

CY15FRAMKIT-002

Serial F-RAM Development Kit User Guide

Document Number: 002-23147 Rev. *A

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



© Cypress Semiconductor Corporation, 2018-2019. This document is the property of Cypress Semiconductor Corporation and its subsidiaries ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress, production, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. No computing device can be absolutely secure. Therefore, despite security measures implemented in Cypress hardware or software products, Cypress shall have no liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. CYPRESS DOES NOT REPRESENT, WARRANT, OR GUARANTEE THAT CYPRESS PRODUCTS, OR SYSTEMS CREATED USING CYPRESS PRODUCTS, WILL BE FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (collectively, "Security Breach"). Cypress disclaims any liability relating to any Security Breach, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from any Security Breach. In addition, the products described in these materials may contain design defects or errors known as errata which may cause the product to deviate from published specifications. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. "High-Risk Device" means any device or system whose failure could cause personal injury, death, or property damage. Examples of High-Risk Devices are weapons, nuclear installations, surgical implants. and other medical devices. "Critical Component" means any component of a High-Risk Device whose failure to perform can be reasonably expected to cause, directly or indirectly, the failure of the High-Risk Device, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from any use of a Cypress product as a Critical Component in a High-Risk Device. You shall indemnify and hold Cypress, its directors, officers, employees, agents, affiliates, distributors, and assigns harmless from and against all claims, costs, damages, and expenses, arising out of any claim, including claims for product liability, personal injury or death, or property damage arising from any use of a Cypress product as a Critical Component in a High-Risk Device. Cypress products are not intended or authorized for use as a Critical Component in any High-Risk Device except to the limited extent that (i) Cypress's published data sheet for the product explicitly states Cypress has qualified the product for use in a specific High-Risk Device, or (ii) Cypress has given you advance written authorization to use the product as a Critical Component in the specific High-Risk Device and you have signed a separate indemnification agreement.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.

Contents



1.	Introduction	4
	1.1 Kit Contents	4
	1.2 Getting Started	4
	1.3 Pre-requisite Software	4
	1.4 Pre-requisite Hardware	4
	1.5 Additional Resources	4
2.	Kit Overview	5
	2.1 CY15FRAMKIT-002 Kit Overview	5
	2.2 Kit Introduction and Configuration Guide	6
	2.3 F-RAM and MCU Kit Connection	
3.	Kit Operation	11
	3.1 Programming the NUCLEO-433LC-P Evaluation Board	11
	3.2 Quad SPI Configuration for STM32L433RC	17
4.	Firmware Details	20
4.		
4. A.	Appendix	
		24
	Appendix	24 24
	Appendix CY15FRAMKIT-002 Kit Block Diagram	24
	Appendix CY15FRAMKIT-002 Kit Block Diagram CY15FRAMKIT-002 Kit Component Placements	24 24 25 26
	Appendix CY15FRAMKIT-002 Kit Block Diagram CY15FRAMKIT-002 Kit Component Placements CY15FRAMKIT-002 Kit Schematic	24
	Appendix CY15FRAMKIT-002 Kit Block Diagram CY15FRAMKIT-002 Kit Component Placements CY15FRAMKIT-002 Kit Schematic Pin Assignment Table	24 24 25 26 27 30
	Appendix CY15FRAMKIT-002 Kit Block Diagram CY15FRAMKIT-002 Kit Component Placements CY15FRAMKIT-002 Kit Schematic Pin Assignment Table. Use of Zero-ohm and No-Load Resistors	
	Appendix CY15FRAMKIT-002 Kit Block Diagram CY15FRAMKIT-002 Kit Component Placements CY15FRAMKIT-002 Kit Schematic Pin Assignment Table Use of Zero-ohm and No-Load Resistors Bill of Materials (BOM)	24 24 25 26 27 30 31 31
Α.	Appendix CY15FRAMKIT-002 Kit Block Diagram CY15FRAMKIT-002 Kit Component Placements CY15FRAMKIT-002 Kit Schematic Pin Assignment Table Use of Zero-ohm and No-Load Resistors Bill of Materials (BOM) CY15FRAMKIT-002 Board - Jumper Details	24 24 25 26 27 30 31 31 31 33

1. Introduction



Thank you for your interest in the CY15FRAMKIT-002 F-RAM[™] 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 circuitry, and provides both Arduino and Morpho connectivity. This kit supports 3.3-V or 1.8-V power supplies.

1.1 Kit Contents

The CY15FRAMKIT-002 Kit includes the following contents:

- CY15FRAMKIT-002 DVK board with CY15B104QSN-108SXI, 108-MHz 4-Mbit Quad SPI F-RAM
- Quick Start Guide
- Arduino Stackable Headers for connector expansion

1.2 Getting Started

This guide helps you to get acquainted with the CY15FRAMKIT-002 DVK. The Pre-requisite Software section describes the software required to use the driver set provided for this kit. The Kit Overview chapter explains the features of the kit. The Kit Operation chapter explains how to program and run the kit. The Appendix B explains the GUI to test out the QSPI F-RAM features. The Firmware Details chapter the F-RAM access API and a link to download the example project. Appendix A provides the kit block diagram, schematics, pin assignment, and the bill of materials (BOM)

1.3 Pre-requisite Software

- Keil µVision 5 software (for development and programming)
- STSW-LINK009 ST-LINK, ST-LINK/V2, ST-LINK/V2-1 USB driver signed for Windows 7, Windows 8, Windows 10
- 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 example project and firmware files are part of the compressed folder downloaded from the Cypress Community.
- The datasheet for Cypress Quad SPI F-RAM mounted on the kit can be downloaded from here.

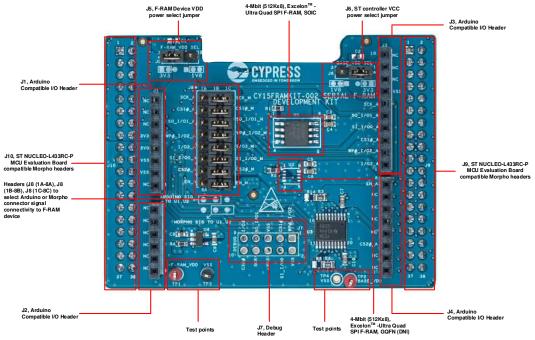




2.1 CY15FRAMKIT-002 Kit Overview

The CY15FRAMKIT-002 kit can be used to understand the features of serial F-RAMs (SPI and QSPI). The DVK is an addon board, which contains a 4-Mbit Excelon[™] -Ultra QSPI F-RAM. It has four connectors that are Arduino UNO-compatible and connect to either an ST NUCLEO-L433RC-P MCU Evaluation Board or an Arduino UNO R3 compatible board for SPI evaluation. It also provides two 19x2 Morpho headers (J9 and J10) 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. The 3.3-V supply comes from the base board and the 1.8-V supply is generated by an on-board regulator. The CY15FRAMKIT-002 Kit consists of the following blocks.

