CY15FRAMKIT-002

Excelon™ Ultra QSPI F-RAM Development Kit User Guide

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
www.cypress.com

Contents

1. Introduction

Thank you for your interest in the CY15FRAMKIT-002 FRA-M Kit (DVK). The kit (shield) is designed as an easy-to-use and inexpensive development kit as well as evaluation platform for Cypress's latest Quad SPI enabled Serial F-RAM. This kit works in conjunction with the NUCLEO-L433RC-P MCU Evaluation Board, a starter kit for ARM® Cortex®-M4-based ST Microelectronics devices. You will need both kits to demonstrate the operation described in this guide. This board features a 4-Mbit Quad SPI F-RAM, associated circuit, Arduino as well as Morpho connectivity. This kit supports 3.3-V or 1.8V power supply.

1.1 Kit Contents

The CY15FRAMKIT-002 Kit includes following contents,

- CY15FRAMKIT-002 DVK board with CY15B104QSN-108SXI, 108-MHz 4-Mbit Quad SPI F-RAM
- Quick Start Guide

1.2 Getting Started

This guide helps you to get acquainted with the CY15FRAMKIT-002 DVK. The Kit Installation chapter on page 10 describes the software required to utilize the driver set provided for this kit. The Kit Overview chapter on page 14 explains the features of the kit. The Kit Operation chapter on page 20 explains how to program and run the kit. The Kit Software chapter on page 32 explains the GUI to test out the QSPI F-RAM features. The Kit Example Firmware chapter on page 41 explains the F-RAM access APIs and a link to download the test project. The Appendix on page 23 provides the kit block diagram, schematics, pin assignment, and the bill of materials (BOM)

1.3 Pre-requisite software

- Keil uVision 5 software (for development and programming)
- STSW-LINK009 - ST-LINK, ST-LINK/V2, ST-LINK/V2-1 USB driver signed for Windows7, Windows8, Windows10
- STSW-LINK007 - ST-LINK, ST-LINK/V2, ST-LINK/V2-1 firmware upgrade

1.4 Pre-requisite Hardware

- NUCLEO-L433RC-P: STM32 Nucleo64 Evaluation Board (details)
- CY15FRAMKIT-002 DVK board

1.5 Additional Resources

- Details about the ST Micro-electronics Kit can be found here
- Details about Keil µVision installation can be found here (Download MDK-Arm tool chain)
- The test project and driver files are part of the compressed folder downloaded from Cypress community
- The datasheet for Cypress Quad SPI F-RAM mounted on the kit can be downloaded from here

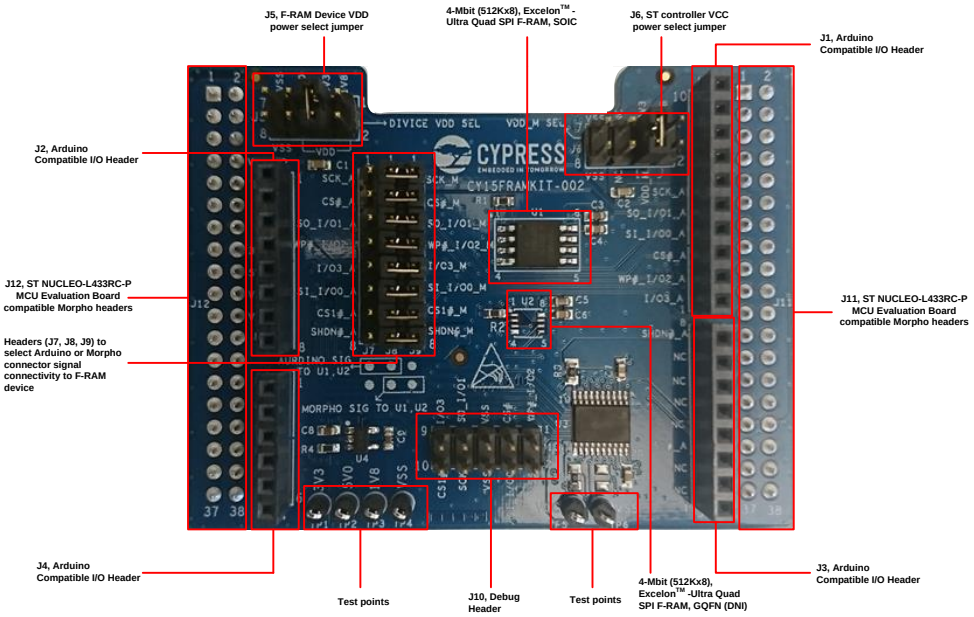
2. Kit Overview

2.1 CY15FRAMKIT-002 Kit Overview

The CY15FRAMKIT-002 can be used to understand the features of the serial F-RAMs (SPI and QSPI). The DVK is an add-on board, which contains a 4-Mbit Excelon™ -Ultra QSPI F-RAM. It has four connectors that are Arduino UNO-compatible and connect to either ST NUCLEO-L433RC-P MCU Evaluation Board or Arduino UNO R3 board for SPI evaluation. It also provides two 19x2 Morpho headers (J11 and J12) compatible with an ST NUCLEO-L433RC-P MCU Evaluation Board for QSPI evaluation. The kit operates using a 1.8-V/3.3-V power supply from the baseboard. The CY15FRAMKIT-002 Kit consists of the following blocks as shown in Figure 1.

- 4-Mbit (512Kx8), Excelon™ -Ultra Quad SPI F-RAM
- ST NUCLEO-L433RC-P MCU Evaluation Board compatible Morpho headers (J11, J12)
- Arduino Compatible I/O Header (J1, J2, J3, J4)
- Headers (J7, J8, J9) to select Arduino or Morpho connector signal connectivity to F-RAM device
- J6 header for 3.3-V/5.0-V F-RAM device VCC power supply selection
- J5 header for 1.8-V/3.3-V/5.0-V ST controller VCC power supply
- J10 debug header for probing the SPI/QSPI signals
- Test points for supported voltage (3V3/5V0/1V8) and ground signals (VSS)
- Do Not Install (DNI) footprint for the Excelon™ -Ultra Quad SPI F-RAM in 8-pin Grid-Array Quad Flat No-Lead
- GQFN package

Figure 1. CY15FRAMKIT-002 Board Markup



2.2 Kit Introduction and Configuration Guide

The following sections describe various aspects of the kit hardware and the kit setup.

2.2.1 F-RAM Device Power Supply Jumper

The CY15FRAMKIT-002 Board hardware operates at 1.8 V/3.3 V, selected through header J5, as shown in Figure 2. The factory default jumper setting is 3.3 V (short pins 3 and 4 of J5). User can short the pins 1 and 2 of J5 for 1.8 V selection.

CAUTION

- Do not power the CY15FRAMKIT-002 board through an external power source. The board is designed to be powered by the ST NUCLEO-L433RC-P MCU Evaluation base board.
- The CY15FRAMKIT-002 operates from 1.8 V to 3.6 V. Exceeding the maximum voltage limit (3.6 V) can damage the board.

Figure 2. F-RAM Device VDD Select Jumper

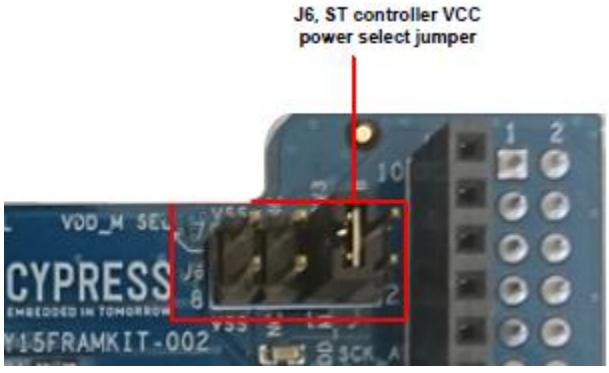
2.2.2 ST Controller Device Power Supply Jumper

The ST NUCLEO-L433RC-P MCU Evaluation Board hardware operates at 1.8 V/3.3 V, selected through header J6, as shown in Figure 3. The factory default jumper setting is 3.3 V (short pins 3 and 4 of J6). User can short the pins 1 and 2 of J6 for 1.8 V selection.

CAUTION

- Do not power the ST NUCLEO-L433RC-P MCU Evaluation board through an external power source. The board is designed to be powered by the Regulator output on the board.
- The ST NUCLEO-L433RC-P MCU Evaluation operates from 1.8 V to 3.6 V. Exceeding the maximum voltage limit (3.6 V) can damage the board.

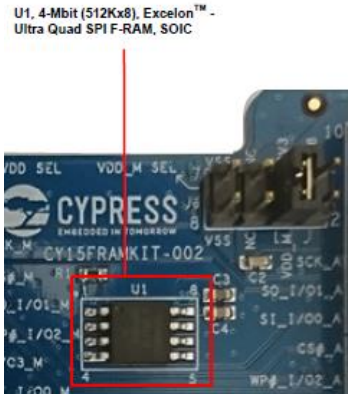
Figure 3. ST Controller Device VDD_M Select Jumper



2.2.3 Excelon™ -Ultra Quad SPI F-RAM

Figure 4 shows the 4-Mbit (512Kx8), 3.3 V, Excelon™ -Ultra Quad SPI F-RAM part in the 8-pin SOIC package option.

