reference clock out
TM_EXT_RESET	O	1	External reset pin; used to hold external devices in reset until all clocks are up
TM_TESTMODE_N	I	1	Test mode pins. See Section 17.4 Test Mode Configuration.
TM_BSCN_JTAG	I	1	
Inputs Recognized at Reset Time (these are multiplexed Expansion Bus address lines)			

Table 3-20 Test and Miscellaneous

Symbol	Dir.	Size	Name/Function
TM_PCIE0_MODE	I	1	Sets up PCIe mode for specific lane as Root mode. Chapter 7, PCI Express Interface.
TM_GEM0_MODE	I	2	xMII0 interface mode
TM_GEM1_MODE	I	2	xMII1 interface mode
TM_BOOT_OP[1:0]	I	2	Selects boot source. See Section 19.3 Boot Sequence.
TM_TDM_CLKSRC_EN	I	1	When high, the LS10xMA device provides TDM_CK to external circuitry.
TM_GPBT_OP	I	2	Inputs can be defined by software configuration. See Section 14.4 General Purpose Boot Option Inputs.
TM_EXP_NAND_SEL	I	1	When high, the Expansion Bus is configured for NAND accesses.
TM_PCIe_EXT_REFCLK	I	1	If the PCI Express interface is used, TM_PCIe_EXT_REFCLK must be high at reset time. If the PCI Express interface is not used, TM_PCIe_EXT_REFCLK can be pulled low with an external resistor or with the internal pull-down resistor that is built into the I/O cell.

3.5 Power and Ground

See Section 19.1 Power Supply Sequencing for more details.

Table 3-21 Power and Ground Contacts

Symbol(s)	Name/Function
Power Inputs	
VDD	Vcore (+0.98 V)
VDD_PCIE_1	PCIe (+0.98 V)
VDDO_PCIE_2.5	PCIe Interface (+2.5 V)
VDDO_EXP	Expansion Interface (+3.3 V)
VDDO_ETH0	WAN Interface — Ethernet0 (+3.3 or +2.5 V)[1]
VDDO_ETH1	LAN Interface — Ethernet1 (+3.3 or + 2.5 V)[1]
VDDO_DDR2	External Memory Interface (+1.8 V)
VDD_REF	SSTL reference voltage (+0.5* VDDO DDR2)
VDDA33_USB	USB 3.3V Supply (+3.3V VDD)
VDDA25_USB	USB 2.5V supply (+2.5V VDD)
VDD_PLL_0, VDD_PLL_1, VDD_PLL_2	PLL Supply (+0.98 V)
Grounds	
VSS	Ground

Note:
1. RGMII was verified with 2.5V devices and supplies. GMII, MII, and RMII were verified with 3.3V devices and supplies.

4 Expansion Bus Interface

This section provides details on the LS10xMA Expansion bus interface.

4.1 Features

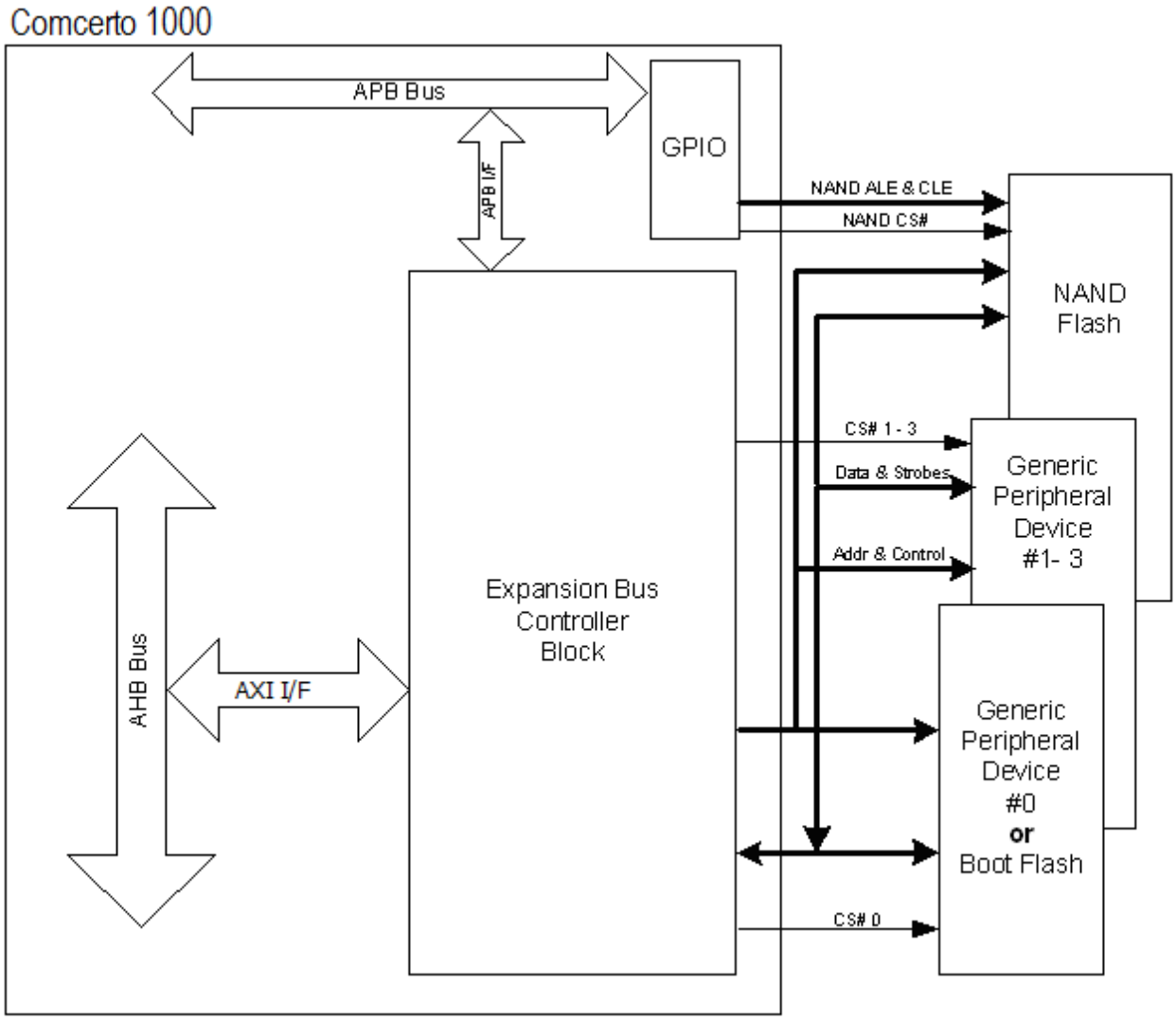
- Support for up to 5 peripheral devices: NAND flash and 4 general-purpose device interfaces, that is NOR flash devices. NAND flash chip-select is controlled by software and driven on a GPIO pin.
- Support for 8-Bit, 16-Bit, 32-Bit, and 64-Bit AHB transactions.
- Support for 8-Bit and16-Bit bus interfaces; configurable per chip select.
- AHB transaction size that is larger than the external bus width will be split into separate transactions by the Expansion Bus Interface block, meanwhile halting the AHB interface.
- Programmable chip select level, configurable per chip select.
- Each chip-select supports up to 16M of memory address space.
- Chip-select 0 is dedicated to external boot flash when used, enabled by default. Otherwise, it can be configured to support any generic peripheral device.
- An external input pin is used to set the default bus size of chip-select 0 device, 8-Bit or 16-Bit data bus can be selected. This bus size may be overwritten by software to support.
- Provides support for multiplexed address-data mode with address latch enable.
- Programmable timings per chip select; counts in terms of expansion clock cycles.
- Support for byte-enable writes, up to 2 bytes on the Expansion Interface.
- Provides support for Ready/Busy# acknowledge signal for terminating transactions. Detecting the de-assertion of Read/Busy# signal will override the Chip-Select and Write Enable/ Read Enable to be de-asserted for the current transaction. Detection of Ready signal on rising/falling edge is configurable.
- Synchronous interface. The AHB clock can be a divide down by 3, 4, 5, 6 or 7. The divide integer is configurable by software, default is set to 7. Sync clock maximum frequency will be 66 MHz (AHB Clock 200 MHz divide by 3).
- External expansion clock is provided and maybe suppressed by software, when not used by any of the peripherals. All programmable timing parameters are synchronized to this expansion clock.
- Two separate sets of read/write controls are provided; one set is dedicated for NAND flash: WE_NAND#/RE_NAND#, and the other set is for the generic devices: EXP_RE/ EXP_WE.
- Two command modes are supported. Signals assertion level is configurable.
  - Provide support for separate EXP_RE and EXP_WE signals.
  - Provide support for EXP_RW and EXP_STRB signals. The EXP_RW signal will be driven on the EXP_WE pin, and the EXP_STRB signal will be driven on the EXP_RE pin.
- Extended support for External flash devices are provided through GPIO pins when enabled.
  - EXP_WP# and EXP_RP# **(*optional*)**.
  - Peripheral interrupt support.
  - NAND Flash controls and chip select support: CS, ALE, CLE, and RDY.

4.2 Expansion Bus Functional Description

For a view of the Expansion Bus block within the top-level block diagram, see Figure 4-1.

The Expansion Interface controls data flow to or from up to four generic memories and one dedicated NAND Flash device. Initially CS0, which is part of the generic interface, can be configured to be used for the external boot flash, occupying address space zero of the expansion memory map; two pins determine the external boot flash bus size.

Figure 4-1 Expansion Bus Block Diagram



When the Expansion Bus Interface block is accessed by the system, the Expansion controller will fetch/store data from/to these memory flash devices until de-selected. Depending on the memory device type being accessed, the required timing parameters are configured to comply with the device timing-requirements, before initiating any transactions. RDY pulse sampling, when supported, will be synchronized to AHB clock (HClk) before causing the de-assertion of the CS and the RE or WE control signals. The read data will be latched upon de-assertion of the CS or RE signal, which ever comes first. A configuration bit per chip-select is provided to enable/disable detection of the RDY signal and should be disabled for devices that do no provide the RDY signals. The RDY signal, when enabled, is not detected, will cause an error response on the AHB interface.

4.2.1 NAND Flash Access

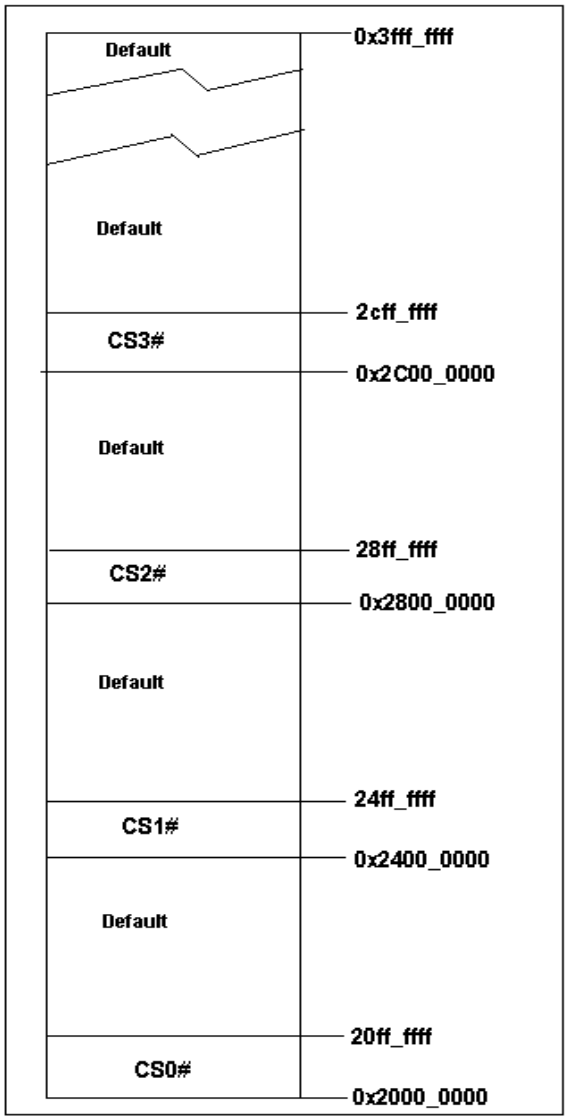
As for the NAND Flash, when accessed, the software controls the CLE and ALE signals as well as the device Chip-select through GPIO dedicated pins. The Expansion Bus Controller provides the flash address/data/command and drive the NAND_WE # /NAND_RE# signals. Each NAND flash access phase will be considered a full expansion bus access cycle.

4.2.2 AHB Address Decoding

Accesses to the Expansion Interface block are decoded internally to assert the appropriate chip-select of the targeted device. 64-Bit AHB input data will be latched in case of write operations.

Depending on the peripheral bus size used, the latched data will be transferred over decoded number of expansion cycles. The maximum number of expansion cycles to write 64-Bit data over an 8-Bit bus peripheral will be 8 cycles, meanwhile the AHB bus is halted until all data bytes are transferred. In case of read operations, the input data from the peripheral device is latched and formatted over a 64-Bit AHB bus according to the peripheral bus size selected. The read data is duplicated over the 64-Bit AHB data bus, that is, if MEM_SIZE is set to 16, the read byte will be latched over all half-words of AHB data bus. Figure 4-1 illustrates the Expansion Bus chip select region.

Figure 4-2 Expansion Bus Chip Select Region



Depending on the memory size and the access size, the expansion address bus is decoded from the internal AHB address bus. The lowest bit of the AHB bus is dropped when a half word access is performed, and the lowest two bits are dropped when a word access is performed. Hence, word accesses are always aligned to word boundaries in memory. Depending on the memory size, the interface will complete the Expansion bus address with the Expansion cycle count bits, and decode the DM signals.

NOTE: AHB accesses outside the expansion chip-selects allocated memory segments will cause an AHB Error response.

4.3 Signal Description

The control signals unique to NAND FLASH interfacing are controlled by software through GPIO pins. Table 4-1 and Table 4-2 describe the expansion bus interface signals.

Table 4-1 Expansion Bus Interface Signals (Hardware Controlled)

Signal Name	Dir	Size	Signal Description
EXP_CLK	O	1	Expansion Bus Clock — to the peripheral devices. This clock is generated in the Expansion Block by dividing down the AHB clock by any of the following integers: 3,4, 5, 6 or 7. Frequency ranges from 28.5 to 66 MHz. This clock is gated-off when accessing asynchronous devices by setting EXP_CLK_EN to 0.
EXP_CSx	O	4	Expansion Bus Chip Select - Chip Selects with programmable active level (active low by default), enable and disable selection of the targeted peripheral devices. Chip Selects #0-3 are supported by default. Active level is configurable by software. Note: EXP_CS3 is muxed with GPIO 05.
EXP_NAND_CS	O	1	NAND Flash Chip Enable: This gates transfers between the LS10xMA device and the NAND Flash device.
EXP_WE# / EXP_RW#	O	1	Write Enable / Read-Write Normal Mode: Write Enable, when asserted, indicates a write operation to peripheral. Strobe Mode: Read/Write Enable signal. Assertion level of this signal is configurable.
EXP_RE# /EXP_STRB#	O	1	Read Enable / Strobe Normal Mode: Read Enable, when asserted, indicates a read operation from peripheral. Strobe Mode: Command Strobe signal. Assertion level of this signal is configurable.
EXP_NAND_RE#	O	1	Read Enable dedicated for NAND Flash — When asserted low, indicates a read operation from flash device. Assertion level of this signal is configurable. NAND flash chip-select is driven by software through GPIO pin.
EXP_NAND_WE#	I	1	Write Enable dedicated for NAND Flash — When asserted low, indicates a write operation to the flash device. Assertion level of this signal is configurable. Note: NAND flash chip-select is driven by software through a GPIO pin.
EXP_DM[1:0]	O	2	Data Byte Mask — Input data byte to the peripheral device is masked when DM is sampled high during a write access. One DM pin per data byte. LS10xMA drives EXP_DM while driving the data bus. In all other cases, DM is tri-stated.
EXP_RDY_BSY#	I	1	Peripheral Device Ready / Busy Status — When set low indicates that the current transaction is in progress. Set high when the device is ready for a new transaction. Detection edge of this signal is configurable (high->low) or (low->high). When detected will cause de-assertion of the CS and RE/WE for the current transaction.

Table 4-1 Expansion Bus Interface Signals (Hardware Controlled) (continued)

Signal Name	Dir	Size	Signal Description
EXP_A[23:0]	O	24	Expansion Address Bus - Specifies one Memory location along with the DM[1:0] when supported. Will address 16M of memory address space, per chip-select. • EXP_A13/GPIO11 • EXP_A14/GPIO14 • EXP_A15/GPIO8 • EXP_A16/GPIO9 • EXP_A17/GPIO10 **NOTE:** EXP_A[17:13] defaults to GPIOs configured as inputs and have internal pull-down. Other devices on the board that are connected to these signals should not drive the signals during boot-up and hence it is suggested to implement an external pull-down to ensure zero state during boot-up. When TM_EXP_NAND_SEL (bootstrap) or nand_sel config (GPIO space) is set, the following upper expansion bus address pins will be dedicated to expansion NAND controls: • EXP_A18 / EXP_NAND_CS / GPIO29 • EXP_A19 / EXP_NAND_ALE / GPIO30 • EXP_A20 / EXP_NAND_CLE / GPIO31 • EXP_A21 / EXP_NAND_RDY / GPIO6 • EXP_A22 / EXP_NAND_RE_N • EXP_A23 / EXP_NAND_WE_N In such case, a smaller NOR can be used.
EXP_ALE	O	1	Address Latch Enable — The ALE, active high output, controls address latching to the peripheral device when multiplexed address / data bus mode is used.
EXP_DQ[15:0]	B	16	Data Bus — Will be configured to support 8 or 16 active bits. For 8-Bit, EXP_DATA[7:0] and for 16-Bit EXP_DATA[15:0] are active. Least Significant Bits (LSB) are always active.

Table 4-2 Expansion Bus Interface Signals (Software Controlled)

Signal Name	Dir	Size	Signal Description
EXP_NAND_CS#	O	1	NAND Chip Select—Active low CS# enables and disables selection of the targeted NAND flash devices.
EXP_NAND_CLE	O	1	NAND Command Latch Enable —The CLE output controls writing to the command register. When CLE is high, the command is loaded on the rising edge of NAND_WE#.
EXP_NAND_ALE	O	1	Address Latch Enable—The ALE, active high output, controls writing data to the flash device address register. When ALE is high, the address is loaded on the rising edge of NAND_WE#. NAND ALE must remain high during the entire address sequence.
EXP_NAND_RDY/BSY#	I	1	Ready/Busy NAND Device Status—When high indicates that the device is ready for access. When low indicates no access will be granted.
EXP_WP#	O	1	Write Protect—The WP# output provides protection when programming or erasing the flash. The internal voltage regulator is reset when WP# is low, preventing any program or erase operations. Supported by both: NAND and NOR types of Flash devices. Not assigned a dedicated GPIO pin.
EXP_RP#	O	1	Reset/Power-Down—When set low, will place the flash memory in a sleep mode and will tri-state its outputs. For normal operation, RP# must be set high. Not used by the NAND. Not assigned a dedicated GPIO pin.

4.4 Modes of Operation

Expansion block is highly configurable per chip select to accommodate for most common types of memory/flash. Four modes of operation can be configured to support the targeted peripheral timings:

- Normal mode (default mode for CS0-CS3)
- ALE mode
- Strobe mode
- NAND mode (GPIO29 used as NAND CS)

These modes support both, synchronous and asynchronous devices. By default, synchronous support is set and the output expansion clock is provided by setting EXP_CLK_EN global bit to 1 (high). When a asynchronous device is accessed, EXP_CLK_EN global bit should be set to 0 (low), which will suppress expansion clock output. The GPIO29 is multiplexed with EXP_NAND_CS and supports NAND mode.

4.4.1 Normal Mode

This mode is supported by all chip selects and is set by programming the following configuration bits:

- ALE_MODE = 0
- STRB_MODE = 0

Table 4-3, Normal Mode Signal List, Figure 4-3 and Figure 4-4 depict the list of external pins that are valid when the Normal mode is set.

Table 4-3 Normal Mode Signal List

Signal Name	Dir	Size	Signal Description
EXP_CLK	O	1	Expansion Bus Clock — to the peripheral devices. This clock is generated in the Expansion Block by dividing down the AHB clock by any of the following integers: 3,4, 5, 6 or 7. Frequency ranges from 28.5 to 66 MHz. This clock is gated-off when accessing asynchronous devices by setting EXP_CLK_EN to 0.
EXP_CSx	O	4	Expansion Bus Chip Select - Chip Selects with programmable active level (active low by default), enable and disable selection of the targeted peripheral devices. Chip Selects #0-3 are supported by default. Active level is configurable by software. Note: EXP_CS3 is muxed with GPIO05.
EXP_RE	O	1	Read Enable — Read Enable. When asserted indicates a read operation from peripheral. Active level is configurable by software.
EXP_WE	O	1	Write Enable — When asserted, indicates a write operation to peripheral. Active level is configurable by software.
EXP_DM	O	2	Data Byte Mask — Input data byte to the peripheral device is masked when EXP_DM is sampled high during a write access. One EXP_DM pin per data byte. LS10xMA drives EXP_DM while driving the data bus. In all other cases, DM is tri-stated. The timing of these signals is configurable by writing to DM_MODE field: 0: DM timing will mimic the EXP_CS. 1: DM timing will mimic the EXP_WE command.
EXP_RDY	I	1	Peripheral device ready/busy status. When set low indicates that the current transaction is in progress. Set high when the device is ready for a new transaction. Detection edge of this signal is configurable (high->low) or (low->high). When Detected will cause de-assertion of the CS and RE/WE for the current transaction
EXP_ADDR	O	24	Expansion Address Bus—Specifies one memory location along with the DM[1:0] when supported. This bus will address 16M of memory address space, per chip-select. When NAND mode is used by setting the exp_nand_sel, NOR memory address space allocated for NOR will be less than 16M. It will be 256K.
EXP_DATA	B	32	Data bus—MEM_BUS_SIZE is used to configure the peripheral interface width: • 00: Peripheral has an 8-Bits data bus interface. • 01: Peripheral has a 16-Bits data bus interface. • 10: Reserved. • 11: Reserved. Only lower 16-Bits are used.

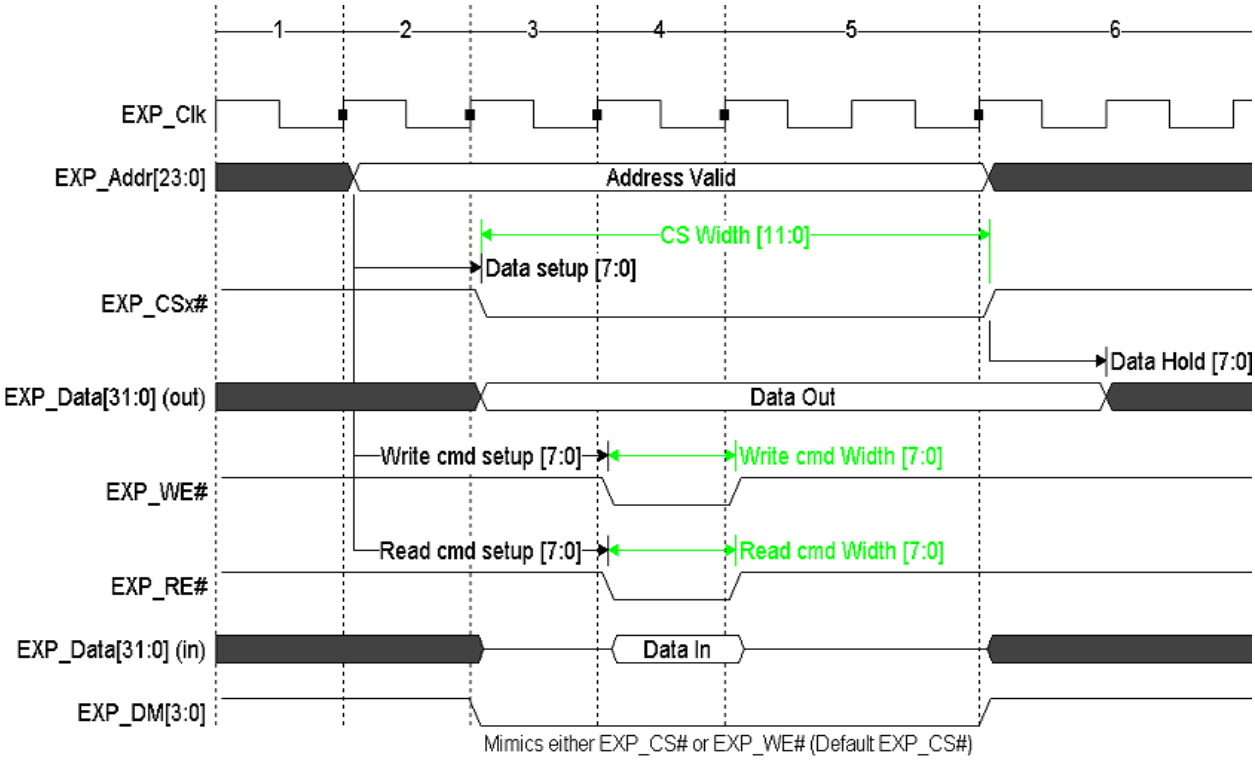
Figure 4-3 shows the Normal Mode relative signals and their programmable fields. CS and the RE or WE command will be asserted, depending on the access type, after a programmed number of expansion clock cycles from the active address. The expansion clock is a divide down frequency from the AHB clock; it is generated from dividing down hclk by one of the configured divide integers (3, 4, 5, 6, or 7).

In the read cycle, the data bus is tri-stated and data is expected to be ready sometime after the assertion of the RE command. The expansion controller will latch the data upon de-assertion of either the RE command or the CS signal, whichever comes first. The CS is terminated after the expiration of the CS width count or upon detection of the de-assertion of the RDY signal provided by the device (when enabled by setting EXP_RDY_EN to 1).

During the Write cycle, CS and WE command are asserted some programmable cycles after the valid address. The external device is expected to latch the data by the de-assertion of the WE command. The data will be held valid for some programmable cycles after the de-assertion of the CS. Next Read/Write transaction can only be started after the expiration of the Data-hold time count.

When EXP_RDY_EN bit is set high, detected RDY pulse will be synchronized to AHB clock (HClk) before causing the de-assertion of the CS and the RE or WE control signals, as was mentioned above, that the read data will be latched upon de-assertion of the CS or RE signal, whichever comes first. A configuration bit per chip-select is provided to enable/disable detection of the RDY signal and should be disabled for devices that do not provide the RDY signals. If case of RDY is enabled and was not detected by the expansion controller before the de-assertion of the CS, an error response will be indicated on the AHB interface for this transaction.

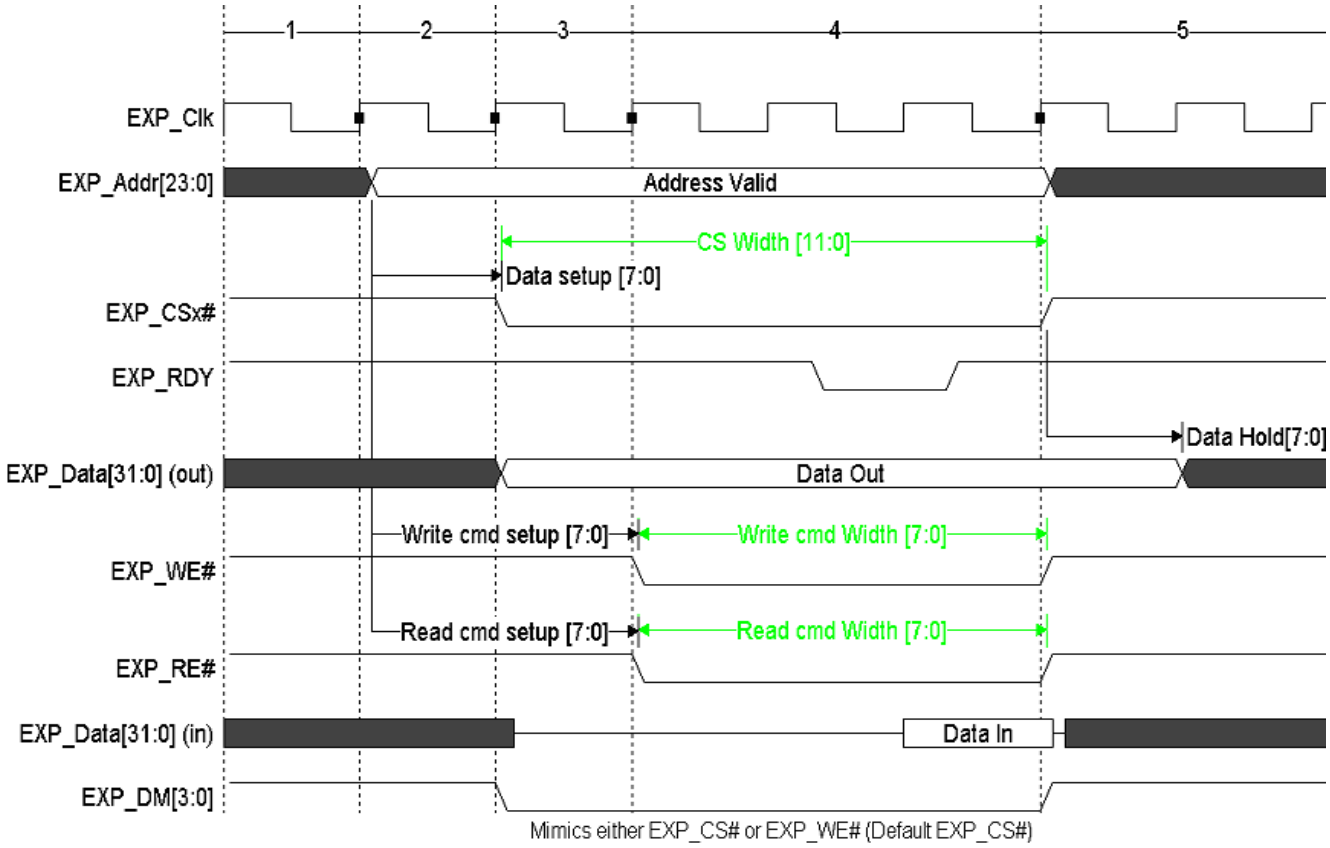
Figure 4-3 Normal Mode Timing Diagram (RDY_EN = 0)



In Figure 4-4 illustrates the read and write commands, when a read access is being performed, only RE signal will toggle. Also, when a write command is being performed, only WE command will toggle, not both.

NOTE: AHB accesses outside the expansion chip-selects allocated memory segments will cause an AHB Error response.

Figure 4-4 Normal Mode Timing Diagram (RDY_EN = 1)



4.4.2 ALE Mode

This mode is supported by all chip selects, and is set by programming the following configuration bits:

- ALE_MODE = 1
- STRB_MODE = 0

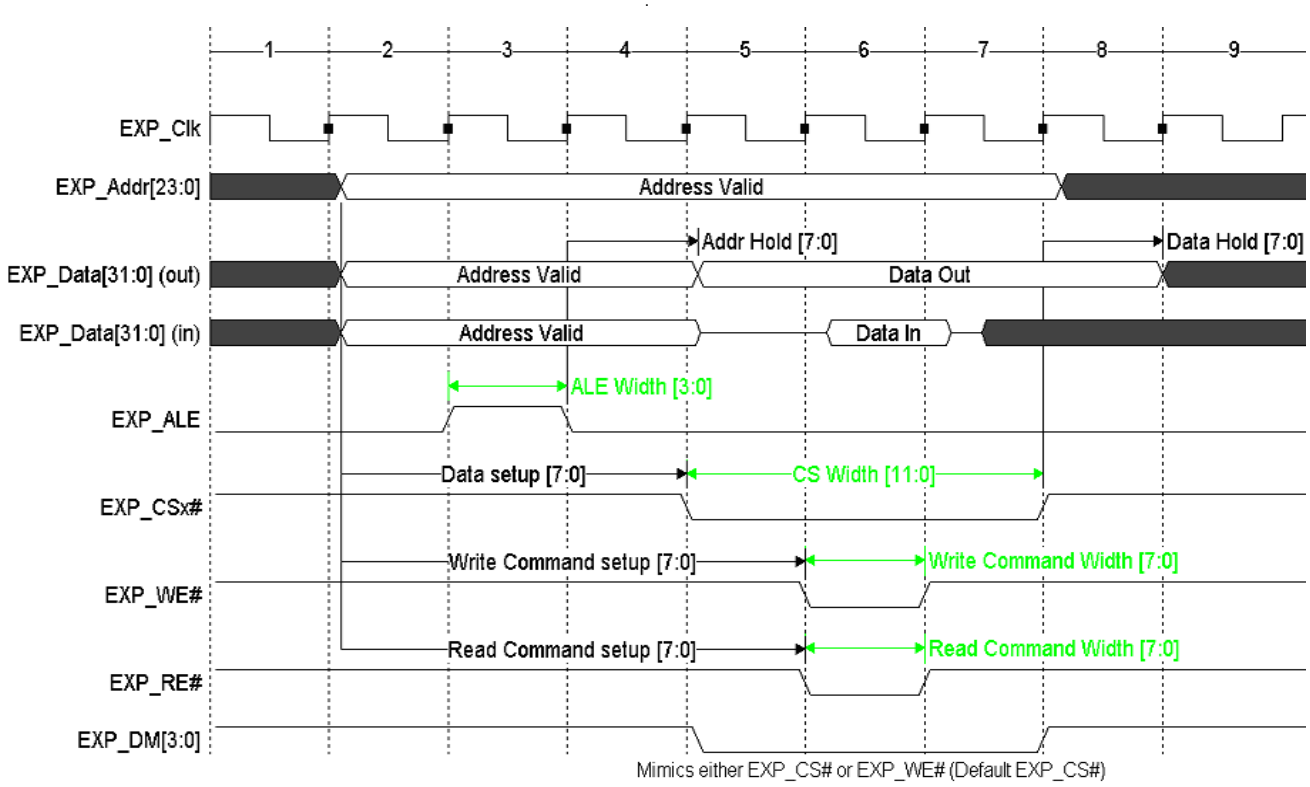
ALE Mode can be supported by chip selects 0-3 by setting ALE_MODE bit to 1. In this mode, the data bus will be used to transfer both the valid address and valid data, Active high EXP_ALE will be used by the peripheral device to latch the valid address. The valid address should be stable for the configured address hold time before placing valid data on the bus in case of writes. Read data will be latched by the Expansion controller upon de-assertion of the EXP_RE signal or EXP_CS, whichever comes first. Valid address will also be driven on the address bus, to provide support for devices that may require it.