- 4-Mbit (512Kx8), Excelon-Ultra Quad SPI F-RAM
- ST NUCLEO-L433RC-P MCU Evaluation Board with compatible Morpho headers (J9, J10)
- Arduino-compatible I/O Headers (J1, J2, J3, J4)
- Headers (J8 (1A-8A), J8 (1B-8B), J8 (1C-8C)) to select Arduino or Morpho connector signal connectivity to the F-RAM device
- J5 header for 1.8-V/3.3-V F-RAM device VDD power supply selection
- J6 header for 1.8-V/3.3-V selection for level shifters. Set this to the base board's power supply value.
- J7 debug header for probing SPI/QSPI signals
- Test points for the supported voltage (F-RAM_VDD and BASE_VDD) 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





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 1. The factory default jumper setting is 3.3 V (short pins 1 and 2 of J5). You can short pins 2 and 3 of J5 for 1.8-V operation.

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 or other Arduino UNO R3-compatible base board.
- The CY15FRAMKIT-002 board operates with 1.8 V or 3.3 V supply. Max operating voltage of the device is 3.6 V. Exceeding the maximum voltage limit (3.6 V) may damage the board.

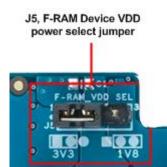


Figure 1. F-RAM Device VDD (F-RAM_VDD) Selection Jumper

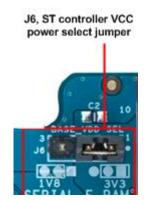
2.2.2 ST Controller Device Power Supply Jumper

The ST NUCLEO-L433RC-P MCU Evaluation Board hardware operates at 1.8 V/3.3 V. The value selected on the base board must be selected on the shield to enable appropriate level shifters through header J6, as shown in Figure 2. The factory default jumper setting is 3.3 V (short pins 1 and 2 of J6). You can short pins 2 and 3 of J6 for the 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 with 1.8 V or 3.3 V supply. Exceeding the maximum voltage limit (3.6 V) may damage the board.

Figure 2. ST Controller Device VDD (BASE_VDD) Selection Jumper



2.2.3 Valid Power Selection for Voltage Level Translator (BASE_VDD) and F-RAM Device VDD (F-RAM VDD)

Table 1 shows the valid power selection for voltage level translator (BASE_VDD) and F-RAM supply (FRAM_VDD) for a proper operation of the board.



BASE_VDD (J6)	F-RAM_VDD (J5)
1V8	1V8
1V8	3V3
3V3	3V3

 Table 1. Valid Power Selection for Voltage Level Translator through Jumpers

2.2.4 Excelon-Ultra Quad SPI F-RAM

Figure 3 shows the 4-Mbit (512Kx8), 3.3 V, Excelon-Ultra Quad SPI F-RAM part in the 8-pin SOIC package option.

Figure 3. Cypress Serial Quad SPI F-RAM Device on the Board



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

Figure 4 shows the Morpho headers to the ST NUCLEO-L433RC-P MCU Evaluation Board for QSPI interface evaluation – J9 and J10. You can plug the CY15FRAMKIT-002 board onto the ST NUCLEO-L433RC-P MCU Evaluation Board through these connectors. For the schematic, see Appendix.

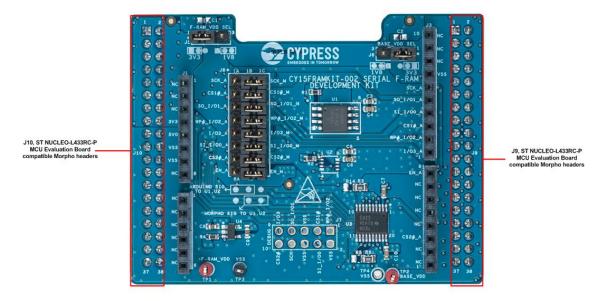


Figure 4. Morpho Headers to ST NUCLEO-L433RC-P MCU Evaluation Board



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

Figure 5 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, see Appendix.

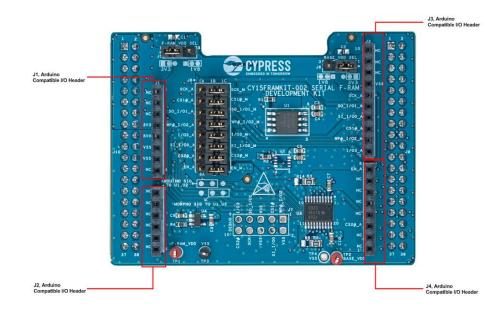
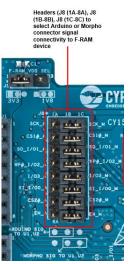


Figure 5. Arduino Headers to ST NUCLEO-L433RC-P MCU Evaluation Board

2.2.7 Headers to Select Arduino or Morpho Connector Signal Connectivity to F-RAM Device

Figure 6 shows the headers – J8 (1A-8A), J8 (1B-8B) and J8 (1C-8C). You can short F-RAM device signals to the Arduino connectors by shorting the pins of J8 (1A-8A) to the corresponding pins of J8 (1B-8B) or to the Morpho connectors by shorting J8 (1B-8B) pins to J8 (1C-8C) pins. For the schematic, see Appendix.

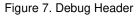
Figure 6. Headers to Select Arduino or Morpho Connector Signals

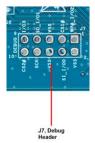


2.2.8 Debug Header

Debug header J7 is provided for easy access to the Quad SPI F-RAM communication pins as shown in Figure 7. For the schematic, see Appendix.







2.2.9 Test Points

The CY15FRAMKIT-002 board provides FRAM_VDD, VSS, and BASE_VDD test points as shown in Figure 8.

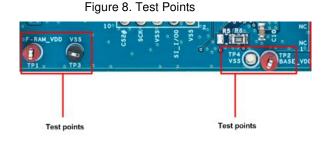


Table 2. Test Points

Test Points	Description	
BASE_VDD	ASE_VDD Test point for the Voltage Level Translator to match th base board power supply	
VSS	Test point for the kit board ground	
F-RAM_VDD	Test point for the F-RAM device (U1, U2) power supply	

CAUTION:

Power supply test points are only for probing and measurement purposes. Do not power the kit using this test point to avoid damage to the board.

2.2.10 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 enables access to the second device on the board with an appropriate firmware modification.



Figure 9. 8-pin GQFN - Not Populated by Default



2.3 F-RAM and MCU Kit Connection

The CY15FRAMKIT-002 is plugged into the ST MCU Evaluation Board through Arduino (J1, J2, J3 and J4) and Morpho (J9, J10), connectors as shown in Figure 10.

Figure 10. CY15FRAMKIT-002 Mounted on ST MCU Evaluation Board



CAUTION:

The CY15FRAMKIT-002 board is plugged into the ST MCU Evaluation Board through its compatible headers J1, J2, J3, J4, J9 and J10. Because J9 and J10 are high pin-count connectors, removing the CY15FRAMKIT-002 board can be difficult and may lead to bending on the connector pins. Therefore, take care when removing the CY15FRAMKIT-002. Sliding the CY15FRAMKIT-002 board out of the connectors slowly and evenly will reduce the risk of bending the connector pins.



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

3.1 Programming the NUCLEO-433LC-P Evaluation Board

The NUCLEO-L433RC-P board is an STM32 development board based on the Nucleo-64 platform. The board offers several features including 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 11. 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. Drivers and example project have been written with the STM32L433RC controller as the target, but they can be easily modified to be compatible with any other ST microcontroller with a Quad SPI interface. In the firmware section of this document, appropriate sections will be highlighted to ensure compatibility with other ST microcontrollers.