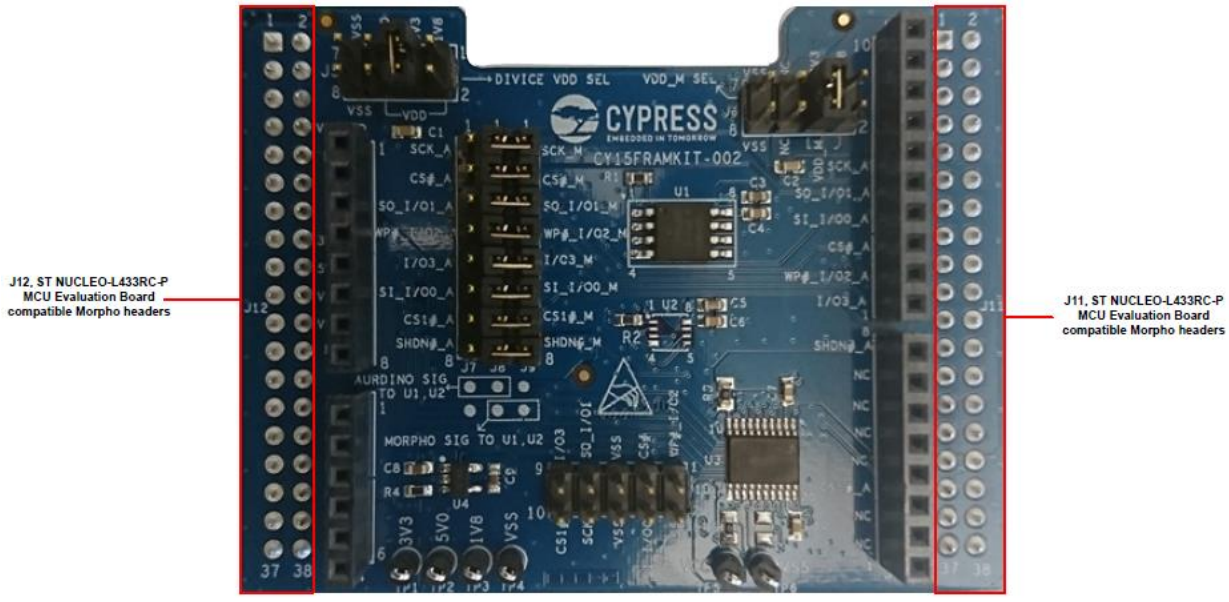
Figure 4. Cypress Serial Quad SPI F-RAM Device on the Board



2.2.4 Morpho Headers to ST NUCLEO-L433RC-P MCU Evaluation Board

Figure 5 shows the Morpho headers to the ST NUCLEO-L433RC-P MCU Evaluation Board for QSPI interface evaluation – J11 and J12. You can plug the CY15FRAMKIT-002 board onto the ST NUCLEO-L433RC-P MCU Evaluation Board through these connectors. For the schematic, refer to the Appendix on page 23.

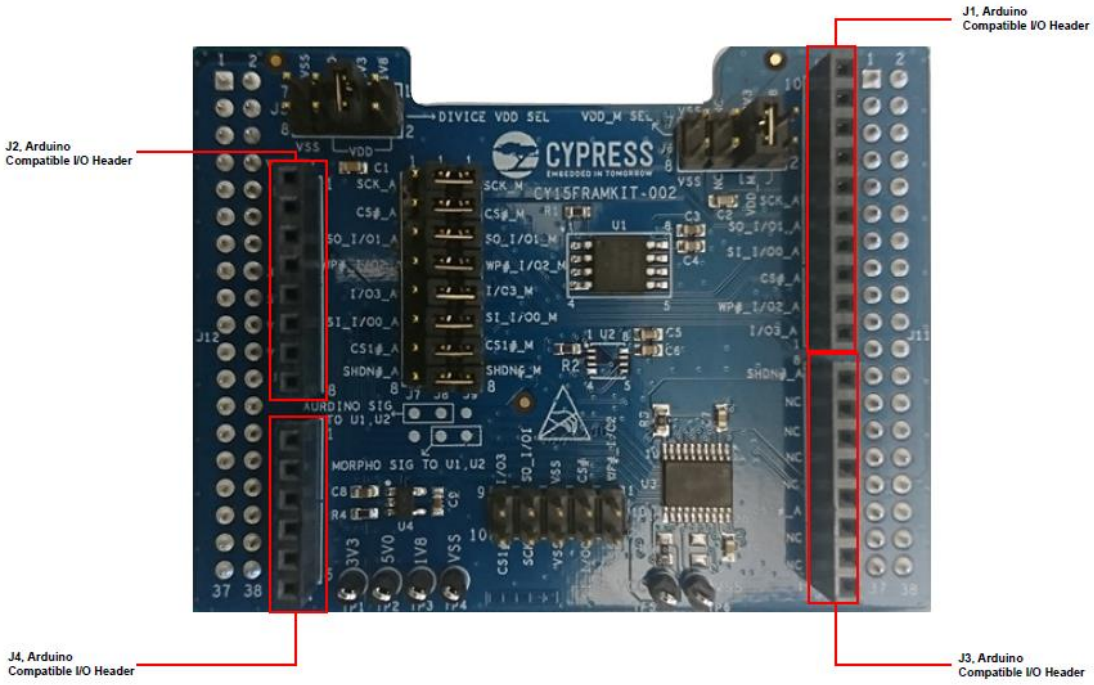
Figure 5. Morpho Headers to ST NUCLEO-L433RC-P MCU Evaluation Board



2.2.5 Arduino Headers to ST NUCLEO-L433RC-P MCU Evaluation Board

Figure 6 shows the Arduino headers to the ST NUCLEO-L433RC-P MCU Evaluation Board for SPI Interface evaluation – J1, J2, J3 and J4. You can plug the CY15FRAMKIT-002 board onto the ST NUCLEO-L433RC-P MCU Evaluation Board through these connectors. For the schematic, refer to the Appendix on page 23.

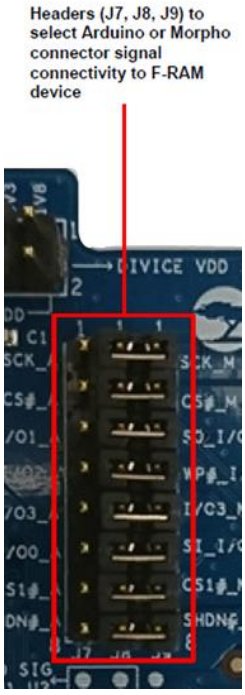
Figure 6. Arduino Headers to ST NUCLEO-L433RC-P MCU Evaluation Board



2.2.6 Headers to select Arduino or Morpho connector signal connectivity to F-RAM device

Figure 7 shows the headers – J7, J8 and J9. User can short F-RAM device signals to Arduino connector by shorting J7 and J8 or to Morpho connector by shorting J9 and J8 header through jumper. For the schematic, refer to the Appendix on page 23.

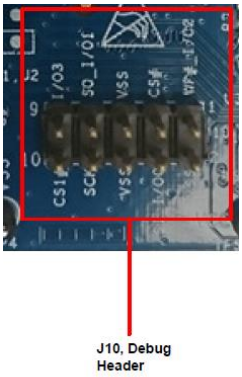
Figure 7. Headers to select Arduino or Morpho connector signals



2.2.7 Debug Header

Debug header is provided for easy access to the Quad SPI F-RAM communication pins. As shown in Figure 8, header J10 provides access to the SPI F-RAM device communication signals. For the schematic, refer to the Appendix on page 23.

Figure 8. Debug Header



2.2.8 Test Points

The CY15FRAMKIT-002 board provides 3V3, 5V0, 1V8, VSS, and VCC test points as shown in Figure 9. Test Points. These test points are loaded by default.

Figure 9. Test Points

Table 1. Test Points

Test Points	Description
3V3	Test point for the kit board power supply
5V0	Test point for the kit board power supply
1V8	Test point for the kit board power supply
VSS	Test point for the kit board ground
VCC	Test point for the F-RAM device (U1, U2) power supply

CAUTION: The VCC test point is only for probing and measurement purposes. Do not power the kit using this test point to avoid any damage to the board.

2.2.9 8-pin Grid-Array Quad Flat No-Lead GQFN package – Not Populated by Default

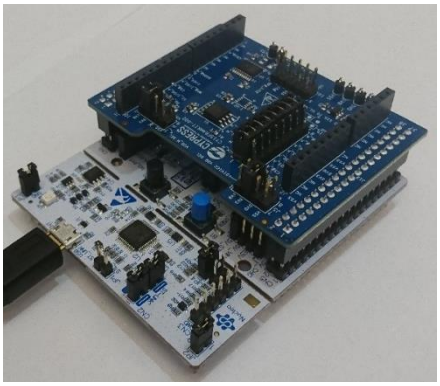
The CY15FRAMKIT-002 board provides a footprint option for the Excelon™ -Ultra Quad SPI F-RAM in 8-pin GQFN package. You can mount a 4-Mbit (512Kx8), Excelon™ -Ultra Quad SPI F-RAM device to evaluate the GQFN package in addition to the default 8-pin SOIC option on the CY15FRAMKIT-002 board. An independent chip select control is provided for the 8-pin GQFN package, which enable access to the second device on the board with an appropriate firmware modification.

Figure 10. 8-pin GQFN – Not Populated by Default

2.3 F-RAM and MCU Kit Connection

The ST MCU Evaluation Board is plugged onto the CY15FRAMKIT-002 through Arduino and Morpho connectors: Arduino (J1, J2, J3 and J4) and Morpho connectors (J11, J12), as shown in Figure 11. CY15FRAMKIT-002 mounted on ST MCU Evaluation Board.

Figure 11. CY15FRAMKIT-002 mounted on ST MCU Evaluation Board



CAUTION: The ST MCU Evaluation Board is plugged onto the CY15FRAMKIT-002 through its compatible headers J1, J2, J3, J4, J11 and J12. Because J11 and J12 are high pin-count connectors on the ST4 MCU Evaluation Board, removing the ST MCU Evaluation Board can be difficult and may lead to bending its J1, J2, J3, J4, J11 and J12 connector pins. Therefore, take care when removing the ST MCU Evaluation Board from the setup. Sliding the ST MCU Evaluation Board out of headers slowly and evenly will reduce the risk of bending connector pins.

3. Kit Operation

This section describes the setup needed to program the base board (NUCLEO-L433RC-P) to access the Quad SPI F-RAM on CY15FRAMKIT-002 DVK

3.1 Programming the NUCLEO-433LC-P evaluation board

The NUCLEO-L433RC-P board is STM32 development board based on a Nucleo-64 platform. The board offers several features including an Arduino and Morpho connectivity. The controller is supported in a wide variety of Integrated Development Environments (IDEs) including IAR™, Keil® and other GCC based IDEs for ARM™ architectures. Details about this platform can be found here

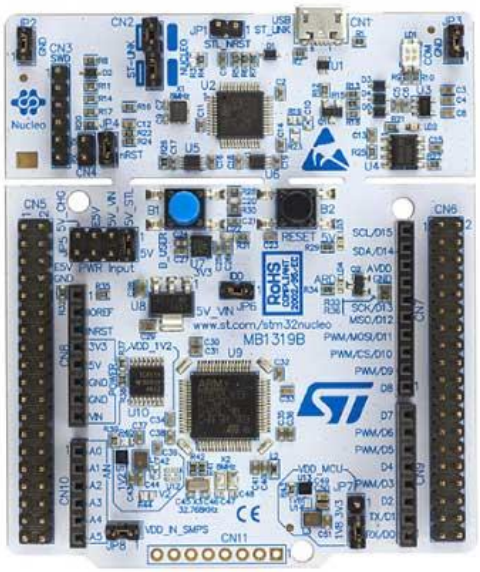


Figure 12: NUCLEO-L433RC-P Evaluation Board

The STM32L433RC MCU on the board supports an industry standard, high speed Quad SPI interface. This interface is routed on to the Morpho connectors of the NUCLEO board. The drivers and sample project has been written with STM32L433RC controller as target but can be easily modified to be compatible with any other ST microcontroller with Quad SPI interface. In firmware section of this document, appropriate sections will be highlighted to ensure compatibility with other ST microcontrollers.

