# CYPRESS
## EMBEDDED IN TOMORROW™

# CY15FRAMKIT-001

## Serial F-RAM™ Development Kit Guide

**Copyrights**

# Contents

## Regulatory Compliance

The CY15FRAMKIT-001 Serial F-RAM™ Development Kit is intended for use as a serial memory development platform for hardware or software in a laboratory environment. The board is an open-system design, which does not include a shielded enclosure. Therefore, the board may cause interference with other electrical or electronic devices in close proximity.

Attaching additional wiring to this product or modifying the product operation from the factory default may affect its performance and cause interference with other apparatus in the immediate vicinity. If such interference is detected, suitable mitigating measures should be taken.

The CY15FRAMKIT-001, as shipped from the factory, has been verified to meet with the requirements of CE as a Class A product.

| | |
|---|---|
| | The CY15FRAMKIT-001 contains ESD-sensitive devices. Electrostatic charges readily accumulate on the human body and any equipment, and can discharge without detection. Permanent damage may occur to devices subjected to high-energy discharges. Proper ESD precautions are recommended to avoid performance degradation or loss of functionality. Store unused CY15FRAMKIT-001 boards in the protective shipping package. |
| | End-of-Life/Product Recycling<br><br>The end-of-life cycle for this kit is five years from the date of manufacture mentioned on the back of the box. Contact your nearest recycler to discard the kit. |

# General Safety Instructions

### ESD Protection

ESD can damage boards and associated components. Cypress recommends that the user perform procedures only at an ESD workstation. If an ESD workstation is not available, use appropriate ESD protection by wearing an antistatic wrist strap attached to the chassis ground (any unpainted metal surface) on the board when handling parts.

### Handling Boards

CY15FRAMKIT-001 boards are sensitive to ESD. Hold the board only by its edges. After removing the board from its box, place it on a grounded, static-free surface. Use a conductive foam pad if available. Do not slide the board over any surface.

# 1. Introduction

Thanks for your interest in the CY15FRAMKIT-001 Serial F-RAM Development Kit (DVK). The kit (shield) is designed as an easy-to-use and inexpensive development kit, showcasing the features of Cypress serial ferroelectric random access memory (F-RAM). Designed for flexibility, this kit offers footprint-compatibility with CY8CKIT-042 PSoC® 4 Pioneer Kit, CY8CKIT-040, CY8CKIT-042 BLE, third-party Arduino™ UNO R3, and several other Arduino kits. This board features a 256-Kbit SPI F-RAM, a 256-Kbit I$^2$C F-RAM, three-pole DIP switch to control the I$^2$C F-RAM device select pins, Arduino UNO compatible headers, and two debug headers. This kit supports power supply voltages of either 5.0 V or 3.3 V.

The CY15FRAMKIT-001 is a development kit for the Cypress Serial F-RAM memories. Cypress F-RAM is built on ferroelectric technology, which combines the nonvolatile data storage with the high-performance RAM. Serial F-RAMs provide fast writes at full bus speed. They do not have any write delays and data is instantly nonvolatile. They consume as low as 300 µA active and 6 µA standby current. Because of fast write speeds, serial F-RAMs need to be active for short periods, yielding very low energy consumption. Endurance is virtually unlimited offering 100 trillion read/write cycles.

## 1.1 Kit Contents

The CY15FRAMKIT-001 Serial F-RAM DVK includes the following contents, as shown in Figure 1-1:

- CY15FRAMKIT-001 serial F-RAM DVK board
- Quick start guide

Figure 1-1. CY15FRAMKIT-001 Serial F-RAM DVK Contents



Visit www.cypress.com/shop for more information. Inspect the contents of the kit; if any parts are missing, contact your nearest Cypress sales office for help.

## 1.2 PSoC Creator™

PSoC Creator is a state-of-the-art, easy-to-use integrated design environment (IDE). It introduces revolutionary hardware and software codesign, powered by a library of preverified and precharacterized PSoC Components™.