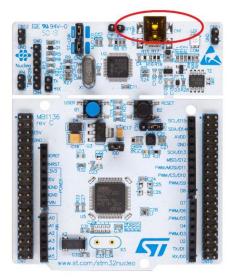
3.1.1 System Requirements

Ensure that all the prerequisite from Section 1.3 are installed on the system. Download the latest example project from Cypress community.

Establish the connection with the STM32 Nucleo board by connecting CN1 of the Nucleo board to the USB port of the PC. The board should be detected correctly under Device Manager.

Refer to the guide here to resolve issues related to installation of Nucleo 64 board drivers.

Figure 12. Connector CN1 for Nucleo 64 Board



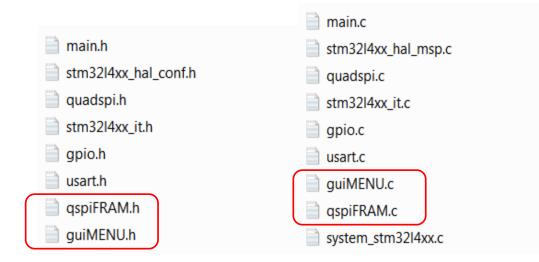
3.1.2 Compiling the Example Project

This user guide is accompanied by a example 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 *Excelon_QSPI* folder contains an MDK-Arm project along with the required source/include files. The contents of the zip file are shown here:

👢 Inc	2/5/2018 5:21 PM	File folder
👢 Src	2/5/2018 5:21 PM	File folder
👢 MDK-ARM	2/5/2018 5:18 PM	File folder
👢 Drivers	2/5/2018 5:18 PM	File folder

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





- 1. The *qspiFRAM(.h/.c)* files provide all the functions to access the QSPI F-RAM device. Include these (with the appropriate code modification) in your project to access the QSPI F-RAM.
- The guiMENU(.h/.c) files are provided to allow evaluation of the F-RAM by sending commands over a UART terminal (115200 bps, 8-NoParity-1). These files need not be included in the final project.

Open the μ Vision project from the MDK-Arm folder. If the tool (Keil) looks for a board support package, press the ESC key (There is no native package for the NUCLEO-L433RC-P board in the Keil library).

Your workspace should have following structure at this stage:

File Edit View Project Flash Debug Peripherals Tools SVCS Window	w Help
□ 22 22 24 24 24 24 24 24 24 24 24 24 24	🏙 f_open 🛛 🔻 🗟 🖑 🔍 🔍 🔹 🔿 🗞 📰 🔍
🔗 🕮 🥔 🤐 🛛 🙀 🖉 QSPI 🛛 🕞 🐼 🖓 🌚	
Project 🛛 🗸 🖬	
Project: QSPI QSPI CMSIS Drivers/CMSIS Application/MDK-ARM Application/User Drivers/STM32L4xx_HAL_Driver	

3. Before compiling, appropriate properties for the target must be selected. Press ALT+F7 or select **Project** > **Options for Target**, or click the button circled in the figure above.



STM32L431 STM32L432 STM32L433 STM32L433CB STM32L433CC STM32L433RB STM32L433RC STM32L433RC STM32L433RC STM32L433RC STM32L433RC STM32L433RC	ST has built a new architecture to reach best-in-class ultra-low-power figures thanks to its high flexibility. STM32L4 MCUs have scored 176.7 in the standardized EEMBC ULPBench tests that compare the efficiency of ultra-low-power microcontrollers. Moreover, the STM32L4 series shatters performance limits in the ultra-low-power world. It delivers 100 DMIPS based on its ARM Cortex-M4 core with FPU and ST ART Accelerator at 80 MHz. STM32L4 microcontrollers offer dynamic voltage scaling to balance power consumption with processing demand, low-power peripherals (LP UART, LP timers) available in Stop mode, safety and security features, smart and numerous peripherals, advanced and low-power analog peripherals such a op amps, comparators, LCD, 12-bit DACs and 16-bit ADCs (hardware	Ξ
[OK Cancel He	lp

- 4. Update the clock speed to 80.00MHz in the Target tab under Xtal (MHz).
- 5. On the **Debug** tab, ST-Link Debugger should be populated automatically if the Nucleo board is plugged into a USB port on the computer and the drivers are installed correctly.

		Debug Utilities	
 Use Simulato Limit Speed to 		OUse: ST-Lin	k Debugger Settings
 Load Application Initialization File: 	ion at Startup 🔽 Run to main()	 ✓ Load Application at Startup ✓ Run to main() Initialization File: 	
	Edit		Edit
Restore Debug Session Settings		Restore Debug Session Settings Image: Seakpoints Image: Toolbox Image: System Viewer Image: System Viewer	
CPU DLL:	Parameter:	Driver DLL:	Parameter.
SARMCM3.DLL	-REMAP -MPU	SARMCM3.DLL	-MPU
Dialog DLL:	Parameter:	Dialog DLL:	Parameter:
DCM.DLL	-pCM4	TCM.DLL	-pCM4
	Manage Component Vi	ewer Description File	B5

6. Click **Settings** on the **Debug** tab to select the appropriate Flash Programming Algorithm for this device.



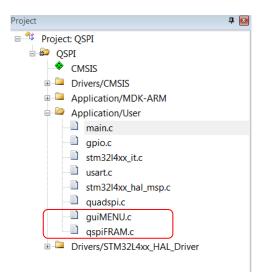
51 @ E	rase Full Chip 🔽 Pro- rase Sectors 🔽 Veri		RAM for Algorithm Start 0x20000000	Size	c 0x1000
ogramming Algor	ithm				
Description	Device Size	Device Type	Address Range		
			Start	Size	c 🔽
	\leq	Add	Remove		

- 1. Enable the **Reset and Run** option on **Target Driver Setup** window to ensure that the device resets on successful program.
- 2. By default, there will be no Programming Algorithm on first time selection of this controller. Click **Add** to open a list of available algorithms.
- 3. Select the appropriate programming algorithm as shown below, and click **Add**. The STM32L433RC MCU has 256 KB On-Chip flash.

Description	Flash Size	Device Type	Origin	
MX25LM51245G_STM32L	64M	Ext. Flash SPI	Device Family Package	
STM32L4xx 256 KB Flash	256k	On-chip Flash	Device Family Package	
STM32L4xx single bank F	20B	On-chip Flash	Device Family Package	
A2FxxxM3 128kB Flash	128k	On-chip Flash	MDK Core	
A2FxxxM3 256kB Flash	256k	On-chip Flash	MDK Core	
A2FxxxM3 512kB Flash	512k	On-chip Flash	MDK Core	
ADuCRF101 128kB Flash	128k	On-chip Flash	MDK Core	
ADuCM360 128kB Flash	128k	On-chip Flash	MDK Core	
AM29x128 Flash	16M	Ext. Flash 16-bit	MDK Core	
ATSAM3N 128kB Flash	128k	On-chip Flash	MDK Core	
ATSAM3N 16kB Flash	16k	On-chip Flash	MDK Core	
ATSAM3N 256kB Flash	256k	On-chip Flash	MDK Core	
ATSAM3N 32kB Flash	32k	On-chip Flash	MDK Core	
ATSAM3N 64kB Flash	64k	On-chip Flash	MDK Core	
ATSAM3N GPNVM bits	16B	On-chip Flash	MDK Core	
ATSAM3S 128kB Flash	128k	On-chip Flash	MDK Core	
	050	A 11 EL 1	UDIZ O	
			ash\STM32L4xx_256.FLM	

4. Add the *qspiFRAM.c* file in the Application/User space. Add the *guiFRAM.c* file if you are evaluating the device features using a UART terminal.