3.1.1 System Requirements

Before starting the user should:

1. Install the MDK-ARM v5.17 or later Integrated Development Environment (IDE) from Keil
2. ST-LINK/V2-1 driver will be installed automatically. In case of problem, the user can proceed with manual installation of the driver, from toolchains install directory
3. Download the STM32 Nucleo firmware from the www.st.com/stm32nucleo webpage.
4. Establish the connection with the STM32 Nucleo board, by connecting CN1 of the Nucleo board to the USB port of the PC
5. Please refer to the guide here to resolve issues related to installation of Nucleo board drivers

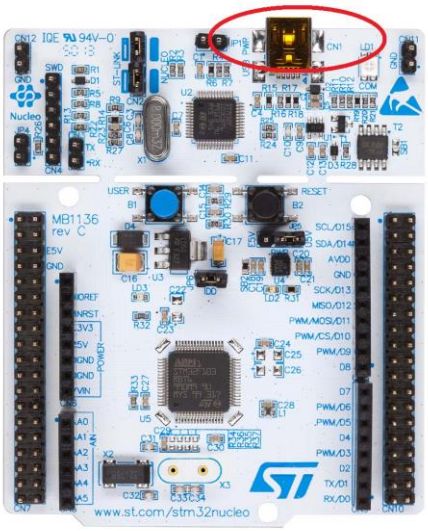


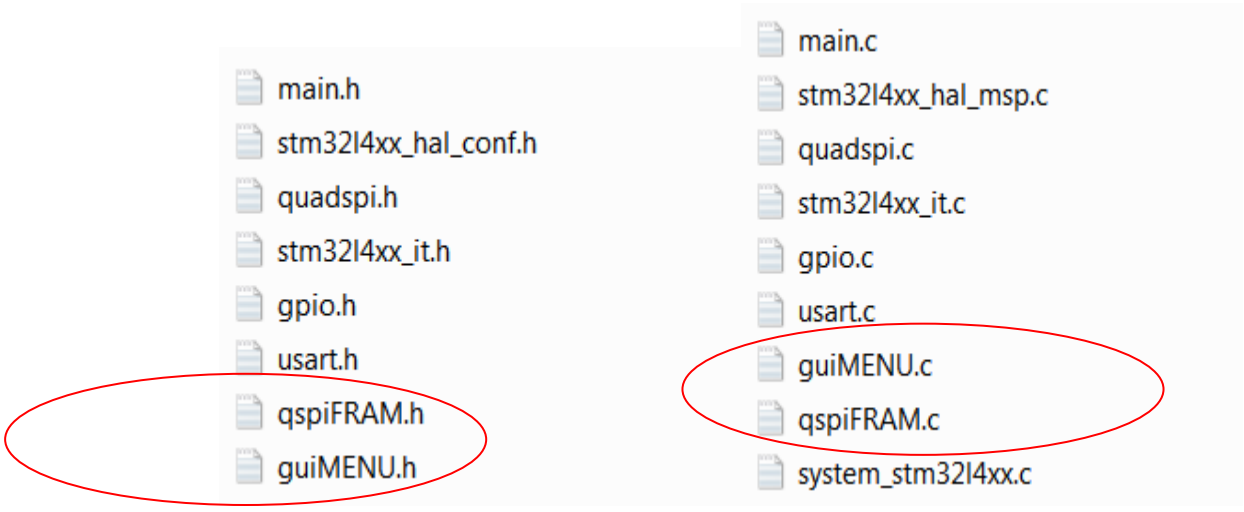
Figure 13: Connector CN1 for Nucleo 64 board

3.1.2 Compiling Test Project

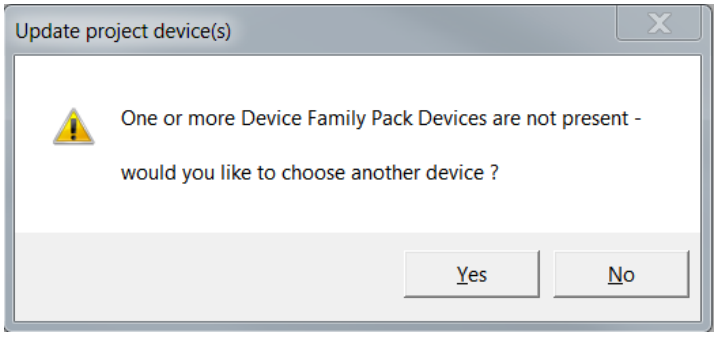
This user guide is accompanied by a test project (Excelon_QSPI.zip) that can be quickly compiled and programmed on to the Nucleo board to evaluate the Quad SPI F-RAM features

The folder Excelon_QSPI contains a MDK-ARM project along with the required source/include files.

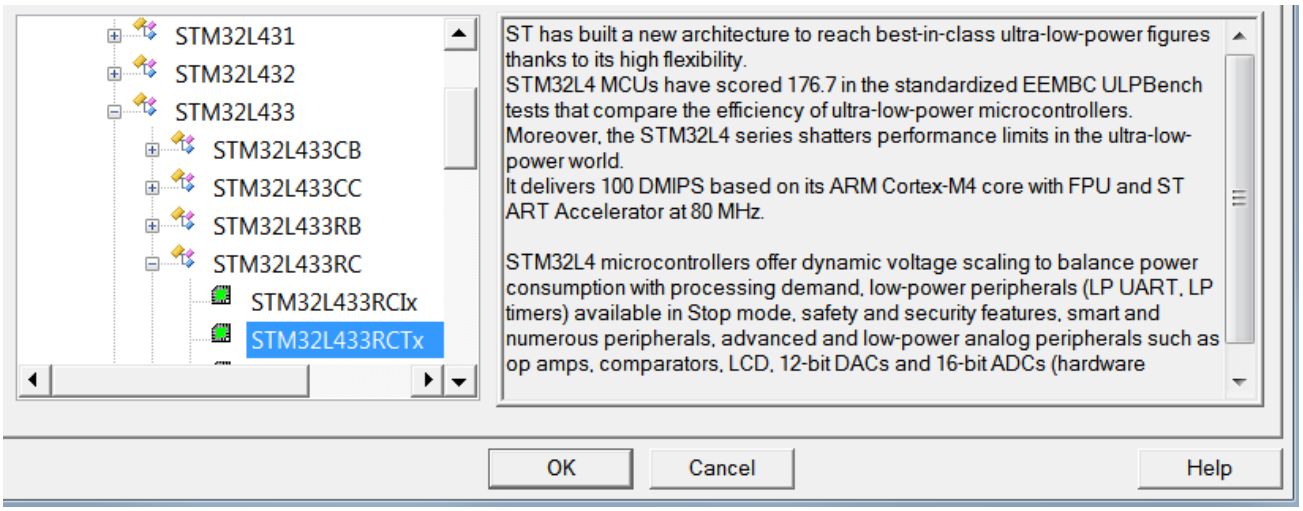
The Excelon_QSPI/Inc and Excelon_QSPI/Src folders contains header files critical for initializing the QSPI and LPUART modules of the ST controller. The function declarations for accessing Cypress's QSPI F-RAM are also provided in this folder



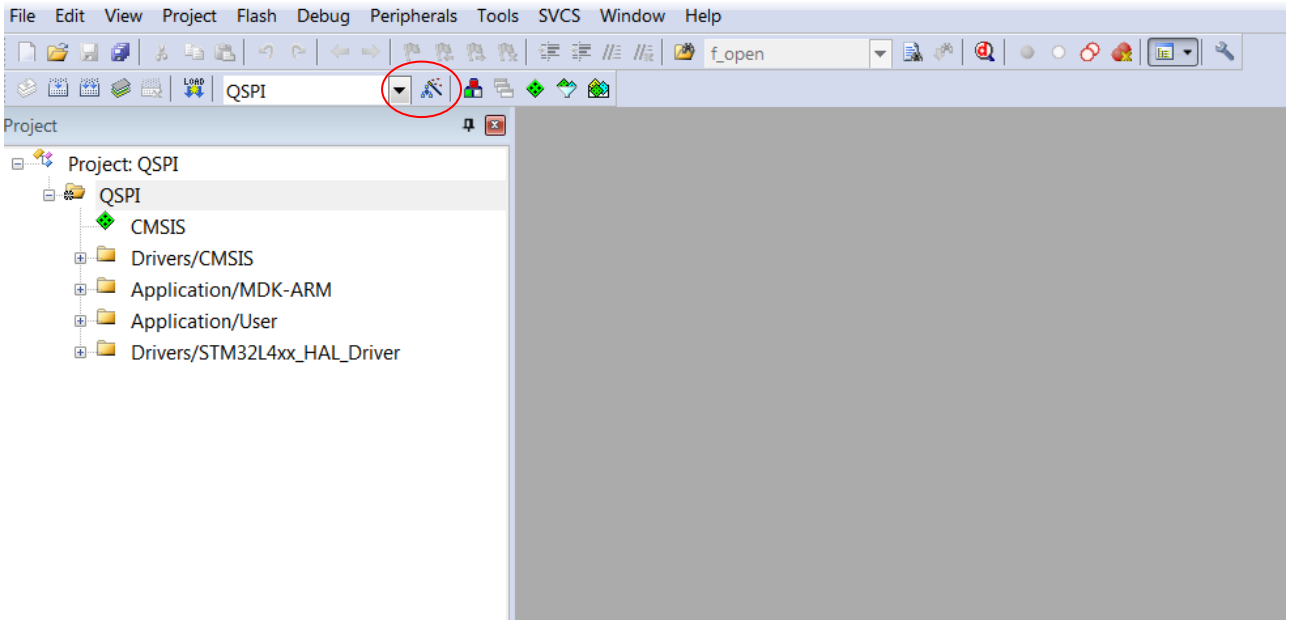
1. The files qspiFRAM(.h/.c) provide all the API accesses to the QSPI F-RAM device. User should include these (with appropriate code modification) in their project to access the QSPI F-RAM
2. The files guiMENU(.h/.c) are provided to allow evaluation of the F-RAM by sending commands over UART terminal (115200 bps, 8-NoParity-1). These files need not be included in the final project
3. Open the uVision project in MDK-ARM folder. If the installer looks for Board support package, press ESC key (There is no native package for NUCLEO-L433RC-P board in Keil library)