Table 4-4 and Figure 4-5 depict the list of signals that are valid when the ALE mode is set:

Table 4-4 ALE Mode Signal List

Signal Name	Dir	Size	Signal Description
EXP_CLK	O	1	Expansion Bus Clock — to the peripheral devices. This clock is generated in the Expansion Block by dividing down the AHB clock by any of the following integers: 3,4, 5, 6, or 7. Frequency ranges from 28.5 to 66 MHz. This clock is gated-off when accessing asynchronous devices by setting EXP_CLK_EN to 0.
EXP_CSx	O	4	Expansion Bus Chip Select - Chip Selects with programmable active level (active low by default), enable and disable selection of the targeted peripheral devices. Chip Selects #0-3 are supported by default. Active level is configurable by software. Note: EXP_CS3 is muxed with GPIO05.
EXP_RE	O	1	Read Enable — Read Enable. When asserted indicates a read operation from peripheral. Active level is configurable by software.
EXP_WE	O	1	Write Enable — When asserted, indicates a write operation to peripheral. Active level is configurable by software.
EXP_DM	O	2	Data Byte Mask — Input data byte to the peripheral device is masked when EXP_DM is sampled high during a write access. One EXP_DM pin per data byte. LS10xMA drives EXP_DM while driving the data bus. In all other cases, DM is tri-stated. The timing of these signals is configurable by writing to DM_MODE field: 0: DM timing will mimic the EXP_CS. 1: DM timing will mimic the EXP_WE command.
EXP_RDY	I	1	Peripheral device ready/busy status. When set low indicates that the current transaction is in progress. Set high when the device is ready for a new transaction. Detection edge of this signal is configurable (high->low) or (low->high). When Detected will cause de-assertion of the CS and RE/WE for the current transaction
EXP_ADDR	O	24	Expansion Address Bus—Specifies one memory location along with the DM[1:0] when supported. This bus will address 16M of memory address space, per chip-select. When NAND mode is used by setting the exp_nand_sel, NOR memory address space allocated for NOR will be less than 16M. It will be 256K.
EXP_DATA	B	32	Data bus—MEM_BUS_SIZE is used to configure the peripheral interface width: • 00: Peripheral has an 8-Bits data bus interface. • 01: Peripheral has a 16-Bits data bus interface. • 10: Reserved • 11: Reserved. Only lower 16-Bits are used.
EXP_ALE	O	1	Address latch enable pulse, used by the peripheral device to latch the valid address.

Figure 4-5 ALE Mode Timing Diagram



4.4.3 Strobe Mode

This mode is supported by all chip selects, and is set by programming the following configuration bits:

- ALE_MODE = 0
- STRB_MODE = 1

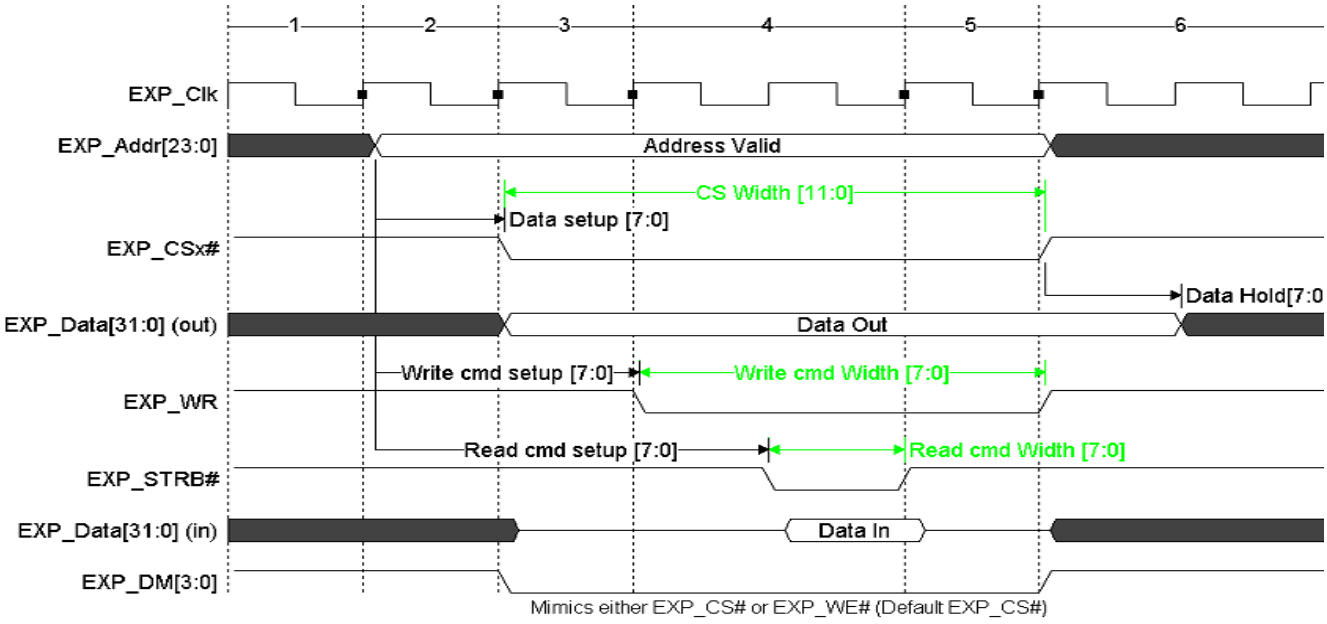
This mode can be supported by chip selects 0-3 by setting STRB_MODE bit to 1. In this mode, EXP_WE# pin will be used by the peripheral device as EXP_RW# command signal. EXP_RE# pin will be used by the peripheral as the EXP_STRB command strobe signal. The active level of these two signals is configurable, depending on the polarity supported by the peripheral device.

Table 4-5 and Figure 4-6 depict the list of signals that are valid when the Command Strobe mode is set.

Table 4-5 Command Strobe Mode Signal List

Signal Name	Dir	Size	Signal Description
EXP_CLK	O	1	Expansion Bus Clock — to the peripheral devices. This clock is generated in the Expansion Block by dividing down the AHB clock by any of the following integers: 3,4, 5, 6 or 7. Frequency ranges from 28.5 to 66 MHz. This clock is gated-off when accessing asynchronous devices by setting EXP_CLK_EN to 0.
EXP_CSx	O	4	Expansion Bus Chip Select - Chip Selects with programmable active level (active low by default), enable and disable selection of the targeted peripheral devices. Chip Selects #0-3 are supported by default. Active level is configurable by software. Note: EXP_CS3 is muxed with GPIO05.
EXP_RE/EXP_STRB	O	1	Command Strobe—By default, when set low, this signal is used by the peripheral for command strobing. Active level is configurable by software.
EXP_WE/EXP_RW	O	1	Read/Write Enable. By default, when set high, indicates a read operation from peripheral. When asserted low, indicates a write transaction to the peripheral. Active level is configurable by software.
EXP_DM	O	2	Data Byte Mask — Input data byte to the peripheral device is masked when EXP_DM is sampled high during a write access. LS10xMA drives EXP_DM while driving the data bus. In all other cases, DM is tri-stated. One EXP_DM pin per data byte. The timing of these signals is configurable by writing to DM_MODE field: 0: DM timing will mimic the EXP_CS. default 1: DM timing will mimic the EXP_WE command.
EXP_RDY	I	1	Peripheral device ready/busy status. When set low indicates that the current transaction is in progress. Set high when the device is ready for a new transaction. Detection edge of this signal is configurable (high->low) or (low->high). When Detected will cause de-assertion of the CS and RE/WE for the current transaction
EXP_ADDR	O	24	Expansion Address Bus—Specifies one memory location along with the DM[1:0] when supported. This bus will address 16M of memory address space, per chip-select. When NAND mode is used by setting the exp_nand_sel, NOR memory address space allocated for NOR will be less than 16M. It will be 256K.
EXP_DATA	B	32	Data bus—MEM_BUS_SIZE is used to configure the peripheral interface width: • 00: Peripheral has an 8-Bits data bus interface. • 01: Peripheral has a 16-Bits data bus interface. • 10: Reserved • 11: Reserved Only lower 16-Bits are used.

Figure 4-6 Strobe Mode Timing



NOTE: In this mode, both EXP_RW# and EXP_STRB# signals shall be switching within read and write transactions. The read data will be sampled by the Expansion Bus Interface block upon de-assertion of the EXP_CS or EXP_STRB, whichever comes first.

4.4.4 NAND Mode

GPIO 29 is multiplexed and used as EXP_NAND_CS. By default this signal is used as NAND flash write enable signal (Exp_nand_we_n) when NAND_MODE is set high. This mode is not supported by the chip selects #0-3. This mode is set by programming the following configuration bits (set by default):

- ALE_MODE = 0
- STRB_MODE = 0

The chip select and latch controls for this mode are provided by a dedicated GPIO pin. NAND_RE/NAND_WE commands, address and data are initiated by the expansion block as an expansion cycle. Each NAND transaction should be broken up into several expansion (Normal Mode type) transactions. The software has the ultimate control over NAND transactions. NAND ECC support is also provided by software only.

Table 4-6, NAND Mode Signal List, Figure 4-7, Figure 4-8, and Figure 4-9 depict the list of signals that are valid when the NAND mode is set:

Table 4-6 NAND Mode Signal List

Signal Names	Dir	Size	Signal Description
NAND_CS_N (GPIO)	O	1	Chip Selects. Enables and disables selection of the targeted NAND peripheral device. Driven by software.
NAND_RE_N	O	1	Read Enable dedicated for NAND flash, when asserted low, indicates a read operation from flash device. Assertion level of this signal is configurable.
NAND_WE_N	O	1	Write Enable dedicated for NAND flash, when asserted low, indicates a write operation to the flash device. When NAND is not used, this pin can be configured as a peripheral chip-select. Assertion level of this signal is configurable.
NAND_RDY (GPIO)	I	1	NAND operating condition Ready/Busy#. When high indicates that the device is ready for access. When set low indicates no access will be granted.
NAND_ALE (GPIO)	O	1	NAND Address Latch Enable. The NAND_ALE output controls writing to the address register. When ALE is high, the address is loaded on the rising edge of WE#. NAND_ALE must remain high during the entire address sequence.
NAND_CLE (GPIO)	O	1	NAND Command Latch Enable. The CLE output controls writing to the command register. When CLE is high, the command is loaded on the rising edge of WE#.
EXP_DATA	B	32	Expansion multiplexed Address/Data Bus. During the address phase, NAND_ALE pulse is used by the peripheral to latch the address. After address hold-time the data will be valid on the bus. MEM_BUS_SIZE is used to configure the peripheral interface width: • 00: Peripheral has an 8-Bits data bus interface. • 01: Peripheral has a 16-Bits data bus interface. • 10: Reserved • 11: Reserved Only lower 16-Bits are used.

Figure 4-7 NAND Command Latch Timing

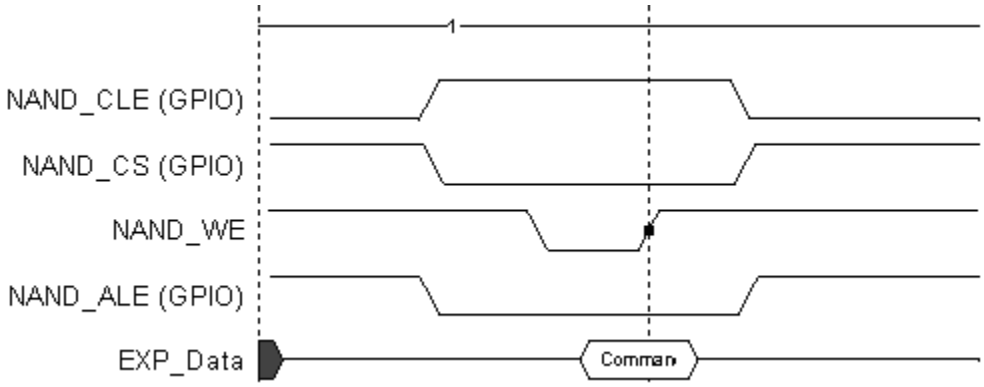


Figure 4-8 NAND Address Latch Timing

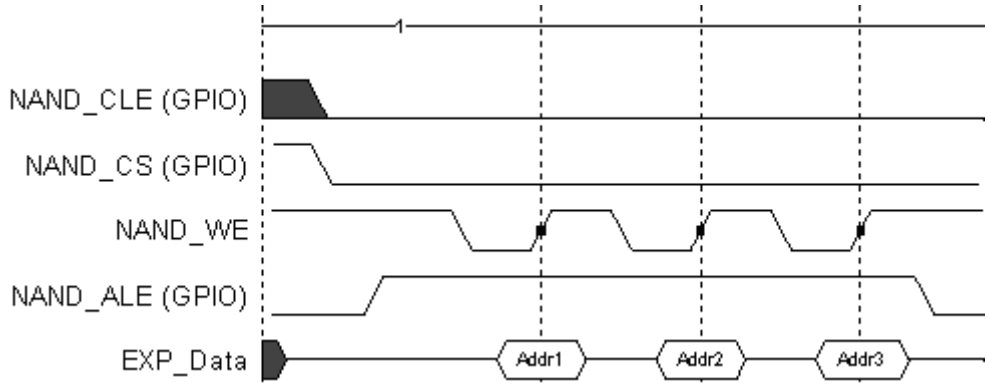
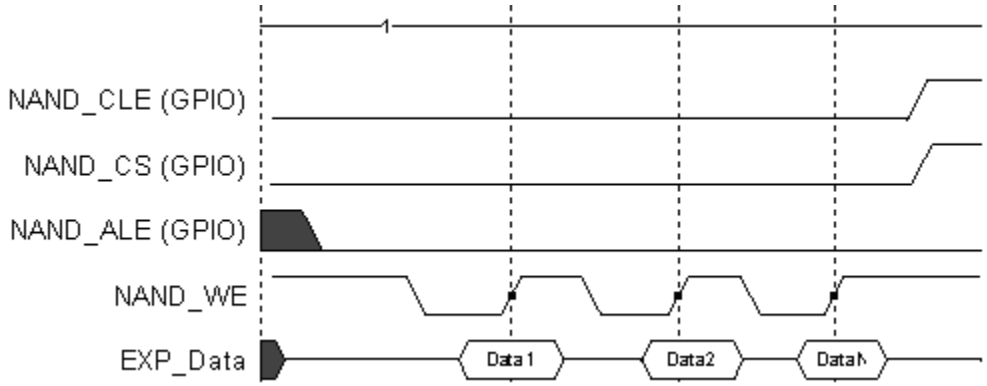


Figure 4-9 NAND Data Latch Timing



Each toggle of WE or RE to access the NAND flash will be executed over a full expansion cycle. The software initiates a transaction in the NAND allocated address space, which will cause an expansion cycle to be started. During the NAND Address cycle or the NAND Command cycle, the software will initiate the write transaction to the NAND address space placing the NAND address or command type value on the data bus. The expansion controller will issue a write access to the NAND flash, as if accessing a generic device. The software at the meantime will drive the NAND chip-select and the controls (CLE/ALE) through dedicated GPIO pins.

NOTE: NAND_RE and NAND_WE will not be toggled when there is an access to CS#[3:0].

4.5 Boot Flash Configurations

Chip-select 0 is dedicated for external boot when used. By default, EXP_CS0 is enabled and configured to operate in the Normal Mode. The External flash bus width is determined by the two input pins to the expansion block. Mode fields are configured as the following:

- ALE_MODE = 0
- STRB_MODE = 0

4.6 Clocks

One clock is input to the expansion block running at AHB clock rate (200 MHz). The expansion block has a built in ExpClk Generator, which is responsible of generating a divided synchronous clock to the external peripherals with a maximum of 66 MHz. The AHB clock can be a divide down by 3, 4, 5, 6 or 7. The divide integer is configurable by software, default is set to 7. A clock switch selects between AHB clock and the divider output, and defaults to AHB Clock upon reset and will switch to the divided AHB clock right after reset ends. Power down logic shall be default to activate the clock. When reset is asserted, output clock might have problems when switching to reference clock.

External expansion clock is provided and maybe suppressed by software, when not used by any of the peripherals. All programmable timing parameters are synchronized to this expansion clock.

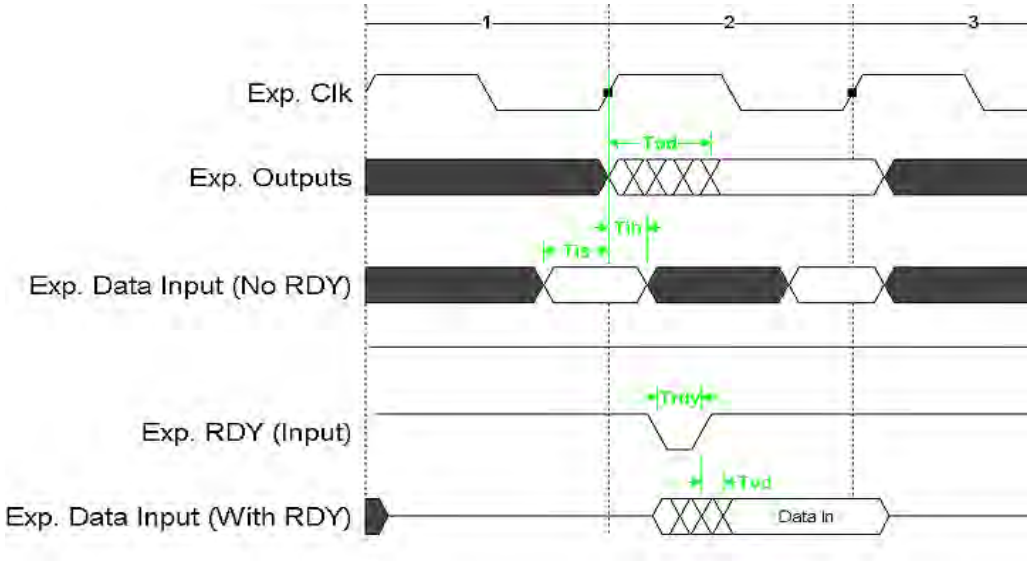
4.7 Reset

Asynchronous reset will be used to reset the block's internal logic. This reset may be a combination of the HW and SW resets provided by the chip and is synchronized externally to the AHB clock. A self-clear soft reset can be generated by the block's internal configuration logic when asserted by software. This self-clear reset will reset the entire block's logic synchronously. Configuration register are not affected by the block's soft-reset.

4.8 Expansion Interface AC Timing

Figure 4-10 and Table 4-7 describes the expansion interface AC timing parameters.

Figure 4-10 Expansion Bus Interface AC Timing



NOTE: These timing parameters are with respect to the LS10xMA. Timing requirements for peripheral devices may vary.

Table 4-7 AC Timing Parameters

Symbol	Parameter	Min.	Max.	Units	Notes
	EXP_CLK (Frequency)	23.6	66	MHz	
	EXP_CLK Duty cycle	40	60	%	
Tod	Expansion Outputs relative the rising edge of the EXP_CLK (Output Delay Time)				
	valid to valid	0	6	ns	(1)
	tri-state to valid	0	6	ns	(1)
	valid to tri-state	0	10	ns	(1)
Tz	Expansion Outputs Valid to tri-state				
Tis	Expansion Inputs Setup Time for read data with respect to the rising edge (When Trdy is not used)	1		Cycle	(2)
Tih	Expansion Inputs Hold Time for read data with respect to the rising edge (When Trdy is not used)	0		ns	(3)
Trdy	Ready Input required pulse width	9		ns	(4)
Tvd	Expected valid data after the rising/falling edge of the RDY Pulse. Since the active low RDY edge detection is configurable.		0	ns	(5)

NOTES:
1) Test Load 50pF.
2) Tis (min) value should be configured to be at least 1 EXP_CLK cycle.
3) Tih timing is relevant to the cycle in which CS and/or RE is de-asserted.
4) RDY is sampled and synchronized by HCLK (200 MHz).
5) Due to RDY synchronization logic before valid input data latching.

4.9 Asynchronous Interface

For an example of asynchronous timing, see Figure 4-4, Normal Mode Timing Diagram (RDY_EN = 1).

1. When Expansion output signals are changing on the same clock edge, the delay between these signals is considered to be between 0 -11 (ns).
2. When Expansion output signals are changing on different clock cycles, the maximum delay between these signals is considered to be: Unexpired of EXP_CLK)-11 (ns) to Unexpired of EXP_CLK)+11 (ns) where N depends on the configured parameters.
3. The read data (if expired is not used) relative to rising edge of CS or RE (whichever de-asserted first) should be at least (EXP_CLK Period)+11 (ns).
4. Upon EXP_RDY de-assertion state, expansion transaction may be completed within 2 EXP_CLK cycles when relative configuration parameters are set to minimum values.
5. The cycle in which the read data is sampled upon de-assertion of RE and/or CS whichever comes first.
6. While in Asynchronous mode (when RDY is used), the design does not guarantee the relationship between the CS and RE/WE signals. Since the detection of RDY pulse will cause the de-assertion of the CS and RE/WE signals simultaneously.

5 DDR2 SDRAM Interface

This section provides details on the LS10xMA DDR2 SDRAM controller.

5.1 DDR2 SDRAM Features

- Differential clock (CK and CK_N) commands provided on each clock edge. Data is fetched/stored on both edges of CK.
- Support of clock frequencies from 125 to 400 MHz.
- 16-Bit or 32-Bit data bus interfacing to x8 of x16 memories (x4 not supported). 16-Bit interface intended for low-cost applications using a single x16 memory.
- Bi-directional Differential Data Strobe (DQS) with data, source-synchronous data capture.
- Address bus supports up to 1 GB of memory with one chip select signal.
- Four or eight internal banks for concurrent operation.
- Data Mask (DM) for masking write data.
- Supports burst lengths of 4 cycles.
- Supports programmable auto-refresh.
- Configurable terminations: no termination, 75 Ω, 150 Ω.
- Configurable drive strength:13.4mA and 6.7mA (Applicable for 200 MHz and lower).
- 1.8 V I/O (SSTL_18 compatible).
- 256 megabit to 4 gigabit devices with x8/x16 data ports (4 gigabit support for x16 devices only)
- Full initialization of memory on memory controller reset.
- A programmable register interface to control memory device parameters and protocols.
- Built-in adjustable Delay Compensation Circuitry (DCC) for reliable data send and capture timing.
- Support for DDR2 power saving modes: Power down mode and self refresh mode.
- Pipelined command, read and write data interfaces to the memory controller.
- Advanced bank look-ahead features for high memory throughput.
- AHB multi-port design supporting bandwidth allocation arbitration scheme.

5.2 Signal Descriptions

Table 5-1 lists the DDR2 signal descriptions.

Table 5-1 DDR2 Signals

Signal Name	Dir	Size	Signal Description
DDR_CLK, DDR_CLK#	O	2	Differential Clock Outputs—All address and control signals are sampled by the DDR2 on the crossing of the positive edge of CK and negative edge of CK#. Input data (DQ and DQS) is referenced to the crossings of CK and CK#.
DDR_CLKE	O	1	Clock Enable Active High—When set low, memory DDR2 internal clock, input buffers and output drivers are de-activated. CKE must be maintained HIGH throughout read and write accesses.

Table 5-1 DDR2 Signals (continued)

Signal Name	Dir	Size	Signal Description
DDR_CS#	O	1	Chip Select—Active low CS# enables and disables selection of the targeted memory module. One chip select for each DDR2.
DDR_RAS#	O	1	Memory Row Address Strobe—Active low define the raw address for the required data.
DDR_CAS#	O	1	Memory Column Address Strobe—Active low define the column address for the required data.
DDR_WE#	O	1	Memory Write Enable—When set low, indicates a write operation to memory, otherwise indicates read.
DDR_DM[1]	O	4	Data Byte Mask—Input data byte to the memory is masked when DM is sampled high along with that input data during a write access. The DDR2 samples the DM signal on both edges of DS. One DM pin per data byte.
DDR_BA	O	3	Bank Select Address—Selects DDR2 bank for command access.
DDR_A	O	15	Address Bus—Provides row and column addresses for RAS and CAS commands.
DDR_DQ[1]	B	32	Data Bus—Bi-directional data bus with DDR2 memory.
DDR_DQS, DDR_DQS#[1]	B	8	Differential Data Strobe—Used to latch data read from the DDR2 memory for reads and used by the memory to sample data writes. DS is edge-aligned with read data and centered with respect to the write data. One differential DS pair per data byte.
DDR_ODT	O	1	On-Die Termination.

Note:
1. In 16- Bit mode DDR_DM[3:2], DDR_DQ[31:16], DDR_DQS[3:2], and DDR_DQS#[3:2] are not used.

5.3 DDR2 SDRAM Interface Bus Timing

Figure 5-1and Figure 5-2 illustrate the DDR2 write cycle and read cycle timing respectively. Table 5-2 lists the DDR2 timing parameters.

Figure 5-1 DDR2 Write Cycle Timing

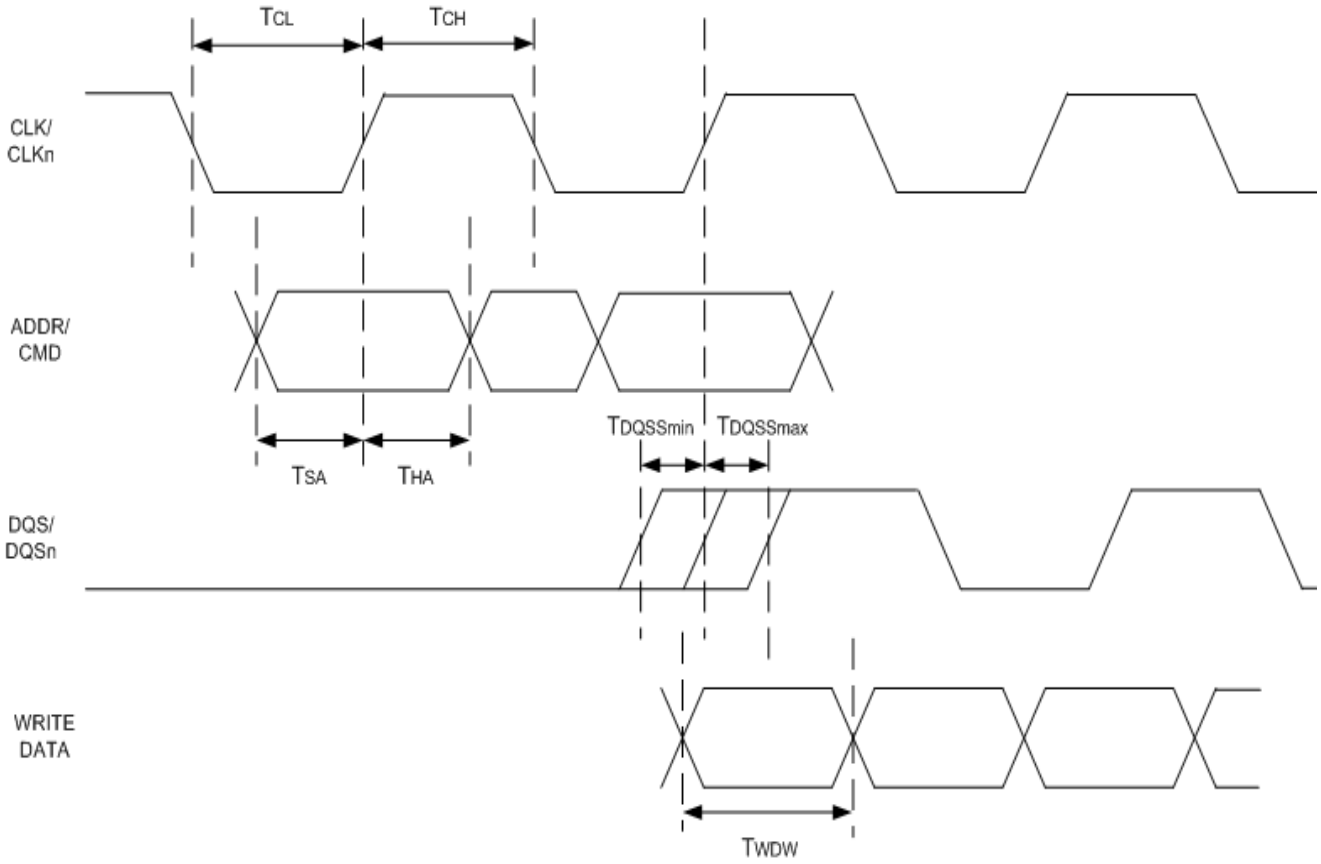


Figure 5-2 DDR2 Read Cycle Timing

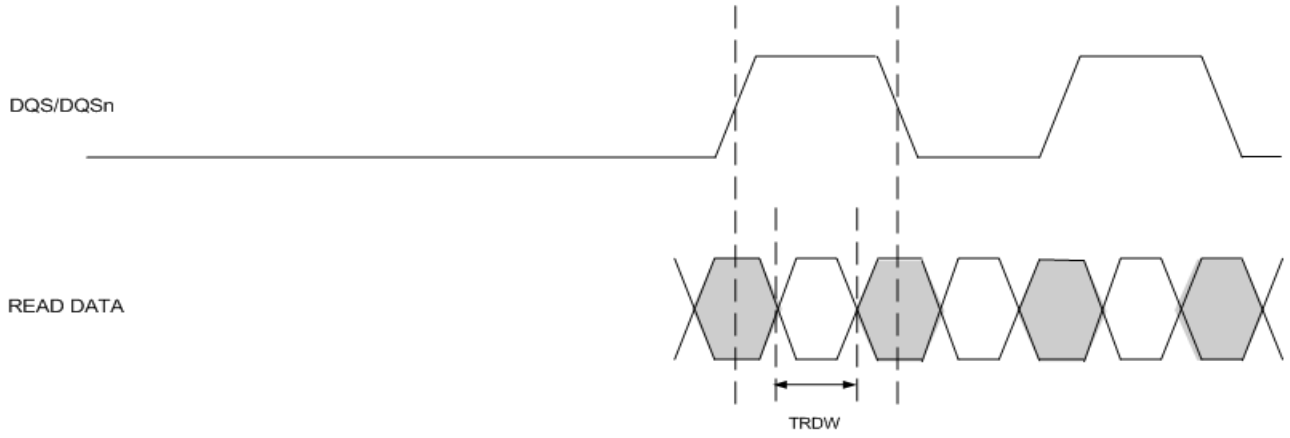


Table 5-2 DDR2 Timing Parameters

Symbol	Parameter	Min.	Max.	Units	Notes
TCH/TCL	Clock high/low level width 330MHz/400 MHz	48	52	%tck	
TDQSH/ TDQSL	Write DQS high/low level width	35	65	%tck	
TIS	Setup time for address and command signals to rising edge of clock 330 MHz 400 MHz	 0.7 0.46		ns	1
TIH	Hold time for address and command signals to rising edge of clock 330 MHz 400 MHz	 1.1 0.9		ns	1
TWDW	Write Data Window – the minimum time the output data is stable 330 MHz 400 MHz	 0.8 0.52		ns	2, 3
TDQSS	First DQS latching transition to associated clock edge	25	25	%tck	3
TRDW	Read Data Window - the minimum time read data must be stable 330MHz 400MHz	 0.65 0.54		ns	4, 5,6
TRTRIP	Round trip from chip to memory (CLK) and back (DQS) 330MHz 400MHz	 0.7 0.7	 2.2 2.2	ns	7

General Notes

- Specified timing is for full drive strength operation.
- All values were measured from vref to vref, unless otherwise is specified.
- All timing parameters with CLK signal are defined on CLK-CLKn crossing point.
- All timing parameters with DQS signal are defined on DQS-DQSn crossing point.

Notes

1. Assuming 1V/ns slew rate for address and command signals at the input to the DRAM.
2. The timing is for each 8 data bits and 1 mask bit, measured at the input to the DRAM with 2 inch trace length.
3. The timing of each DQS and its corresponding data and mask bits is determined during boot timing by executing the DDR training sequence function. This function is responsible for fine tuning the delay of Write Data and Write DQS relative to each other and to CLK.
4. At input of the LS10xMA device.
5. The training sequence function is responsible for fine tuning the delay of each read DQS to adjust to the memory DQ-DQS skew.
6. The read data window combines the setup and hold timing requirement of the device.
7. The round trip calculation does not include the maximal skew inside the memory from clock to DQS.

6 Ethernet Interface

This section provides details on the LS10xMA Ethernet interface. For register-level details to support software driver development, see *LS10xMA Register Reference Guide*.

6.1 Overview

The LS10xMA device supports two Ethernet Media Access Controllers (MACs). For a view of the Ethernet blocks within the top-level block diagram, see Figure 2-1. Both Ethernet0 (WAN) and Ethernet1 (LAN) ports can be configured as GMII / RGMII / MII / RMII and are supported by one MDIO interface.

6.2 Features

The following are the Ethernet MAC features:

- Compatible with IEEE 802.3 standards for 10/100/1000 Mbps Ethernet
- Supports half-duplex or full-duplex operation in all three speeds
- Statistics counter registers for RMON/MIB
- Embedded DMA
- Interrupt generation on packet receive and transmit completion, errors, latency timeout from packet received or *N* packets received
- Automatic pad and cyclic redundancy check generation on transmitted frames
- Frame extension and frame bursting at 1000 Mbps in half duplex mode
- Automatic discard of frames received with errors
- Receive IP, TCP, and UDP checksum off-load
- Support for jumbo frames up to 2 KB
- Rx packet parsing, capable of recognizing the following protocols: 802.3 Ethernet, VLAN, Q-in-Q (stacked VLANs) PPPoE, MPLS, IPv4, and IPv6
- L3/L4 offset detection and indication through descriptor status
- Full utilization of the TX 1 Gbps line
- Full-duplex flow control with recognition of incoming pause frames and hardware generation of transmitted pause frames
- Compatible with RMII standard 1.2 and RGMII standard 1.3—RGMII electrical characteristics compliant with RGMII standard 1.3 (based on 2.5V CMOS interface voltages as defined by JEDEC EIA/JESD805), and are not compliant with RGMII version 2.0 (specifically not based on 1.5V HSTL interface voltages as defined by JEDEC EIA JESD8-6)
- 8-entry EtherType match lookup table

- Support for MAC to MAC direct communication assuming (for connecting to another Ethernet MAC)
  - MDIO is not used for communication between MACs
  - When RGMII mode is used, the opposite MAC must be able to ignore the in-band link/speed duplex information (LS10xMA can optionally ignore this information)
  - When RMII mode is used, a direct connection is possible
  - When GMII (1G mode) is used a direct connection is possible
  - When MII (10/100) is used, a direct connection is typically not possible
  - External delay is applied.