5. 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
```

 Press F8 to program the controller with the test program. Open a terminal emulator program of your choice (PuTTY, uCON, etc.) and establish a UART connection at 115200 bps with 8 data bits, no parity, and 1 stop bit. Look under Ports (COM & LPT) to determine the COM port number to use.

The controller will be waiting for the appropriate GUI command. See Appendix B for GUI commands.



3.2 Quad SPI Configuration for STM32L433RC

The Arduino/Morpho QSPI F-RAM Kit is designed interface with both the SPI port on the Arduino interface as well as the Quad SPI port on the Morpho interface. You can use either interface with the help of jumper selections. The drivers/API provided with example project are for using the Quad SPI port of controller only.



Figure 13. Complete Setup

3.2.1 Quad SPI Initialization

The initialization of the Quad SPI block in the ST controller is done by calling the QUADSPI_Init() function. The function is defined in *quadspi.c* but can be added to your main application file for a smaller project. The 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 the Quad SPI interface of this controller, refer to AN4760 from STMicroelectronics. The critical variables for evaluating the QSPI F-RAM are "ClockPrescaler" and "FlashSize"

ClockPrescaler: The default value in the example project is 32, which corresponds to approximately a 5-MHz clock frequency on the QSPI interface. You should modify this variable for achieving the required clock frequency. The controller on the Nucleo board supports a maximum of 48 MHz on the Quad SPI Clock, while the Morpho connecter is expected to support up to 30 MHz.

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

FlashSize: The default value in the example project is 24, which corresponds to the total number of address bits needed to address the memory attached to the interface. The Cypress F-RAM on the kit is a 4-Mbit device but has 3-byte addressing. The default value of 24 should remain unchanged in your application.

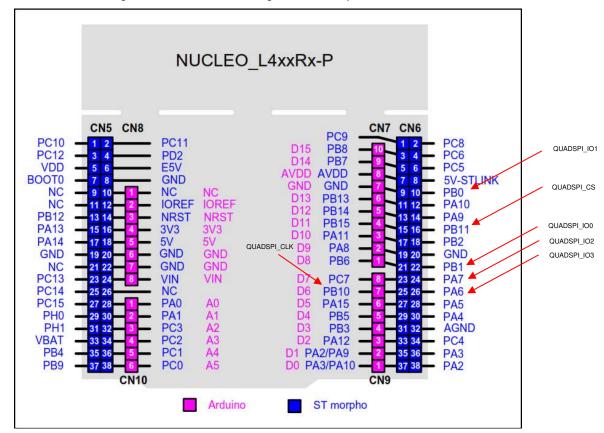


3.2.2 Quad SPI Interface on STM32L433RC

The Quad SPI interface for this controller is routed to the 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 14. 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.

In addition to initializing the Quad SPI block as described earlier, the QSPI_Init() function also ensures initialization of the controller pins as Quad SPI pins.

Figure 15. Pin Initialization as Quad SPI

/**QUADSPI GPIO Configuration PA6 -----> QUADSPI BK1 IO3 PA7 -----> QUADSPI_BK1_IO2 PB0 -----> QUADSPI_BK1_IO1 ----> QUADSPI_BK1_IO0 PB1 PB10 -----> QUADSPI_CLK ----> QUADSPI BK1_NCS PB11 */ 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.PIN = GPIO_PIN_0\GPIO_PIN_1\GPIO_PIN_1\GPIO_PIN_1\GPIO_PIN_1\,
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);
```

4. Firmware Details



This section summarizes the functions written in the *qspiFRAM* (.*h*/.*c*) file along with the μ Vision example project provided by Cypress Semiconductor. Each function has been supplemented with comments to help understand the usage. Revision 1.0 of the functions covers limited features of the device. The detailed feature set of Cypress' Quad SPI F-RAM can be found in the device datasheet. The functions 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 Semiconductor. You can leverage the API for building your end applications.

The example project is stored in the *Excelon_QSPI* folder and has following structure:

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

QSPI F-RAM functions are declared in *Excelon_QSPI**Inc\qspiFRAM.h;* implemented in *Excelon_QSPI**Src\qspiFRAM.c.* The following table provides a summary of all the supported functions for Revision 1.0 of the API. Unless specified explicitly the description is applicable only for the Quad SPI F-RAM device on the kit.

No.	Function Name	Description	Notes
1	bool FRAM_Interface_Reset (void)	This function resets the operating mode of the F-RAM 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 function is written specifically to implement a "Go Home" feature in case the controller faces a mismatch of the operating mode. For example, 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 a similar function in the end application.
2	<pre>void Read_Device_Status(void)</pre>	This function reads the register values of the F-RAM and stores them in a runtime structure variable.	Assumes that the controller is in a known operating mode with respect to the F-RAM.
3	bool Read_ID(uint8_t IOMode, uint8_t Reg_lat)	This function reads the ID register and compares it with the default value of 0x50518206000000.	This function is used internally by the FRAM_Interface_Reset (void) function. Read_ID () returns a success or a fail status based on whether the ID read from device matches the default value. The end user can leverage this function to identify the device operating mode (SPI, DPI, or QPI).
4	Void FRAM_WREN(void)	Issues the WREN (0x06) opcode.	
5	Void FRAM_WRDI (void)	Issues the WRDI (0x04) opcode.	



No.	Function Name	Description	Notes
6	<pre>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);</pre>	These functions can be used to read the Status/Configuration registers of the F-RAM and store the register value in the pointer argument.	These functions assume that the F-RAM is in a known operating mode.
7	<pre>void FRAM_RDSN (void); void FRAM_WRSN (uint8_t *SNreg);</pre>	These functions can be used to read or write the Serial Number Register.	SN Register is an 8-byte register. The Read function will read the device's serial number and store it in the SN_Reg array of the operating_mode structure. The argument passed to Write function must be a pointer to an 8-byte array containing the new Serial number value.
8	<pre>bool FRAM_Write(uint32_t address, uint8_t *write_data, uint32_t write_length);</pre>	This function performs a Write operation (Opcode 0x02).	Device must be write enabled (FRAM_WREN()) prior to writing into the device. This function performs a write of length write_length bytes stored in array pointed to by *write_data, starting at location address in the F-RAM. The function assumes that the controller is in a known operating mode (SPI, DPI, or QPI).
9	<pre>bool FRAM_Fast_Write(uint32_t address, uint8_t *write_data, uint32_t write_length);</pre>	This function performs a Fast Write operation (Opcode 0xDA).	Device must be write enabled (FRAM_WREN()) prior to writing into the device. This function performs a fast write of length write_length bytes stored in array pointed by *write_data, starting at location "address" in the F-RAM. The function assumes that the controller is in a known operating mode (SPI, DPI, or QPI).
10	<pre>bool FRAM_DIW(uint32_t address, uint8_t *write_data, uint32_t write_length);</pre>	This function performs a Write operation using opcode 0xA2.	Device must be write enabled (FRAM_WREN()) prior to writing into the device.
11	<pre>bool FRAM_DIOW(uint32_t address, uint8_t *write_data, uint32_t write_length);</pre>	This function performs a Write operation using opcode 0xA1.	This function performs a write of length write_length bytes stored in array pointed to by *write_data, starting at location address in the F-RAM. The function assumes that the controller and the F-RAM device is in SPI operating mode.
12	<pre>bool FRAM_QIW(uint32_t address, uint8_t *write_data, uint32_t write_length);</pre>	This function performs a Write operation using opcode 0x32.	Device must be write enabled (FRAM_WREN()) prior to writing into the device.
13	<pre>bool FRAM_QIOW(uint32_t address, uint8_t *write_data, uint32_t write_length);</pre>	This function performs a Write operation using opcode 0xD2.	This function performs a write of length write_length bytes stored in array pointed to by *write_data, starting at location address in the F-RAM. The function assumes that the controller and the F-RAM device is in SPI operating mode. QUAD bit must be set prior to executing this function.