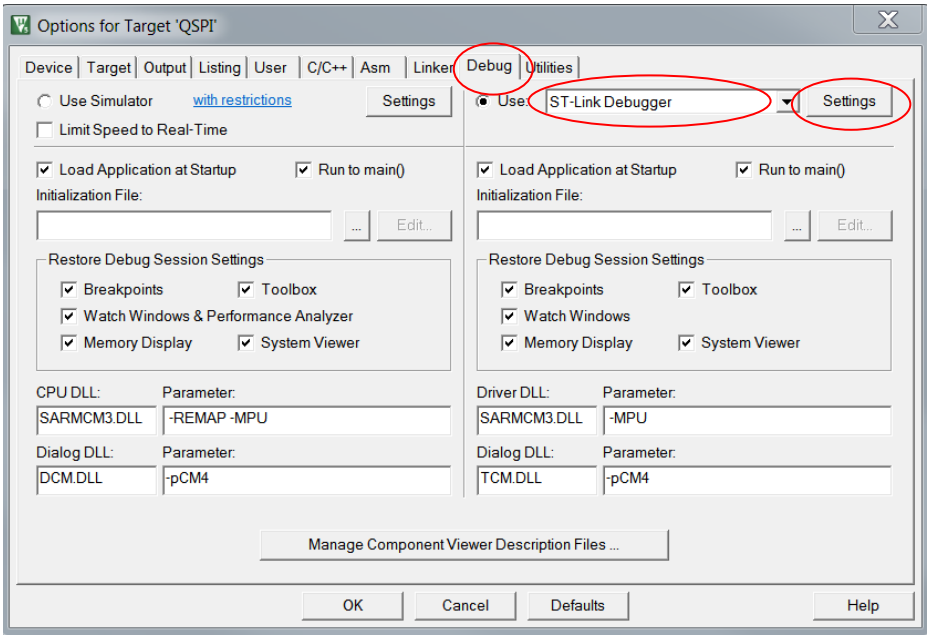
4. Click "Yes" on seeing this message and browse to the correct device (STM32L433RCTx) shown below



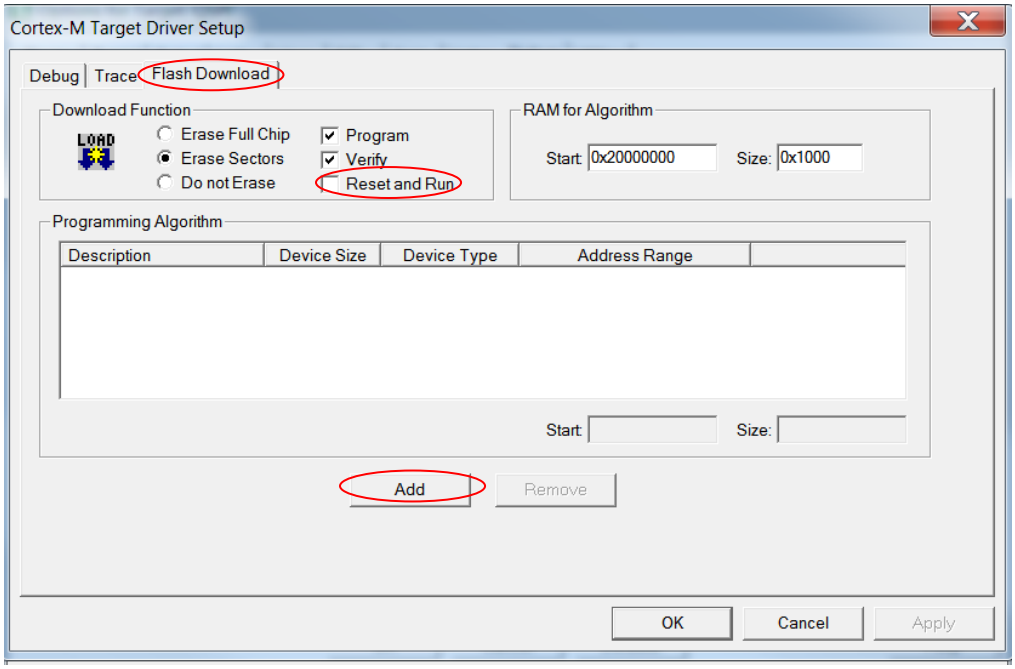
5. Your workspace should have following structure at this stage



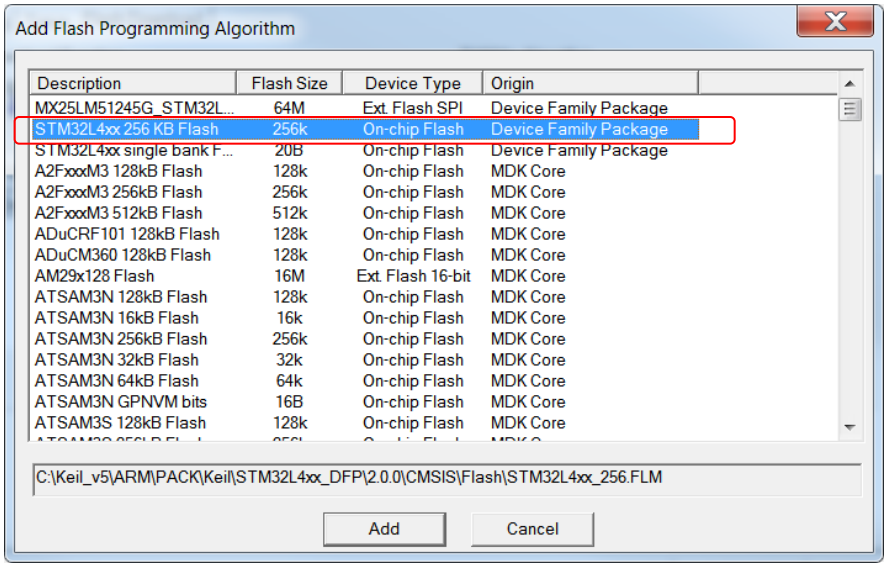
6. Before compiling, appropriate properties for target must be selected. Press “ALT+F7” or browse to “Options for Target” in “Project” menu
7. Update the clock speed to 80.00MHz in Target - Xtal (MHz)
8. On “Debug” page, ST-Link Debugger should get populated automatically if the Nucleo board is plugged into USB port of the computer and the drivers are installed correctly



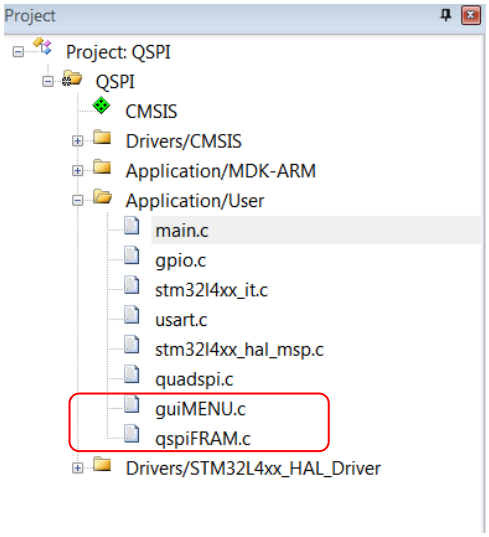
9. Click “Debugger – Settings” to select the appropriate Flash Programming Algorithm for this device



10. By default, there will be no Programming Algorithm on first time selection of this controller.
11. Click Add button to open a list of available algorithms.
12. The STM32L433RC MCU has 256Kbytes on On-Chip Flash. Select the appropriate programming algorithm



13. Click Add. Also “Check” the “Reset and Run” option on Target Driver Setup window to ensure that the device resets on successful program
14. Add qspiFRAM.c file in Application/User space. Also add guiFRAM.c file if you are evaluating the device features on HyperTerminal



15. Press F7 to build the project. There should be no errors or warning at this stage

```
linking...
Program Size: Code=23476 RO-data=1024 RW-data=20 ZI-data=9476
"QSPI\QSPI.axf" - 0 Error(s), 0 Warning(s).
Build Time Elapsed:  00:00:21
```

16. Press F8 to program the controller with default test program. Open a Hyper terminal program of your choice (puTTY, uCON etc) and establish a UART connection at 115200 bps with 8-bits message, no parity and 1 stop bit.
17. The controller will be waiting for appropriate GUI command. GUI commands will be explained in Appendix on page 32.

3.2 Quad SPI Configuration for STM32L433RC

The Arduino/Morpho QSPI F-RAM Kit is designed interface with both the SPI port on Arduino interface as well as the Quad SPI port on the Morpho interface. With help of jumper, user can select either of the interface. The drivers/API provided with test project are for initializing the Quad SPI port of controller only.