The following are the traffic management features:

- Rx port shaping from 48Kb/s to 1Gb/s through packet discard at line or PAUSE frame based flow control
- Ingress Congestion Control (ICC), prioritizing defined “managed and reserved” flows.
- Access Control List (ACL)
- Tx port supports up to eight hardware managed queues
- Tx port supports hardware based scheduling (priority and/or DWRR)
- Tx port support hardware based rate shaping at port and queue levels

6.3 Functional Description

6.3.1 MAC Operation

The LS10xMA Ethernet MAC module implements a 10/100/1000 Mbps MAC compatible with the IEEE 802.3 standard. This MAC can operate in either half or full duplex at all three speeds.

The MAC transmit block takes data from the external FIFO interface, adds preambles, and, if necessary, add pad and Frame Check Sequence (FCS). The MAC receive block checks for valid preamble, FCS, alignment and length, and presents received frames to the MAC offload interfaces. Accepted frames are transferred to memory.

The Rx packet will be parsed by the hardware and some information will be extracted into the Rx descriptor. The parse-able protocols include 802.3 Ethernet, VLAN, Q-in-Q (stacked VLANs), PPPoE, MPLS, IPv4, and IPv6. The information extracted from these pockets will be: broadcast match, multicast/unicast hash match, specific Ethernet DA match, VLAN and priority information, L3/L4 offsets within the frame, packet type information, and EtherType match lookup table.

6.3.2 Traffic Management

The Ethernet ports are supported with hardware aimed at controlling packet congestion as described in Section 6.3.2.1 to Section 6.3.2.3.

6.3.2.1 Rx Shaping

Ingress traffic is shaped by discarding packets at the line when insufficient credit is available or by generating Ethernet PAUSE frames to flow-control the upstream traffic source. The following mechanisms generates pause logic.

- Rx rate monitor — created by a statically configurable value every clock tick and debited according to the number of bytes received. This is illustrated in Figure 6-1

- In-use counter — incremented when an Rx descriptor is being used up (by hardware) and decremented when a data buffer is de-queued off the descriptor list (by software). This is illustrated in Figure 6-2.

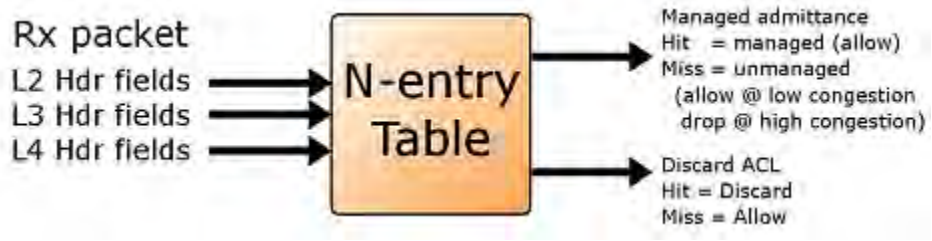
Figure 6-1 Rx Rate Monitor

Figure 6-2 In-Use Counter

6.3.2.2 Rx Admittance Control

Each Ethernet port implements two admittance control policies. The first is admittance based on priority during periods of ingress congestion. This policy monitors for average ingress queue depth and determines congestion based upon predefined queue depth thresholds. Depending on the degree of congestion a probabilistic discard scheme is employed for dropping unmanaged (or low-priority) traffic. The second policy is based on a discard-only ACL where flows identified to the hardware by software are detected as they come and discarded. Appropriate configuration of the Rx packet classification hardware by software is required for flow definition for both policies. A small table can hold entries to support both policies. Figure 6-3 provides a conceptual overview of this classification function.

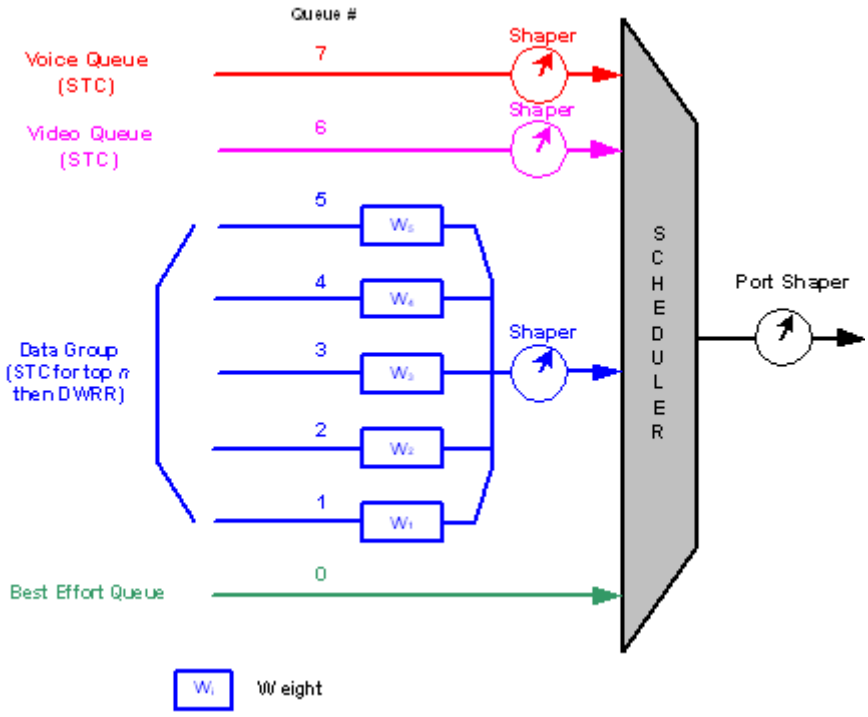
Figure 6-3 Rx Admittance Control



6.3.2.3 Tx Scheduling and Shaping

Logic associated with each Ethernet port can be configured to schedule and shape egress traffic according to QoS or type of packet flow. Figure 6-4 illustrates the MII interface timing

Figure 6-4 MII Interface Timing



6.4 Ethernet Configuration Options

The inputs at reset, TM_ETH[1:0]_MODE[1:0] define Ethernet0 (WAN) and Ethernet1 (LAN) configuration:

Table 6-1 Ethernet Configuration Options

Ethernet Configurations	TM_ETH[1:0]_MODE1	TM_ETH[1:0]_MODE0
RGMII	0	0
RMII	0	1
MII	1	0
GMII	1	1

6.5 Signal Descriptions

See Section 6.6 for Ethernet0 mode selection details.

6.5.1 MII Mode

For AC timing specification, see Section 6.7.2. Table 6-2 lists the MII signal description.

Table 6-2 MII Signals

Signal Name	Dir	Size	Signal Description
MII_REFCLK	O	1	Reference Clock for external PHY/Switch. 25 MHz reference clock, with ±50 ppm accuracy.
MII_TX_CLK	I	1	Transmit Clock—25 MHz in 100-Base mode, and 2.5 MHz in 10-Base mode, with ±100 ppm accuracy.
MII_TXD[3:0]	O	4	MII Transmit Data—Nibble-wide transmit data stream. Synchronous with MII_TX_CLK.
MII_TX_EN	O	1	MII Transmit Enable—Indicates data on MII_TXD is valid.
MII_TX_ER	O	1	MII Transmit Error—Request out for a transmit error condition.
MII_RX_CLK	I	1	MII Receive Clock—25 MHz in 100-Base mode, and 2.5 MHz in 10-Base mode.
MII_RXD[3:0]	I	4	MII Receive Data—Nibble-wide receive data stream. Synchronous with MII_RX_CLK.
MII_RX_DV	I	1	MII Receive Data Valid—Indicates receive frame in progress, and data on MII_RXD pins is valid.
MII_RX_ER	I	1	MII Received Error Detected—Indicates an error is occurring during a receive frame.
MII_CRS	I	1	MII Carrier Sense—Indicates traffic on link. Asynchronous signal.
MII_COL	I	1	MII Collision Detect—Indicates a collision has occurred. Asynchronous signal.

6.5.2 RMII Mode

For AC timing specification, see Section 6.7.3. Table 6-3 lists the RMII signal description.

Table 6-3 RMII Signals

Signal Name	Dir	Size	Signal Description
RMII_CLK	I	1	Reference Clock Input—Continuous 50 MHz reference clock, with ±50 ppm accuracy.
RMII_REFCLK	O	1	Reference Clock Output—Output reference clock for external PHY/Switch. Continuous 50 MHz reference clock, with ±50 ppm accuracy.
RMII_TX_EN	O	1	Transmit Enable—When high, indicates that MAC is presenting di-bits. Asserted synchronously with first nibble of the preamble and remains asserted while all di-bits to be transmitted are present. Synchronous to REFCLK.
RMII_TXD[1:0]	O	2	Transmit Data—TXD1 is the most significant. Synchronous to REFCLK.
RMII_CRS_DV	I	1	Carrier Sense / Receive Data Valid—Asserted by the PHY when the medium is non-idle. Data on RXD is considered valid when CRS_DV is asserted. During a false carrier event, CRS_DV shall remain asserted for the duration of the carrier activity. Not synchronous to REFCLK.
RMII_RX_ER	I	1	Receive Error Detected—High for one or more REFCLK cycles to indicate that an error was detected somewhere in the frame presently being transferred. Synchronous to REFCLK.
RMII_RXD[1:0]	I	2	Receive data—RXD1 is the most significant. Synchronous to REFCLK.

Table 6-3 RMII Signals (continued)

Signal Name	Dir	Size	Signal Description
RMII_PHY_LINK	I	1	Link indication
RMII_PHY_DPX	I	1	Duplex indication
RMII_PHY_SPEED	I	1	Speed indication

6.5.3 GMII Mode

For AC timing specification, see Section 6.7.4. Table 6-4 lists the GMII signal description.

Table 6-4 GMII Signals

Signal Name	Dir	Size	Signal Description
GMII_REFCLK	O	1	Reference Clock for external PHY/Switch. 25 MHz reference clock, with ±50 ppm accuracy.
GMII_GTX_CLK	O	1	GMII Gigabit Transmit Clock—125 MHz in 1000-Base mode.
GMII_TX_CLK	I	1	GMII Transmit Clock—This input is not required by 802.3 in GMII mode. It needs to be driven for transmit side to operate. The frequency of this clock need not be 125 MHz because the only reason we need this clock is to be able to switch from it to the real 125 MHz clock.
GMII_TXD[7:0]	O	8	GMII Transmit Data—Byte-wide transmit data stream. Synchronous with GMII_GTX_CLK.
GMII_TX_EN	O	1	GMII Transmit Enable—Indicates data on GMII_TXD is valid.
GMII_TX_ER	O	1	GMII Transmit Error—Request out for a transmit error condition.
GMII_RX_CLK	I	1	GMII Receive Clock—50 MHz
GMII_RXD[7:0]	I	8	GMII Receive Data—Byte-wide receive data stream. Synchronous with GMII_RX_CLK.
GMII_RX_DV	I	1	GMII Receive Data Valid—Indicates receive frame in progress, and data on GMII_RXD pins is valid.
GMII_RX_ER	I	1	GMII Received Error Detected—Indicates an error is occurring during a receive frame.
GMII_CRS	I	1	GMII Carrier Sense—Indicates traffic on link. Asynchronous signal.
GMII_COL	I	1	GMII Collision Detect—Indicates a collision has occurred. Asynchronous signal.

6.5.4 RGMII Mode

For AC timing specification, see Section 6.7.5. Table 6-5 lists the RGMII signal description.

Table 6-5 RGMII Signals

Signal Name	Dir	Size	Signal Description
RGMII_REFCLK	O	1	Reference Clock for external PHY/Switch. 25 MHz reference clock, with ±50 ppm accuracy.
RGMII_TXC	O	1	RGMII Transmit Clock—Depending on speed, this is a 125 MHz, 25 MHz, or 2.5 MHz clock.
RGMII_TXD[3:0]	O	4	RGMII Transmit Data—Interface is run at double data rate, with bits [3:0] presented on the rising edge of TXC, and bits [7:4] presented at the falling edge of TXC.

Table 6-5 RGMII Signals

Signal Name	Dir	Size	Signal Description
RGMII_RXC	I	1	RGMII Receive Clock—Depending on speed, this is a 125 MHz, 25 MHz, or 2.5 MHz clock, with ±50 ppm accuracy.
RGMII_RX_CTL	I	1	RGMII Receive Control—RX_DV is presented on rising edge of RXC. A logical derivative of RX_DV and RX_ER is presented on the falling edge of RXC.
RGMII_RXD[3:0]	I	4	RGMII Receive Data—Interface is run at double data rate, with bits [3:0] presented on the rising edge of RXC, and bits [7:4] presented at the falling edge of RXC.
RGMII_TX_CTL	O	1	RGMII Transmit Control—TX_EN is presented on the rising edge of TXC. A logical derivative of TX_EN and TX_ER is presented on the falling edge of TXC.

6.5.5 Management Interface

Table 6-6 lists the management interface signal description.

Table 6-6 Management Interface Signals

Signal Name	Dir	Size	Signal Description
GEM_MDC	O	1	Serial Management Interface Clock—A continuous clock stream is not expected.
GEM_MDIO	B	1	Serial Management Interface Data—Transfers data in and out of the device synchronously to MDC.

6.6 Ethernet Signal Multiplexing

Logic levels latched at reset on TM_ETH[1:0]_MODE[1:0] for Ethernet0 and Ethernet1 determine the MAC configuration as shown in Table 6-7.

Table 6-7 Ethernet Configuration

TM_ETH[1:0]_MODE[0] [=] 1 TM_ETH[1:0]_MODE[1] [=] 1 GMII	TM_ETH[1:0]_MODE[0] [=] 1 TM_ETH[1:0]_MODE[1] [=] 0 RMII	TM_ETH[1:0]_MODE[0] [=] 0 TM_ETH[1:0]_MODE[1] [=] 0 RGMII	TM_ETH[1:0]_MODE[0] [=] 0 TM_ETH[1:0]_MODE[1] [=] 1 MII
GMII[1:0]_RXD0	RMII[1:0]_RXD0	RGMII[1:0]_RXD0	MII[1:0]_RXD0
GMII[1:0]_RXD1	RMII[1:0]_RXD1	RGMII[1:0]_RXD1	MII[1:0]_RXD1
GMII[1:0]_RXD2	RMII[1:0]_PHY_DPX	RGMII[1:0]_RXD2	MII[1:0]_RXD2
GMII[1:0]_RXD3	RMII[1:0]_PHY_SPEED	RGMII[1:0]_RXD3	MII[1:0]_RXD3
GMII[1:0]_TXD0	RMII[1:0]_TXD0	RGMII[1:0]_TXD0	MII[1:0]_TXD0
GMII[1:0]_TXD1	RMII[1:0]_TXD1	RGMII[1:0]_TXD1	MII[1:0]_TXD1
GMII[1:0]_TXD2	RMII[1:0]_RX_ER	RGMII[1:0]_TXD2	MII[1:0]_TXD2
GMII[1:0]_TXD3	RMII[1:0]_PHY_LINK	RGMII[1:0]_TXD3	MII[1:0]_TXD3
GMII[1:0]_RX_CLK	RMII[1:0]_CLK	RGMII[1:0]_RXC	MII[1:0]_RX_CLK
GMII[1:0]_RX_DV	RMII[1:0]_CRS_DV	RGMII[1:0]_RX_CTL	MII[1:0]_RX_DV

TM_ETH[1:0]_MODE[0] [=] 1 TM_ETH[1:0]_MODE[1] [=] 1 GMII	TM_ETH[1:0]_MODE[0] [=] 1 TM_ETH[1:0]_MODE[1] [=] 0 RMII	TM_ETH[1:0]_MODE[0] [=] 0 TM_ETH[1:0]_MODE[1] [=] 0 RGMII	TM_ETH[1:0]_MODE[0] [=] 0 TM_ETH[1:0]_MODE[1] [=] 1 MII
GMII[1:0]_TX_EN	RMII[1:0]_TX_EN	RGMII[1:0]_TXC	MII[1:0]_TX_EN
GMII[1:0]_TX_CLK	—	—	MII[1:0]_TX_CLK
GMII[1:0]_COL	—	—	MII[1:0]_COL
GMII[1:0]_CRS	—	—	MII[1:0]_CRS
GMII[1:0]_RX_ER	—	—	MII[1:0]_RX_ER
GMII[1:0]_TX_ER	—	RGMII[1:0]_TX_CTL	MII[1:0]_TX_ER
GMII[1:0]_REFCLK	RMII[1:0]_REFCLK	RGMII[1:0]_REFCLK	MII[1:0]_REFCLK

6.7 Ethernet Interface Bus Timing

NOTE:

1. RGMII was verified with 2.5V devices and supplies.
2. GMII, MII, and RMII were verified with 3.3V devices and supplies.

6.7.1 MDIO Interface Timing

Figure 6-5 illustrates the MDIO interface timing.

Figure 6-5 MDIO Interface Timing

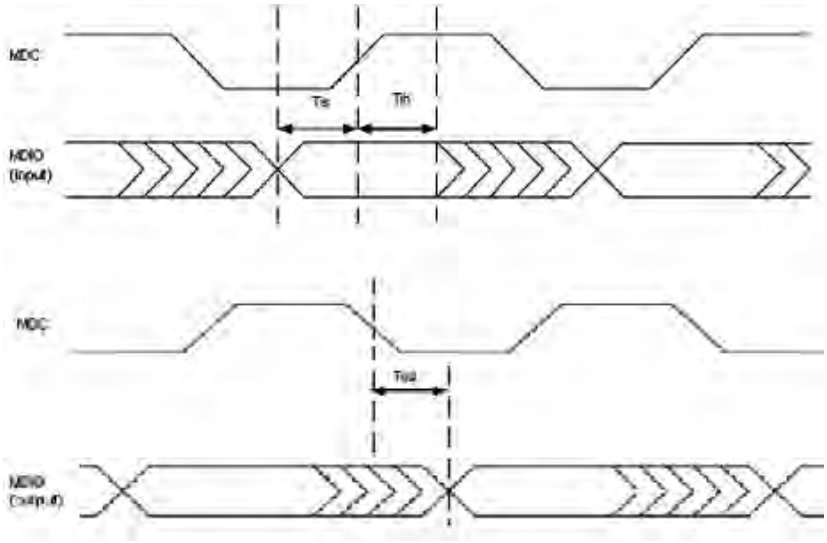


Table 6-8 lists the MDIO interface timing parameters.

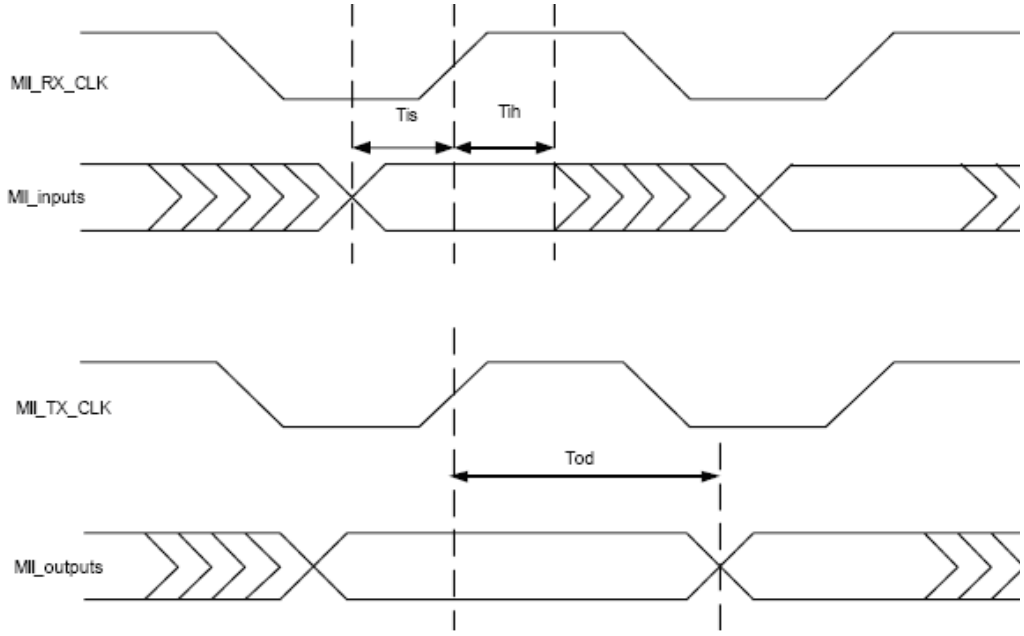
Table 6-8 MDIO Interface Timing Parameters

Symbol	Parameter	Min.	Typ.	Max.	Units	Notes
	Clock edge rate (All Clocks)	0.25		2	V/ns	1
	MDC Frequency			2.5	MHz	
	MDC Duty Cycle	40		60	%	
Tis	MDIO Input Setup Time	20			ns	
Tih	MDIO Input Hold Time	0			ns	
Tod	MDIO Output Delay Time	-10		10	ns	
NOTES: 1. The rise and fall times are determined by the edge rate in V/ns. A “Max” edge rate is the fastest rate at which a clock transitions.						

6.7.2 MII Interface Timing

Figure 6-6 illustrates the MII interface timing.

Figure 6-6 MII Interface Timing



*MII_COL and MII_CRS are specified as asynchronous

Table 6-9 lists the MII interface timing parameters.

Table 6-9 MII Interface Timing Parameters

Symbol	Parameter	Min.	Typ.	Max.	Units	Notes
	Clock edge rate (All Clocks)	0.25		2	V/ns	
	MII_CLK Frequency, 100 Mbps		25		MHz	
	MII_CLK Frequency, 10 Mbps		2.5		MHz	
	MII_CLK Accuracy	-100		100	ppm	
	MII_CLK Duty Cycle	35		65	%	
Tis	Input Setup Time	10			ns	1, 3
Tih	Input Hold Time	10			ns	2, 3
Tod	Output Delay Time	1		15	ns	2

NOTES:
1. Referenced to clock level of 0.8V.
2. Referenced to clock level of 2.0V.
3. Not including MII_CRS and MII_COL. These signals are not required to transition synchronously.

6.7.3 RMII Interface Timing

Figure 6-7 illustrates the RMII interface timing.

Figure 6-7 RMII Interface Timing

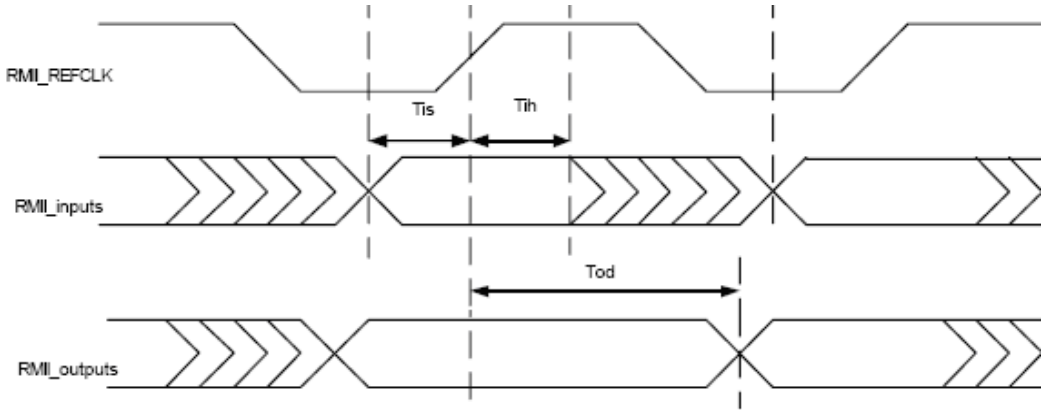


Table 6-10 lists the RMII interface timing parameters.

Table 6-10 RMII Interface Timing Parameters

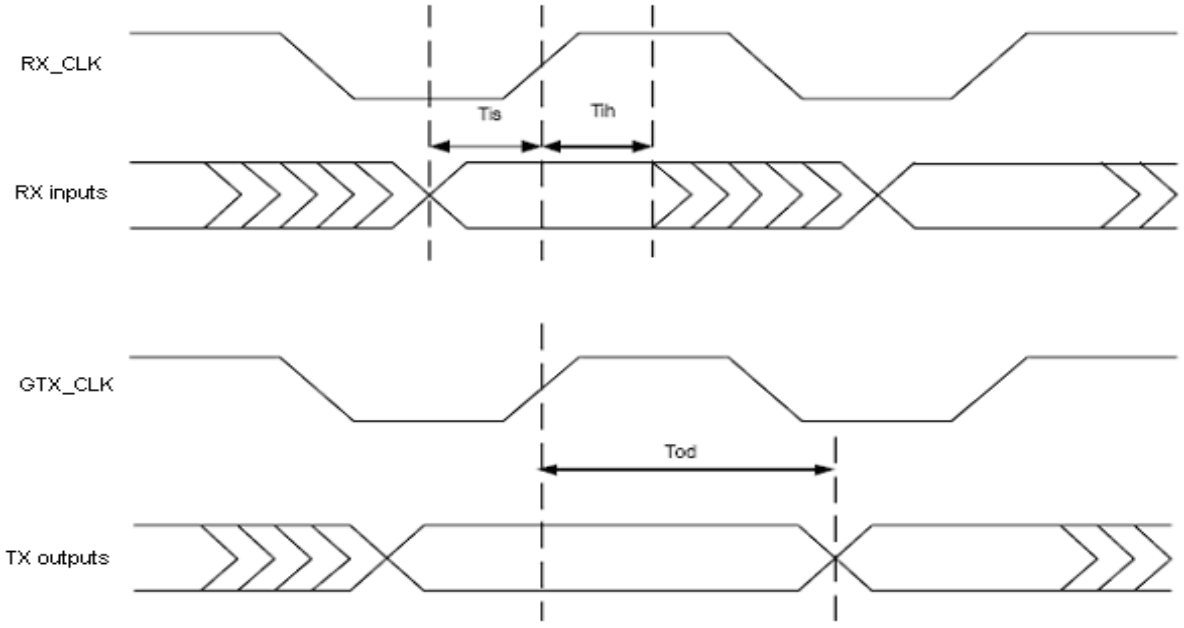
Symbol	Parameter	Min.	Typ.	Max.	Units	Notes
	Rise and fall times (signals and clock)	1		5	ns	1
	RMII_REFCLK Frequency		50		MHz	
	RMII_REFCLK Accuracy	-50		50	ppm	
	RMII_REFCLK Duty Cycle	35		65	%	
Tis	Input Setup Time	4			ns	2
Tih	Input Hold Time	2			ns	2
Tod	Output Delay Time	2		14	ns	2,3

NOTES:
1. Measured between the points on the waveform, which cross 0.8V and 2.0V.
2. Reference to clock level of 1.4V.
3. Output drivers shall be capable of meeting the output requirements while driving a 25pF or greater load.

6.7.4 GMII Interface Timing

Figure 6-8 illustrates the GMII interface timing.

Figure 6-8 GMII Interface Timing



* GMII_COL and GMII_CRS are specified as asynchronous.

Table 6-11 lists the GMII interface timing parameters.

Table 6-11 GMII Interface Timing Parameters

Symbol	Parameter	Min.	Typ.	Max.	Units	Notes
	Clock rise/fall time (all Clocks)			1	ns	1,5
	Clock slew rate (all clocks)	0.6			V/ns	1,2,5
	GTX_CLK frequency		125		MHz	
	GTX_CLK accuracy	-100		100	ppm	
	GTX_CLK period	7.5		8.5	ns	
	RX_CLK period	7.5			ns	
	RX_CLK and GTX_CLK high/low time	2.5			ns	
Tis	Input setup time	2			ns	3,7
Tih	Input hold time	0			ns	4,7
Tod	Output delay time	0.5		4.5	ns	4,6

Table 6-11 GMII Interface Timing Parameters (continued)

Symbol	Parameter	Min.	Typ.	Max.	Units	Notes
1. Reference levels of 0.7v and 1.9v. 2. Clock Skew rate is the instantaneous rate of change of the clock potential with respect to time (dV/dt), not an average value over the entire rise or fall time interval. Conformance with this specification guarantees that the clock signals will rise and fall monotonically through the switching region. 3. Referenced to clock level of 0.7v 4. Referenced to clock level of 1.9v 5. Must be measured using a test circuit defined in Figure 35-20 in the standard. 6. Must be measured using a test circuit defined in Figure 35-21 in the standard. 7. Not including GMII_CRS and GMII_COL. These signals are not required to transition synchronously.						

6.7.5 RGMII Interface Timing

Figure 6-9 illustrates the RGMII interface timing.

Figure 6-9 RGMII Interface Timing

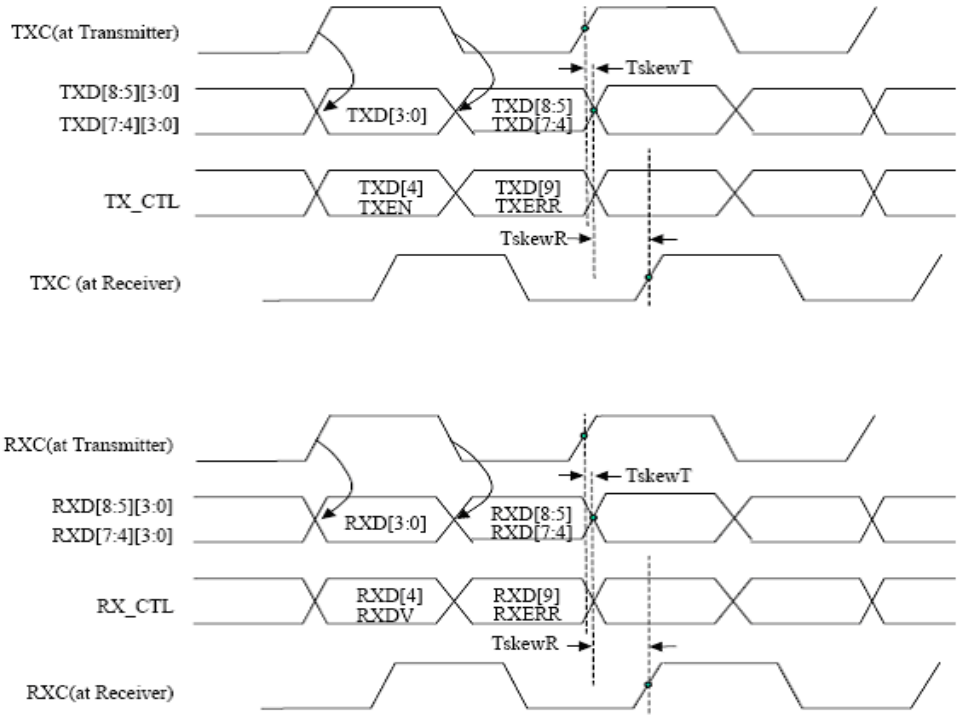


Table 6-12 lists the RGMII interface timing parameters.

Table 6-12 RGMII Interface Timing Parameters

Symbol	Parameter	Min.	Typ.	Max.	Units	Notes
TskewT	Data to Clock output Skew (at Transmitter)	-500	0	500	ps	1,6

Table 6-12 RGMII Interface Timing Parameters (continued)

Symbol	Parameter	Min.	Typ.	Max.	Units	Notes
TskewR	Data to Clock input Skew (at Receiver)	1		2.6	ns	2,6
Tcyc	Clock Cycle Duration (1000 Mbps)	7.2	8	8.8	ns	3,6
Tcyc	Clock Cycle Duration (100 Mbps)	36	40	44	ns	3,6
Tcyc	Clock Cycle Duration (10 Mbps)	360	400	440	ns	3,6
	Clock Accuracy	-50		50	ppm	6
Duty_G	Duty Cycle for Gigabit	45	50	55	%	4,5,6
Duty_T	Duty Cycle for 10/100T	40	50	60	%	4,5,6
Tr / Tf	Rise / Fall Time (20-80%)			.75	ns	6

NOTES:

1. The skew at the transmitter is specified for 15pF clock and data loads. If the load is significantly different on the board, it is the responsibility of the board designer to characterize the output skew and to account for it. Parallel and serial terminations are required and circuit simulation is highly recommended (application note should be made available on this subject).
2. It is recommended that the PHY/switch that is connected to this interface includes internal clock delays (for both receive and transmit directions), in order to overcome the difference introduced between TskewT and TskewR. An alternative option of additional trace delay on the board is not recommended due to the additional load it introduces. For 10/100, the Maximum value is unspecified.
3. For 10 Mbps and 100 Mbps, Tcyc will scale to 400 ns ± 40 ns and 40 ns ± 4 ns respectively.
4. Duty cycle may be stretched/shrunk during speed changes or while transitioning to a received packet's clock domain as long as minimum duty cycle is not violated and stretching occurs for no more than three Tcyc of the lowest speed transitioned between.
5. Cycle values are defined in percentages of the nominal clock period so to make this table speed independent.
6. Needs to be compliant with RGMII v1.3 (thus based upon 2.5 V CMOS interface voltages as defined by JEDEC EIA/JESD8-5), and not compliant with RGMII v2.0 (specifically, not based upon 1.5 V HSTL interface voltages as defined by JEDEC EIA/JESD8-6).

7 PCI Express Interface

This section provides details on the LS10xMA PCIe interface.

7.1 Overview

The LS10xMA device two includes PCI Express (PCIe) interface controllers, each of which supports a single lane. The PCIe interfaces are compatible with the PCIe Base Specification rev 1.1. The two PCIe SERDES share a common reference clock, which must be provided externally.

NOTE: LS101MAXE7DFA device supports only one PCIe.

If the bootstrap, TM_PCIe_EXT_REFCLK (EXP_A[12]), is asserted then the two SERDES look for a clock to be input on the pcie_refclk pins of the device. If no external clock is supplied in this scenario then the PCIe interfaces will not operate. When the PCI Express interface is used, the bootstrap, TM_PCIe_EXT_REFCLK (EXP_A[12]), must be asserted at reset time

7.2 PCI Express Features

The following are the controller features:

- Compliant with PCIe rev 1.1.
- PCI Express port with bootstrap input selection between root port.
- On-chip PCI Express SerDes.
- Flexible AHB Bursting mechanism to maximize efficiency when Accessing AHB Fabric and specifically DDR2 SDRAM.
- Requester DMA with two independent prioritized DMA channels:-
  - Three modes of operation:-
    - Direct memory-mapped mode – 256 MB memory space whereby AHB reads and writes are translated directly into PCIe transactions (Channel 1 only).
    - Bulk transfer descriptor queue mode – description of a bulk transfer up to 4 KBytes read from system memory and appropriate PCIe packet commands pushed into channel command queue (Channel 0 only).
    - Indirect register mode – queuing of commands through zero wait state register interface, with polling for command completion (Channel 0 and 1).
  - Transfers up to Max Payload Length (128 bytes) of data.
  - Can be used to generate configuration requests when PCIe is configured as a root port.
- Completer DMA channel maps internal AHB memory space into external PCIe memory space.
  - Programmable internal local address offset per BAR.
  - Completion generation based on RCB for received memory read requests.
- Programmable arbitration scheme for transmit channel based on DMA/virtual channels and available buffer space for each type of request.