No.	Function Name	Description	Notes
14	<pre>bool FRAM_DDRWrite(uint32_t address, uint8_t *write_data, uint32_t write_length);</pre>	This function is not tested for Revision 1.0 of this API.	
	<pre>bool FRAM_DDRQIOW(uint32_t address, uint8_t *write_data, uint32_t write_length);</pre>		N/A
	<pre>bool FRAM_DDRFastWrite(uint32_t address, uint8_t *write_data, uint32_t write_length);</pre>		
15	<pre>bool FRAM_Read(uint32_t address, uint8_t *read_data, uint32_t read_length);</pre>	This function performs a Read operation (Opcode 0x03).	This function performs a read of length read_length bytes and stores it in array pointed to by *read_data, starting at location "address" in the controller.
			The function assumes that the controller is in a known operating mode (SPI, DPI, or QPI).
16	<pre>bool FRAM_FastRead(uint32_t address, uint8_t *read_data, uint32_t read_length);</pre>	This function performs a Fast Read operation (Opcode 0x0B).	This function performs a fast read of length read_length bytes and stores it in array pointed to by *read_data, starting at location address in the controller.
			The function assumes that the controller is in a known operating mode (SPI, DPI, or QPI).
17	<pre>bool FRAM_DORRead(uint32_t address, uint8_t *read_data, uint32_t read_length);</pre>	This function performs a Fast Read operation using opcode 0x3B.	This function performs a fast read of length "read_length" bytes and stores it in array pointed to by *read_data,
18	<pre>bool FRAM_DIORRead(uint32_t address, uint8_t *read_data, uint32_t read_length);</pre>	This function performs a Fast Read operation using opcode 0xBB.	starting at location address in the controller. The function assumes that the controller and the F-RAM is in SPI operating mode.
19	<pre>bool FRAM_QORRead(uint32_t address, uint8_t *read_data, uint32_t read_length);</pre>	This function performs a Fast Read operation using opcode 0x6B.	This function performs a fast read of length read_length bytes and stores it in array pointed to by *read_data,
20	<pre>bool FRAM_QIORRead(uint32_t address, uint8_t *read_data, uint32_t read_length);</pre>	This function performs a Fast Read operation using opcode 0xEB.	starting at location address in the controller. The function assumes that the controller and the F-RAM is in SPI operating mode. The QUAD bit must be set prior to executing this function.
21	<pre>bool FRAM_DDRFastRead(uint32_t address, uint8_t *read_data, uint32_t read_length);</pre>	This function is not tested for Revision 1.0 of this API.	N/A
22	<pre>device_mode Detect_Mode(void);</pre>	This function is not tested for Revision 1.0 of this API.	N/A
23	<pre>void FRAM_WRR(uint8_t *Register, uint8_t No_of_bytes);</pre>	This function can perform writes to one or all registers in the device. Refer to	The WREN command must be issued prior to executing this function.
		the device datasheet for a detailed explanation.	A single command is used to update Status register 1, Configuration Registers 1, 2, 4 and 5.
			The Argument No_of_bytes determines the number of registers that are updated with this function.
			It is recommended to avoid using this function to ensure that unrelated register bits are not affected.
			It is recommended to instead use the individual register bit manipulation functions included in the API.

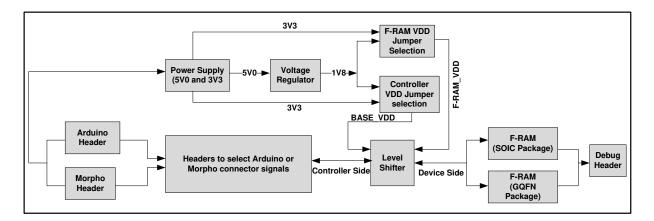


No.	Function Name	Description	Notes	
23	<pre>void FRAM_WRAR(uint32_t Reg_address, uint8_t *New_Reg_Data);</pre>	This function updates the register addressed by Reg_address with the value pointed to by *New_Reg_Data.	The WREN command must be issued prior to executing this function.	
24	<pre>void FRAM_RDAR(uint32_t Reg_address, uint8_t *New_Reg_Data);</pre>	This function reads the register addressed by "Reg_address" into the variable pointed to by *New_Reg_Data.		
25	<pre>bool FRAM_Set_Memory_Latency(uint8_t New_latency);</pre>	This function allows updating of the memory latency values of the F-RAM.	This function is self-sufficient. No other register bits are affected by this function. Memory latency is updated to New_latency.	
26	<pre>bool FRAM_Set_Register_Latency(uint8_t New_latency);</pre>	This function allows updating of the register latency values of the F-RAM.	This function is self-sufficient. No other register bits are affected by this function. Register latency is updated to New_latency.	
27	<pre>void FRAM_Get_Memory_Latency(void);</pre>	This function reads the memory latency bits and stores the corresponding decimal value in the Mem_Latency variable of the operating_mode structure.	These functions assume the controller is in a known operating mode (SPI, DPI, or QPI).	
28	<pre>void FRAM_Get_Register_Latency(void);</pre>	This function reads the register latency bits and stores the corresponding decimal value in the Reg_Latency variable of the operating_mode structure.		
29	<pre>void FRAM_QPIEN(void);</pre>	Changes the operating mode of the F- RAM and controller to Quad SPI.		
30	<pre>void FRAM_DPIEN(void);</pre>	Changes the operating mode of the F- RAM and controller to Dual SPI.	These functions are self-sufficient. No other register bits are affected by these functions.	
31	<pre>void FRAM_SPIEN(void);</pre>	Changes the operating mode of the F- RAM and controller to Standard SPI.		
32	<pre>void FRAM_QUAD_ENABLE(void);</pre>	Sets the QUAD bit.	The QUAD bit must be set prior to executing any of the extended SPI opcodes. Refer to F-RAM datasheet for details. This function is self-sufficient. No other register bits are affected by this function.	
33	<pre>void FRAM_QUAD_DISABLE(void);</pre>	Clears the QUAD bit.	This function is self-sufficient. No other register bits are affected by this function.	