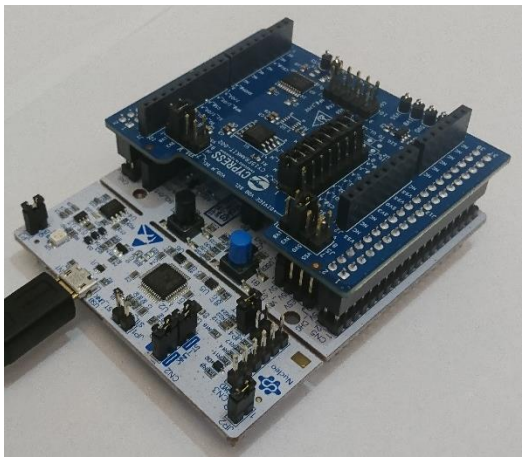


Figure 14: Complete Setup

3.2.1 Quad SPI initialization

The initialization of Quad SPI block of ST controller is done by calling QUADSPI_Init() function. The function is defined in “quadspi.c” but can be added to user’s main application file for smaller project. QUADSPI_Init() function initializes the “*hqspi*” structure.

```
hqspi.Instance = QUADSPI;
hqspi.Init.ClockPrescaler = 32;
hqspi.Init.FifoThreshold = 4;
hqspi.Init.SampleShifting = QSPI_SAMPLE_SHIFTING_NONE;
hqspi.Init.FlashSize = 24;
hqspi.Init.ChipSelectHighTime = QSPI_CS_HIGH_TIME_1_CYCLE;
hqspi.Init.ClockMode = QSPI_CLOCK_MODE_0;
hqspi.Init.FlashID = QSPI_FLASH_ID_1;
hqspi.Init.DualFlash = QSPI_DUALFLASH_DISABLE;
```

For details on Quad SPI interface of this controller, refer to AN4760 from STMicroelectronics. The critical variables for evaluating QSPI F-RAM are “ClockPrescaler” and “FlashSize”

ClockPrescaler = The default value in test project is 32. It corresponds to approximately 5MHz of clock frequency on QSPI interface. Users need to modify this variable for achieving the desired clock frequency. The Controller on the Nucleo board supports a maximum of 48MHz on Quad SPI Clock, while the Morpho connecter is expected to support up-to 30MHz.

Note: The limitation of clock speed is due to the low speed connectors used for design of Nucleo board as well as DVK kit. The Cypress F-RAM is capable of running at up to 108-MHz in Quad SPI mode.

“**FlashSize**” = The default value in test project is 24. The value corresponds to total number of address bits needed to address the memory attached to the interface. The Cypress F-RAM on the kit is a 4Mbit device but has 3-byte addressing. The default value of 24 can remain unchanged in user’s application

3.2.2 Quad SPI interface on STM32L433RC

The Quad SPI interface for this controller is routed to following GPIOs

Sr. No.	Quad SPI Interface Pin	STM32L433RC Pin
1	QUADSPI_CS#	PB11
2	QUADSPI_CLK	PB10
3	QUADSPI_IO0	PB1
4	QUADSPI_IO1	PB0
5	QUADSPI_IO2	PA7
6	QUADSPI_IO3	PA6

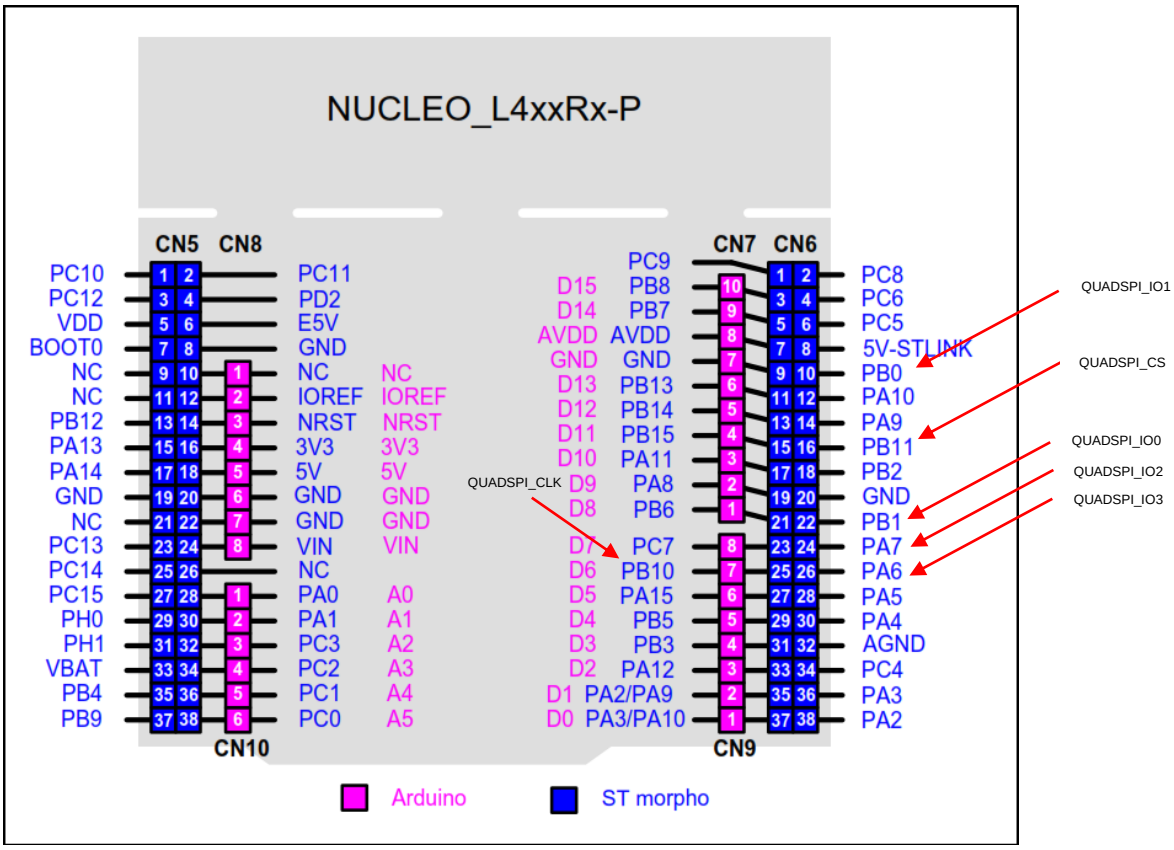


Figure 15: Quad SPI Pin Assignment on Morpho Connector

The CY15FRAMKIT-002 board plugs into the Nucleo board. Both Arduino and Morpho connectors need to be plugged in. The QSPI_Init() function also ensures initialization of the controller pins as Quad SPI pins.

```
/**QUADSPI GPIO Configuration
PA6     ------> QUADSPI_BK1_IO3
PA7     ------> QUADSPI_BK1_IO2
PB0     ------> QUADSPI_BK1_IO1
PB1     ------> QUADSPI_BK1_IO0
PB10     ------> QUADSPI_CLK
PB11     ------> QUADSPI_BK1_NCS
*/
GPIO_InitStruct.Pin = GPIO_PIN_6|GPIO_PIN_7;
GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;
GPIO_InitStruct.Pull = GPIO_PULLUP;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_VERY_HIGH;
GPIO_InitStruct.Alternate = GPIO_AF10_QUADSPI;
HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);

GPIO_InitStruct.Pin = GPIO_PIN_0|GPIO_PIN_1|GPIO_PIN_10|GPIO_PIN_11;
GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;
GPIO_InitStruct.Pull = GPIO_NOPULL;
GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_VERY_HIGH;
GPIO_InitStruct.Alternate = GPIO_AF10_QUADSPI;
HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);
```

Figure 16: Pin Initialization as Quad SPI

4. Firmware Details

This section summarizes the APIs written in qspiFRAM (.h/.c) file along with the uVision Example project provided by Cypress Semiconductors. Each function has been supplemented by user comments to help understand the usage. Revision 1.0 of this release covers limited features of the device. The detailed feature set of Cypress' Quad SPI F-RAM can be found in the device datasheet. The APIs listed here are for enabling easy usage of the Quad SPI F-RAM features and are not an official release of driver support from Cypress Semiconductors. Users are encouraged to leverage these APIs for building their end applications.

The Example project is stored in Excelon_QSPI folder and has following structure:

Project Files	Description
\Excelon_QSPI\MDK-ARM	Example project in uVISION 5
\Excelon_QSPI\Inc	Include files for Example project.
\Excelon_QSPI\Src	Source files for Example project
\Excelon_QSPI\Drivers	HAL Drivers for STM32

The QSPI F-RAM APIs are declared in \Excelon_QSPI\Inc\qspiFRAM.h file and declared in \Excelon_QSPI\Src\qspiFRAM.c. Following table provides summary of all the supported APIs for Revision 1.0 of this release. Unless specified explicitly the description is applicable only for the Quad SPI F-RAM device on the kit.

Sr. No	API Name	Description	Notes
1	bool FRAM_Interface_Reset (void)	This API resets the operating mode of the F-RAM device to SPI. All registers are reset to their default values	The F-RAM device can be configured in several operating modes (SPI, DPI or QPI) and can have several register settings. This API is written specifically to implement a "Go Home" feature in case the controller faces a miss-match of operating mode. For eg: during a sudden power cycle, the F-RAM device will retain its state but the controller will execute a power-up routine and will not be able to communicate with the F-RAM device. It is recommended to implement similar function in end application
2	void Read_Device_Status(void)	This API will read all the register values of the F-RAM device and stores it in run time structure variable	Assumes that Controller is in a known operating mode with respect to F-RAM
3	bool Read_ID(uint8_t IOMode, uint8_t Reg_lat)	This API reads the ID register of the device and compares it with default value 0x50518206000000	This function is utilized internally by FRAM_Interface_Reset (void) function. Read_ID () API returns a success or a fail status based on whether the ID read from device matches the default value. End user can leverage this API to identify the device operating mode (SPI, DPI or QPI)
4	Void FRAM_WREN(void)	Issues WREN (0x06) opcode to the device	
5	Void FRAM_WRDI (void)	Issues WRDI (0x04) opcode to the device	
6	void FRAM_RDSR1 (uint8_t *StatusReg); void FRAM_RDSR2 (uint8_t *StatusReg); void FRAM_RDCR1 (uint8_t *ConfigReg); void FRAM_RDCR2 (uint8_t *ConfigReg); void FRAM_RDCR4 (uint8_t *ConfigReg); void FRAM_RDCR5 (uint8_t *ConfigReg);	These APIs can be used to read the status/Configuration registers of the F-RAM and store the register value in the pointer argument	The APIs assume that the F-RAM is in a known operating mode
7	void FRAM_RDSN (void); void FRAM_WRSN (uint8_t *SNreg);	These APIs can be used to read or write Serial Number Register	SN Register is an 8-byte register. Read function will read the device's serial number and store it in SN_Reg array of operating_mode structure