The following are the interface features:

- Support for all PCIe Link power management states.

- Common 100 RefClk frequency.
- Transmit path:
  - Differential Tx amplitude with less than ± 10% variation across temperature, voltage, and process.
  - Programmable transmit level.
  - Programmable Tx boost of 0- 5.75 dB in increments of ~ .37 dB.
- Receive path:
  - Pin-programmable equalization of .5-4 dB in .5-dB increments.
  - Programmable Rx Clock Data Recovery (CDR) based on a digital PLL (DPLL) for fast startup and dynamic characteristics independent of silicon processing for high yielding, robust jitter tolerance.
  - Rx CDR that can tolerate run lengths of thousands of bits.
  - Programmable Loss of Signal (LOS) threshold.
- Test features:
  - Built-in support for Byte Error Rate Testing (BERT) per lane.
  - Built-in support high-resolution scope acquisition of eye patterns of known periodicity per Rx signal pair.

7.3 PCIe Configuration Options

The inputs at reset, TM_PCIE[1:0]_Mode independently define the two PCIe interfaces.

Table 7-1 PCIe Configuration Options

PCIe Configuration	TM_PCIe1_MODE	TM_PCIe0_MODE
PCIE[1:0] Root Node	0	0

7.4 PCIe Signals

Table 7-2 lists the PCIe signals.

Table 7-2 PCI Express Signal

Symbol	Dir.	Size	Name/Function
Lane 1			
PCIE0_HSO_P PCIE0_HSO_M	O	1	Lane 1, High-speed differential transmit output pair
PCIE0_HSI_P PCIE0_HSI_M	I	1	Lane 1, High-speed differential transmit input pair
Lane 2			
PCIE1_HSO_P PCIE1_HSO_M	O	1	Lane 2, High-speed differential transmit output pair
PCIE1_HSI_P PCIE1_HSI_M	I	1	Lane 2, High-speed differential transmit input pair

Table 7-2 PCI Express Signal (continued)

Symbol	Dir.	Size	Name/Function
Clock Module			
PCIE_RESREF	B	1	Reference resistor connection.Connect to ground with a 191R 1% resistor.
PCIE_REFCLK_P	I	1	Reference clock input, differential high
PCIE_REFCLK_N	I	1	Reference clock input, differential low

8 Serial Peripheral Interface

This section provides details on the LS10xMA SPI interface.

8.1 Overview

On LS10xMA device, the SPI typically works with TDM and interfaces with SLIC/SLAC device to connect to standard POTS telephone equipment. It can also be used to load boot code from an external ROM.

The multi-chain SPI provides a common interface mechanism to SLICs from multiple vendors and is used to setup such parameters as μ-law / a-law PCM conversion. For a view of the Serial Peripheral Interface within the top-level block diagram, see Figure 8-1.

8.2 Features

- Serial interface operation
- Clock bit-rate - User can dynamically control the serial bit rate of the data transfer.
- Serial Master - Enables serial communication with serial slave peripheral devices.
- Independent masking of interrupts - transmit FIFO overflow, transmit FIFO empty, receive FIFO full, receive FIFO underflow, and receive FIFO overflow interrupts can all be masked independently.
- Data frame size (4 to 16 bits) - The frame size of each data transfer is under the control of the programmer.
- FIFO depth - The depth of the transmit and receive FIFO buffers is 8 words deep. The FIFO width is fixed at 16 bits.
- Number of slave select outputs - 3 serial slave select output signals can be generated.
- Hardware/software slave-select - Dedicated hardware slave-select lines can be used or software control can be used to target the serial slave device.

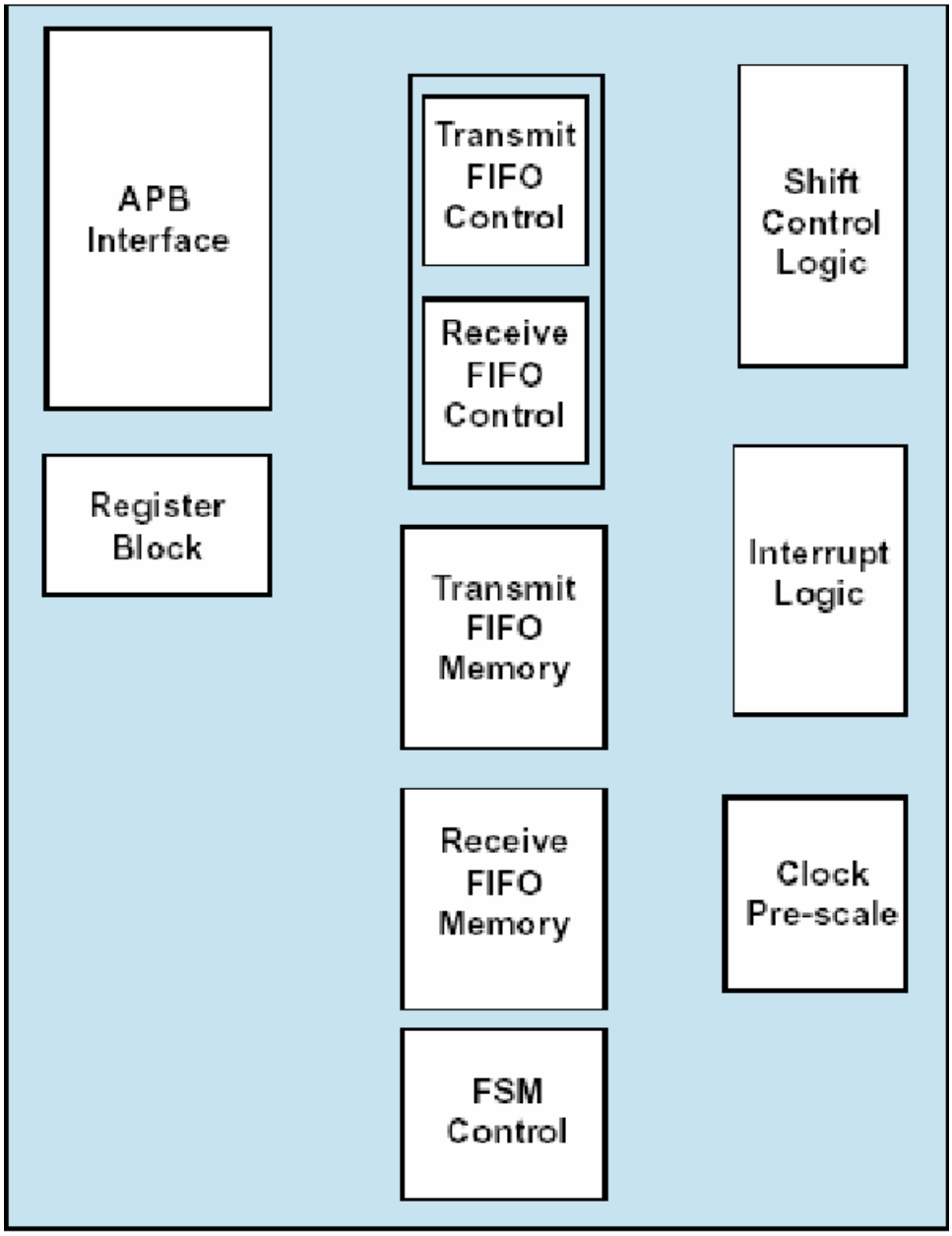
NOTE: The SPI interlace signals are multiplexed with GPIOs:

- SPI_SCLK (muxed with GPIO24)
- SPI_TXD (muxed with GPIO25)
- SPI_RXD (muxed with GPIO12)
- SPI_SS0# (muxed with GPIO13)
- SPI_SS1# (muxed with GPIO26)
- SPI_SS2# (muxed with GPIO27)

8.3 Functional Description

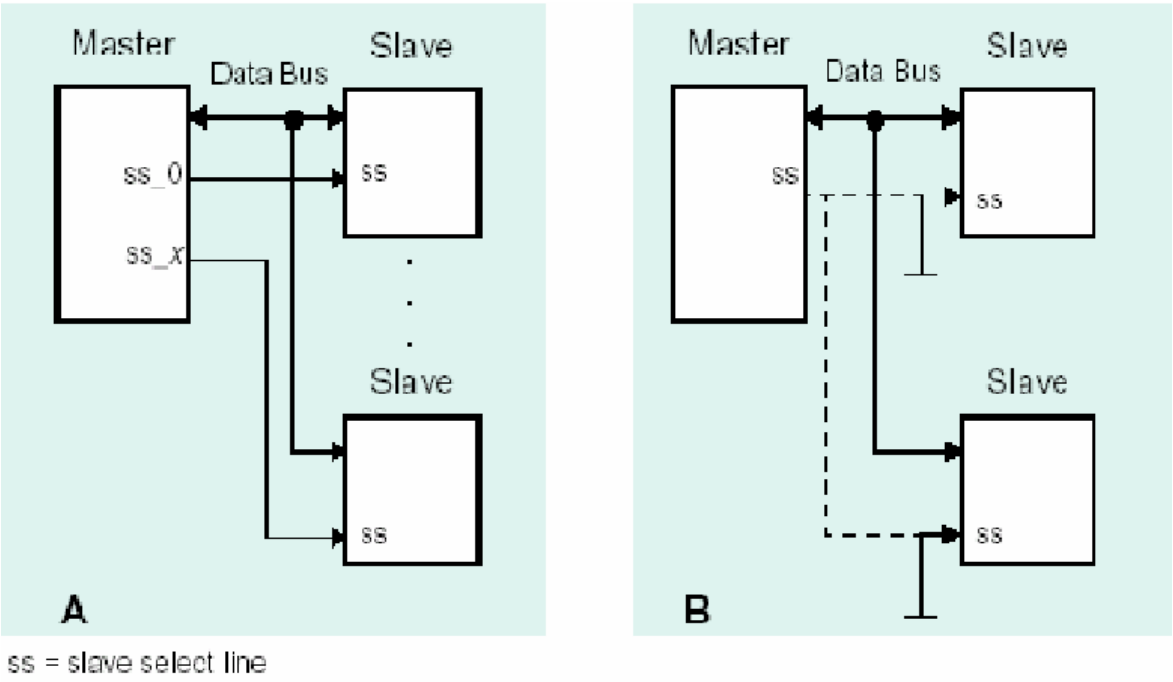
The LS10xMA SPI interface can connect to any serial slave peripheral device that supports SPI – a four-wire, full-duplex, serial protocol. There are four possible combinations for the serial clock's phase and polarity. The clock phase (SCPH) determines whether the serial transfer begins with the falling edge of the slave select signal or the first edge of the serial clock. The slave select line is held high when the SPI interface is idle or disabled. Figure 8-1 presents a high-level block diagram of the LS10xMA SPI Interface.

Figure 8-1 Block Diagram of SPI block



The LS10xMA SPI interface allows for serial slave to be selected/addressed using either hardware or software. When implemented in hardware, serial slaves are selected under the control of dedicated hardware select lines. The number of select lines generated from the serial master is equal to the number of serial slaves present on the bus. The serial master device asserts the select line of the target serial slave before data transfer begins. This architecture is illustrated in Figure 8-2 A.

Figure 8-2 Hardware/Software Slave Selection



When implemented in software, the input select line of each serial slave can either originate for a single slave select output pin on the serial master (user must configure the master to have one slave select output) or be permanently grounded. The main program in the software domain controls the selection of the target slave device. This architecture is illustrated in Figure 8-2 B. The LS10xMA SPI interface does not enforce hardware or software control, the user can configure the interface for either implementation. Figure 8-3 presents an example of an LS10xMA using SPI to interface to multiple SLIC devices.

8.4 Signal Descriptions

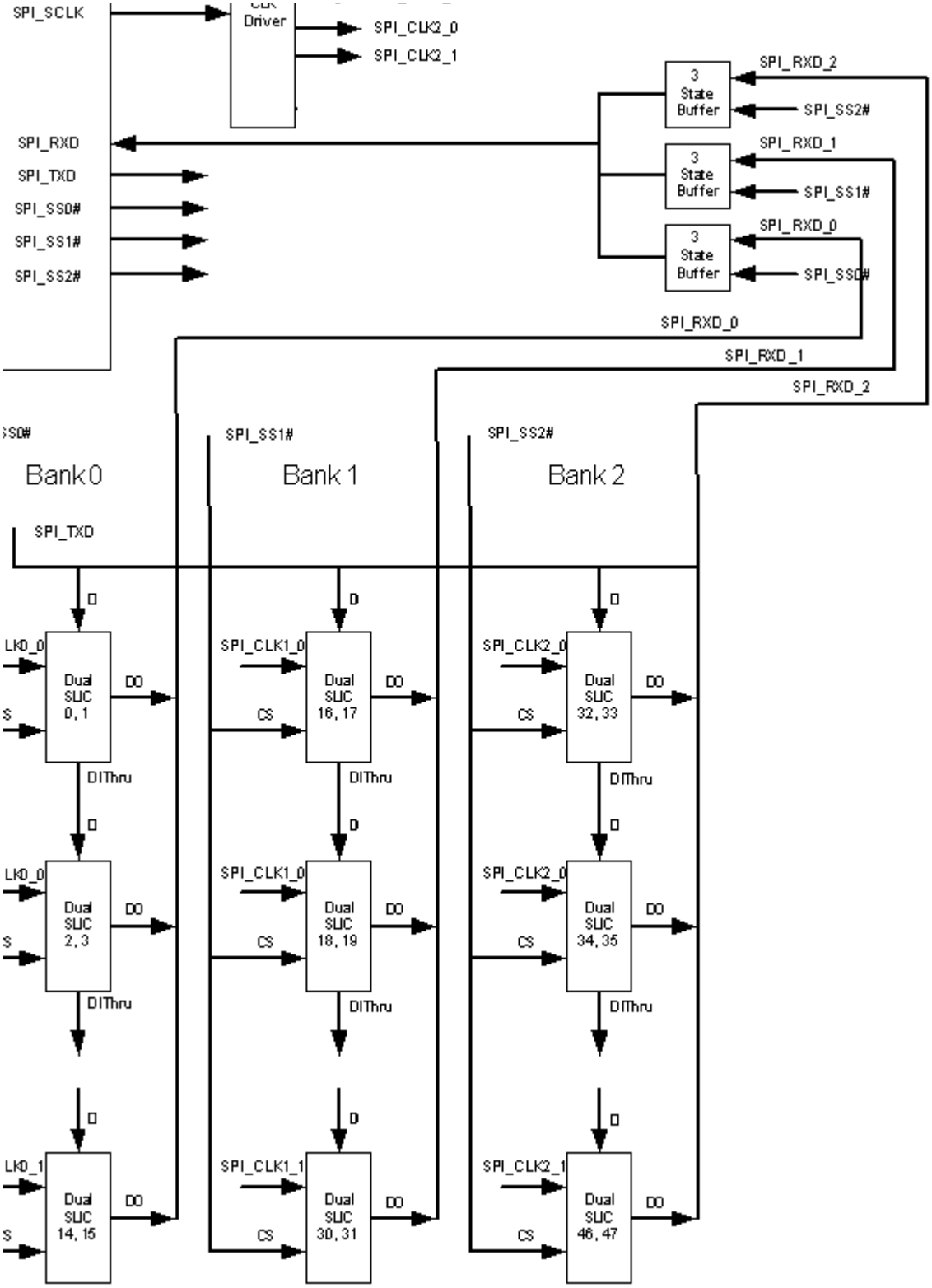
Table 8-1lists the SPI interface signal descriptions.

Table 8-1 LS10xMA SPI Interface Signals

Signal Name	Dir.	Size	Description
SPI_SCLK	O	1	Serial Bit-rate Clock. The sclk_out signal is synchronous to the ssi_clk signal
SPI_TXD	O	1	Transmit Data Signal. Output data from the serial master or serial slave is transmitted on this line.
SPI_RXD	I	1	Receive Data Signal. Input data from a serial master or serial slave device is received on this line.
SPI_SS#	O	3	Slave Select Output. Active Low. Hardware slave select signal. The width of the signal is equal to the number of slaves present on the serial bus. SS0 is used for boot device, when configured for SPI boot.

Figure 8-3 illustrates the connection between the SPI bus and the SLIC devices.

Figure 8-3 SPI Bus Connection to SLIC Devices



8.5 SPI Timing

Figure 8-4 illustrates the SPI timing waveforms.

Figure 8-4 SPI Timing Waveforms

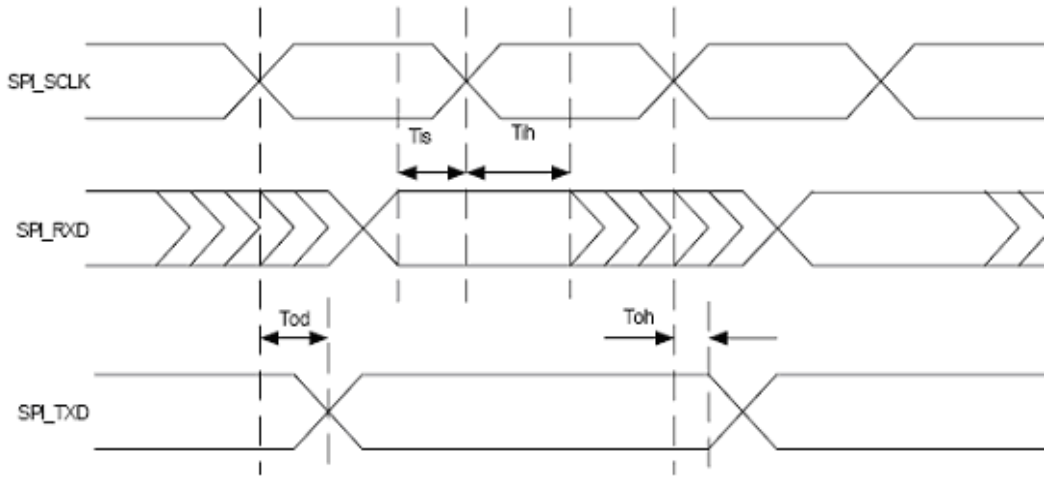


Table 8-2 lists the SPI setup and hold times.

Table 8-2 SPI Setup and Hold Times

Symbol	Parameter	Min.	Typ.	Max.	Units	Notes
	SPI_SCLK Frequency			4	MHz	
	Clock Duty Cycle (All Clocks)	35		65	%	
Tis	Input Setup Time SPI_RXD	20			ns	1
Tih	Input Hold Time SPI_RXD	2			ns	1
Tod	Output Delay Time SPI_TXD			8	ns	1, 2
Toh	Output Hold Time SPI_TXD	-5			ns	1, 2

1. Input can be referenced to either the rising or falling edge of the clock. Output is always reference to the opposite edge then that of the input.
2. SPI_SS# signals are configured when output clock is not active, thus have no AC requirements.
3. Maximum capacitive load is 54 pF.
4. Maximum capacitive load for SPI_SSn# is 36 pF.

8.6 Transmit and Receive FIFO Buffers

Two FIFO memories are used to buffer the transmit and receive data paths. The FIFO depth is 8, allowing 8x16-bit values to be stored independently in both transmit and receive modes. Transmit data is loaded from the transmit

FIFO, serialized, and sent over the serial line (txd) to the target peripheral. Receive data from the peripheral (on rxd) is converted to parallel words and loaded into the receive FIFO buffer. Serial data is propagated on one edge of the serial clock and captured on the opposite edge when the data is stable. Each data word that is transferred on the serial bus is referred to as a *data frame*. The serial data frame may range from 4 to 16 bits in length.

NOTE: The transmit and receive FIFO buffers are cleared when the SPI interface is disabled (SSI_EN=0) or reset.

8.7 SPI Interrupts

The LS10xMA SPI interface supports combined interrupt requests. The combined interrupt request is the ORed result of all other SPI interrupts after masking. All SPI interrupts have the same active high polarity level.

- **Transmit FIFO Empty Interrupt:** Set when the transmit FIFO is equal to or below its threshold value and requires service to prevent an under-run. The threshold value, set through a Software-programmable register, is used to determine the level of transmit FIFO entries at which an interrupt is generated. This interrupt is cleared by hardware when data is written into the transmit FIFO buffer, bringing it over the threshold level.
- **Transmit FIFO Overflow Interrupt:** Set when a processor access attempts to write into the transmit FIFO after it has been completely filled. When set, data written from the processor is discarded. This interrupt remains set until the user reads the transmit FIFO overflow interrupt clear register (TXOICR).
- **Receive FIFO Full Interrupt:** Set when the receive FIFO is equal to or above its threshold value plus 1 and requires service to prevent an overflow. The threshold value, set through a Software-programmable register, is used to determine the level of receive FIFO entries at which an interrupt is generated. This interrupt is cleared by hardware when data is read from the receive FIFO buffer, bringing it below the threshold level.
- **Receive FIFO Overflow Interrupt:** Set when the receive logic attempts to place data into the receive FIFO after it has been completely filled. When set, newly received data is discarded. This interrupt remains set until the user reads the receive FIFO overflow interrupt clear register (RXOICR).
- **Receive FIFO Underflow Interrupt:** Set when a processor access attempts to read from the receive FIFO when it is empty. When set, zeros are read back from the receive FIFO. This interrupt remains set until the processor reads the receive FIFO underflow interrupt clear register (RXUICR).
- **Combined Interrupt Request:** This interrupt is the ORed result of all the above interrupt requests after masking. To mask this interrupt signal the user must mask all other SPI interface interrupt requests.

8.8 Transfer Modes

The LS10xMA SPI interface operates in three modes:

- Transmit and Receive
- Transmit Only
- Receive Only

To set the transfer mode (TMOD), write to control register 0 (CTRLR0).

NOTE: The transfer mode setting does not affect the duplex of the serial transfer.

8.8.1 Transmit and Receive

When TMOD = 2‘b00, both transmit and receive logic are valid. The data transfer occurs as normal according the selected frame format (that is, serial protocol). Transmit data is popped from the transmit FIFO and sent through the txd line to the target device, which replies with data on the rxd line. The receive data from the target device is moved from the receive shift register into the receive FIFO at the end of each data frame. Error flags related to the transmit and receive FIFO logic are enabled in this mode.

8.8.2 Transmit Only

When TMOD = 2‘b01, the receive data is invalid and should not be stored in the receive FIFO. The data transfer occurs as normal, according to the selected frame format (that is, serial protocol). Transmit data is popped from the transmit FIFO and sent through the txd line to the target device, which replies with data on the rxd line. At the end of the data frame, the receive shift register does not load its newly received data into the receive FIFO. The data in the receive shift register is overwritten by the next transfer. Error flags related to the receive FIFO logic are disabled in this mode. Interrupts originating from the receive logic should be masked by the software when this mode is entered.

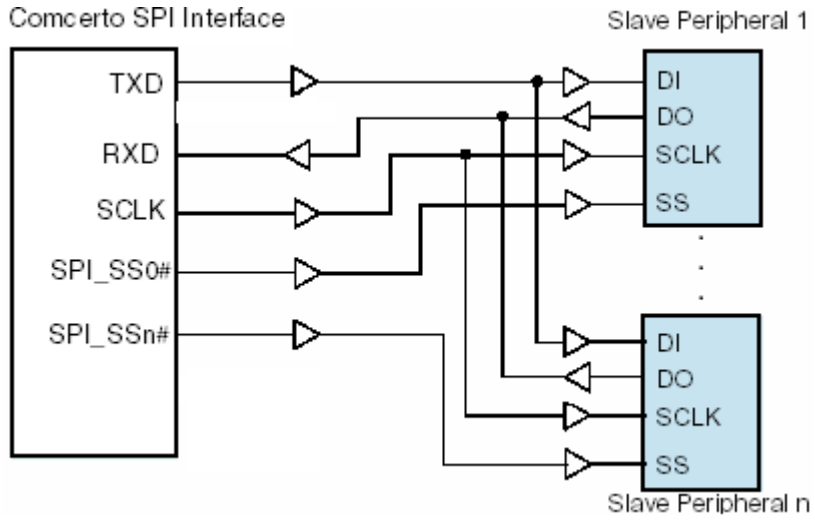
8.8.3 Receive Only

When TMOD = 2‘b10, the transmit data is invalid. When configured as a slave, the transmit FIFO is never popped in Receive Only mode. Data from a previous transfer is re-transmitted from the Shift Register. The data transfer occurs as normal according to the selected frame format (serial protocol). The receive data from the target device is moved from the receive shift register into the receive FIFO at the end of each data frame. Error flags related to the transmit FIFO logic are disabled in this mode. Interrupts originating from the transmit logic should be masked by the user when this mode is entered.

8.9 Operational Modes

The LS10xMA SPI interface operates only in “Serial Master Mode”. This mode enables serial communication with serial slave peripheral devices. Figure 8-5 shows an example of the SPI interface configured as a serial master with all other devices on the serial bus configured as serial slaves.

Figure 8-5 LS10xMA Configured as Master Device



As a serial master device, the LS10xMA SPI interface generates the serial clock that regulates the flow of data bits. The serial clock, when driven from the SPI interface master, only toggles during active data transfers. The number of active clock edges is equal to the number of bits driven on the data lines. At other times, the serial clock is held in an inactive state as defined by the protocol under which it operates. The serial master may transmit data at a variety of baud rates, which are under the control of a software programmable control register.

The baud rate is calculated from the following formula:

```
AHB Clock / SCKDV
```

where AHB clock is the frequency of the internal AHB clock (controlled by the clock and reset block), and SCKDV is any even value between 2 and 65534. SCKDV is set by writing to the BAUDR register.

The LS10xMA SPI interface can assert all of its slave select output lines (when hardware is used to select target slaves) to *broadcast* to all slaves on the bus. In response, only one serial slave drives data back onto the master's RXD line. The serial master can transmit data to all serial slaves on the bus simultaneously when the slave peripherals select lines have been enabled in the SER register. Throughout this section, the term *broadcast mode* will be referred to the situation when the master is transmitting data to all serial slaves on the bus.

Data transfers are started by the SPI interface when at least one valid data entry is present in the transmit FIFO and a serial slave device is selected. When actively transferring data, the busy flag (BUSY) in the Status Register (SR) is set. The software driver must wait until the busy flag is cleared before attempting to begin a new serial transfer.

When the transfer mode is transmit and receive (TMOD=2'b00) or transmit only (TMOD=2'b01), transfers are terminated by the shift control logic when the transmit FIFO is empty. For continuous data transfers, the user must ensure that the transmit FIFO buffer does not become empty before all the data has been transmitted. The transmit FIFO threshold level (see TXFLR), can be used to early interrupt the ACP, indicating that the transmit FIFO buffer is nearly empty. The user may also write a block of data (at least two FIFO entries) into the transmit FIFO before enabling a serial-slave. This ensures that serial transmission does not begin until the number of data frames that make up the continuous transfer are present in the transmit FIFO.

A typical software flow for completing a continuous serial transfer from the LS10xMA SPI interface to a serial-slave peripheral, when TMOD=2'b00 or TMOD=2'b01 is described as follows:

1. If the SPI interface is enabled, disable it by writing ‘0’ to SSIENR.
2. Write to the CTRL0 register, setting the frame format, transfer mode, data frame size, and so on.
3. Set the baud rate of the serial transfer by writing to the BAUDR register.
4. Not Used. (If the serial protocol is Microwire, set Microwire transfer options by writing to MWCR register.)
5. Set the transmit and receive FIFO threshold levels by writing to the TXFTLR and RXFTLR registers.
6. Enable the target slave by writing to the SER register.
7. Set the interrupt masks by writing the IMR register.
8. Enable the SPI interface by writing ‘1’ to the SSIENR register.
9. Fill the transmit FIFO with the data for serial transfer. The serial transfer will begin as soon as the first data word is present in the transmit FIFO.
10. Poll the BUSY bit by reading the SR register, or you can, use interrupts. If a transmit FIFO empty interrupt request is made, write the transmit FIFO. If a receive FIFO full interrupt request is made, read the receive FIFO.
11. The transfer is stopped by the shift control logic when the transmit FIFO is empty. When the transfer is done, the BUSY status is reset to 0.
12. Read the receive FIFO until empty— if TMOD!= 2‘b01.

When the transfer mode is receive only (TMOD=2‘b10), a serial transfer is started by writing one “dummy” data word into the transmit FIFO when a serial-slave is elected. If the serial transfer is continuous, this same data word is retransmitted until the serial transfer is completed.

The transmit FIFO is popped only once at the beginning and may remain empty for the duration of the serial transfer. The end of the serial transfer is controlled by the number of frames (NDF) field in control register 1 (CTRLR1). If for example the user wishes to receive 24 data frames from a serial-slave peripheral, the NDF field should be programmed with the value 23. The receive logic will terminate the serial transfer when the number of frames received is equal to the NDF value + 1.

This transfer mode increases the bandwidth of the internal peripheral bus because the transmit FIFO will never need to be serviced during the transfer. The receive FIFO buffer should be read each time the receive FIFO generates a FIFO full interrupt request to prevent an overflow. The receive FIFO threshold level (RXFTLR) can be used to give early indication that the receive FIFO is nearly full.

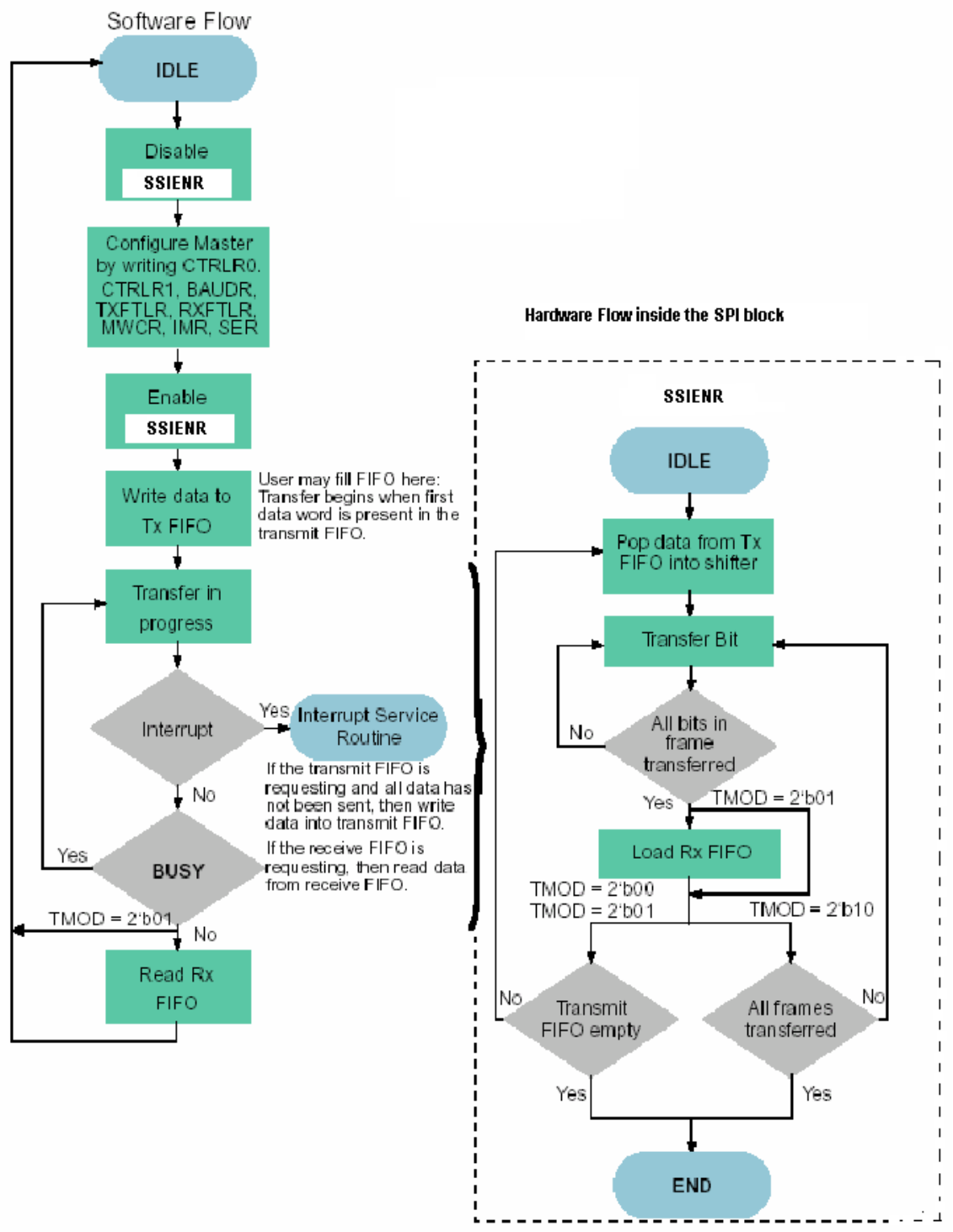
A typical software flow for completing a continuous serial transfer from the SPI interface to a serial-slave peripheral, when TMOD=2‘b10, is described as follows:

1. If the SPI interface is enabled, disable it by writing ‘0’ to the SSIENR register.
2. Write to the CTRLR0 register, setting the frame format, transfer mode, data frame size, and so on.
3. Set the baud rate of the serial transfer by writing to the BAUDR register.
4. Not used. (If the serial protocol is Microwire, set Microwire transfer options by writing to MWCR register.)
5. Set the transmit and receive FIFO threshold levels by writing to the TXFTLR and RXFTLR registers.
6. Set the NDF field in control register 1 (CTLR1) to the number of data frames that are to be received (1).
7. Set the interrupt masks by writing the IMR register.
8. Enable the target slave by writing to the SER register.
9. Enable the SPI interface by writing ‘1’ to the SSIENR register.

10. Write a dummy data word into the transmit FIFO. Serial transfers begin as soon as the word is present in the transmit FIFO.
11. Poll the BUSY signal by reading the SR register. If a receive FIFO full request is made, read the receive FIFO.
12. The transfer is stopped by the shift control logic when the number of words received is equal to NDF–1. When the transfer is done, the BUSY status is reset to 0.
13. Read the receive FIFO until empty.

Figure 8-6 shows a typical software flow for starting a SPI master serial transfer.