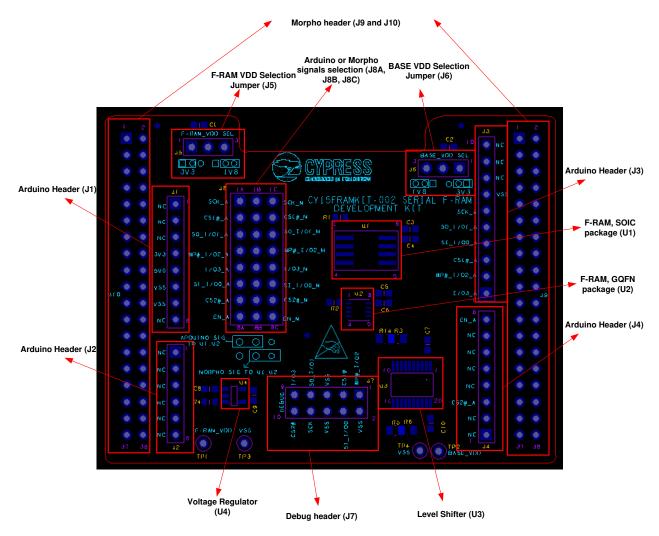


CY15FRAMKIT-002 Kit Block Diagram



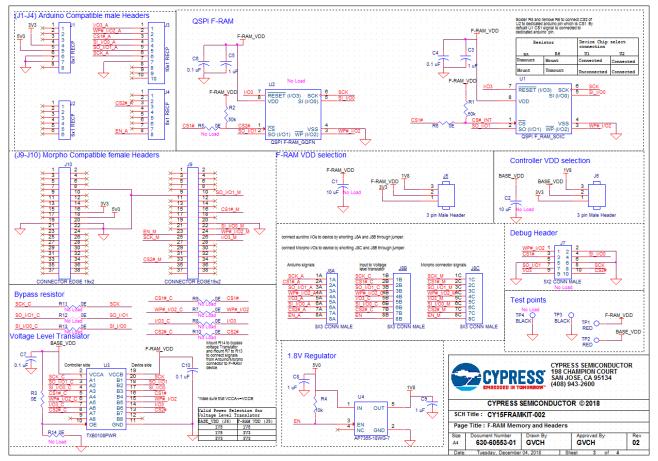


CY15FRAMKIT-002 Kit Component Placements





CY15FRAMKIT-002 Kit Schematic





Pin Assignment Table

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

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

	J1 Connector (8 pins)			
Pin	Arduino Signal	ST Controller Signals	CY15FRAMKIT-002 Kit Signals	
J1_1	NC	-	NC	
J1_2	IOREF	-	NC	
J1_3	NRST	NRST	NC	
J1_4	3V3	-	3V3	
J1_5	5V	-	5V0	
J1_6	GND	-	GND	
J1_7	GND	-	GND	
J1_8	VIN	-	NC	

	J2 Connector (6 pins)			
Pin	Arduino Signal	ST Controller Signals	CY15FRAMKIT-002 Kit Signals	
J2_1	A0	PA0	NC	
J2_2	A1	PA1	NC	
J2_3	A2	PC3	NC	
J2_4	A3	PC2	NC	
J2_5	A4	PC1	NC	
J2_6	A5	PC0	NC	

	J3 Connector (10 pins)			
Pin	Arduino Signal	ST Controller Signals	CY15FRAMKIT-002 Kit Signals	
J3_1	D8	PB6	I/O3_A	
J3_2	PWM / D9	PA8	WP#_I/O2_A	
J3_3	PWM / CS / D10	PA11	CS1#_A	
J3_4	PWM / MOSI/D11	PB15	SI_I/O0_A	
J3_5	MISO / D12	PB14	SO_I/O1_A	
J3_6	SCK / D13	PB13	SCK_A	
J3_7	GND	-	GND	
J3_8	AVDO	-	NC	
J3_9	SDA / D14	PB7	NC	
J3_10	SCL / D15	PB8	NC	

	J4 Connector (8 pins)			
Pin	Arduino Signal	ST Controller Signals	CY15FRAMKIT-002 Kit Signals	
J4_1	RX/D0	PA3/PA10	NC	
J4_2	TX/D1	PA2/PA9	NC	
J4_3	D2	PA12	CS2#_A	
J4_4	PWM / D3	PB3	NC	
J4_5	D4	PB5	NC	



	J4 Connector (8 pins)			
Pin	Pin Arduino Signal ST Controller Signals		CY15FRAMKIT-002 Kit Signals	
J4_6	PWM/D5	PA15	NC	
J4_7	PWM/D6	PB10	NC	
J4_8	D7	PC7	EN_A	

Morpho-Compatible Headers (J9, J10)

	J9 Connector (38 pins)			
Pin	ST Controller Signals	CY15FRAMKIT-002 Kit Signals		
J9_1	PC9	NC		
J9_2	PC8	NC		
J9_3	PB8	NC		
J9_4	PC6	NC		
J9_5	PB7	NC		
J9_6	PC5	NC		
J9_7	AVDD	NC		
J9_8	5V-STLINK	NC		
J9_9	GND	GND		
J9_10	PB0	SO_I/O1_M		
J9_11	PB13	NC		
J9_12	PA10	NC		
J9_13	PB14	NC		
J9_14	PA9	NC		
J9_15	PB15	NC		
J9_16	PB11	CS1#_M		
J9_17	PA11	NC		
J9_18	PB2	NC		
J9_19	PA8	NC		
J9_20	GND	GND		
J9_21	PB6	NC		
J9_22	PB1	SI_I/O0_M		
J9_23	PC7	EN_M		
J9_24	PA7	WP#_I/O2_M		
J9_25	PB10	SCK_M		
J9_26	PA6	I/O3_M		
J9_27	PA15	NC		
J9_28	PA5	NC		
J9_29	PB5	NC		
J9_30	PA4	NC		
J9_31	PB3	NC		
J9_32	AGND	NC		
J9_33	PA12	CS2#_M		
J9_34	PC4	NC		
J9_35	PA2/PA9	NC		
J9_36	PA3	NC		



	J9 Connector (38 pins)			
Pin	Pin ST Controller CY15FRAMKIT-002 Kit Signals			
J9_37	PA3/PA10	NC		
J9_38	PA2	NC		

J10 Connector (38 pins)			
Pin	ST Controller Signals	CY15FRAMKIT-002 Kit Signals	
J10_1	PC10	NC	
J10_2	PC11	NC	
J10_3	PC12	NC	
J10_4	PD2	NC	
J10_5	VDD	NC	
J10_6	E5V	NC	
J10_7	BOOT0	NC	
J10_8	GND	GND	
J10_9	NC	NC	
J10_10	NC	NC	
J10_11	NC	NC	
J10_12	IOREF	NC	
J10_13	PB12	NC	
J10_14	NRST	NC	
J10_15	PA13	NC	
J10_16	3V3	3V3	
J10_17	PA14	NC	
J10_18	5V	5V0	
J10_19	GND	GND	
J10_20	GND	GND	
J10_21	NC	NC	
J10_22	GND	GND	
J10_23	PC13	NC	
J10_24	VIN	NC	
J10_25	PC14	NC	
J10_26	NC	NC	
J10_27	PC15	NC	
J10_28	PA0	NC	
J10_29	PH0	NC	
J10_30	PA1	NC	
J10_31	PH1	NC	
J10_32	PC3	NC	
J10_33	VBAT	NC	
J10_34	PC2	NC	
J10_35	PB4	NC	
J10_36	PC1	NC	
J10_37	PB9	NC	
J10_38	PC0	NC	