Sr. No	API Name	Description	Notes
			The argument passed to Write function must be a pointer to an 8-byte array containing value of new Serial number
8	bool FRAM_Write(uint32_t address, uint8_t *write_data, uint32_t write_length);	This API performs write operation (Opcode 0x02) to the device	Device must be write enabled (FRAM_WREN()) prior to writing into the device. This API performs write of length “write_length” bytes stored in array pointed by “*write_data”, starting at location “address” in the F-RAM. The API assumes that the controller is in a known operating mode (SPI, DPI or QPI)
9	bool FRAM_Fast_Write(uint32_t address, uint8_t *write_data, uint32_t write_length);	This API performs Fast write operation (Opcode 0xDA) to the device	Device must be write enabled (FRAM_WREN()) prior to writing into the device. This API performs fast write of length “write_length” bytes stored in array pointed by “*write_data”, starting at location “address” in the F-RAM. The API assumes that the controller is in a known operating mode (SPI, DPI or QPI)
10	bool FRAM_DIW(uint32_t address, uint8_t *write_data, uint32_t write_length);	This API performs write operation to the device using opcode 0xA2	Device must be write enabled (FRAM_WREN()) prior to writing into the device. This API performs write of length “write_length” bytes stored in array pointed by “*write_data”, starting at location “address” in the F-RAM. The API assumes that the controller and the F-RAM device is in SPI operating mode
11	bool FRAM_DIOW(uint32_t address, uint8_t *write_data, uint32_t write_length);	This API performs write operation to the device using opcode 0xA1	
12	bool FRAM_QIW(uint32_t address, uint8_t *write_data, uint32_t write_length);	This API performs write operation to the device using opcode 0x32	Device must be write enabled (FRAM_WREN()) prior to writing into the device. This API performs write of length “write_length” bytes stored in array pointed by “*write_data”, starting at location “address” in the F-RAM. The API assumes that the controller and the F-RAM device is in SPI operating mode. QUAD bit must be set prior to executing this API
13	bool FRAM_QIOW(uint32_t address, uint8_t *write_data, uint32_t write_length);	This API performs write operation to the device using opcode 0xD2	

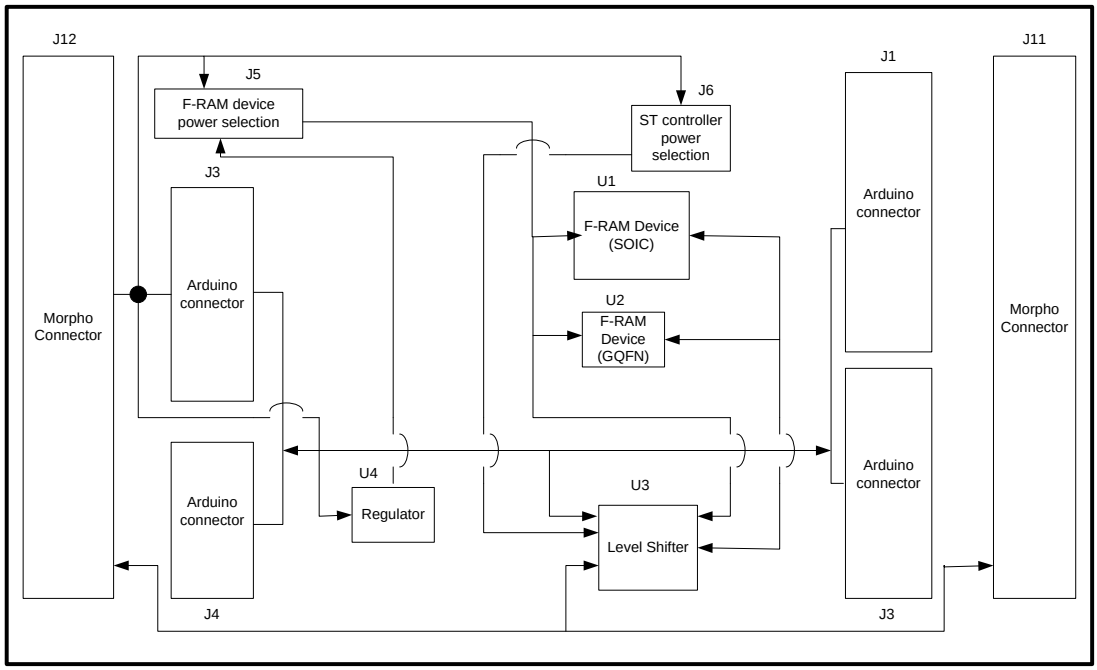
14	bool FRAM_DDRWrite(uint32_t address, uint8_t *write_data, uint32_t write_length);
	bool FRAM_DDRQIOW(uint32_t address, uint8_t *write_data, uint32_t write_length);
	bool FRAM_DDRFastWrite(uint32_t address, uint8_t *write_data, uint32_t write_length);
		This API is not tested for Revision 1.0 of this release	N/A

15	bool FRAM_Read(uint32_t address, uint8_t *read_data, uint32_t read_length);	This API performs read operation (Opcode 0x03) to the device	This API performs read of length “read_length” bytes and stores it in array pointed by “*read_data”, starting at location “address” in the controller. The API assumes that the controller is in a known operating mode (SPI, DPI or QPI)
16	bool FRAM_FastRead(uint32_t address, uint8_t *read_data, uint32_t read_length);	This API performs read operation (Opcode 0x0B) to the device	This API performs fast read of length “read_length” bytes and stores it in array pointed by “*read_data”, starting at location “address” in the controller. The API assumes that the controller is in a known operating mode (SPI, DPI or QPI)
17	bool FRAM_DORRead(uint32_t address, uint8_t *read_data, uint32_t read_length);	This API performs read operation on the device using opcode 0x3B	This API performs fast read of length “read_length” bytes and stores it in array pointed by “*read_data”, starting at location “address” in the controller. The API assumes that the controller and the F-RAM is in SPI Operating mode
18	bool FRAM_DIORRead(uint32_t address, uint8_t *read_data, uint32_t read_length);	This API performs read operation on the device using opcode 0xBB	
19	bool FRAM_QORRead(uint32_t address, uint8_t *read_data, uint32_t read_length);	This API performs read operation on the device using opcode 0x6B	This API performs fast read of length “read_length” bytes and stores it in array pointed by “*read_data”, starting at location “address” in the controller. The API assumes that the controller and the F-RAM is in SPI Operating mode The QUAD bit must be set prior to executing this API
20	bool FRAM_QIORRead(uint32_t address, uint8_t *read_data, uint32_t read_length);	This API performs read operation on the device using opcode 0xEB	
21	bool FRAM_DDRFastRead(uint32_t address, uint8_t *read_data, uint32_t read_length);	This API is not tested for Revision 1.0 of this release	N/A
22	device_mode Detect_Mode(void);	This API is not tested for Revision 1.0 of this release	N/A
23	void FRAM_WRR(uint8_t *Register, uint8_t No_of_bytes);	This API can perform writes to one or all registers in the device. Refer device datasheet for detailed understanding of the opcode	WREN command must be issued prior to executing this API. Single command be used to update Status register 1, Configuration Registers 1, 2, 4 and 5. The Argument “No_Of_bytes” determines the number of registers that get update with this API. It is recommended to avoid using this API to ensure that unrelated register bits are not affected. It is recommended to leverage the individual register bit manipulation APIs of this driver.
23	void FRAM_WRAR(uint32_t Reg_address, uint8_t *New_Reg_Data);	This API updates the register addressed by “Reg_address” with value pointed by “*New_Reg_Data”	WREN command must be issued prior to executing this API.

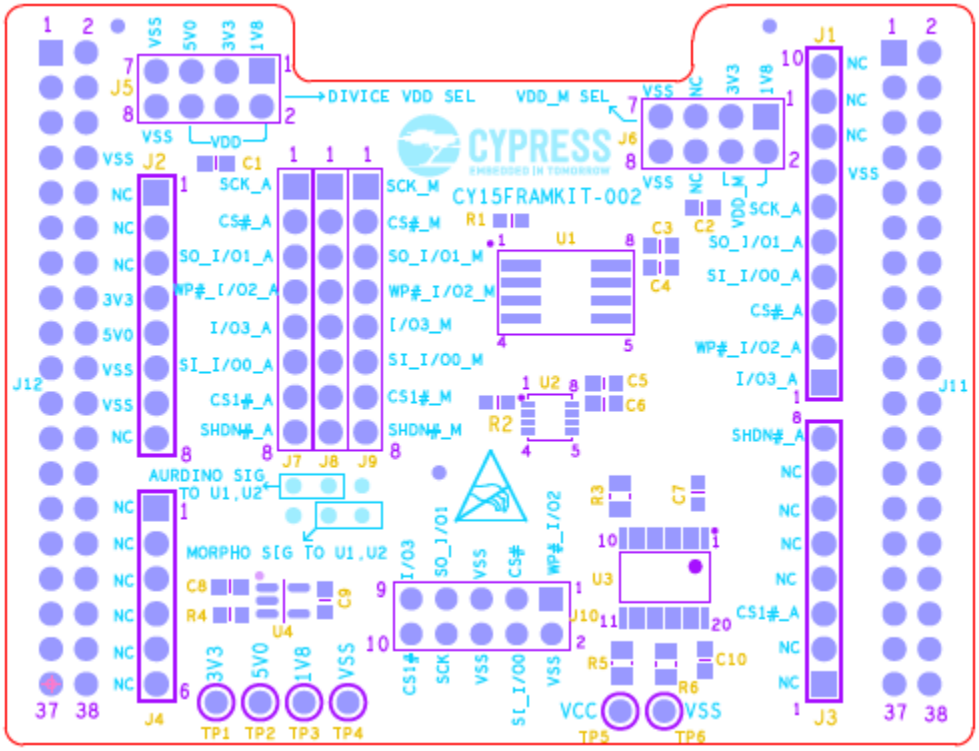
Sr. No	API Name	Description	Notes
24	void FRAM_RDAR(uint32_t Reg_address, uint8_t *New_Reg_Data);	This API reads the register addressed by “Reg_address” into the variable pointed by “*New_Reg_Data”	
25	bool FRAM_Set_Memory_Latency(uint8_t New_latency);	This API allows updating the memory latency values of F-RAM	This API is self-sufficient in terms of issues appropriate write enable commands. No other register bits are affected by this API
26	bool FRAM_Set_Register_Latency(uint8_t New_latency);	This API allows updating the register latency values of F-RAM	This API is self-sufficient in terms of issues appropriate write enable commands. No other register bits are affected by this API
27	void FRAM_Get_Memory_Latency(void);	This API reads the memory latency bits from device and stores the corresponding decimal value in Mem_Latency variable of operating_mode structure	API assumes the controller is in known operating mode (SPI, DPI or QPI)
28	void FRAM_Get_Register_Latency(void);	This API reads the register latency bits from device and stores the corresponding decimal value in Reg_Latency variable of operating_mode structure	
29	void FRAM_QPIEN(void);	Changes the operating mode of the F-RAM and controller to Quad SPI	This API is self-sufficient in terms of issues appropriate write enable commands. No other register bits are affected by this API
30	void FRAM_DPIEN(void);	Changes the operating mode of the F-RAM and controller to Dual SPI	
31	void FRAM_SPIEN(void);	Changes the operating mode of the F-RAM and controller to Standard SPI	
32	void FRAM_QUAD_ENABLE(void);	Sets the QUAD bit	QUAD bit must be set prior to executing any o the extended SPI opcodes. Refer F-RAM datasheet for details. This API is self-sufficient in terms of issues appropriate write enable commands. No other register bits are affected by this API
33	void FRAM_QUAD_DISABLE(void);	Resets the QUAD bit	This API is self-sufficient in terms of issues appropriate write enable commands. No other register bits are affected by this API