Figure 8-6 SPI Block Master Transfer Flow



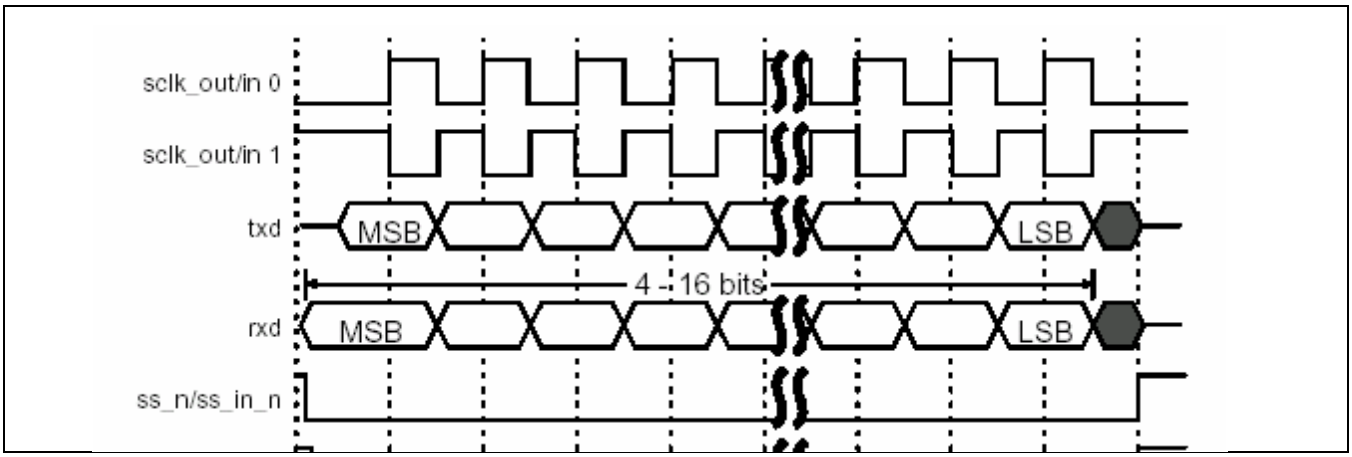
8.10 SPI Clock Polarity and Phase Settings

With the SPI, the clock polarity (SCPOL) determines whether the inactive state of the serial clock is high or low. To transmit data, both SPI devices must have identical SCPH (serial clock phase) and SCPOL values. In the LS10xMA SPI interface, these values are set in the CTLR0 register. The data frame can be 4 to 16 bits in length. When the SCPH = 0, data transmission begins on the falling edge of the slave select signal. The first data bit is captured by the master and slave peripherals on the first edge of the serial clock, therefore, valid data must be present on the txd and rxd lines prior to the first serial clock edge. Figure 8-7 shows a timing diagram for a single SPI data transfer with SCPH = 0. The serial clock is shown for SCPOL = 0 and SCPOL = 1.

The following signals are illustrated in the Figure 8-7 to Figure 8-10:

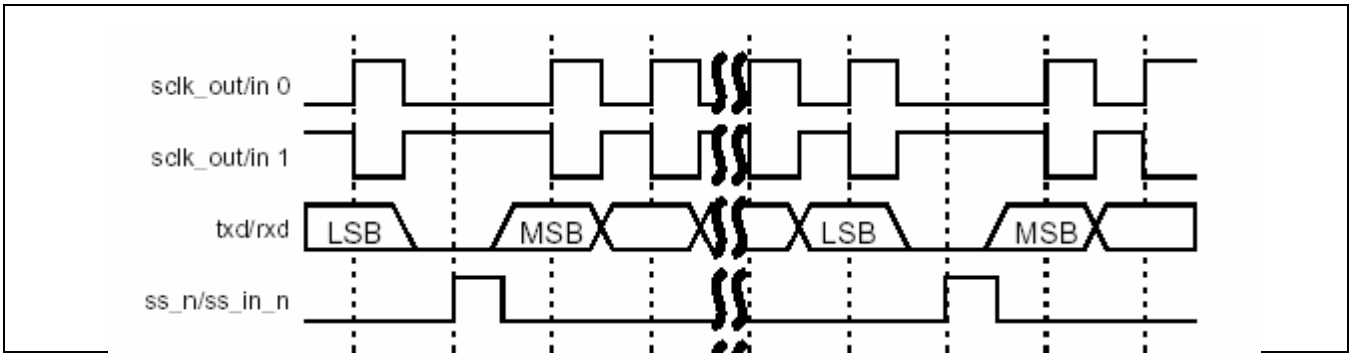
- sclk_out serial clock from the SPI interface.
- ss_n slave select signal from the SPI interface
- txd transmit data line for the SPI interface
- rxd receive data line for the SPI interface

Figure 8-7 SPI Serial Format (SCPH = 0)



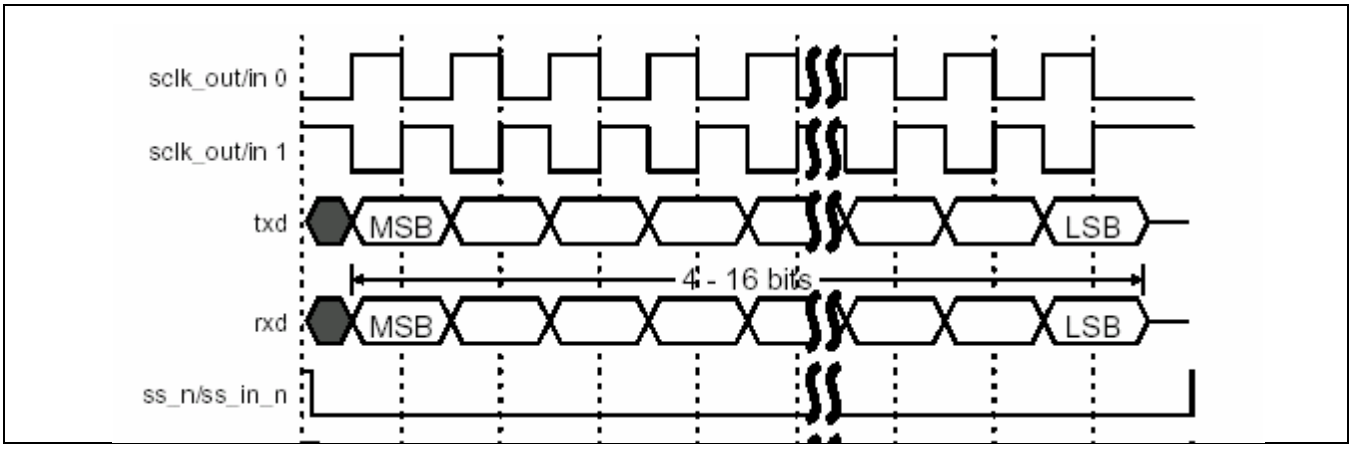
As data transmission starts on the falling edge of the slave select signal when SCPH = 0. Continuous data transfers require the slave select signal to toggle before beginning the next data frame. This is illustrated in Figure 8-8.

Figure 8-8 SPI Serial Format Continuous Transfers (SCPH = 0)



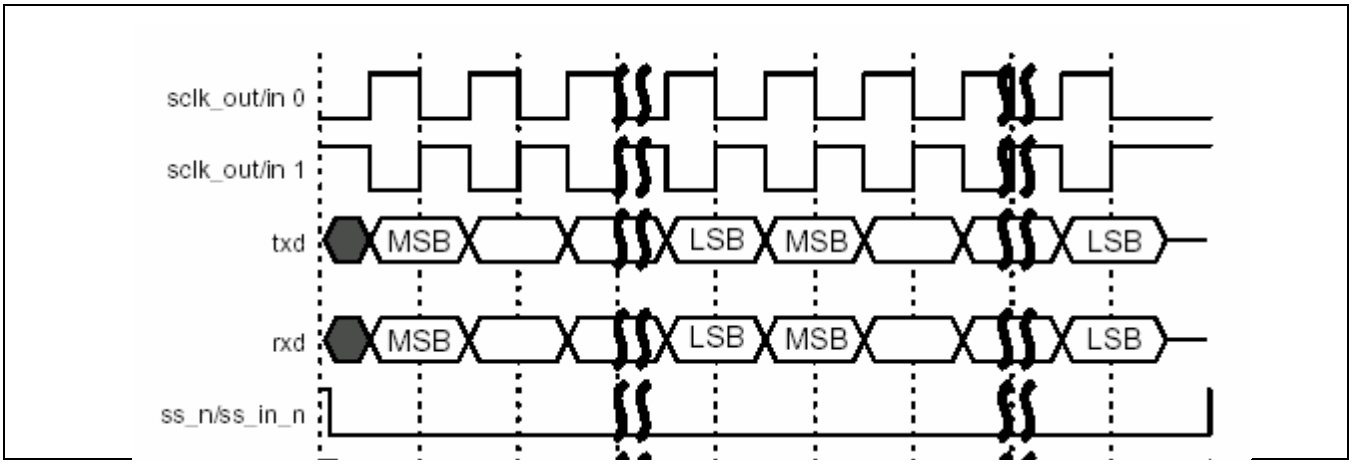
When SCPH = 1, both master and slave peripherals begin transmitting data on the first serial clock edge after the slave select line is activated. The first data bit is captured on the second serial clock edge. During continuous data frame transfers, the slave select line may be held active low until the last bit of the last frame has been captured. Figure 8-9 shows the timing diagram for the SPI format when SCPH = 1.

Figure 8-9 SPI Serial Format (SCPH = 1)



Continuous data frames are transferred in the same way as single frames, with the MSB of the next frame following directly after the LSB of the current frame. The slave-select signal is held active for the duration of the transfer. Figure 8-10 shows the timing diagram for continuous SPI transfers when SCPH=1.

Figure 8-10 SPI Serial Format Continuous Transfer (SCPH = 1)

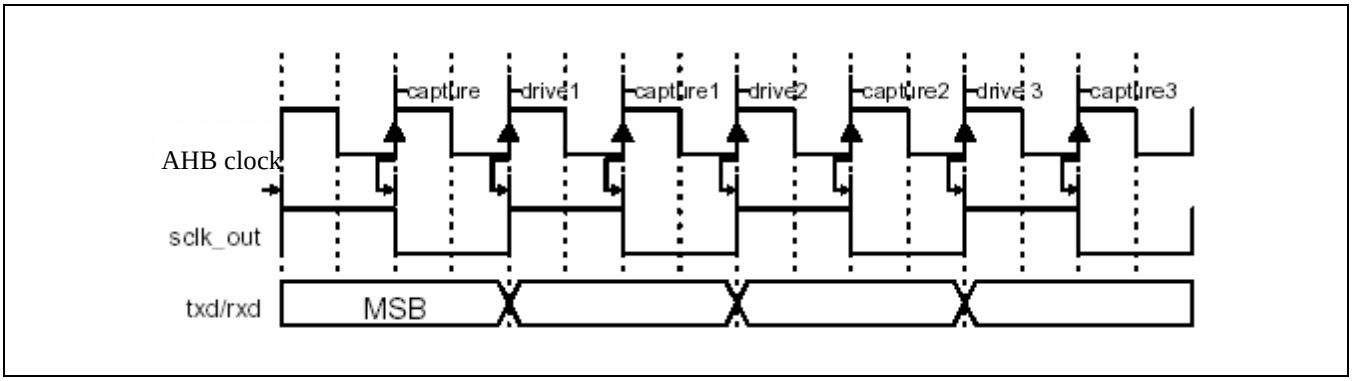


8.11 Clock Ratios

The maximum frequency of the bit rate clock (sclk_out) is one-half the frequency of the internal bus, AHB up to 4 MHz. This allows the shift control logic to capture data on one clock edge of sclk_out and propagate data on the opposite edge. This is illustrated in Figure 8-11.

The sclk_out line only toggles when an active transfer is in progress. At all other times, it is held in an inactive state as defined by the serial protocol under which it operates. SCKDV is a programmable register holding any even value from 0 to 65,534. If SCKDV=0, sclk_out is disabled.ow

Figure 8-11 Maximum sclk_out/AHB Ratio



A summary of the frequency ratio restrictions between the bit-rate clock (sclk_out/sclk_in) and the internal AHB clock are described as:

```
Master: AHB clock >= 2 × (maximum Fsclk_out)
```

9 Security Accelerator

This section provides details on the LS10xMA Security accelerator interface.

9.1 Overview

The IPSec Engine provides cryptographic functionality. It accelerates IPSec protocol processing by combining hash and ciphering engines, a special purpose DMA engine, ESP / AH packet processing logic, and interrupt mitigation. The LS10xMA hardware architecture provides direct access to the IPSec Engine's slave interface, hence supporting software control of the raw cryptographic mode.

NOTE: Security co-processor is not supported on LS101MAXE7DFA device.

9.2 Features

IPSec Features

- Data Encryption Standard Execution
  - AES, DES, and 3DES algorithms
  - ECB and CBC modes for DES and 3DES
  - CTR and CBC mode for AES (128 bit,192 bit, 256 bit)
  - AES-CBC mode cipher. 128, 192, 256 bit key sizes
  - DES-CBC mode cipher. 56 and 168 bit (3DES) key sizes
- Message digest execution
  - SHA1, SHA-256 with 160- or 256-Bit message digest
  - MD5 with 128-bit message digest
  - HMAC with either algorithm
  - HMAC-MD5 mode hash
  - HMAC-SHA1 mode hash
  - HMAC-SHA256 mode hash
- Raw hashing modes: MD5, SHA-1, SHA-256
- AH mode processing
- ESP mode processing
- Transport mode processing
- Tunnel mode processing
- IPv4 and IPv6 processing
- Extended Sequence Numbers
- DDT based packet memory architecture
- SA bundles and nested tunnels are not handled explicitly by the hardware. They may be constructed by sequencing the same packet through the EAOE multiple times via software
- Provides 64 entries long (per IN/OUT direction) control FIFOs holding multiple configurations to be written to IPSEC configurations registers

- Provides 64 entries long (per IN/OUT direction) status FIFOs holding multiple status reads from IPSEC status registers
- ARM interrupt mitigation
- Allows access to slave interface, in order to support software-controlled raw cryptographic mode
- IKEv 2 Support

SSL Support Features

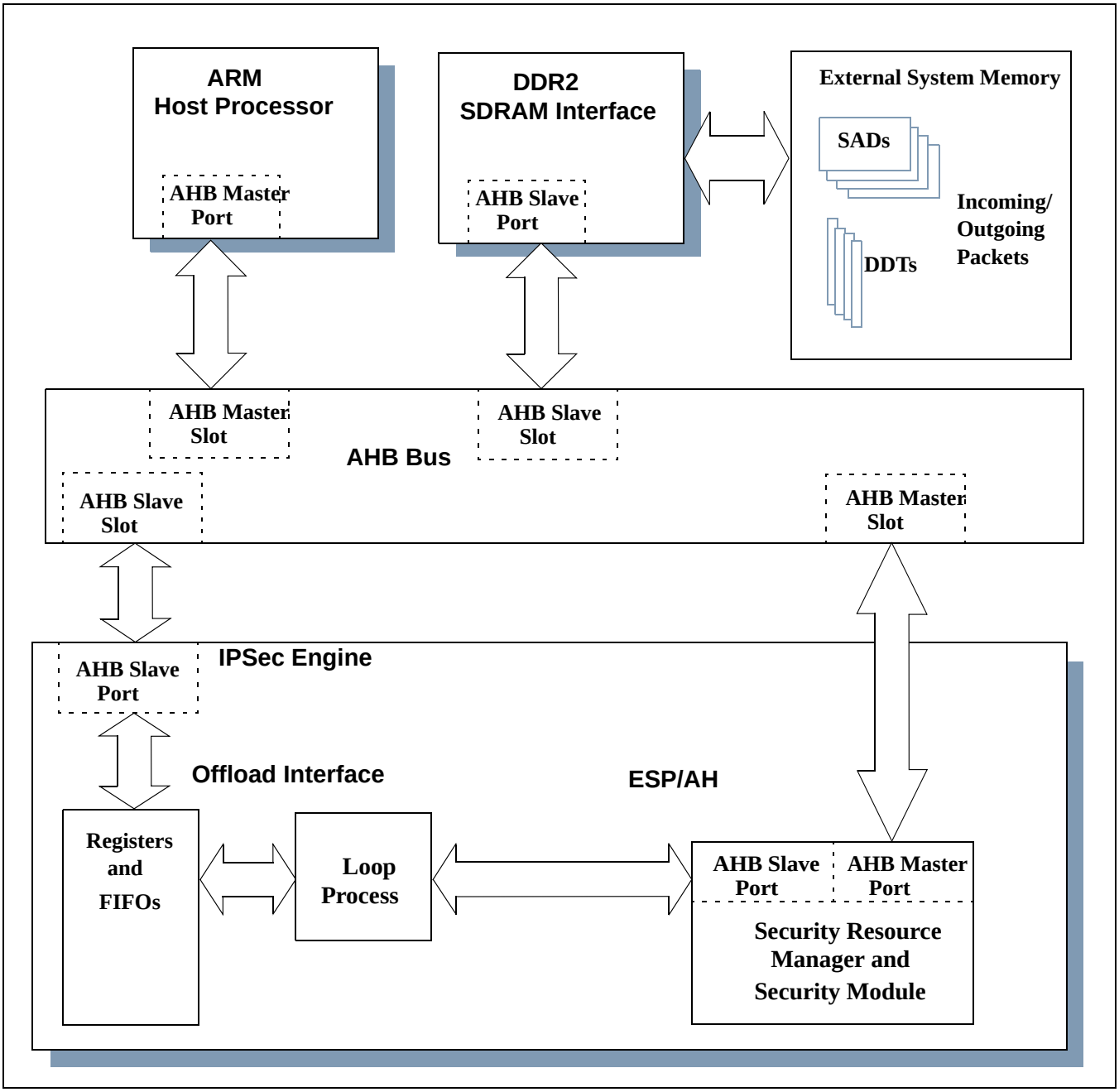
- RC4 and AES support
- Support for simultaneous sessions
- Concurrent support for SSL sessions and IPSec tunnels
- Hardware-based random-number generation
- Hardware-based public key acceleration engine

9.3 Functional Description

As shown in Figure 9-1, the IPSec Engine consists of an Offload Interface and an (ESP/AH) core. The IPSec Engine bridges the gap between raw cryptographic offload and complete IPSec offload. By combining hash and ciphering engines, a special purpose DMA engine, and ESP/AH packet processing logic, the IPSec engine allows of the all mathematically intensive IPSec operations to be off-loaded from the host ARM processor. With the Offload Interface, host processor responsibilities are minimized.

Beside the ESP/AH core for IPSec processing, the LS10xMA provides several hardware-based engines to accelerate the SSL/TCL and other security standards. In addition to AES, DES, and hash cores, the device includes a RC4 engine to support the SSL standard, a PKA engine and a hardware-based true random generator.

Figure 9-1 IPSec Engine Operations Overview



The IPSec Engine works to en-crypt and de-crypt, as well as perform authentication processing for streams of packets moving both in and out of the LS10xMA device simultaneously. The following standard algorithms are supported:

- AH-SHA-1
- AH MD5
- ESP DES/NULL
- ESP DES/SHA-1

- ESP 3DES/MD5
- ESP 3DES/NULL
- ESP 3DES/SHA-1
- ESP 3DES/MD5
- ESP AES128/NULL
- ESP AES128/SHA-1
- ESP 3DES/NULL
- ESP 3DES/SHA-1
- ESP 3DES/MD5
- ESP AES128/NULL
- ESP AES128/SHA-1
- ESP AES256/MD5

10 TDM Bus Interface

This section provides details on the LS10xMA TDM bus interface.

10.1 Overview

The TDM Bus interface provides a full-duplex, serial TDM bus for digital data transfer between the Network Interface Device such as T1/E1 Framer, Time Slot Interchanger, SLIC, and the LS10xMA. This section presents TDM Bus features, functional description, and interface timing.

NOTE: TDM is not supported on LS101MAXE7DFA device.

10.2 Features

- Full-duplex, serial time division multiplexed bus for digital transfer with network interface device.For example, T1/E1 framer, time slot interchanger and so on.
- A single TDM bus providing a maximum of 24/32/64/128 time slots (running at 1.536 or 1.544 MHz, 2.048 MHz, 4.096 MHz, or 8.192 MHz, accordingly), or any configured number when running off another clock.
- Clock rate can be x1 or x2 data rate (going up 16.384 MHz for x2). In x2 mode, data is sampled every two clock cycles.
- Provides a programmable timing parameters interface.
- Data can optionally not be driven during inactive timeslot.
- Internal loopback of TDM DX to TDM DR (uses TDM_CK and TDM_FS).
- External loopback of TDM DR to TDM DX (see Section 10.7 TDM Loopback Timing).
- TDM_CK and TDM_FS may be sourced by the LS10xMA device.

When sourced externally, TDM_CK and TDM_FS must be stable and within specification before reset signal to the TDM block ends.

10.3 Signal Description

The TDM timing signals: Clock and Frame Sync, may be sourced by the LS10xMA device or received from the network. The source of Clock is determined by the logic level of the TM_TDM_CLKSRC_EN input when it is latched at device reset. The source of Frame Sync is determined by a register in the GPIO block.Table 10-1lists the TDM signals.

Table 10-1 TDM Signals

Signal Name	Direction	Description
TDM_CK	B	Data clock input from network, or from device if it is the source of the clock.
TDM_FS	B	Frame synchronization input from the network or from device.
TDM_DX	O	Data transmit output from the device.
TDM_DR	I	Data receive input from the network.

NOTE: When the LS10xMA is configured to output TDM_CK, this signal will stop toggling when the system puts the device into hard reset by asserting RESET_N.

10.4 Functional Description

The TDM bus supports speeds of 1.536 MHz to 8.192 MHz, allowing between 24 and 128 timeslots. The device processing throughput is not designed to uphold processing of this many active channels. Dual rate clock is also supported (up to 16.384 MHz), where data is held for 2 clock cycles. Support is also included for 1.544 MHz clock (193 bit frame) through configuration allowing ignore of the first bit in a frame.

The TDM interface can be set to support both internal and external loopback testing through a GPIO register setting. The parameters of the TDM bus are configurable. The FS, DX, and DR can be sampled on falling or rising edges controlled by parameters FE, XCE and RCE respectively. Data transmission can begin on any clock cycle relative to cycle 0 by programming the appropriate cycle value in the CYCX and CYCR parameters; and the interface can operate in single-clock or double-clock mode where one data bit is transmitted or received every TDM_CK period or every two TDM_CK periods.

The TDM DMA (TDMA) controller manages the flow of data between the TDM bus interface and external system memory through the AMBA AHB Bus and the DDR SDRAM controller. Both the TDM bus interface and the TDMA registers are configured through the APB.

10.5 TDM Bus Interface Timing

Figure 10-1 to Figure 10-8 show the TDM Bus timing and how clock edges are counted in cycles. Cycles are defined as always beginning (with cycle 0) on the rising TDM_CK edge relative to the clock edge on which TDM_FS is sampled. Data transmission can begin on any clock cycle relative to cycle 0 by programming the appropriate cycle value in the CYCX parameter in the Transmitter Operating Configuration Register and CYCR parameter in the Receiver Operating Configuration Register. Each TDM Bus interface can operate in single-clock or double-clock mode. One data bit is transmitted or received every TDM_CK period or every two TDM_CK periods as programmed by the CMSX bit in the Transmitter Operating Configuration Register and CMSR bit in the Receiver Operating Configuration Register.

Figure 10-1 Transmit Timing (FE = 1, CMSX = 0)

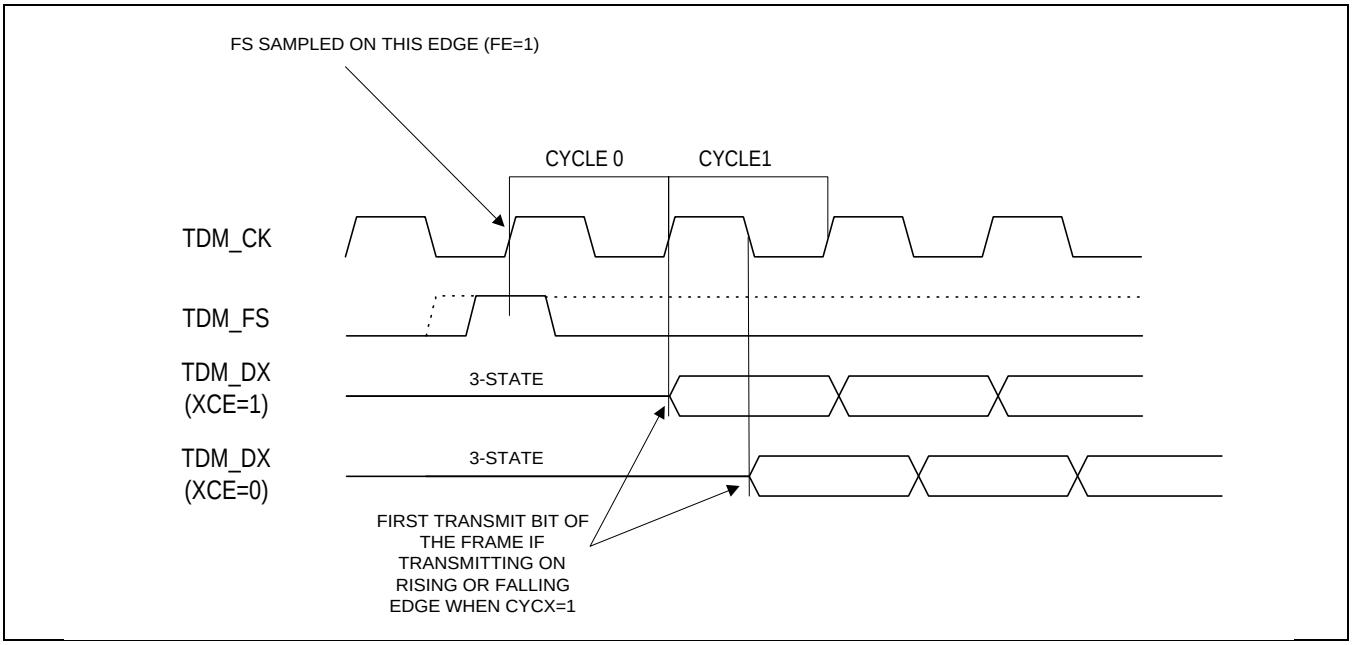


Figure 10-2 Transmit Timing (FE = 0, CMSX = 0)

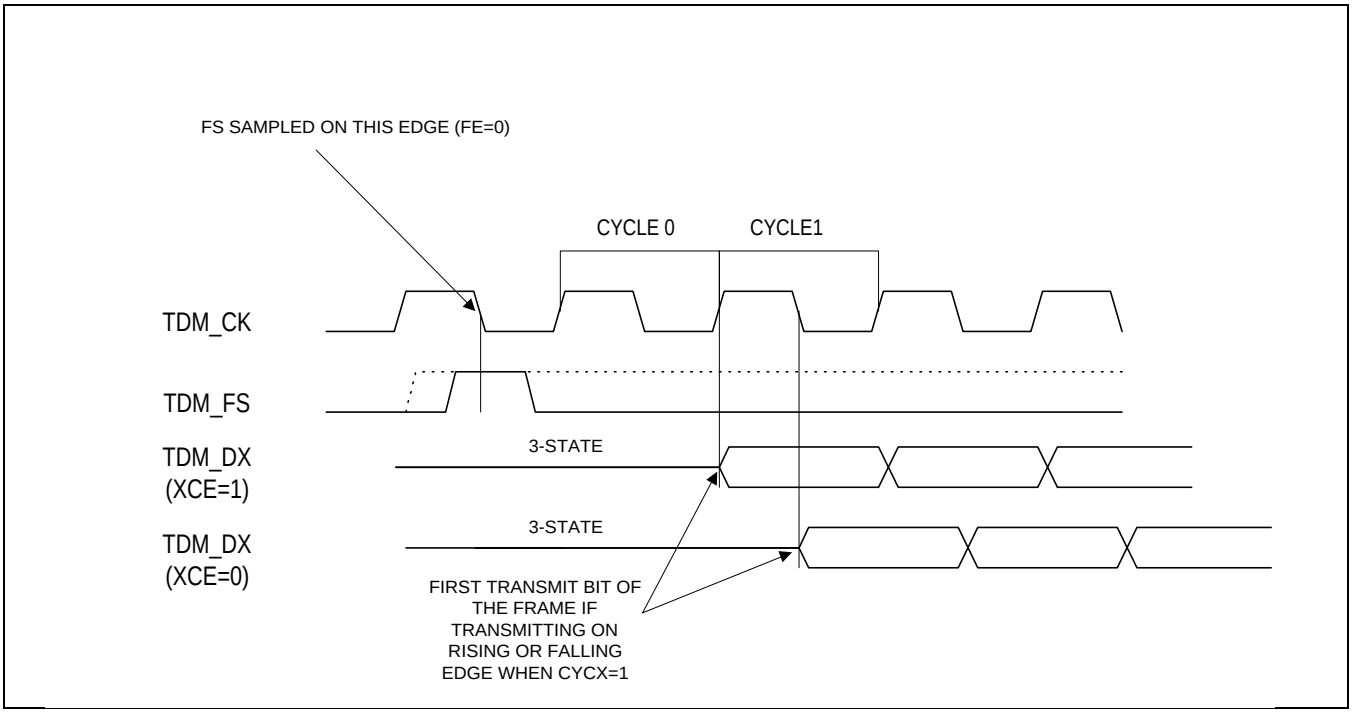


Figure 10-3 Received Timing (FE = 1, CMSR = 0)

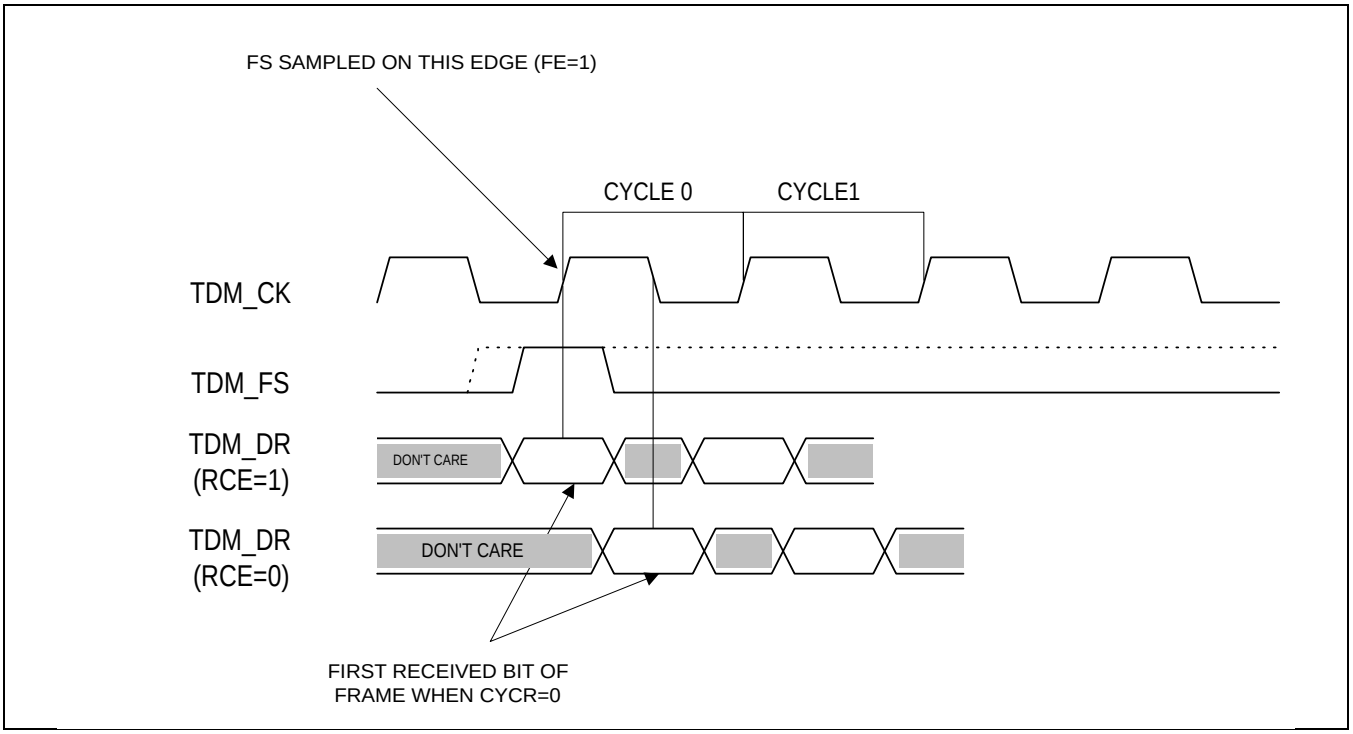


Figure 10-4 Received Timing (FE = 0, CMSR = 0)

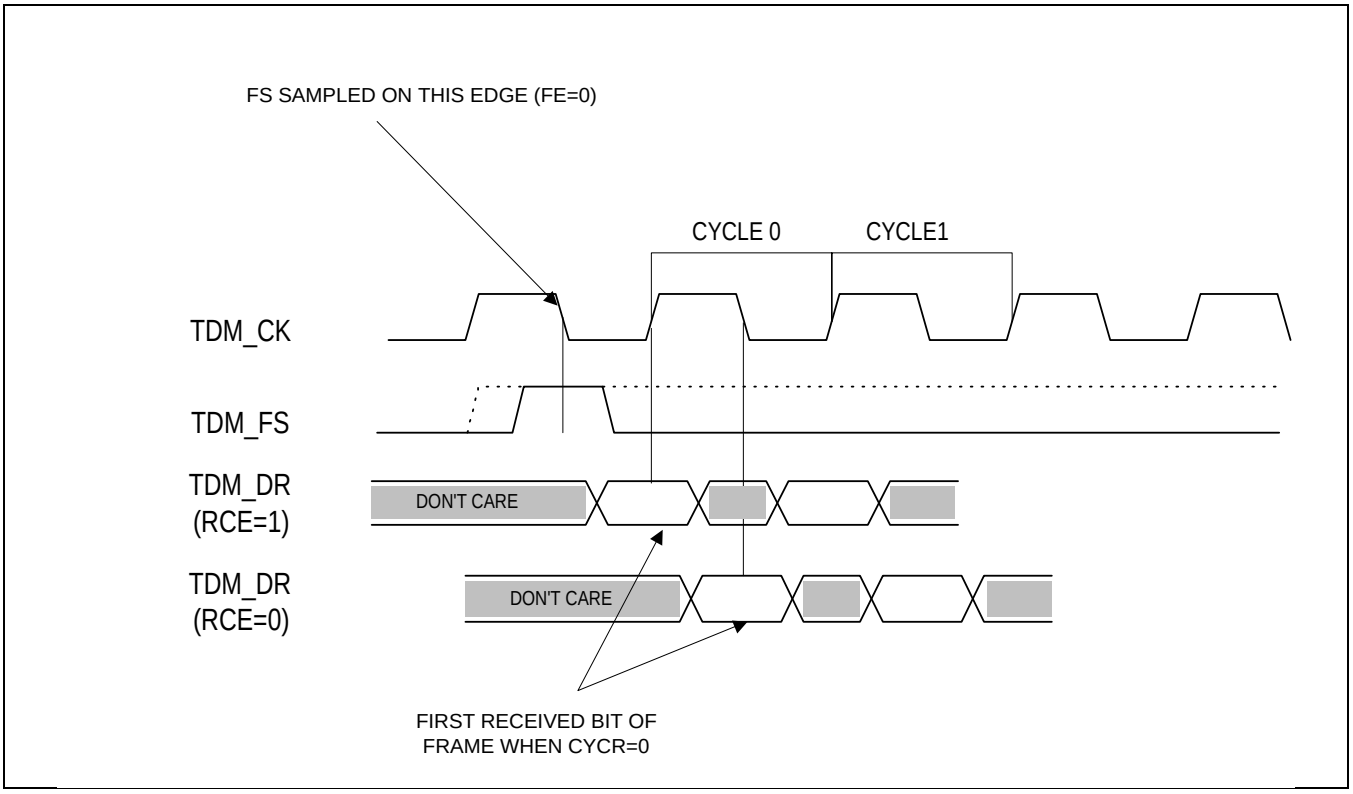


Figure 10-5 Transmit Timing (FE = 1, CMSX = 1)