Debug Header I/Os

Pin	Debug Signals
J7_1	WP#_I/O2
J7_2	GND
J7_3	CS1#
J7_4	SI_I/O0
J7_5	GND
J7_6	GND
J7_7	SO_I/O1
J7_8	SCK
J7_9	I/O3
J7_10	CS2#

Use of Zero-ohm and No-Load Resistors

The current board contains a level shifter (U3) used on the F-RAM signals to allow interfacing the device to base boards in a wide operating voltage (from 1.8 V to 3.3 V). This in turn reduces the maximum operating frequency of the interface due to propagation delay. To achieve a higher operating frequency, mount zero-ohm resistors R7-R13 and disable level shifter (U3) by mounting the R14 resistor and unmounting the R3 resistor. Note that the absolute maximum QSPI frequency that can be achieved with a Nucleo-L433RC-P board is restricted to 60 MHz.

Resistor	Debug Signals	Usage		
R7	WP#_I/O2	Solder R7 which will connect signals from J8 (J8_1B – J8_8B) to F-RAM Device (U1, U2) directly		
R8	I/O3	Solder R8 which will connect signals from J8B (J8_1B – J8_8B) to F-RAM Device (U1, U2) directly		
R9	CS#	Solder R9 which will connect signals from J8B (J8_1B – J8_8B) to F-RAM Device (U1, U2) directly		
R10	CS1#	Solder R10 which will connect signals from J8B (J8_1B – J8_8B) to F-RAM Device (U1, U2) directly		
R11	SCK	Solder R11 which will connect signals from J8B (J8_1B – J8_8B) to F-RAM Device (U1, U2) directly		
R12	SO_I/01	Solder R12 which will connect signals from J8B (J8_1B – J8_8B) to F-RAM Device (U1, U2) directly		
R13	SI_I/O0	Solder R13 which will connect signals from J8B (J8_1B – J8_8B) to F-RAM Device (U1, U2) directly		



Bill of Materials (BOM)

Item	Quantity	Reference	Part	Description	Manufacturer	Manufacturing part_number
1	4	C3,C5,C8,C9	1 uF	CAP CER 1UF 25V X5R 0603	Taiyo Yuden	TMK107BJ105KA-T
2	4	C4,C6,C7,C10	0.1 uF	CAP CER 0.1UF 16V X7R 0603	Murata Electronics	GRM188R71C104KA01D
3	2	2 J1,J4 8x1 RECP		CONN RCPT .100" 8POS SNGL TIN	Samtec inc	SSQ-108-03-T-S
4	1	J2	6x1 RECP	CONN RCPT .100" 6POS SNGL TIN	Samtec Inc	SSQ-106-03-T-S
5	1	13	10x1 RECP	CONN RCPT .100" 10POS SNGL TIN	Samtec Inc	SSQ-110-03-T-S
6	2	15,16	3 pin Male Header	CONN HEADER VERT SGL 3POS GOLD	3M	961103-6404-AR
7	1	81	8X3 CONN MALE	CONN HEADER 24POS .100" AU	Samtec Inc	TSW-108-07-G-T
8	1	9	CONNECTOR EDGE19x2	CONN HEADER FMAL 38PS.1" DL GOLD	Sullins Connector Solutions	PPPC192LFBN-RC
9	1	J10	CONNECTOR EDGE 19x2	CONN HEADER FMAL 38PS.1" DL GOLD	Sullins Connector Solutions	PPPC192LFBN-RC
10	2	R1,R2	50k	RES SMD 51K OHM 5% 1/10W 0603	Panasonic-ECG	ERJ-3GEYJ513V
11	2	R3,R6	OE	RES SMD 0 OHM JUMPER 1/8W 0805	Panasonic - ECG	ERJ-6GEY0R00V
12	1	R4	10k	RES SMD 10K OHM 5% 1/10W 0603	Panasonic - ECG	ERJ-3GEYJ103V
13	2	TP1,TP2	RED	TEST POINT PC MINI .040"D RED	Keystone Electronics	5000
14	1	TP3	BLACK	TEST POINT PC MINI .040"D BLACK	Keystone Electronics	5001
15	1	U1	QSPLF RAM SOIC	F-RAM 4Mb F-RAM 108 MHz QSPI	Cypress Semiconductor	CY15B104QSN-108SXI
16	1	U3	TXB0108PWR	IC TRNSLTR BIDIRECTIONAL 20TSSOP	Texas Instruments	TXB0108PWR
17	1	U4	AP7365-18WG-7	IC REG LIN POS ADJ 600MA SOT25-5	Diodes Incorporated	AP7365-18WG-7
Special 、	Jumper li	nstallation Instructions				
. 17	1	15	Install jumper across	HW, CONN, Rectangular, MINI JUMPER,	Sullins Connector Solutions	STC02SYAN
			pins 1 and 2	6.5mm, CLOSE TYPE, BLACK, NICKEL		
18	1	16	Install jumper across	HW, CONN, Rectangular, MINI JUMPER,	Sullins Connector Solutions	STC02SYAN
	_		pins 1 and 2	6.5mm, CLOSE TYPE, BLACK, NICKEL		
19	8	81	Install jumper across	HW, CONN, Rectangular, MINI JUMPER,	Sullins Connector Solutions	STC02SYAN
	-	-	J8B.1 and J8C.1	6.5mm, CLOSE TYPE, BLACK, NICKEL		
			Install jumper across	,		
			J8B.2 and J8C.2			
			Install jumper across			
			J8B.3 and J8C.3			
			Install jumper across			
			J8B.4 and J8C.4			
			Install jumper across			
			J8B.5 and J8C.5			
			Install jumper across			
			J8B.6 and J8C.6			
			Install jumper across			
			J8B.7 and J8C.7			
			Install jumper across			
			J8B.48and J8C.8			
			Stackable Header			
			should be placed in the			
30	1	N/A	Foam insert	Arduino Stackable Header Kit - R3	Sparkfun	PRT-11417
50	1	IN/A	Foantinsen	Ardunio Stackable Header Kit - KS	Sparkruff	PK1-11417
No load	lo load components					
31		C1,C2	10 uF	CAP CER 10UF 16V X5R 0603	Yageo	CC0603KRX5R7BB106
31		J7	5X2 CONN MALE		Yageo Amphenol FCI	67997-410HLF
32			0E	CONN HEADER 10POS .100 STR TIN		
		R5,R7,R8,R9,R10,R11,R12,R13,R14		RES SMD 0 OHM JUMPER 1/8W 0805	Panasonic - ECG	ERJ-6GEY0R00V
34		TP4	BLACK	TEST POINT PC MINI .040"D BLACK	Keystone Electronics	5001
35	1	U2	QSPI F-RAM_GQFN	F-RAM 4Mb F-RAM 108 MHz QSPI	Cypress Semiconductor	CY15B104QSN-108LPXI