A. Appendix

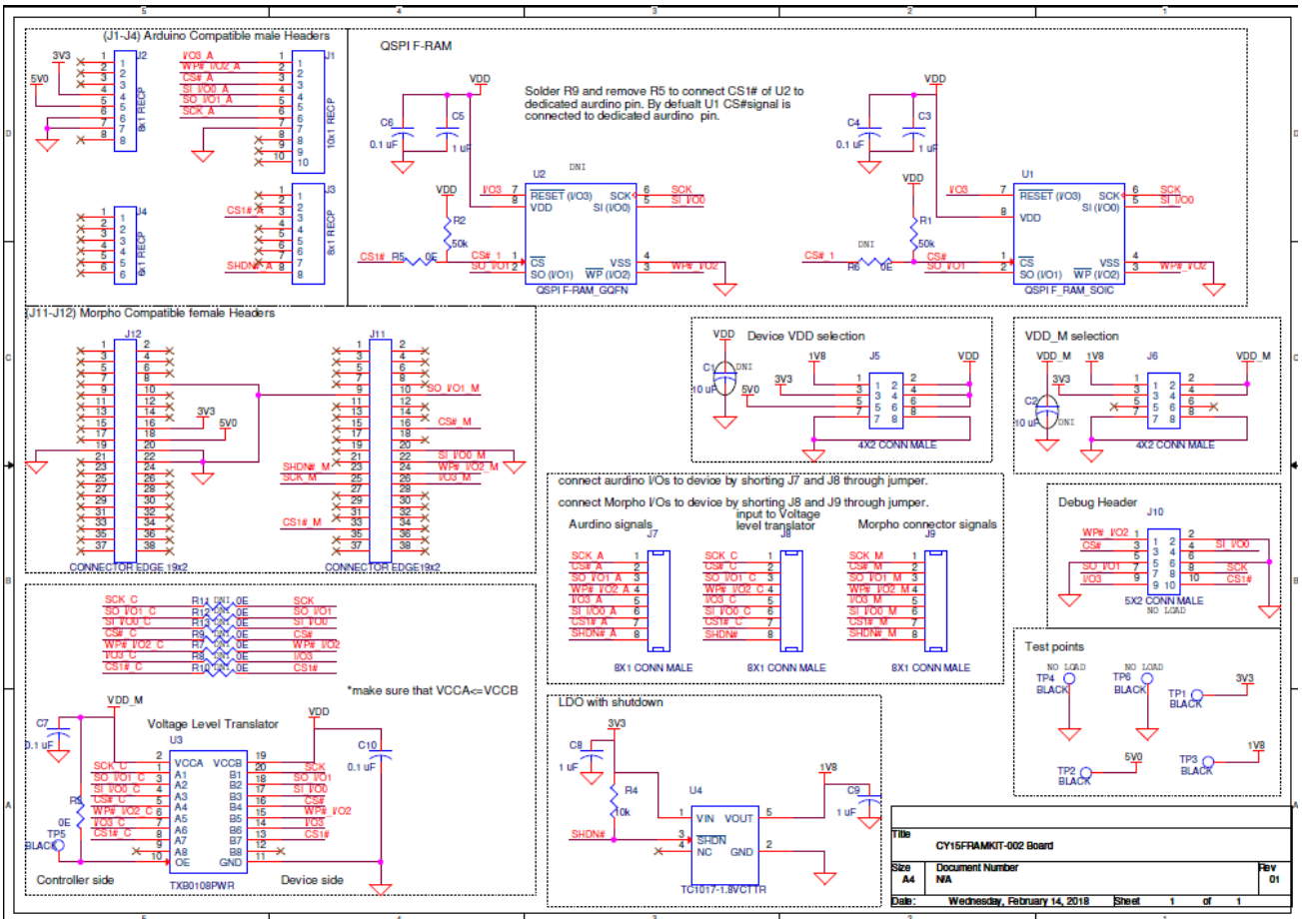
A.1 CY15FRAMKIT-002 Kit Block Diagram



A.2 CY15FRAMKIT-002 Kit Components Placement



A.3 CY15FRAMKIT-002 Kit Schematic



A.4 Pin Assignment Table

This section provides the pin map of the headers and their usage.

Arduino Compatible Headers (J1, J2, J3, J4)

J1			
Pin	**Arduino Signal**	**ST Controller Signals**	**CY15FRAMKIT-002 Kit Signals**
J1_1	D8	PB6	I/O3_A
J1_2	PWM / D9	PA8	WP# I/O2_A
J1_3	PWM / CS / D10	PA11	CS#_A
J1_4	PWM / MOSI/D11	PB15	SI_I/O0_A
J1_5	MISO / D12	PB14	SO_I/O1_A
J1_6	SCK / D13	PB13	SCK_A
J1_7	GND	-	GND
J1_8	AVDO	-	NC
J1_9	SDA / D14	PB7	NC
J1_10	SCL / D15	PB8	NC

J2			
Pin	**Arduino Signal**	**ST Controller Signals**	**CY15FRAMKIT-002 Kit Signals**
J2_1	NC	-	NC
J2_2	IOREF	-	NC
J2_3	NRST	NRST	NC
J2_4	3V3	-	3V3
J2_5	5V	-	5V0
J2_6	GND	-	GND
J2_7	GND	-	GND
J2_8	VIN	-	NC

J3			
Pin	**Arduino Signal**	**ST Controller Signals**	**CY15FRAMKIT-002 Kit Signals**
J3_1	RX/D0	PA3/PA10	NC
J3_2	TX/D1	PA2/PA9	NC
J3_3	D2	PA12	CS1#_A
J3_4	PWM / D3	PB3	NC
J3_5	D4	PB5	NC
J3_6	PWM/D5	PA15	NC
J3_7	PWM/D6	PB10	NC
J3_8	D7	PC7	SHDN#_A

J4			
Pin	**Arduino Signal**	**ST Controller Signals**	**CY15FRAMKIT-002 Kit Signals**
J4_1	A0	PA0	NC
J4_2	A1	PA1	NC
J4_3	A2	PC3	NC
J4_4	A3	PC2	NC
J4_5	A4	PC1	NC
J4_6	A5	PC0	NC

Morpho Compatible Headers (J11, J12)

J11		
Pin	**ST Controller Signals**	**CY15FRAMKIT-002 Kit Signals**
J11_1	PC9	NC
J11_2	PC8	NC
J11_3	PB8	NC
J11_4	PC6	NC
J11_5	PB7	NC
J11_6	PC5	NC
J11_7	AVDD	NC
J11_8	5V-STLINK	NC
J11_9	GND	GND
J11_10	PB0	SO_I/O1_M
J11_11	PB13	NC
J11_12	PA10	NC
J11_13	PB14	NC
J11_14	PA9	NC
J11_15	PB15	NC
J11_16	PB11	CS#_M
J11_17	PA11	NC
J11_18	PB2	NC
J11_19	PA8	NC
J11_20	GND	GND
J11_21	PB6	NC
J11_22	PB1	SI_I/O0_M
J11_23	PC7	SHDN#_M
J11_24	PA7	WP#_I/O2_M
J11_25	PB10	SCK_M
J11_26	PA6	I/O3_M
J11_27	PA15	NC
J11_28	PA5	NC
J11_29	PB5	NC
J11_30	PA4	NC
J11_31	PB3	NC
J11_32	AGND	NC
J11_33	PA12	CS1#_M
J11_34	PC4	NC
J11_35	PA2/PA9	NC
J11_36	PA3	NC
J11_37	PA3/PA10	NC
J11_38	PA2	NC

J12		
Pin	**ST Controller Signals**	**CY15FRAMKIT-002 Kit Signals**
J12_1	PC10	NC
J12_2	PC11	NC
J12_3	PC12	NC
J12_4	PD2	NC
J12_5	VDD	NC
J12_6	E5V	NC
J12_7	BOOT0	NC
J12_8	GND	GND
J12_9	NC	NC
J12_10	NC	NC
J12_11	NC	NC
J12_12	IOREF	NC
J12_13	PB12	NC
J12_14	NRST	NC
J12_15	PA13	NC
J12_16	3V3	3V3
J12_17	PA14	NC
J12_18	5V	5V0
J12_19	GND	GND
J12_20	GND	GND
J12_21	NC	NC
J12_22	GND	GND
J12_23	PC13	NC
J12_24	VIN	NC
J12_25	PC14	NC
J12_26	NC	NC
J12_27	PC15	NC
J12_28	PA0	NC
J12_29	PH0	NC
J12_30	PA1	NC
J12_31	PH1	NC
J12_32	PC3	NC
J12_33	VBAT	NC
J12_34	PC2	NC
J12_35	PB4	NC
J12_36	PC1	NC
J12_37	PB9	NC
J12_38	PC0	NC

Debug Header I/Os

Pin	Debug Signals
J10_1	WP#_IO2
J10_2	GND
J10_3	CS#
J10_4	SI_IO0
J10_5	GND
J10_6	GND
J10_7	SO_IO1
J10_8	SCK
J10_9	IO3
J10_10	CS1#

A.5 Use of Zero-ohm Resistors and No Load

The current board consists of a level shifter (U3) mounted on the F-RAM signals to allow interfacing the device with a wide operating voltage (from 1.8V to 5.5V). But this in turn reduces the maximum operating frequency of the interface due to propagation delay. To achieve a higher operating frequency, mount R7-R13 zero-ohm resistors and un-mount level shifter (U3). Please note that the absolute maximum QSPI frequency that can be achieved with Nucleo-L433RC-P board is restricted to 60MHz.