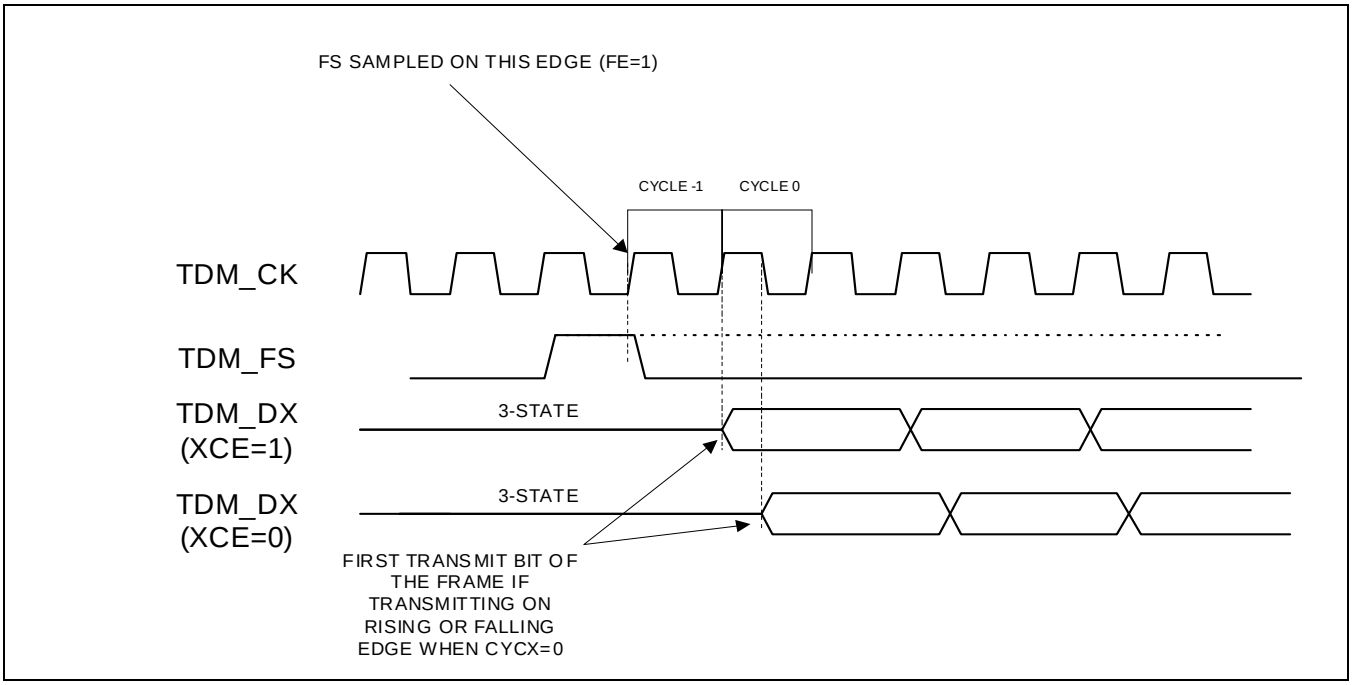


Figure 10-6 Transmit Timing (FE = 0, CMSX = 1)

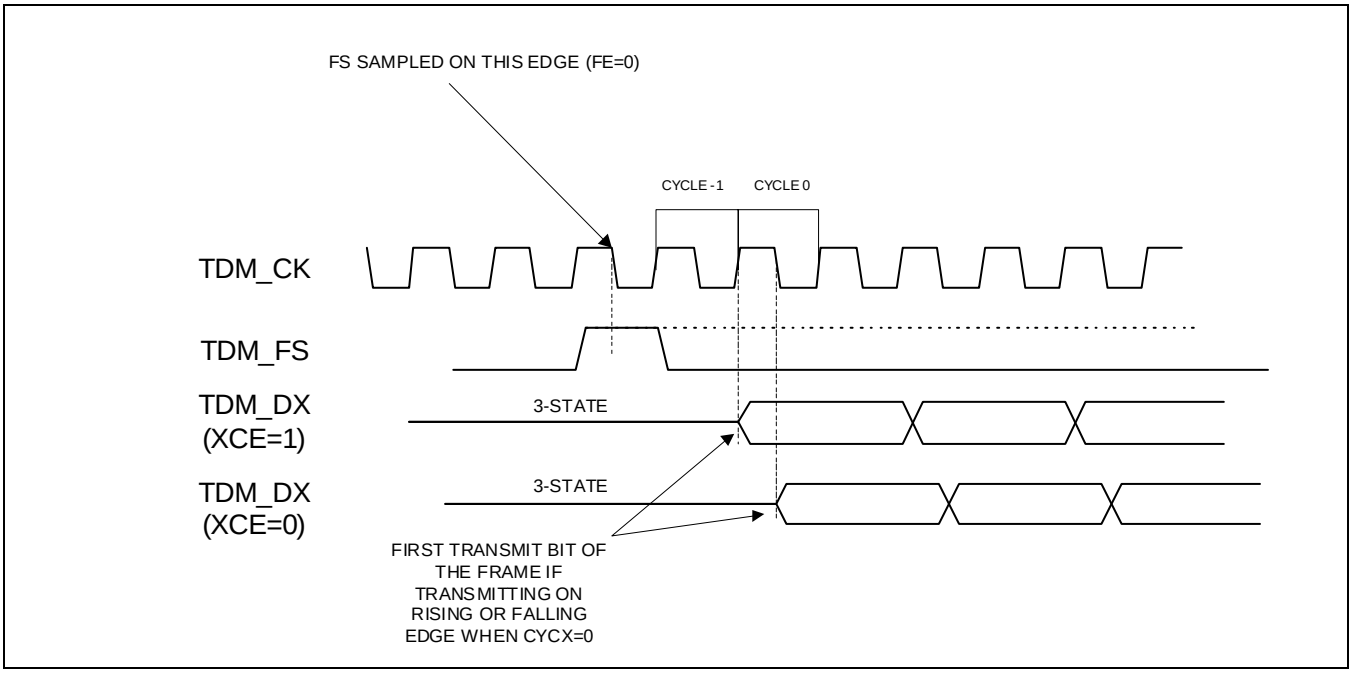


Figure 10-7 Received Timing (FE = 1, CMSR = 1)

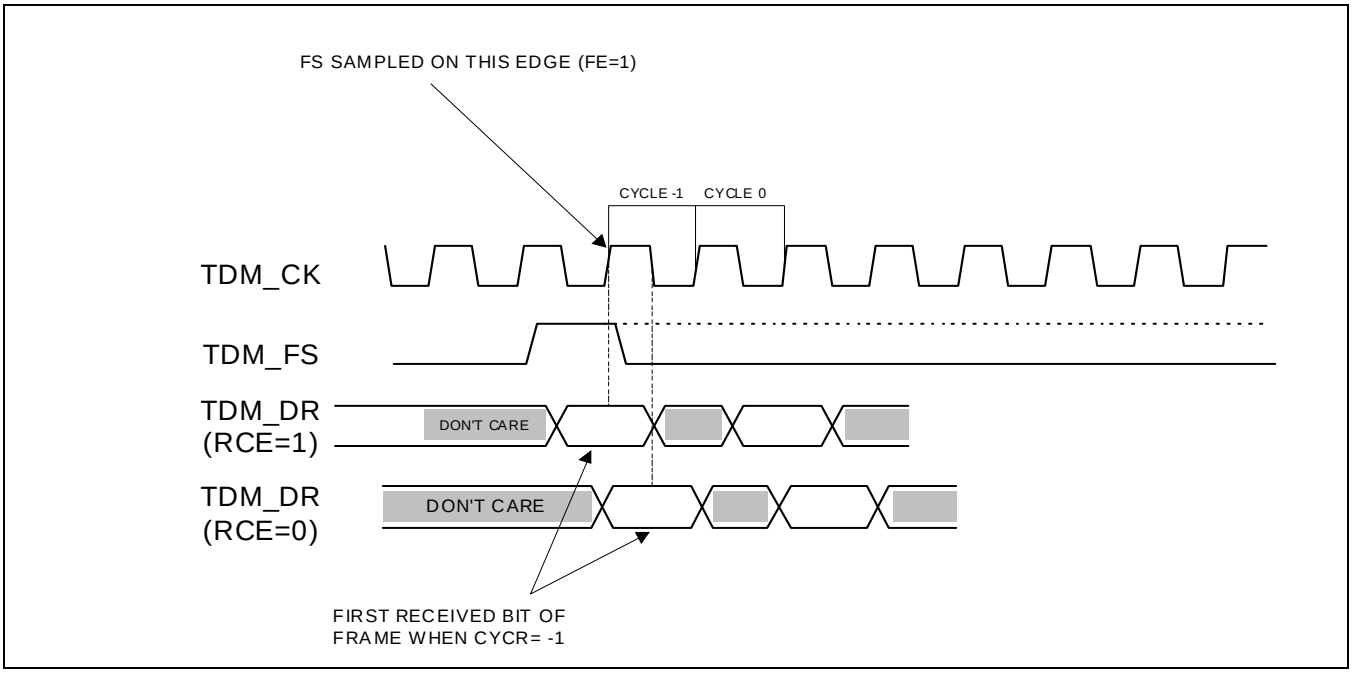
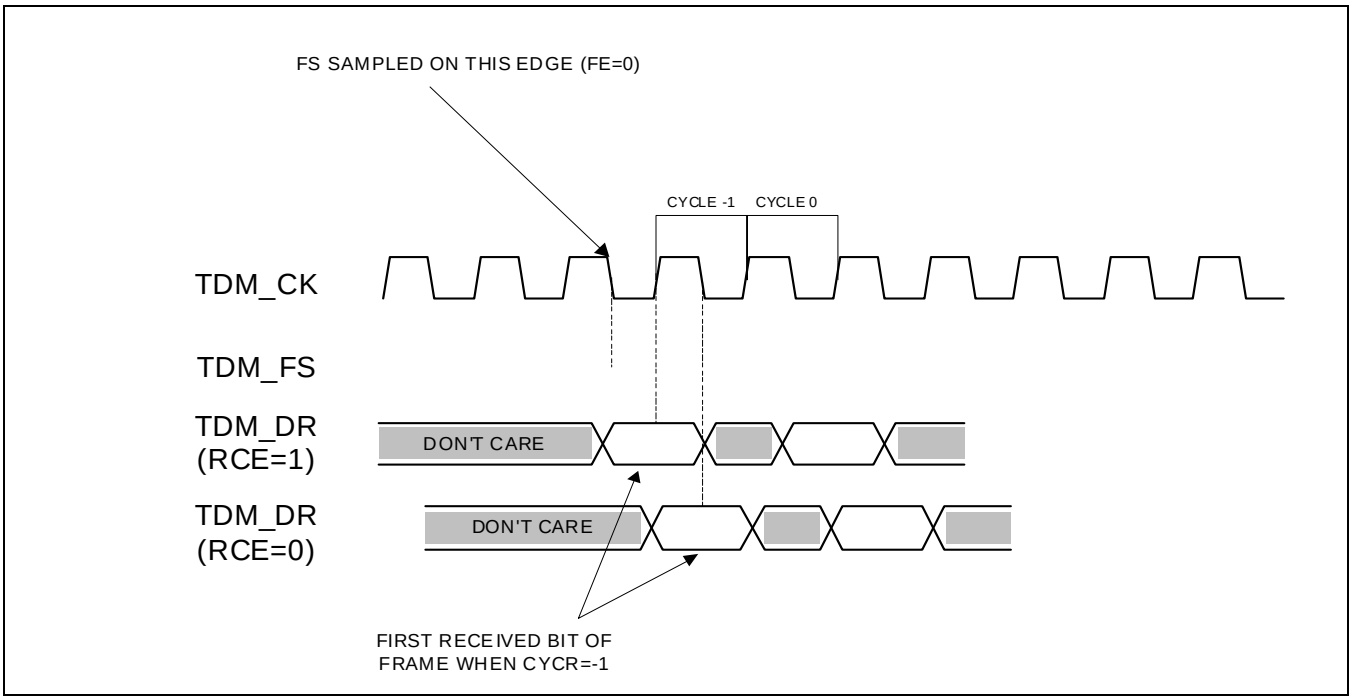


Figure 10-8 Received Timing (FE = 0, CMSR = 1)



10.6 TDM Interface Timing

The TDM Interface timing parameters are programmable and can be configured for H.100, H110, and H-MVIP bus standards using the SUPVSR_SET_TDM_PARAMS message. Figure 10-9 illustrates the TDM bus timing.

Figure 10-9 TDM Bus Timing

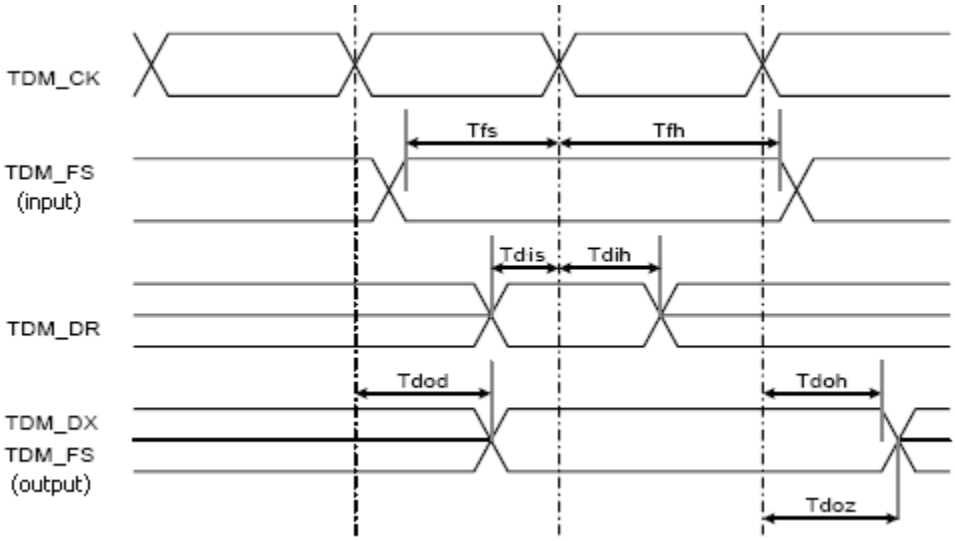


Table 10-2 describes the TDM bus timing details.

Table 10-2 TDM Bus Timing

Symbol	Parameter	Min.	Max.	Units	Notes
	Clock edge rate (All Clocks)	0.25	2	V/ns	1
Tcp	CK Clock Period 1:1 clock to data rate ratio 2:1 clock to data rate ratio	 122 61		 ns	
Tch	CK Clock Duty Cycle	40	60	%	
Tfs	Frame Synch Input Setup Time	20		ns	4, 5, 6
Tfh	Frame Synch Input Hold Time	20		ns	4, 5, 6
Tdis	DR Input Setup Time	25		ns	3, 4
Tdih	DR Input Hold Time	4		ns	2, 4
Tdod	DX/FS Output Delay Time	—	20	ns	2,4,6
Tdoh	DX/FS Output Hold Time	0		ns	2,4,6
Tdoz	DX Output Hi Z Time	2	20	ns	2,4,6

Table 10-2 TDM Bus Timing (continued)

Symbol	Parameter	Min.	Max.	Units	Notes
NOTES: 1. The rise and fall times are determined by the edge rate in V/ns. A “Max” edge rate is the fastest rate at which a clock transitions. 2. Measured at the transmitter. 3. Measured at the receiver. 4. All inputs can be programmed to be sampled at either rising or falling edge of the clock. All outputs can be independently programmed to be generated at either rising or falling edge of the clock 5. The Frame Synch can be programmed as either active low or active high. 6. TDM_FS may be either input or output. If input, Tfs and Tth apply. If output, Tdod and Tdoh apply.					

10.7 TDM Loopback Timing

This is an external, remote loopback. When it is enabled. TDM_DR is looped back to TDM_DX without any sampling. Figure 10-10 illustrates the TDM lookpback timing.

Figure 10-10 TDM Loopback Timing

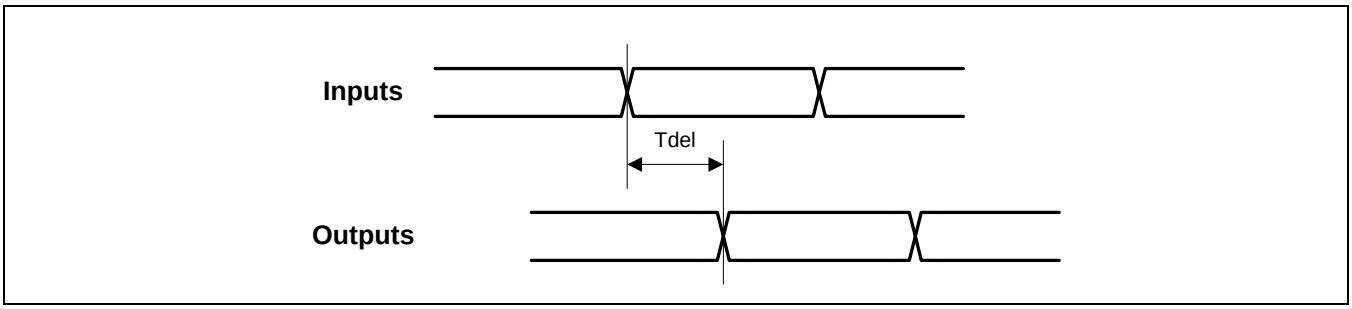


Table 10-3 TDM Loopback Setup and Hold Times

Symbol	Parameter	Min.	Typ.	Max.	Units	Notes
Tdel	Input to output delay	2	—	20	ns	

11 Universal Asynchronous Receiver Transmitter

This section provides details on the LS10xMA UART interface.

11.1 Overview

The LS10xMAx devices provide two UART interfaces, with FIFOs. The signals for the second UART are multiplexed with GPIOs.

The UART module contains two programmable blocks that are modeled after the industry-standard 16550. Only the Rx and Tx signals are provided which can be connected to an RS232 driver for standard serial port operation. The UART interfaces provide a useful console port and can also be used as a software debug port.

NOTE: This section does not provide register-level details to support software driver development.

11.2 Features

- Two UARTs modeled after the industry standard 16550 architecture.
- Asynchronous mode.
- The receiver is normally programmed to automatically determine the baud rate of the transmitter. The receiver relocates the center of the start bit of each new word or byte. This feature enables the asynchronous mode to operate without sharing the actual baud rate clock.
- Transmitter adds start, stop, parity bits to the serial data.
- Receiver rejects false start bits.
- 5, 6, 7, or 8 bit characters.
- Even, odd, or no parity bit generation or detection.
- 1 or 2 stop bit generation, programmable short stop bit control.
- Transmitter buffer empty flag or under-run status generation.
- Receiver buffer full detection.
- Receiver status error flags: over-run, parity, frame, and break interrupt.
- FIFO depth of16 bytes.
- Internal loopback from TX to RX.

NOTE: The second UART interface signals are multiplexed with GPIOs:

- UART1_RX (muxed with GPIO22)
- UART1_TX (muxed with GPIO23)

11.3 Functional Description

The LS10xMA UART serial receives and transmits data to a peripheral. The transmitter performs the parallel to serial conversion adding start, stop, and parity bits to the serial data. The receiver uses the stop/start transition to approximate the center of the data, parity, and stop bits that follow.

In the asynchronous mode**,** the transmitter and receiver are set to operate at a given baud rate from a table of standard baud rates. Other than the baud information, no clock is passed between them. The receiver is normally programmed to automatically determine the baud rate of the transmitter. Absolute baud rate accuracy is not required because the receiver relocates the center of the start bit of each new word or byte. It is this feature that enables the asynchronous mode to operate without sharing the actual baud rate clock. The receiver rejects false start bits.

The asynchronous mode is programmable as follows:

- 5-, 6-, 7-, or 8-bit characters
- Even, odd, or no parity bit generation and detection
- 1 or 2 stop bit generation, programmable short stop bit control
- Transmitter buffer empty flag and under-run status generation
- Receiver buffer full detection
- Receiver status error flags: over-run, parity, frame and break interrupt
- Interrupt enable control bits

On the Rx direction, there is a read-only register (called RBR) containing the data byte (data character) received on the Serial Input Port (SIN). there are also FIFOs on the receive side and the FIFOs can be enabled or disabled by software. When FIFOs are disabled, the data in RBR must be read before the next data arrives, otherwise it will be overwritten, resulting in an overrun error. When FIFOs are enabled, RBR accesses the head of the receiver FIFO. If the receive FIFO is full and RBR is not read before the next data character arrives, then the data already in the FIFO will be preserved but any incoming data will be lost. An overrun error will also occur.

On the TX direction, there is a transmit holding register (called THR) that contains the data to be transmitted on the serial output port (SOUT). There are also FIFOs on the transmit side and FIFOs can be enabled or disabled by the software. When FIFOs are disabled, a single character will be written to THR when THR is empty. Any additional writes to the THR before it is empty again cause the previous data in THR to be overwritten. When FIFOs are enabled, data characters can be written to the transmit FIFO when it is not full. Any attempt to write data when the FIFO is full results in loosing the written data.

11.4 Signal Description

Table 11-1lists the UART interface signals.

Table 11-1 LS10xMA UART Interface Signals

Signal Name	Dir.	Size	Signal Description
UART[1:0]_RX	I	2	UART serial input.
UART[1:0]_TX	O	2	UART serial output

12 Inter-IC Interface

This family of devices include a standard I^2C to load boot code and to provide system management communications with other devices.

12.1 Features

- Conforms to version 2.1 of the I^2C specification.
- Master or slave operation.
- Multi-master systems supported.
- Supports both 7-bit and 10-bit addressing.
- Own address and general call (data broadcast) address detection.
- Built-in collision detection.
- Performs arbitration and clock arbitration.
- Interrupt on address detection.
- Supports High Speed (HS), Fast and Standard transfer rates.
- Allows operations from a wide range on input clock frequencies.

NOTE: The I^2C interface signals are multiplexed with GPIOs:

- I^2C SCL (muxed with GPIO18)
- I^2C_SDA (muxed with GPIO19)

12.2 Functional Description

I^2C is a low-bandwidth, short distance protocol for on-board communications. All devices are connected through two wires: serial data (SDA) and serial clock (SCL). Available Freescale drivers allow LS10xMA device to boot from a serial EEPROM through the I^2C interface. All I^2C devices must have a unique address to identify it on the bus. Slave devices have a predefined address, but the lower bits of the address can be assigned to allow for multiples of the same devices on the bus.

The LS10xMA device operates as a master, a slave, or in multi-master mode. It supports all I^2C speeds: standard (100 kbps), fast (400 kbps) and high speed (3.4 Mbps). The I^2C interface includes a programmable clock divider to allow adjustment of data speed over a wide range. However, the I^2C interface does not include an output signal for direct control of a bridge between fast/standard and high-speed bus segments. Thus, the I^2C interface is typically configured to accommodate the slowest device on the bus.

12.3 Signal Descriptions

Table 12-1lists the I^2C interface signals.

Table 12-1 LS10xMA I^2C Interface Signals

Signal Name	Dir	Size	Signal Description
I2C_SCL	B	1	I^2C Input/Output Clock Line
I2C_SDA	B	1	I^2C Input/Output Data Line

12.4 I^2C Timing

Figure 12-1illustrates the I^2C timing waveform. Table 12-2 lists the 12c timing parameters.

Figure 12-1 I^2C Timing Waveforms

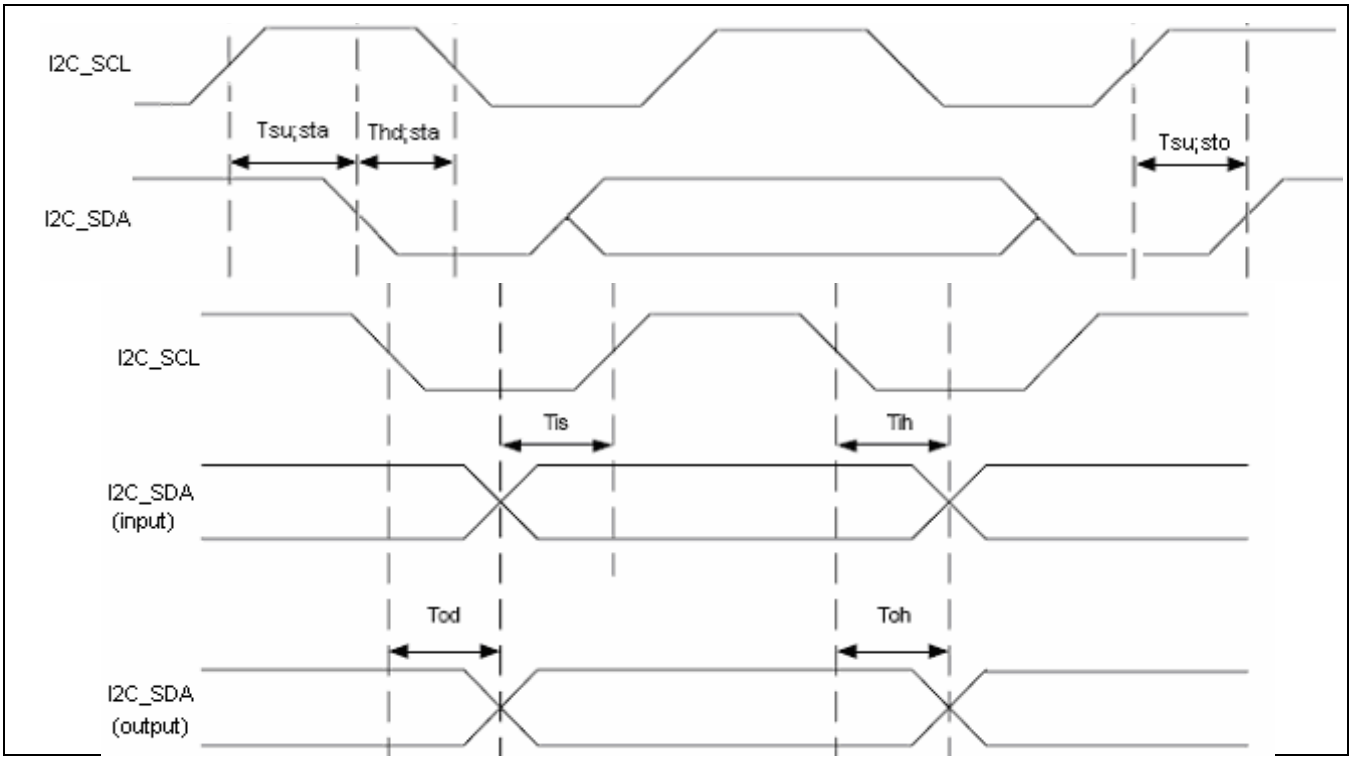


Table 12-2 I^2C Timing Parameter

Symbol	Parameter	Min.	Typ.	Max.	Units	Notes
	Clock Frequency			3.4	MHz	
	Clock Duty Cycle Low period High period	 160 60			 ns ns	
Tsu;sta	(Repeated) START condition setup time	160			ns	1, 2
Ths;sta	(Repeated) START condition hold time	160			ns	1, 2
Tsu;sta	STOP condition setup time	160			ns	1, 2
Tis	Input Setup Time (I^2C_input)	10			ns	

Table 12-2 I^2C Timing Parameter

Symbol	Parameter	Min.	Typ.	Max.	Units	Notes
Tih	Input Hold Time (I^2C_input)	0			ns	
Tod	Output Delay Time (I^2C_output)			60	ns	3, 4
Toh	Output Hold Time (I^2C_output)	0			ns	

NOTES:
1. These numbers can be extended when using a slower output clock (master mode).
2. This number increases as clock frequency is slower, according to the following formula:
   $Tsu;sta(min) = Thd;sta(min) = Tsu;sto(min) = 2^N * T_{AHB}$
   where T_{AHB} is the AHB clock period, and N is a parameter that is used to set the I^2C clock frequency according to the AHB clock frequency
3. Output delay time is not defined in standard. LS10xMA value should guarantee meeting the timing at the receiver.
4. This number increases as clock frequency is slower, according to the following formula:
   $Tod(max) = 2^{N+1} * (M+1) * T_{AHB}$
   where T_{AHB} is the AHB clock period, and N and M are parameters that are used to set the I^2C clock frequency according to the AHB clock frequency

12.5 High Speed Mode

A Master may arbitrate and select High speed operation by transmitting one of the reserved 7-bit addresses of the form 00001*xxx*b. The first five bits of this address are significant: the remaining three bits may be used to identify different Hs-mode masters on the same I^2C bus.

On detecting a code of this form, the I^2C interface enters High speed mode. It will revert to F/S mode on detection of a STOP condition on the I^2C bus.

12.6 Clock Synchronization

If another device on the I^2C bus drives the clock line when the I^2C interface is in master mode, the I^2C interface will synchronize its clock to the I^2C bus clock. The high period of the clock will be determined by the device that generates the shortest high clock period. The low period of the clock will be determined by the device that generates the longest low clock period.

When the I^2C interface is in master mode and is communicating with a slow slave, the slave can stretch each bit period by holding the SCL line low until it is ready for the next bit. When the I^2C interface is in slave mode, it will hold the SCL line low after each byte has been transferred until the IFLG has been cleared in the CNTL register.

12.7 Bus Arbitration

In master mode, the I^2C interface will check that each transmitted logic 1 appears on the I^2C bus as a logic 1. If another device on the bus over-rules and pulls the SDA line low, arbitration is lost. If arbitration is lost during the transmission of a data byte or a Not ACK bit, the I^2C interface will return to the idle state. If arbitration is lost during the transmission of an address, the I^2C interface will switch to slave mode so that it can recognize its own slave address or the general call address.

13 USB Interface

This section provides details on the LS10xMA USB interface. This section does not include register-level descriptions for host or device modes. For a view of the USB interface within the top-level block diagram, see Figure 2-1.

13.1 Overview

The Universal Serial Bus (USB) interface complies with the USB 2.0 standard. The LS10xMA USB controllers handle control flow, data flow, status collection, and power management. The USB PHY is integrated into the LS10xMA device.

13.2 USB Features

- USB 2.0 compliance—Support for both host and device modes
- Support up to four endpoints in device mode.
- Line interface: Differential D+/D-
- Integrated termination
- Data and clock recovery
- Bit stuffing/unstuffing
- Sync detection
- NRZI encoding/decoding
- Support for USB 2.0 LS, FS, and HS modes

USB interface controller implements an EHCI-aware link-list DMA engine to reduce software intervention

13.3 Signal Description

Table 13-1lists the USB interface signal description.

Table 13-1 USB Interface Signals

Signal Name	Dir	Size	Signal Description
USB_DP	B	1	Positive channel connected to the serial USB cable. 3.3V analog signal.
USB_DM	B	1	Negative channel connected to the serial USB cable. 3.3V analog signal.
USB_VBUS	—	1	5V, 500 mA power supply. Required by the USB specification. **NOTE:** Not connected to the LS10xMA device, but necessary to support external interface.
USB_GND	—	1	Ground for USB. NOTE: Not connected to the LS10xMA device, but necessary to support external interface

Table 13-1 USB Interface Signals (continued)

Signal Name	Dir	Size	Signal Description
USB_REF	B	1	External resistor connection for current reference.
USB_VBUS_STAT	I	1	VBUS Status is an indicator of the internal USB VBUS. When the LS10xMA USB controller is in host mode this signal goes low to indicate a power fault condition (such as over-current) on the sourced VBUS. When the LS10xMA USB controller is in device mode this signal goes low to indicate a device mode error. **NOTE:** This I/O is +5 VDC tolerant.

13.4 USB Design and Layout Requirements

USB 2.0 requires careful layout requirements. Successful board-level implementation must respond to two challenges:

- Capacitance on data lines must be kept to a minimum
  - Capacitance from each data line to ground with the other data line disconnected must not exceed 10 pF.
  - Matching of capacitances to ground must be within 1 pF.
  - The differential capacitance across transceiver inputs must be no more than 5.0 pF.
- The impedance of data lines with respect to Ground (GND) must vary with Low Speed, Full Speed, and High Speed Modes

13.5 USB Interface Timing

Figure 13-1illustrates the USB interface timing waveform. lists the USB interface timing Parameters.

Figure 13-1 USB External Bypass Timing Waveform

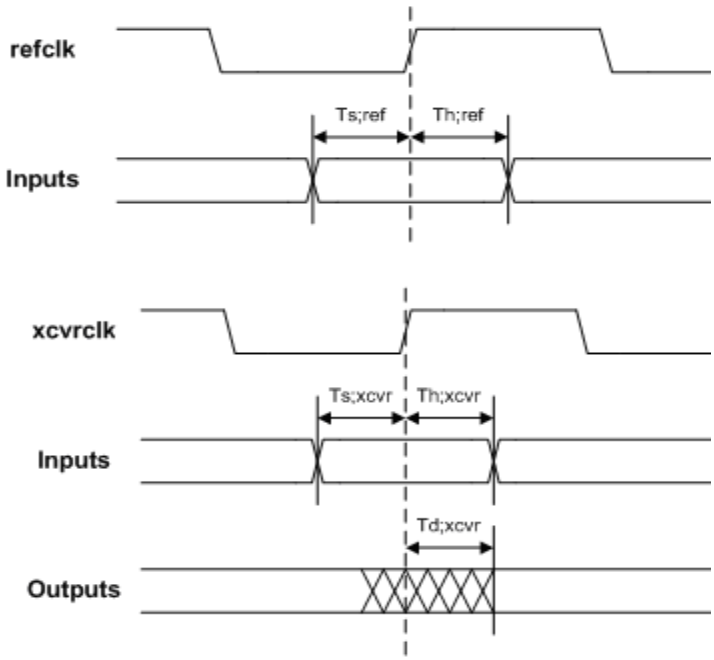


Table 13-2 USB Timing Parameters

Symbol	Parameter	Min.	Typ.	Max.	Units	Notes
	refclk (input) frequency		24		MHz	
	refclk clock duty cycle		50		%	
	xcvrclk (output) frequency		30		MHz	
	xcvrclk duty cycle		50		%	
Ts;ref	Inputs setup time versus refclk	6			ns	1,4
Th;ref	Inputs hold time versus refclk	2			ns	1,4
Ts;xcvr	Inputs setup time versus xcvrclk	16			ns	2,4
Th;xcvr	Inputs hold time versus xcvrclk	-4			ns	2,4
Td;xcvr	Outputs delay time versus xcvrclk	-4		8	ns	3,4

1. These input signals clocked by reference clock include: VLOAD, ONBIST, VCONTROL[3:0]
2. These input signals clocked by UTMI clock include: TERMSEL, DATAIN, TXVALID, TXVALIDH.
3. These output signals clocked by UTMI clock include: DATAOUT, RXVALID, RXVALIDH, RXERROR, RXACTIVE, HOSTDISCONNECT, TXREADY, LINESTATE.
4. The following signals are considered asynchronous and have no timing requirements. Inputs: UTMI_RESET, SUSPEND, DATABUS16_8, DPPULLDOWN, DMPULLDOWN, XCVRSELECT, OPMODE. Outputs: VSTATUS.