CY15FRAMKIT-002 Board - Jumper Details

Headers to select Arduino connector signal connectivity to F-RAM device (J8 (1A-8A), J8 (1B-8B))

Place jumper on J8 (1A-8A) and J8 (1B-8B) horizontally

Pin	Pin	CY15FRAMKIT-002 Kit Signals (Arduino Signal)
J8_1A	J8_1B	SCK_A
J8_2A	J8_2B	CS1#_A
J8_3A	J8_3B	SO_I/O1_A
J8_4A	J8_4B	WP#_I/O2_A
J8_5A	J8_5B	I/O3_A
J8_6A	J8_6B	SI_I/O0_A
J8_7A	J8_7B	CS2#_A
J8_8A	J8_8B	EN_A



Headers to select Morpho connector signal connectivity to F-RAM device (J8 (1B-8B), J9 (1C-8C))

Place jumper on J8 (1B-8B) and J8 (1C-8C) horizontally

Pin	Pin	CY15FRAMKIT-002 Kit Signals (Morpho Signal)
J8_1B	J8_1C	SCK_M
J8_2B	J8_2C	CS1#_M
J8_3B	J8_3C	SO_I/O1_M
J8_4B	J8_4C	WP#_I/O2_M
J8_5B	J8_5C	I/O3_M
J8_6B	J8_6C	SI_I/O0_M
J8_7B	J8_7C	CS2#_M
J8_8B	J8_8C	EN_M

B. Appendix



This section is applicable only if the user includes the *guiMENU* (.*c/.h*) files in the project. The API provided in these files allows for testing the QSPI F-RAM device mounted on the CY15FRAMKIT-002 board by sending commands over the serial interface to the 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 the controller.

GUI Menu

For ease of evaluating the features of the QSPI F-RAM, the function void GUI_Menu(void); has been provided. The example project provided with the drivers calls this function in an infinite loop. It allows you to send appropriate commands over the serial interface to communicate with the F-RAM. The following table summarizes the command set for the GUI.

Command/ Opcode	Description	Notes
	QSPI Opcode based commands	3
RDSR1 (05)	Prints out SR1 value on the terminal. Stores the read value in the operating_mode structure.	
RDSR2 (07)	Prints out SR2 value on the terminal. Stores the read value in the operating_mode structure.	
RDCR1 (35)	Prints out CR1 value on the terminal. Stores the read value in the operating_mode structure.	
RDCR2 (3F)	Prints out CR2 value on the terminal. Stores the read value in the operating_mode structure.	
RDCR4 (45)	Prints out CR4 value on the terminal. Stores the read value in the operating_mode structure.	
RDCR5 (5E)	Prints out CR5 value on the terminal. Stores the read value in the operating_mode structure.	
WRAR (71)	Write Any register.	The function waits for the user input on register address and new register value. The register address value should range from 00 to 06 in decimal. Data to be written in the register must be in Hexadecimal. See the device datasheet for details of legal register values.
RDAR (65)	Read Any register.	The function waits for user input on register address. The register value is printed out on the terminal. The register address value should range from 00 to 06 in decimal.
DID (9F)	Read ID register and stores the value in the operating_mode structure.	Default value is 0x5051820600000000.
RUID (4C)	Read Unique ID register and stores the value in the operating_mode structure.	
WRSN (C2)	Update Serial Number register.	The function waits for 8-bytes of new serial number.Enter 8 bytes in hexadecimal separated by space between each byte. For example: <i>Opcode/Command: c2</i> <i>Current SN Reg: 1123445677080c12</i> <i>Enter New SN Value: 11 22 33 44 55 66 77 88</i>

Entering "000" in a terminal window will provide a summary of supported functions.



Command/ Opcode	Description	Notes	
	QSPI Opcode based commands	5	
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 a write operation to the device using the	Refer to datasheet for details on Write and Read	
FAST_WRITE (DA)	opcode that is used as a command. Write functions will trigger user input for address location, data	functionality of the F-RAM. All the address fields are in decimal.	
DIW (A2)	pattern, and the number of bytes to be written.		
DIOW (A1)	-		
QIW (32)			
QIOW (D2)			
DDRWRITE (DE)	Not supported in Revision 1.0.		
DDRFWRITE (DD)			
DDRQIOW (D1)			
READ (03)	Performs a read operation on the device using the		
FAST_READ (0B)	opcode that is used as a command. Read functions will trigger user input for address location, data pattern, and the number of bytes to be read.		
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	
"222"	Changes the operating mode of device to Dual SPI.	register latency settings before performing Read operations.	
"111"	Changes the operating mode of device to Standard SPI.		
		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 I 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.	



Appendix

Figure 16. GUI Menu

code: 000 llowing Commands aı	e 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 9F	Read Any Register
63	Read ID Register Read Serial Number Register
C2	Write Serial Number Register.
62	
	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 ØxDA
A2	Initiate Memory Write in DIW with Opcode 0xA2
A1	Initiate Memory Write in DIOW with Opcode Ox
32	Initiate Memory Write in QIW with Opcode 0x32
D2	Initiate Memory Write in QIOW with Opcode ØxI
03	Initiate Memory Read with Opcode 0x03
ØB	Initiate Memory Fast Read with Opcode ØxØB
3B	Initiate Memory Read in DOR with Opcode 0x3B
BB	Initiate Memory Read in DIOR with Opcode ØxBl
6B	Initiate Memory Read in QOR with Opcode 0x6B
EB	Initiate Memory Read in QIOR with Opcode ØxEl

******	Special Functions ************************************
Special Commands	Description
999	Performs Interface Reset.
	All Registers restored to default
111	Changes the interface mode to SPI
	for ŠT 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 Básic F-RÁM Testing



Examples of GUI commands:

Figure 17. RDSR1 - Command: 05

Enter an Opcode/Command (000 for supported Features): Opcode/Command: 05 SR1= 00

Figure 18. Read ID - Command: 9F

Enter an Opcode/Command (000 for supported Features): Opcode/Command: 9f ID Reg: 5051820600000000

Figure 19: Write/Read Example

Enter an Opcode/Command (000 for supported Features): Opcode/Command: 02 Write Address:<mark>16</mark> Length to write:8 Select data pattern 0: Solid O 1: Solid 1 2: 0xAA 3: 0x55 4: AA->55->AA->55 5: 55->AA->55->AA 6: Add=Data Waiting for Input: 4 Enter an Opcode/Command (000 for supported Features): Opcode/Command: 03 Read Address:12 Length to read:16 Data:Oc Od Oe Of <mark>aa 55 aa 55 aa 55 aa 55</mark> 18 19 1a 1b Enter an Opcode/Command (000 for supported Features): Opcode/Command:

In this write/read example, 8 bytes are written starting from address location 16. Next, a read operation is performed for 16 locations starting from address 12.

Revision History



Document Revision History

Document Title: CY15FRAMKIT-002 Serial F-RAM Development Kit User Guide			
Document Number: 002-23147 Rev. *A			
Revision	Issue Date	Origin of Change	Description of Change
**	12/14/2018	NILE/GVCH	New Spec
*A	05/23/2019	GVCH	Section 2.1: Updated J8 description Section 2.2.7 and Figure 6: Updated J8 description