Resistor	Debug Signals	Usage
R7	WP#_I/O2	Solder R7 which will connect signals from J8 to F-RAM Device (U1, U2) directly
R8	I/O3	Solder R8 which will connect signals from J8 to F-RAM Device (U1, U2) directly
R9	CS#	Solder R9 which will connect signals from J8 to F-RAM Device (U1, U2) directly
R10	CS1#	Solder R10 which will connect signals from J8 to F-RAM Device (U1, U2) directly
R11	SCK	Solder R11 which will connect signals from J8 to F-RAM Device (U1, U2) directly
R12	SO_I/O1	Solder R12 which will connect signals from J8 to F-RAM Device (U1, U2) directly
R13	SI_I/O0	Solder R13 which will connect signals from J8 to F-RAM Device (U1, U2) directly

A.6 Bill of Materials (BOM)

Item	Quantity	Reference	Part	Manufacturer	Manufacturing part_number
1	2	C1,C2	10 uF	Yageo	CC0603KRX5R7BB106
2	4	C3,C5,C8,C9	1 uF	Taiyo Yuden	TMK107BJ105KA-T
3	4	C4,C6,C7,C10	0.1 uF	Murata Electronics	GRM188R71C104KA01D
4	1	J1	10x1 RECP	Samtec Inc	SSQ-110-03-T-S
5	2	J2,J3	8x1 RECP	Samtec Inc	SSQ-108-03-T-S
6	1	J4	6x1 RECP	Samtec Inc	SSQ-106-03-T-S
7	2	J5,J6	4X2 CONN MALE	Amphenol FCI	67997-408HLF
8	3	J7,J8,J9	8X1 CONN MALE	Amphenol FCI	68001-408HLF
9	1	J10	5X2 CONN MALE	Amphenol FCI	67997-410HLF
10	1	J11	CONNECTOR EDGE19x2	Sullins Connec	PPPC192LFBN-RC
11	1	J12	CONNECTOR EDGE 19x2	Sullins Connec	PPPC192LFBN-RC
12	2	R1,R2	50k	Panasonic-ECG	ERJ-3GEYJ513V
13	2	R3,R5	0E	Panasonic - ECG	ERJ-6GEY0R00V
14	1	R4	10k	Panasonic - ECG	ERJ-3GEYJ103V
15	6	TP1,TP2,TP3,TP4,TP5,TP6	BLACK	Keystone Electronics	5001
16	1	U1	QSPI F_RAM_SOIC	Cypress	CY15B104QSN-108SXI
17	1	U3	TXB0108PWR	Texas Instrume	TXB0108PWR
18	1	U4	TC1017-1.8VCTTR	Microchip	TC1017-1.8VCTTR
No Load Components					
19	8	R6,R7,R8,R9,R10,R11,R12,R	0E	Panasonic - ECG	ERJ-6GEY0R00V
20	1	U2	QSPI F-RAM_GQFN	Cypress	CY15B104QSN-108LPXI

A.7 CY15FRAMKIT-002 Board - Jumper Details

Headers to select Arduino connector signal connectivity to F-RAM device (J7, J8)

Place jumper on J7 and J8 horizontally

Pin	Pin	CY15FRAMKIT-002 Kit Signals (Arduino Signal)
J7_1	J8_1	SCK_A
J7_2	J8_2	CS#_A
J7_3	J8_3	SO_I/O1_A
J7_4	J8_4	WP#_I/O2_A
J7_5	J8_5	I/O3_A
J7_6	J8_6	SI_I/O0_A
J7_7	J8_7	CS1#_A
J7_8	J8_8	SHDN#_A

Headers to select Morpho connector signal connectivity to F-RAM device (J8, J9)

Place jumper on J7 and J8 horizontally

Pin	Pin	CY15FRAMKIT-002 Kit Signals (Morpho Signal)
J9_1	J8_1	SCK_M
J9_2	J8_2	CS#_M
J9_3	J8_3	SO_I/O1_M
J9_4	J8_4	WP#_I/O2_M
J9_5	J8_5	I/O3_M
J9_6	J8_6	SI_I/O0_M
J9_7	J8_7	CS1#_M
J9_8	J8_8	SHDN#_M

B. Appendix

This section is applicable only if user includes guiMENU (.c/.h) files in the project. The APIs written in these files allows for testing the QSPI F-RAM device mounted on the CY15FRAMKIT-002 board by sending commands over the serial interface of controller. On the NUCLEO board, LPUART1 is connected to the virtual COM port and it will be used as the serial interface to communicate with controller.

B.1 GUI Menu

For ease of evaluating the features of QSPI F-RAM as master function “void GUI_Menu(void);” has been provided. The test project provided with the drivers calls this function in an infinite loop. It allows user to send appropriate commands over serial interface in order communicate with the F-RAM. Following table summarizes the command set for the GUI.

Entering “000” at terminal will provide a summary of supported functions

Command/ Opcode	Description	Notes
QSPI Opcode based commands		
RDSR1 (05)	Prints out SR1 value on terminal. Stores the read value in operating_mode structure	
RDSR2 (07)	Prints out SR2 value on terminal. Stores the read value in operating_mode structure	
RDCR1 (35)	Prints out CR1 value on terminal. Stores the read value in operating_mode structure	
RDCR2 (3F)	Prints out CR2 value on terminal. Stores the read value in operating_mode structure	
RDCR4 (45)	Prints out CR4 value on terminal. Stores the read value in operating_mode structure	
RDCR5 (5E)	Prints out CR5 value on terminal. Stores the read value in operating_mode structure	
WRAR (71)	Write any register	The function waits for user input on register address and new register value
RDAR (65)	Read any register	The function waits for user input on register address. The register value is printed out on the terminal
DID (9F)	Read ID register and stores the value in operating_mode structure	Default value is 0x5051820600000000
RUID (4C)	Read Unique ID register and stores the value in operating_mode structure	
WRSN (C2)	Update Serial Number register	The function waits for 8-bytes of new serial number
RDSN (C3)	Read Serial Number register	The function prints 8-byte serial number register of the device
WREN (06)	Issues Write enable command to device	No message is displayed on terminal
WRDI (04)	Issues Write disable command to device	
WRITE (02)	Performs write operation to device using the opcode that is used as a command. Write functions will trigger user input for address location, data pattern and number of bytes to be written	Refer to datasheet for details on Write and Read functionality of the F-RAM
FAST_WRITE (DA)		
DIW (A2)		
DIOW (A1)		
QIW (32)		
QIOW (D2)		
DDRWRITE (DE)	Not supported in Revision 1.0 of the release	
DDRFWRITE (DD)		

DDRQIOW (D1)		
READ (03)	Performs read operation on device using the opcode that is used as a command. Read functions will trigger user input for address location, data pattern and number of bytes to be read	
FAST_READ (0B)		
DOR_READ (3B)		
DIOR_READ (BB)		
QOR_READ (6B)		
QIOR_READ (EB)		
DDRFAST_READ (0D)	Not supported in Revision 1.0 of the release	
Special Commands		
“444”	Changes the operating mode of device to Quad SPI	Refer datasheet for appropriate Memory and register latency settings before performing Read operations
“222”	Changes the operating mode of device to Dual SPI	
“111”	Changes the operating mode of device to Standard SPI	
“999”	Resets the Operating mode to Standard SPI. Updates all the registers to default value.	Use this function get out of an unknown operating mode
“801”	Prints out the current Memory latency setting	
“802”	Updates the Memory latency value with new value	Function will fail if user attempts to set latency higher than 15
“803”	Prints out current Register Latency setting	
“804”	Updates the Register latency value with new value	Function will fail if user attempts to set the latency higher than 3

```
Enter an Opcode/Command (000 for supported Opcodes):
Opcode: 000
Following Commands are supported in GUI

*************************************************************************
                    QSPI Opcodes Supported
*************************************************************************

      Opcodes                Description

        05                Read Status Register 1
        07                Read Status Register 2
        35                Read Configuration Register 1
        3F                Read Configuration Register 2
        45                Read Configuration Register 4
        5E                Read Configuration Register 5
        71                Write Any Register
        65                Read Any Register
        9F                Read ID Register
        C3                Read Serial Number Register
        C2                Write Serial Number Register.
                          Enter each byte seperated by a space

        4C                Read Unique ID Register
        06                Issue Write Enable
        04                Issue Write Disable
        02                Initiate Memory Write with Opcode 0x02
        DA                Initiate Memory Fast Write with Opcode 0xDA
        A2                Initiate Memory Write in DIW with Opcode 0xA2
        A1                Initiate Memory Write in DIOW with Opcode 0xA1
        32                Initiate Memory Write in QIW with Opcode 0x32
        D2                Initiate Memory Write in QIOW with Opcode 0xD2
        03                Initiate Memory Read with Opcode 0x03
        0B                Initiate Memory Fast Read with Opcode 0x0B
        3B                Initiate Memory Read in DOR with Opcode 0x3B
        BB                Initiate Memory Read in DIOR with Opcode 0xBB
        6B                Initiate Memory Read in QOR with Opcode 0x6B
        EB                Initiate Memory Read in QIOR with Opcode 0xEB

*************************************************************************
                    Special Functions
*************************************************************************

  Special Commands              Description
        999               Performs Interface Reset.
                          All Registers restored to default

        111               Changes the interface mode to SPI
                          for ST controller and F-RAM

        222               Changes the interface mode to DPI
                          for ST controller and F-RAM

        444               Changes the interface mode to QPI
                          for ST controller and F-RAM

        801               Read Memory Latency value in F-RAM
        802               Set Memory Latency value in F-RAM
        803               Read Register Latency value in F-RAM
        804               Set Memory Latency value in F-RAM
        123               Perform Basic F-RAM Testing

*************************************************************************
```

Figure 17: GUI Menu

Revision History

Document Revision History

Document Title: ExcelonTM Ultra QSPI F-RAM Development Kit User Guide Document Number: 002-23147 Rev **			
Revision	**Issue Date**	**Origin of Change**	**Description of Change**
**		NILE	New Spec