14 General Purpose Input Output

This section provides details on the LS10xMA GPIO interface.

14.1 Overview

The devices support up to 31 GPIO interface signals that are utilized by both the MSP and ACP. Of those 31, only 7 can be programmed to create interrupts with rising or falling or both edges of input signals. The interface signals are also level sensitive and can generate interrupt as long as the input is high. Input signals are synchronized for the generation of the interrupts.

The GPIO block holds several configurations for other blocks. These include the following:

- APB Bus access Wait state register: List of the APB bus slaves that the added wait state is applied to.
- System configuration status register: Include all input signals that affect the configuration of the Chip.
- TDM Mux control register: TDM configuration setting, such as loopback, internal sampling and so on.
- ARM ID Register - Indicates which ARM is the master of the current AHB data phase.
- GPIO muxing
- Bootstrap Software override

NOTE: Freescale firmware utilities and example software drivers use GPIOs.

To make optimal use of the GPIOs, review planned use of Freescale utilities and drivers with your Freescale representative.

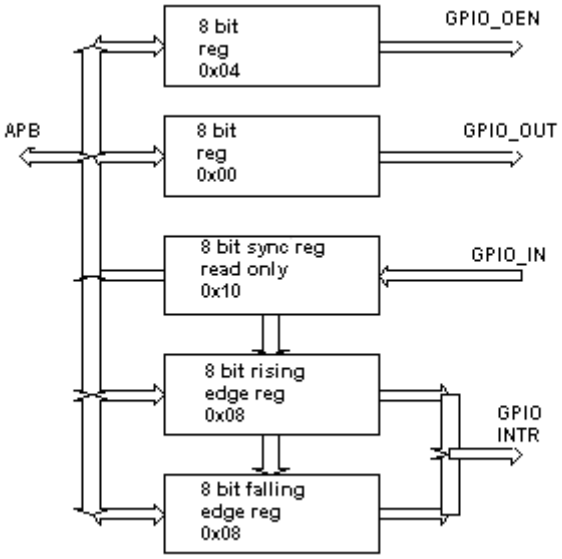
14.2 GPIO Features

- Up to 27 general purpose I/O interface signals for specific control, monitoring signaling purpose.
- Many GPIO pins are muxed with other functional pins and can only be used in case the other functionality is not being used. Muxing selection is per signal.
- GPIO[7:5] and GPIO[3:0] - 7 bits configured to generate level or edge (rising/falling) interrupts to the ARM by setting the corresponding GPIO register bits.
- 8 GPIO pins are capable of driving 16mA of current, which is sufficient for direct LED control.

14.3 GPIO Interface Signals

The GPIO[7:5] and GPIO[3:0] input signals can offer programmable interrupts on rising or falling edges of input signals. Figure 14-1 shows the block diagram of GPIO[7:5] and GPIO[3:0].

Figure 14-1 GPIO Block Diagram



On LS10xMA devices, all are bi-directional and provide slew control on outputs and hysteresis (Schmitt triggering) on inputs. GPIO[3:0] and GPIO[7:5] trigger interrupts, can handle 16mA strength, and they are pulled up. GPIO [23:22] signals can handle 16mA strength and the rest of the GPIOs signals handle 8mA strength.

Some of GPIO[31:8] are pulled down internally are some are pulled up. Table 14-1lists the GPIO signal descriptions.

NOTE: GPIO[10:8], GPIO[11], and GPIO[14] configured as inputs have internal pull-down. Other devices on the board that are connected to these signals should not drive the signals during boot-up and hence it is suggested to implement an external pull-down to ensure zero state during boot-up.

Table 14-1 GPIO Interface Signals

Signal Name	Dir	Size	Signal Description
GPIO[7:5]	B	3	General purpose 05 through 07 I/Os. These pins can generate interrupts.
GPIO[3:0]	B	4	General purpose 00 through 03 I/Os. These pins can generate interrupts.
GPIO08 – GPIO26	B	19	General purpose I/Os 08 through 26. These pins cannot generate interrupts. Note that at the block level a total of 31 GPIO pins are supported, but not all these pins are output on the die.

Table 14-2 describes the GPIO multiplexing details

Table 14-2 General Purpose Signal Multiplexing

Signal Name	Multiplexing
GPIO_0	**General Purpose I/O 0.** Not shared or multiplexed with other signals.
GPIO_1	**General Purpose I/O 1.** Not shared or multiplexed with other signals.

Table 14-2 General Purpose Signal Multiplexing (continued)

Signal Name	Multiplexing
GPIO_2	**General Purpose I/O 2.** Not shared or multiplexed with other signals.
GPIO_3	**General Purpose I/O 3.** Not shared or multiplexed with other signals.
GPIO_4	**General Purpose I/O 4.** Not supported.
GPIO_5	**General Purpose I/O 5.** Multiplexed with EXP_CS3_N.
GPIO_6	**General Purpose I/O 6.** Shared with EXP_NAND_RDY/BSY# or EXP_A21. Not a real mux. Specific functionality determined by software driver.
GPIO_7	**General Purpose I/O 7.** Shared with EXP_IRQ. Not a real mux. Specific functionality determined by software driver.
GPIO_8	**General Purpose I/O 8.** Multiplexed with EXP_A15.
GPIO_9	**General Purpose I/O 9.** Multiplexed with EXP_A16.
GPIO_10	**General Purpose I/O 10.** Multiplexed with EXP_A17.
GPIO_11	**General Purpose I/O 11.** Multiplexed with EXP_A13.
GPIO_12	**General Purpose I/O 12.** Multiplexed with SPI_RXD.
GPIO_13	**General Purpose I/O 13.** Multiplexed with SPI_SS0_N.
GPIO_14	**General Purpose I/O 14.** Multiplexed with EXP_A14.
GPIO_15	**General Purpose I/O 15.** Multiplexed with TIM_EVNT0.
GPIO_16	**General Purpose I/O 16.** Multiplexed with TIM_EVNT1.
GPIO_17	**General Purpose I/O 17.** Shared with TM_EXT_RESET. Not a real mux. Specific functionality determined by software driver.
GPIO_18	**General Purpose I/O 18.** Multiplexed with I2C_SCL.
GPIO_19	**General Purpose I/O 19.** Multiplexed with I2C_SDA.
GPIO_20	**General Purpose I/O 20.** Multiplexed with EXP_ALE.
GPIO_21	**General Purpose I/O 21.** Multiplexed with EXP_RDY/BSY#.
GPIO_22	**General Purpose I/O 22.** Multiplexed with UART1_RX.
GPIO_23	**General Purpose I/O 23.** Multiplexed with UART1_TX.
GPIO_24	**General Purpose I/O 24.** Multiplexed with SPI_SCLK.
GPIO_25	**General Purpose I/O 25.** Multiplexed with SPI_TXD.
GPIO_26	**General Purpose I/O 26.** Multiplexed with SPI_SS1_N
GPIO_27	**General Purpose I/O 27.** Multiplexed with SPI_SS2_N.
GPIO_28	**General Purpose I/O 28**. Not shared or multiplexed with other signals.
GPIO_29	**General Purpose I/O 29.** Shared with EXP_. NAND_CS or EXP_A18. Not a real mux. Specific functionality determined by software driver.
GPIO_30	**General Purpose I/O 30.** Shared with EXP_NAND_ALE or EXP_A19. Not a real mux. Specific functionality determined by software driver.
GPIO_31	**General Purpose I/O 31.** Shared with EXP_NAND_CLE or EXP_A20. Not a real mux. Specific functionality determined by software driver.

14.4 General Purpose Boot Option Inputs

The TM_GPBT_OP(1:0) pins are general-purpose, and have no pre-defined function. These configuration pins are muxed with Expansion Bus address lines:

TM_GPBT_OP0 — EXP_A_L_10 — Internally pulled down

TM_GPBT_OP1 — EXP_A_L_11 — Internally pulled down

During reset, the block does not drive these pins, and they can be set to any configured value through pull-ups or pull-downs. The I/O pads have internal pull-up/-down resistors. Hence, an external pull is required only if the default value made by the internal pull is not the desired one.

Following reset, the block drives these pins to support Expansion Bus access. The previous latched value is already stored in a software accessible register from which the value can be read but not changed.

14.5 GPIO Timing

Figure 14-2 illustrates the GPIO timing waveform.Table 14-3 lists the GPIO timing parameters.

Figure 14-2 GPIO Timing Waveform

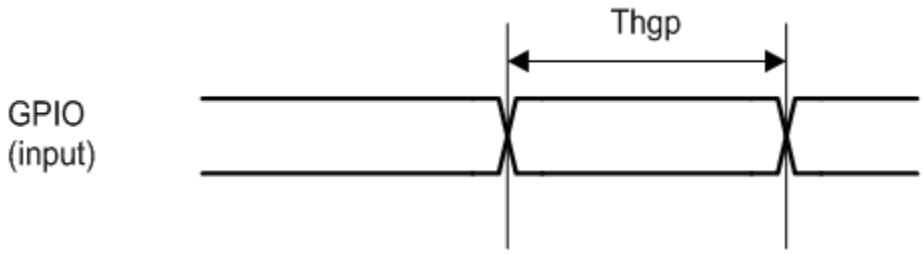


Table 14-3 GPIO Timing Parameters

Symbol	Description	Min	Max	Units	Notes
Thgp	GPIO pulse duration	9		ns	This is the minimal pulse width time for a GPIO (as an input) that is guaranteed to internally register a change.

15 Interrupt Controller

This section provides details on the LS10xMA Interrupt controller.

15.1 Overview

The LS10xMA interrupt controller allows internal and external events to trigger IRQs and FIQs to both ARM 1136J-S processors.

15.2 Interrupt Controller Features

- Status registers that identify specific interrupt conditions to support up to 60 interrupts.
- Mask registers to enable or disable IRQs and FIQs interrupts for the two ARM processors
- Software-based interrupt generation through registers
- Supports ARM's interrupt vector interface with priority and self-clear mechanism
- Inter-processing signaling features
- Hardware support within Interrupt Priority Determination

15.3 Functional Description

The Interrupt Controller provides the status and mask registers to create IRQs and FIQs for the two ARMs. This module also contains inter-processing signaling features.

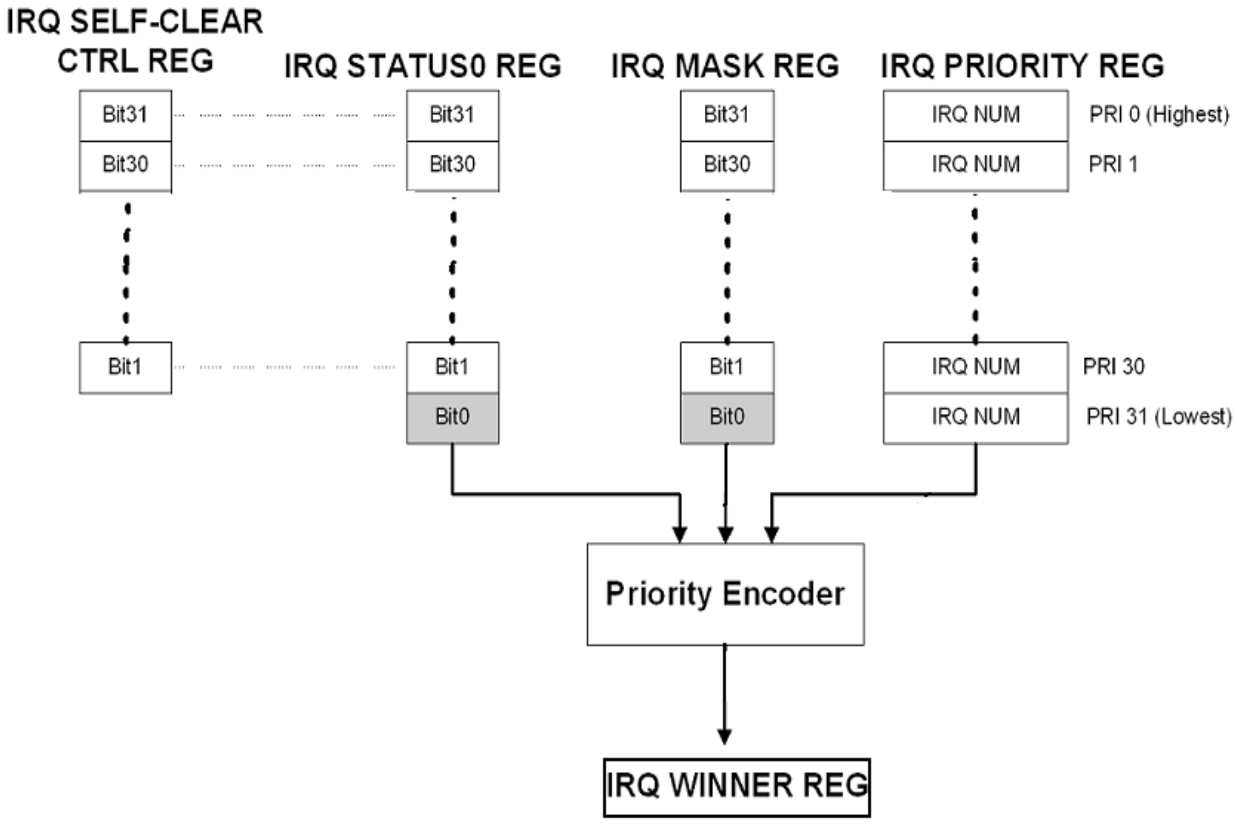
The Interrupt Controller device supports more than 32 peripheral interrupts and therefore, requires multiple interrupt status registers and associated mask registers. To handle over 32 interrupts, two sets of interrupt control registers are required: Status registers control for register 0 and status registers control for register 1. Each includes:

- STATUS_REGISTER
- CLR_STATUS_BIT
- SET_STATUS REGISTER_BIT
- ARM0_IRQ_MASK
- ARM0_FIQ_MASK
- ARM1_IRQ_MASK
- ARM1_FIQ_MASK

IRQ0 (ARM0 IRQ) is created by “ANDING” the STATUS_REGISTER_0 register and ARM0_IRQ_MASK_0 register, STATUS_REGISTER_1 and ARM0_IRQ MASK_1 register then collapsing the result to a single bit. FIQ0 (ARM0 FIQ), IRQ1 (ARM1 IRQ) and FIQ1 (ARM1 FIQ) are created in a similar way with mask registers with offsets respectively.

The mask registers do not affect the reading of the status registers. Writing a ‘1’ to a corresponding bit location clears the status bit. It is important to know whether the source, which affects the status bit, is an event or a level. If the source is a level indication (e.g. PUI_IRQ and EMAC_IRQ) then the source must be cleared before the affected bit can be cleared. Figure 15-1illustrates the interrupt priority.

Figure 15-1 Interrupt Priority



The possible interrupt generation scenarios are as follows:

1. One of the blocks generates an interrupt. The relevant bit in STATUS_REGISTER_0 is being set, which notifies the relevant ARM interrupt service, which than clears the interrupt by setting the CLR_STATUS_0 register.
2. ARM software can simulate a specific interrupt by setting the relevant bit in SET_STATUS_REGSTER_0, which sets the relevant bit in the STATUS_REGISTER_0. The software is also responsible to clear the relevant bit setting the CLR_STATUS_0 register.
3. The ARM processor architecture provides only two interrupts: the IRQ and FIQ Interrupt. Therefore, there are four device level interrupts: ARM0_IRQ, ARM0_FIQ, ARM1_IRQ and ARM1_FIQ. The device permits each peripheral interrupt to be mapped to the IRQ or FIQ interrupt of a specific ARM processor.

16 Timers

This section provides details on the LS10xMA Timers.

16.1 Overview

The LS10xMA timer block includes six general purpose timers each of which can generate an interrupt when the upper limit is reached. Two of the general purpose timers can link to I/O pins and either drive an external output or allow external signals to increment the timer. Each timer consists of two sections, a register and a count-up counter. This block also contains TDM frame sync timer and a watchdog timer.

16.2 Timer Features

General Purpose Timers

- Six 32-bit timers (timer0, timer1, timer2, timer3, timer4, timer5) are used to provide interrupts.
- Six interrupts, one interrupt per timer.
- A secondary interrupt status register to indicate which of the timers had issued a pulse
- Timer3 can drive TIM_EVNT0, timer5 can drive TIM_EVNT1 when these pins are configured as outputs. When a timer reaches its high bound value, the corresponding pin will be switched.
- Timer1 can count the changes on TIM_EVNT0, timer4 can count changes on TIM_EVNT1 when these pins are configured as inputs.
- A timer2 control register can be programmed to chain timer0 and timer2 together
- A timer3 control register can be programmed to chain timer1 and timer3 together
- A timer5 control register can be programmed to chain timer4 and timer5 together
- Timer2, timer3 and timer5 have programmable low and high bound registers.
- Timer0, timer1 and timer4 can be written with specific values if needed.

TDM Frame Sync Timers

- Two 16-bit TDM timers driven by TDM frame sync timing, providing pulses to the interrupt controller.
- One TDM timer interrupt can be programmed from one or more of the two timer pulses
- A secondary TDM timer interrupt status register to indicate which of the timers had issued a pulse

Watch Dog Timer

- One 32-Bit watch dog timer driven by hclk, providing hard reset to the reset block.

16.3 Functional Description

Timer block consists of six general-purpose timers. Each timer consists of two sections, a high/low bound registers and a count-up counter. Every time the high or the low bound registers are configured, the counter is reloaded with the low bound value. Therefore, it is recommended to first configure the high bound register and then the low bound register.

In normal mode, the general purpose counters increment on every rising edge of hclk. Once the counter reaches the high bound value, it will assert the interrupt if the interrupt is not masked and will be reloaded with the low

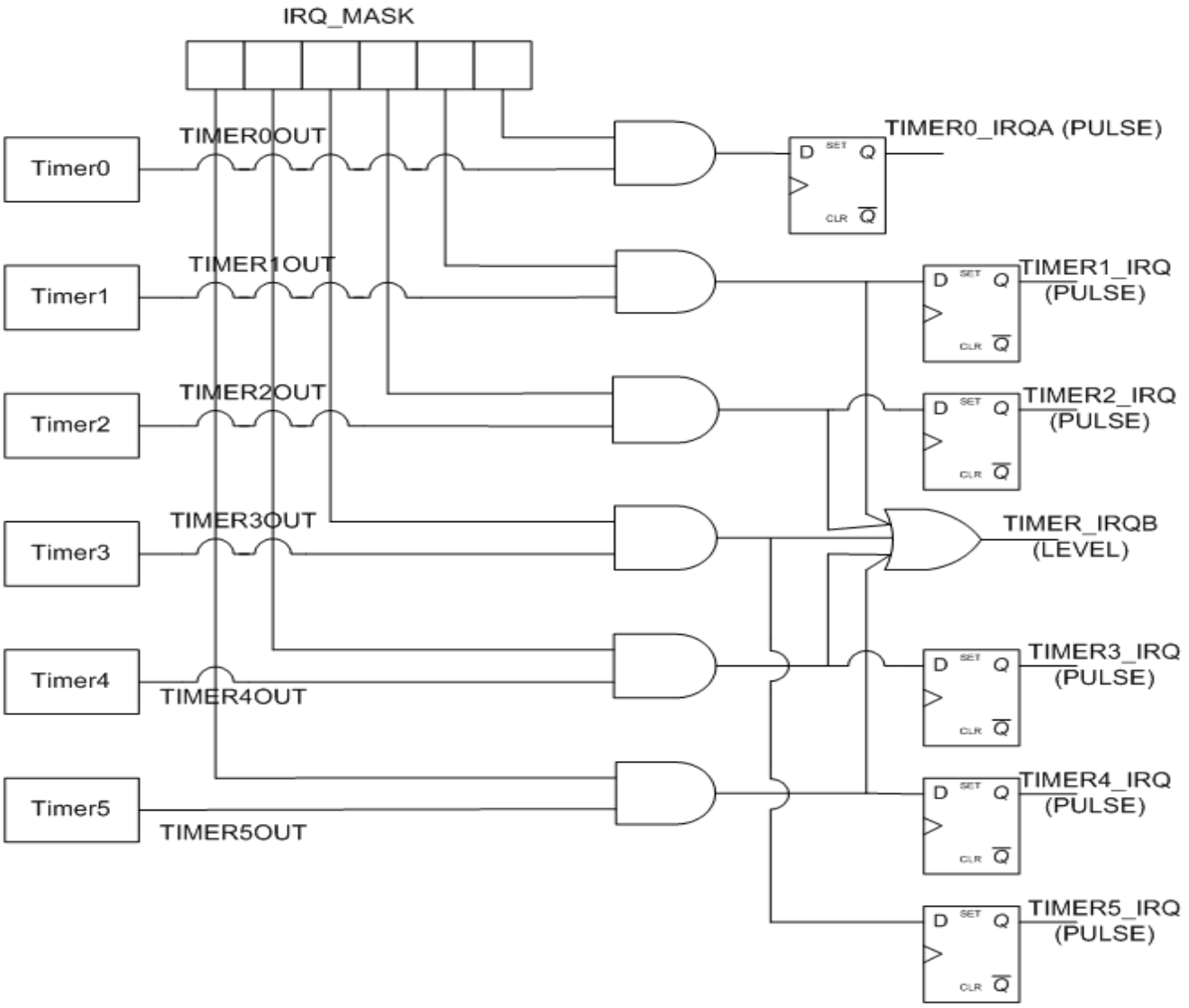
bound value again. The TDM Frame SYNC counters can be configured to count Rx Frame SYNC or Tx frame SYNC. In normal mode, the TDM timers increment on rising edge of HCLK only when TDM Frame Sync is high. For TIM_EVENT mode, see Table 16-1 and Table 16-2.

In chained mode, timer2 increments when timer0 times out, timer3 increments when timer1 times out, timer5 increments when timer4 times out. An interrupt will be asserted when timer2, timer3 or timer5 times out.

16.3.1 Interrupt Generation

As shown in Figure 16-1, the IRQ mask registers are used to selectively enable the timerout pulses to generate the IRQA and timers 1 to 5 IRQ interrupts. An Interrupt status register in the timer block can be read to check which of the timers had generated a timeout. This information can also be read from the interrupt controller, since all timers IRQ are being output.

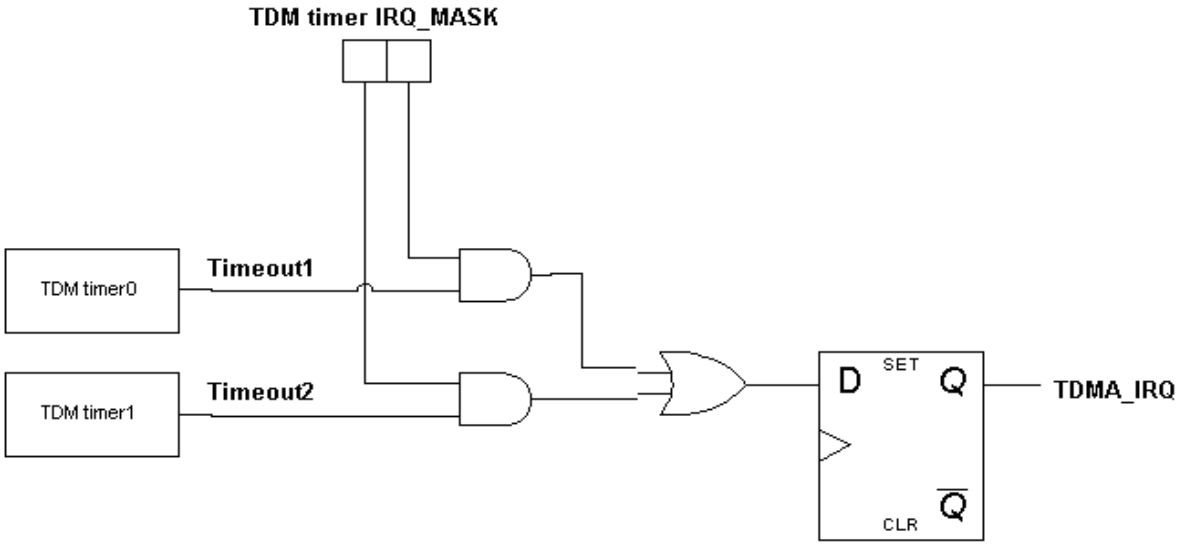
Figure 16-1 General Purpose Timer Top-level Block Diagram



16.3.2 TDM Timer

The TDM timer IRQ mask registers are used to selectively enable the timerout pulses to generate the TDMA IRQ interrupts. The TDM interrupt status register in the timer block can be read to check which of the TDM timers had generated a timeout.

Figure 16-2 TDM Timer Top Level Block Diagram



16.3.3 Counting or Driving TIM_EVNT0 and TIM_EVNT1

TIM_EVNT0 and TIM_EVNT1 pins can be either input or output, depending on the configuration. See Tables 16-1 and Table 16-2 for the timer behavior.

Table 16-1 TIM_EVNT0 Behavior

TIM_EVNT0	Chain Mode	Non-Chain Mode
TIM_EVNT0 is an input and is not ignored by the timer block.	Timer1 counts rising or falling edge on TIM_EVNT0. Timer3 is clocked by Timer1's timeout.	Timer1 counts rising or falling edge on TIM_EVNT0. Timer3 is clocked by hclk.
TIM_EVNT0 is output.	Timer3 is clocked by timer1's timeout. TIM_EVNT0 is toggled by timer3's timeout.	Timer3 is clocked by hclk.TIM_EVNT0 is toggled by timer3's timeout. Timer1 counts with hclk and does not affect TIM_EVNT0.

Table 16-2 TIM_EVNT1 Behavior

TIM_EVNT1	Chain Mode	Non-Chain Mode
TIM_EVNT1 is an input and is not ignored by the timer block.	Timer4 counts rising or falling edge on TIM_EVNT1. Timer5 is clocked by Timer4's timeout.	Timer1 counts rising or falling edge on TIM_EVNT0. Timer3 is clocked by hclk.
TIM_EVNT1 is output.	Timer5 is clocked by timer4's timeout. TIM_EVNT1 is toggled by timer5's timeout.	Timer5 is clocked by hclk.TIM_EVNT1 is toggled by timer5's timeout. Timer4 counts with hclk and does not affect TIM_EVNT1.

NOTE: In order for the timer to catch the rising of falling edge on TIM_EVNT0 or TIM_EVNT1, TIM_EVNT0 or TIM_EVNT1 has to stay high or low for at least 1 hclk cycle plus 3 ns.

16.4 Timer Interface Signals

Table 16-3 lists the timer signals.

Table 16-3 Timer Interface Signals

Signal Name	Dir	Size	Signal Description
TIM_EVNT[1:0]	B	2	Timer Event — These pins can be configured as output, to generate an external clock by the timer, or as input, to count external events (edges).

17 Test and Debug Interface

This section provides details on the LS10xMA Test and Debug interface.

17.1 Overview

The JTAG interface provides access to the ARM embedded ICE unit of both ARM0 (ACP) and ARM1 (MSP). The two ARMs are chained one after the other. The JTAG port supports IEEE 1149.1 Boundary Scan for manufacturing and test. For more information on ARM ICE that uses the JTAG port, see *RealView ICE and RealView Trace User Guide.*

To support test, the TM_BSCANMODE and TM_TESTMODE_N input can be asserted to select test modes. See Section 17.4 Test Mode Configuration.

17.2 Test and Debug Interface Signals

17.2.1 JTAG Interface Signals

To use the JTAG interface for normal and boundary scan set test mode as described in Section 17.4.

Table 17-1lists the JTAG signal description.

Table 17-1 JTAG Signals

Signal Name	I/O	Size	Signal Name/Description
JTAG_TRST#	I	1	JTAG Test Reset. A low signal forces the TAP controller into a logic reset state. TRST# must be open (i.e., unconnected) for normal (non-test) operation. This pin has an internal pull-up, and it conforms to IEEE 1149.1 JTAG specification. **NOTE**: TRST# may be held low for normal operation, but this prevents the use of the TAP as a debug port. Leaving TRST# open (pull-up internal) allows use of the debug port. The TRST# I/O type interfaces well with the corresponding ARM MultiICE output, an open collector driver with a pull-up.
JTAG _TCK	I	1	JTAG Test Clock. This is the JTAG clock signal. This pin has an internal pull-up, and it conforms to IEEE 1149.1 JTAG specification.
JTAG_TMS	I	1	JTAG Test Mode Select. This is the control signal to the TAP controller. This pin has an internal pull-up, and it conforms to IEEE 1149.1 JTAG specification.
JTAG_TDI	I	1	JTAG Test Input Data. This is the boundary scan serial input signal, and it is shifted in on the rising edge of TCK. The pin has an internal pull-up, and it conforms to IEEE 1149.1 JTAG specification.
JTAG_TDO	O	1	JTAG Test Output Data. This is the three-stateable boundary scan data output signal, and it is shifted out on the falling edge of TCK. It conforms to IEEE 1149.1 JTAG specification.

17.2.2 Test Mode Configuration Inputs

Table 17-2 lists the Test signal description.

Table 17-2 Test Signals

Signal Name	I/O	Size	Signal Description
TM_BSCANMODE	I	1	See Section 17.4 for functionality.
TM_TESTMODE_N	I	1	See Section 17.4 for functionality.

17.3 JTAG Interface Timing

The JTAG Interface complies with the IEEE 1149.1 JTAG specification. Figure 17-1 illustrates the JTAG timing waveform and Table 17-3 lists the JTAG timing parameters.

Figure 17-1 JTAG Timing Waveform

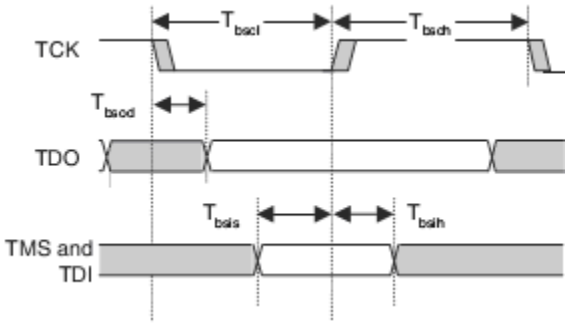


Table 17-3 JTAG Timing Parameters

Symbol	Description	Min	Max	Units
Tbscl	TCK LOW period	50		ns
Tbsch	TCK HIGH period	50		ns
Tbsod	TDO output delay from TCK (falling)	0	25	ns
Tbsis	TDI, TRST_N and TMS setup to TCK (rising)	20		ns
Tbsih	TDI, TRST_N and TMS hold from TCK (rising)	10		ns

17.4 Test Mode Configuration

TM_BSCANMODE and TM_TESTMODE_N inputs configure the LS10xMA device in one of four modes:

- Normal — This is the standard operational mode for the LS10xMA device. In this mode the ARMs can be accessed through the JTAG
- Tri-state — in this mode, for manufacturing test all output-only and bidirectional signals are in high impedance. The device leaves tri-state mode when either TM_BSCANMODE or TM_TESTMODE_N change logic levels. If

RESET_N remains asserted as the device enters this mode, then normal reset hold times and clock stability requirements apply after the device leaves this mode.

NOTE: Few output-only and bidirectional signals, such as USB, TM_PLL_XO, and so on are unique and the tri-state mode does not impact these signals.

- Boundary Scan — This is the JTAG boundary scan mode.
- Internal Scan — This mode is intended for factory use only. This mode should not be used.

Table 17-4 lists the test mode configuration.

Table 17-4 Test Mode Configuration

Input /Test Mode	TM_BSCANMODE	TM_TESTMODE_N
Normal	0	1
Tri-state	0	0
Boundary Scan	1	1
Internal Scan	1	0

18 Power Management

This section provides details on the LS10xMA Power Management.

18.1 Overview

LS10xMA power management options include stop-clock functionality, processor speed control, and sleep, power-down, or suspend modes for specific blocks, including external SDRAM.

18.2 Programmable Stop-Clock Functionality

Significant power savings may be made by stopping clocks to unused blocks within LS10xMA devices. By default, all clocks are enabled as the device boots. Clocks to the following LS10xMA logic cores can be independently stopped:

- ARM0
- ARM1
- Etherenet0 (WAN) Interface
- Ethernet1 (LAN) Interface
- USB Controller and PHY
- TDM Interface and TDM DMA Controller
- UART0/UART1
- I^2C and SPI Controllers
- MDMA Controller
- IPSEC Security Co-Processor
- PCIe Controllers and SERDES

18.2.1 Stop-Clock Procedures

Logic cores with input reference clocks that are driven from clocks generated from the device such as, TDM, Ethernet MACs, can be taken out of reset with the reference clock input stopped or they can be held in software reset. Unless the anticipated power-down period is short, stopping the input reference is more attractive for power savings.

18.2.2 Output Clock Control

The LS10xMA provides several internally generated clocks that can be used as the input clocks for some interfaces. Except for TDM clock (which is routed internally in the device), all of the other interface clocks need to be routed externally from the output to the input.

Clocks sourced by the LS10xMA to external devices internally can be stopped. Logic blocks affected include:

- TDM Interface (TM_TDM_REFCLO, TM_TDM_FSO)
- Ethernet PHY Interface clocks (TM_ETH0_REFCLK, TM_ETH1_REFCLK)
- Expansion Bus Interface (EXP_CLK)

18.3 Automatic Stop-Clock Functionality

The ARM processors automatically stop clocks and other inputs to unused modules. Through the extensive use of this approach, only the logic actively used to perform a calculation consumes power. For more information, refer to the *ARM1136JF-S and ARM1136J-S Technical Reference Manua*l.

18.4 ARM Clock Frequency Control

The clocks driving the two ARM processors can be modified in tandem and independently.

- The clock frequency PLL output may be moved downward from 650 MHz — this will affect the frequency of both ARMs uniformly.
- An integer divisor may be configured to reduce the clock for a specific ARM independent of the other ARM.

CAUTION: **The device is screened for the specified frequency. Freescale cannot guarantee the operation if any changes are made in software. Any changes made without prior approval from Freescale may cause unexpected results and void in the warranty.**

18.5 ARM Sleep Modes

The two ARM processors may be independently moved in and out of power-down states. ARM11 architecture supports the Standby power consumption modes.

For more information, refer to the *ARM1136JF-S and ARM1136J-S Technical Reference Manua*l.

18.6 DDR2 Power Down Modes

The LS10xMA can power-down DDR2 SDRAM. Two modes of SDRAM power down are supported:

18.6.1 Memory Power-Down

The memory controller sets the memory devices into power-down mode. In this mode, the controller and memory clocks are fully operational, but the DDR_CLKE input bit to the memory devices is de-asserted. The controller continues to monitor memory refresh needs and automatically bring the memory out of the power-down mode to perform these refreshes. When a refresh is required, the DDR_CLKE input bit to the memory devices will be re-enabled. This action brings the memory devices out of power-down. Once the refresh has been completed, the memory devices will be returned to power-down mode by de-asserting the DDR_CLKE input bit.

18.6.2 Memory Self-Refresh

The memory controller sets the memory devices into self-refresh mode. In this mode, the controller and memory clocks are fully operational and the DDR_CLKE input bit to the memory devices is de-asserted. Since the memory automatically refreshes its contents, the controller does not need to send explicit refreshes to the memory.

18.7 USB Suspend Mode

When operating in host mode, the USB controller is capable of global and selective Suspend and Resume operations across attached USB devices as described in section 4.3 of the Intel EHCI specification. The USB controller presents a standard EHCI-compatible interface to software drivers.

When operating in device mode, the USB controller supports Suspend and Resume functionality as described in the USB Specification, revision 2.0. When operating in device mode, the USB controller is capable of global and selective Suspend and Resume operations as described in section 7.1.7.6 of the USB 2.0 specification.

18.8 PCIe Power Management

The LS10xMA supports link and device power management functionality that described in PCI Bus Power Management Specification 1.2 and Advanced Configuration/Power Management Specification 2.0.

18.9 Input Output Power Up

In the I/O power up options, the DDR higher voltage supply is first, followed by core supply. The GPIO digital supply is first, followed by analog supply.

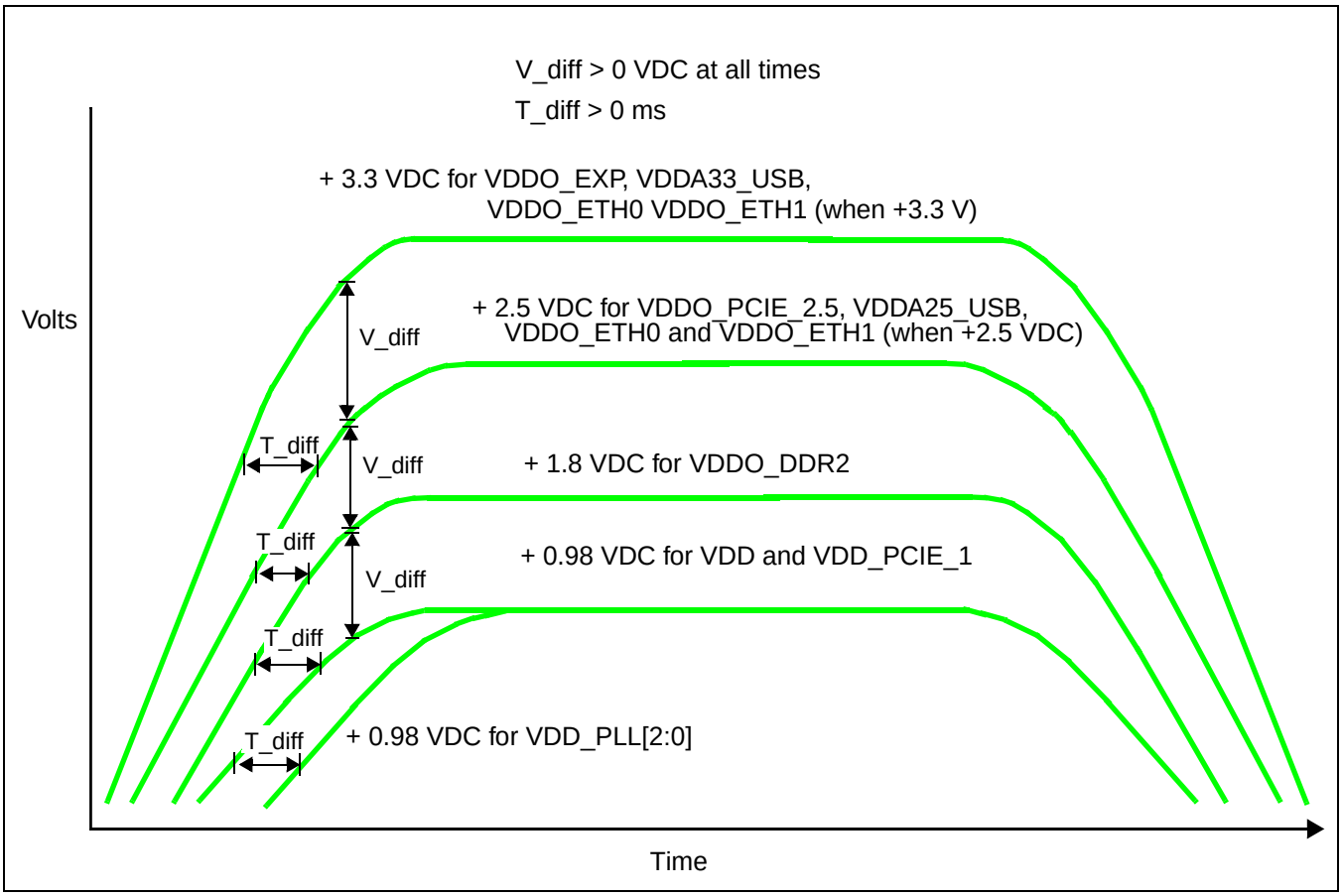
19 Hardware Initialization

This section describes the sequence in which operating voltages must be applied and what conditions must be met before device reset.

19.1 Power Supply Sequencing

To avoid damage to device ESD structures DC power must be applied in a specific order. This is summarized in Figure 19-1.

Figure 19-1 Power up and Power down sequencing



VDDO_ETH0 and VDDO_ETH1 may be configured as +2.5 or +3.3 VDC depending on RGMII mode or GMII/MII/RMII mode that is in use.

The following subsections present power application rules and procedures:

19.1.1 Power Sequencing Rules

Power domains:

- VCORE — VDD (+0.98 V)

- **CMOS** — VDD_PCIE_1 (+0.98 V)
  VDDO_PCIE_2.5 (+2.5 V)
  VDDO_ETH0 (+2.5 or 3.3 V)
  VDDO_ETH1 (+2.5 or 3.3 V)
  VDDO_EXP (+3.3 V)

  CMOS +3.3 V must be powered on before +2.5 V (when used) and both of them before +0.98 V.

- **USB** — VDDA33_USB (+3.3 V)
  VDDA25_USB (+2.5 V)

  USB +3.3 V must be powered on before +2.5 V and both of them before +0.98 V.

- **SSTL** —VDDO_DDR2 (1.8 V)
  VDD_REF (0.9 V)

  SSTL +1.8 V must be powered up before +1.0V.

- **PLL** — VDD_PLL_0 (+0.98 V)
  VDD_PLL_1 (+0.98 V)
  VDD_PLL_2 (+0.98 V)

  PLL +0.98 V must be powered up after all CMOS power.

There are no power-up sequence requirements between the separate power domains: CMOS, SSTL, and USB.

NOTE: Designers must consider the ramp rates of various power supplies and board-level practice to ensure this power-up sequence is met. Try to slow the ramp rate or introduce sufficient delay to meet these requirements.

19.1.2 Power Up Procedure

Initialize the power supplies in the following sequence:

1. **+3.3 V —** VDDO_EXP
   VDDO_ETH0 (when +3.3 V)
   VDDO_ETH1(when +3.3 V)
   VDDA33_USB
2. **+2.5 V —** VDDO_PCIe_2.5
   VDDO_ETH0 (when +2.5 V)
   VDDO_ETH1(when +2.5 V)
   VDDA25_USB
3. **+1.8 V —** VDDO_DDR2
4. **+0.98 V —** VDD
   VDD_PCIe_1
5. **PLL +0.98** —VDD_PLL_0
   VDD_PLL_1
   VDD_PLL_2

19.1.3 Power Down Procedure

The power down sequence is the reverse of the power up sequence.

19.2 Reset Specifications

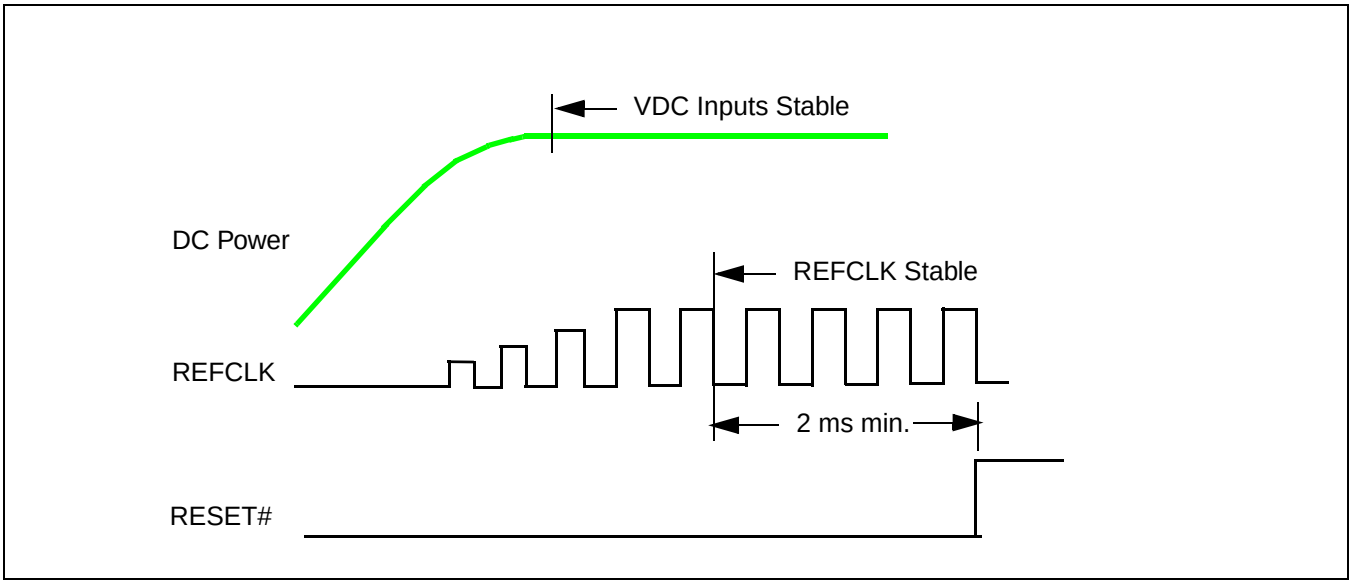
After power up and other pre-requisites are met, the reset input puts the LS10xMA device into a known state for software initialization.

19.2.1 RESET# Timing

Prior to de-assertion of RESET#, the following conditions must be met:

- All DC power inputs must be with their specifications.
- Analog reference voltages and PLL power inputs must be with their specifications.
- REFCLK must be within its specifications.

Figure 19-2 Reset Timing



REFCLK must be within its specifications (see Section 20.4 Reference Clock) for the LS10xMA device to perform a valid reset operation.

19.2.2 Latching of Configuration Inputs after Reset

There are several I/Os which are latched at reset time and affect various modes of operation for the device. These configuration pins are multiplexed with the 13 lower bits of the Expansion Bus addresses.

During reset, the block does not drive these pins, and they can be set to any configured value through external pull-ups or pull-downs. All of these I/Os have internal pull-up/pull-down resistors, so an external pull is required only if the default value (through the internal pull) is not the desired one.

NOTE: To be recognized and latched at reset, these signals must be stable 20 nanoseconds before and after the rising edge of RESET#.

Table 19-1 presents the configuration inputs.

Table 19-1 Configuration Inputs

Signal Name	Signal Name/Description
TM_PCIE0_MODE	Sets up PCIe mode for specific lane as Root mode. Chapter 7, PCI Express Interface.
TM_ETH0_MODE_0	Configure Ethernet MAC type for port 0. See Section 6.6 Ethernet Signal Multiplexing.
TM_ETH0_MODE_1	
TM_ETH1_MODE_0	Configure Ethernet MAC type for port 1. See Section 6.6 Ethernet Signal Multiplexing.
TM_ETH1_MODE_1	
TM_BOOT_OP_0	Selects boot source. See Section 19.3 Boot Sequence.
TM_BOOT_OP_1	
TM_TDM_CLKSRC_EN	When high, the device provides TDM_CK to external circuitry.
TM_GPBT_OP0	These inputs can be defined by software configuration. See Section 14.4 General Purpose Boot Option Inputs.
TM_GPBT_OP1	
TM_EXP_NAND_SEL	When high, the Expansion Bus is configured for NAND accesses.
TM_PCIe_EXT_REFCL	If the PCI Express interface is used, TM_PCIe_EXT_REFCLK must be high at reset time. If the PCI Express interface is not used, TM_PCIe_EXT_REFCLK can be pulled low with an external resistor or with the internal pull-down resistor that is built into the I/O cell.

19.3 Boot Sequence

The LS10xMA supports one stage boot loader through NOR or two stages loader through SPI/I^{2}C EEPROM and through NAND.

19.3.1 Boot Source Options

Two boot options: internal and external.

In the case of external boot, code is executed directly from NOR. The NOR device may be a large one, in which case the full code may be on the NOR device, or it may be a small one, in which case enough code will be on it to setup the NAND interface, copy code from the NAND device to the DDR2 SDRAM, and continue executing directly from DDR2 SDRAM.

In internal boot, a configuration pin selects if a memory device is present on SPI or I^{2}C. Code is executed directly from the IBR. This code in turn copies code from external memory through the SPI or I^{2}C interfaces in ARAM, and it is then executed directly from the ARAM. This ARAM code operates as a secondary boot loader and goes on to copy code from external sources into the DDR2 SDRAM, and then commence execution from the DDR2 SDRAM.

The two configuration pins TM_BOOT_OP[1:0] define the boot sequence. Table 19-2 lists the boot configuration details.

Table 19-2 Boot Configuration Lands

TM_BOOT_OP1	TM_BOOT_OPT0	BOOT OUTCOME
0	0	Internal boot, copy code from SPI device.
0	1	Internal boot, copying code from I^2C device
1	0	External boot, directly from 8-Bit NOR device.
1	1	External boot, directly from 16-Bit NOR device.

20 Electrical and Environmental Specifications

This section describes the electrical and environmental specifications of LS10xMA device.

20.1 Operating Conditions

Table 20-1 and Table 20-2 list the voltage and temperature requirement for LS10xMA device.

NOTE: The specifications in the following tables are based on estimates and simulations. Updates based on characterized data from silicon measurements will be provided later.

Table 20-1 Voltage Requirements

Parameter	Symbols	Min.	Typ.	Max.	Units
Digital					
Core Supply Voltage (0.98V ± 3%) PCIe Supply Voltage (0.98V ± 3%)	VDD VDD_PCIe_1	0.950	0.980	1.010	VDC
External Memory Interface Supply Voltage (1.8V ± 5%)	VDDO_DDR2	1.71	1.80	1.89	VDC
Ethernet Interface Supply Voltage (2.5V ± 5%) PCIe Interface Supply Voltage (2.5V ± 5%) USB 2.5V supply	VDDO_ETH[1:0]* VDDO_PCIe_2.5 VDDA25_USB	2.375	2.50	2.625	VDC
Input-Output (UART,JTAG,I2C,TDM, SPI, and GPIO) Alternate Ethernet Interface Supply Voltage (3.3V ± 5%) USB 3.3V Supply	VDDO_EXP VDDO_ETH[1:0]* VDDA33_USB	3.14	3.30	3.47	VDC
Reference					
SSTL2 Reference Voltage	VDD_REF	0.49_{*VDDQ}	0.50_{*VDDQ}	0.51_{*VDDQ}	V
PLL Reference Voltage (0.98V ± 3%)	VDD_PLL_0 VDD_PLL_1 VDD_PLL_2	0.950	0.980	1.010	V
Note: * VDDO_ETH0 power the MAC0 interface and VDDO_ETH1 power the MAC1 interface. * VDDO_ETHO and VDDO_ETH1 supply pins can be tied to the same supply or to different supply depending on the requirements of the implemented interface * RGMII was verified with 2.5V devices and supplies. GMII, MII, and RMII were verified with 3.3V devices and supplies.					

Table 20-2 Temperature Conditions

Parameter	Symbols	Min.	Typ.	Max.	Units
Operating Temperature	T_A	0	—	+70	°C
		- 45 (LS101MAXE7DFA)		+85 (LS101MAXE7DFA)	°C

Table 20-3 lists the maximum power consumption of LS10xMA devices.

Table 20-3 Maximum Power Consumption

	Maximum Power Consumption			
Parameter	**Symbol**	**LS102MASX7EHA, LS102MASX7BFA**	**LS101MAXE7DFA**	**Units**
Core Supply Power (0.98v + 3%)	VDD_PLL_[2:0], VDD, VDD_PCIe_1	1.20		W
DDR (1.8v + 5%)	VDDO_DDR2	0.45		W
USB PHY and PCIe (2.5v + 5%)	VDDA25_USB, VDDO_PCIe_2.5 VDDO_ETH [1:0]	0.25		W
USB PHY and I/O Supply Power (3.3v + 5%)	VDDO_EXP, VDDA33_USB	0.25		W
Total Power Consumption (operating)		2.15	1.8	W
The maximum power per supply rail might vary slightly per application, but the total maximum power consumption does not change.				

20.2 Absolute Maximum Ratings

Absolute maximum ratings specify conditions under which there will be no device damage, but device operation is not guaranteed. Table 20-4 lists the absolute maximum ratings.

Table 20-4 Absolute Maximum Ratings

Parameter	Symbol	Limits	Units
Core Supply Voltage (+0.98V)	VDD VDD_PCIe_1	-0.3 to +1.1	V
External Memory Interface Supply Voltage (+1.8V)	VDDO_DDR2	-0.5 to +2.3	V
External Interface Supply Voltage (+2.5V)	VDDO_ETH[1:0] [1] VDDO_PCIe_2.5 VDDA25_USB	-0.5 to +3.0	V
I/O and Alternate Ethernet Interface Supply Voltage (3.3V)	VDDO_ETH[1:0][1] VDDO_EXP VDDA33_USB	-0.3 to +3.6	V
Input Voltage	V_{IN} for GEM signals (when VDDO_ETH[1:0] is 3.3V) and other digital signals, not including DDR V_{IN} for GEM signals (when VDDO_ETH[1:0] is 2.5V) V_{IN} DDR signals V_{IN} for other	-0.5 to +5.5 -0.5 to +3.6 -0.5 to +2.3 -0.5 to (VDD + 0.5)	V
Storage Temperature	T_{STG}	-65 to +110	°C
Voltage Applied to Outputs in High Impedance (Off) State	V_{HZ}	-0.5 to (VDD + 0.5)	V
DC Input Clamp Current	I_{IK}	±20	mA

Table 20-4 Absolute Maximum Ratings (continued)

Parameter	Symbol	Limits	Units
DC Output Clamp Current	I_{OK}	±20	mA
Static Discharge Voltage HBM (25°C)	V_{ESD}	±2000	V
Latch-up Current (85°C)	I_{TRIG}	±150	mA
Junction Temperature	T_J	110	°C

Note:
1. RGMII was verified with 2.5V devices and supplies. GMII, MII, and RMII were verified with 3.3V devices and supplies.

20.3 I/O Characteristics

20.3.1 LVCMOS Characteristics

Table 20-5 LVCMOS Characteristics

Parameter	Symbol	Min.	Typ.	Max.	Units	Test Conditions[1]
V_{DDO_EXP}, V_{DDO_ETH0}, V_{DDO_ETH1} = 3.3VDC						
Input Voltage Low	V_{IL}	-0.3	–	0.8	VDC	
Input Voltage High	V_{IH}	2.0	–	5.5	VDC	
Input Current Low [2]	I_{IL}	–	–	-10	µA	V_{IN} = 0
Input Current High [2]	I_{IH}	–	–	+10	µA	V_{IN} = 3.3V
Input Current Low (with internal pull-downs) [2]	I_{IL}	–	–	-10	µA	V_{IN} = 0
Input Current High (with internal pull-downs) [2]	I_{IH}	–	–	+130	µA	V_{IN} = 3.3V
Input Current Low (with internal pull-ups) [2]	I_{IL}	–	–	-100	µA	V_{IN} = 0
Input Current High (with internal pull-ups) [2]	I_{IH}	–	–	+10	µA	V_{IN} = 3.3V
Output Voltage Low	V_{OL}	–	–	0.4	VDC	
Output Voltage High	V_{OH}	2.4	–	VCC	VDC	
Three-State (Off) Current Low[2]	I_{OTL}	–	–	-10	µA	V_{IN} = 0V
Three-State (OFF) Current High[2]	I_{OTH}	–	–	+10	µA	V_{IN} = 3.3V
Threshold point	V_T	1.31	1.41	1.50	V	
Schmitt trigger threshold point Low to High	V_T+	1.57	1.69	1.79	V	
Schmitt trigger threshold point High to Low	V_T-	1.15	1.25	1.34	V	
V_{DDO_ETH0}, V_{DDO_ETH1} = 2.5VDC						
Input Voltage Low	V_{IL}	-0.3	–	0.7	V	
Input Voltage High	V_{IH}	1.7	–	3.6	V	
Input Current Low [2]	I_{IL}	–	–	-10	µA	V_{IN} = 0V
Input Current High [2]	I_{IH}	–	–	+8	µA	V_{IN} = 2.5V
Output Voltage Low	V_{OL}	–	–	0.7	VDC	
Output Voltage High	V_{OH}	1.7	–	–	VDC	
Three-State (Off) Current Low[2]	I_{OTL}	–	–	-10	µA	V_{IN} = 0V

Parameter	Symbol	Min.	Typ.	Max.	Units	Test Conditions[1]
Three-State (OFF) Current High[2]	I_{OTH}	–	–	+10	µA	V_{IN} = 2.5V
Threshold point	V_T	1.01	1.08	1.12	V	

Notes:

Test Conditions (unless otherwise stated):
V_{CORE} = +0.98 ± 0.03 VDC
V_{DDO_DDR2}= +1.8 ± 0.09 VDC
V_{DDO_EXP} = +3.3 ± 0.165 VDC
$V_{DDO\_ETH0,\ ETH1}$ = +3.3 ± 0.165 VDC when +3.3V Ethernet transceiver is used
$V_{DDO\_ETH0,\ ETH1}$ = +2.5 ± 0.125 VDC when +2.5V Ethernet transceiver is used
$V_{IN\ (MAX)}$= +3.6V
Tj = -40°C to 125°C

Current flow out of the device is shown as minus.

20.3.2 SSTL18 Characteristics

These apply to the DDR2 SDRAM interface powered by V_{MEM} (1.8V, operating).

Table 20-6 SSTL2 Characteristics

Parameter	Symbol	Min.	Typ.	Max.	Units	Test Conditions[1]
SSTL18 I/O power	V_{DDQ}	1.71	1.80	1.89	V	
Low level output current	I_{OL}	13.4	–	–	mA	
High level output current2	I_{OH}	-13.4	–	–	mA	
High level input voltage, DC	V_{IH}	V_{REF} + 0.125	–	V_{DDQ} + 0.3	V	
Low level input voltage, DC	V_{IL}	-0.3	–	V_{REF} - 0.125	V	
Input reference voltage	V_{REF}	0.49_{*VDDQ}	0.50_{*VDDQ}	0.51_{*VDDQ}	V	
Termination voltage	V_{TT}	V_{REF} - 0.04	V_{REF}	V_{REF} + 0.04	V	

Notes:

1. Test Conditions (unless otherwise stated):
V_{CORE} = +0.98 ± 0.03 VDC
V_{DDR2} = +1.8 ± 0.09 VDC
V_{DDO_EXP} = +3.3 ± 0.165 VDC
$V_{DDO\_ETH0,\ ETH1}$ = +3.3 ± 0.165 VDC when +3.3V Ethernet transceiver is used
$V_{DDO\_ETH0,\ ETH1}$ = +2.5 ± 0.125 VDC when +2.5V Ethernet transceiver is used
$V_{IN\ (MAX)}$= +3.6V
T_j = -40°C to 125°C

2. Current flow out of the device is shown as minus

20.4 Reference Clock

This applies to the REFCLK input. Sections on specific interfaces specify output reference clocks.

Figure 20-1 Reference Clock Waveform Requirements

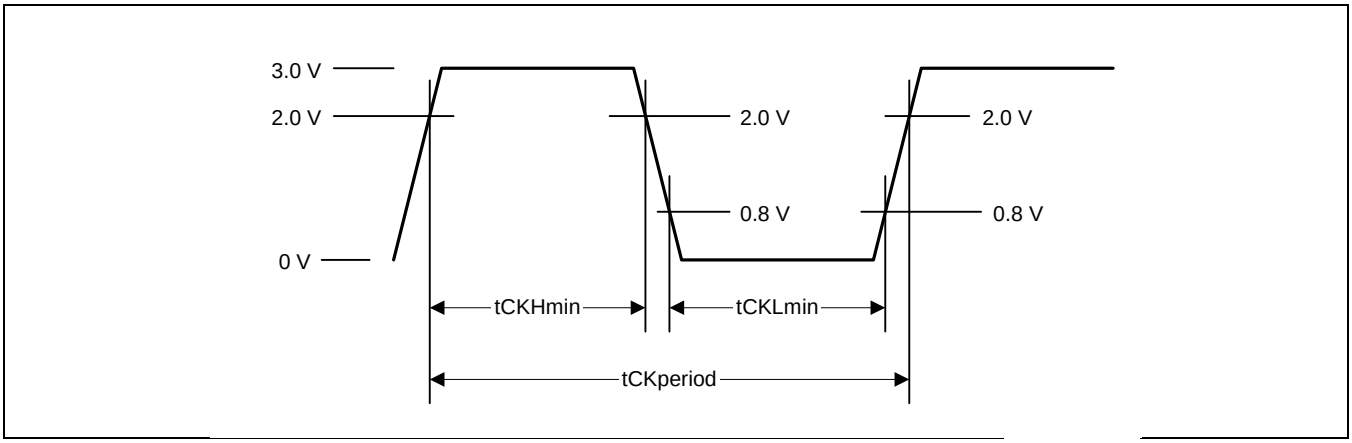


Table 20-7 Reference Clock Timing Requirements

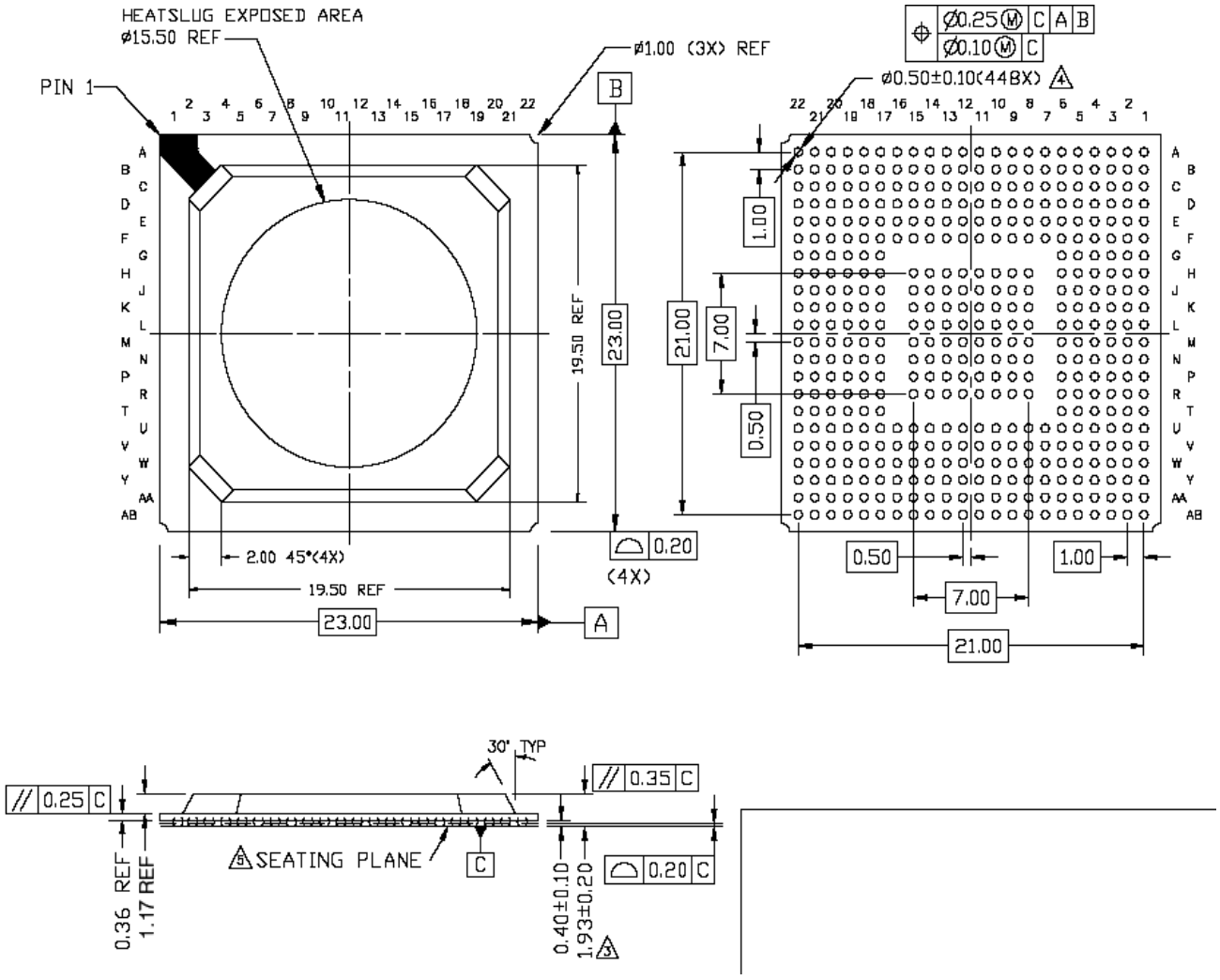
Parameter	Symbol	Min.	Max.	Units
Clock Period	tCKperiod	41.6646	41.6686	ns
Duty Cycle		40	60	Percentage high
Clock Frequency = 1/tCK period = 24 MHz ± 0.005% (50 ppm). Jitter tolerance: the value (in ps) for the maximum tolerated period-jitter for the PLL_REFCLK input is less than 50 ps peak to peak.				

NOTE:	This clock is specified as 50 ppm to support output of RMII and GMII clock signals for which the standard specification is 50 ppm. For designs not requiring that the LS10xMAx device output clocks this precise, 100 ppm applies.

21 Package Dimensions

This section provides details on the LS10xMA Package dimensions. Package dimensions are shown in Figure 21-1. The drawing shows the 23mm x 23mm x 1.93mm package with 448 balls spaced at a 1.0mm pitch.

Figure 21-1 Package Dimension Top



NOTES:

1. Dimensioning and tolerancing per ASME Y14.5M-1994
2. Solder ball position designation per JSED 95-1, SPP-010.
3. This dimension includes stand-off height, package body thickness, and lid height but does not include attached features, such as external heatsink or chip capacitors. An integral heatslug is not considered an attached feature.
4. Dimension is measured at the maximum solder ball diameter, parallel to primary datum C.
5. Primary datum C and seating plane are defined by the spherical crowns of the solder balls.
6. All dimensions are in millimeters.

22 Ordering Information

This section gives the LS10xMA ordering information.

Table 22-1 lists the legacy part numbers for existing designs. New designs should use LS10xMA part numbers as listed in Table 22-2.

Table 22-1 Legacy M83XX Part Information

Production Part Number	VoIP	Security	Ext. temp	CPU freq
LS10xMA single core 533 MHz				
M83150G13	No	No	No	533 MHz
M83151G13	Yes	No	No	533 MHz
M83152G13	No	Yes	No	533 MHz
M83153G13	Yes	Yes	No	533 MHz
M83159G13	Yes	Yes	Yes	533 MHz
LS10xMA single core 650 MHz				
M83160G13	No	No	No	650 MHz
M83161G13	Yes	No	No	650 MHz
M83162G13	No	Yes	No	650 MHz
M83163G13	Yes	Yes	No	650 MHz
LS10xMA dual core 450 MHz				
M83240G13	No	No	No	450 MHz
M83241G13	Yes	No	No	450 MHz
M83242G13	No	Yes	No	450 MHz
M83243G13	Yes	Yes	No	450 MHz
LS10xMA dual core 650 MHz				
M83260G13	No	No	No	650 MHz
M83261G13	Yes	No	No	650 MHz
M83262G13	No	Yes	No	650 MHz
M83263G13	Yes	Yes	No	650 MHz

Table 22-2 LS10xMA Family Ordering Information

Freescale QorIQ Part Number	VoIP	Security	Ext. temp	Legacy Part Number	CPU freq
LS10xMA single core 533 MHz					
LS101MASN7DFA	No	No	No	M83150G13	533 MHz
LS101MASE7DFA	Yes	Yes	No	M83153G13	533 MHz
LS101MAXE7DFA	Yes	Yes	Yes	M83159G13	533 MHz
LS10xMA single core 650 MHz					
LS101MASN7EHA	No	No	No	M83160G13	650 MHz

LS101MASE7EHA	Yes	Yes	No	M83163G13	650 MHz
LS10xMA dual core 450 MHz					
LS102MASN7BFA	No	No	No	M83240G13	450 MHz
LS102MASE7BFA	Yes	Yes	No	M83243G13	450 MHz
LS10xMA dual core 650 MHz					
LS102MASN7EHA	No	No	No	M83260G13	650 MHz
LS102MASE7EHA	Yes	Yes	No	M83263G13	650 MHz

23 Revision History

The following table shows the revision history of this document..

Table 23-1 Document revision history

Revision	Date	Substantive Changes
Rev 1	10/2014	In the Table 1-1 Feature By Device Number, corrected the device number from LS101MASX7DFA to LS101MAXE7DFA.
Rev 0	09/2014	First public revision.

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number: LS10xMA
Rev. 1, 10/2014