With PSoC Creator, you can:

- Drag and drop PSoC Components to build a schematic of your custom design

- Automatically place and route components and configure GPIOs

- Develop and debug firmware using the included component APIs

PSoC Creator also enables you to tap into an entire tools ecosystem with integrated compiler chains and production programmers for PSoC devices. For more information, visit www.cypress.com/Creator.

## 1.3 Getting Started

This guide helps you to get acquainted with the Serial F-RAM DVK. The Kit Installation chapter describes the installation of the kit software. The Kit Overview chapter explains the features of the kit. The Hardware chapter details the hardware operation. The Examples Projects chapter describes the code examples. The Appendix provides the schematics, pin assignment, use of zero-ohm and 30-ohm resistors, and the bill of materials (BOM).

## 1.4 Additional Learning Resources

Visit www.cypress.com/go/F-RAM for additional learning resources in the form of datasheets and application notes. The baseboard-related resources are available in the following links.

- CY8CKIT-042 PSoC 4 Pioneer Kit

- Arduino UNO R3

## 1.5 Technical Support

For assistance, go to our support web page, www.cypress.com/support, or contact our customer support at +1 (800) 541-4736 Ext. 2 (in the USA) or +1 (408) 943-2600 Ext. 2 (International).

## 1.6 Documentation Conventions

Table 1-1. Document Conventions for Guides

| Convention | Usage |
|---|---|
| Courier New | Displays file locations, user-entered text, and source code:<br>`C:\...cd\icc\` |
| *Italics* | Displays file names and reference documentation:<br>Read about the *sourcefile.hex* file in the *PSoC Designer User Guide*. |
| [**Bracketed, Bold**] | Displays keyboard commands in procedures:<br>[**Enter**] or [**Ctrl**] [**C**] |
| File > Open | Represents menu paths:<br>File > Open > New Project |
| **Bold** | Displays commands, menu paths, and icon names in procedures:<br>Click the **File** icon and then click **Open**. |
| Times New Roman | Displays an equation:<br>$2 + 2 = 4$ |
| Text in gray boxes | Describes cautions or unique functionality of the product. |

# 2. Kit Installation

This section describes the installation of the CY15FRAMKIT-001 Serial F-RAM DVK software and prerequisites.

## 2.1 Prerequisite Software

You must install the PSoC Creator IDE or the Arduino IDE before using the kit. All Cypress software installations require administrator privileges. Administrator privileges are not required to execute the software after installation. Close all other Cypress software that is currently running before you install the kit software.

To download the PSoC Creator IDE, go to www.cypress.com/psoccreator.

To download the Arduino IDE, go to arduino.cc/en/Main/Software.

## 2.2 Install Kit Software

Follow these steps to install the CY15FRAMKIT-001 Serial F-RAM DVK software:

1. Download the CY15FRAMKIT-001 DVK software from www.cypress.com/go/CY15FRAMKIT-001. The CY15FRAMKIT-001 software is available in three different formats (see Figure 2-1):

Figure 2-1. Available Formats for Downloading Serial F-RAM DVK Software



- **CY15FRAMKIT-001 Kit Setup**: This executable file installs only the kit contents, which include kit code examples, hardware files, and user documents.

- **CY15FRAMKIT-001 CD ISO**: This file is a complete package, stored in a CD-ROM image format, which you can use to create a CD or extract using ISO extraction programs, such as WinZip or WinRAR. The file can also be mounted like a virtual CD using virtual drive programs such as Virtual CloneDrive and MagicISO. This file includes all the required firmware, hardware files, and user documents.

2. If you have downloaded the CD ISO file, mount it on a virtual drive. Extract the ISO contents if you do not have a virtual drive on which to mount. Double-click *cyautorun.exe* in the root directory of the extracted content or mounted ISO if "Autorun from CD/DVD" is not enabled on the PC. The installation window shown in Figure 2-2 will appear automatically.

   **Note:** If you are using the Kit Setup file, then go to step 3 for installation.

3. The kit startup screen appears as shown in Figure 2-2. The default installation location is **Cypress** under **Program Files** folder.

Figure 2-2. Kit Installer Startup Screen
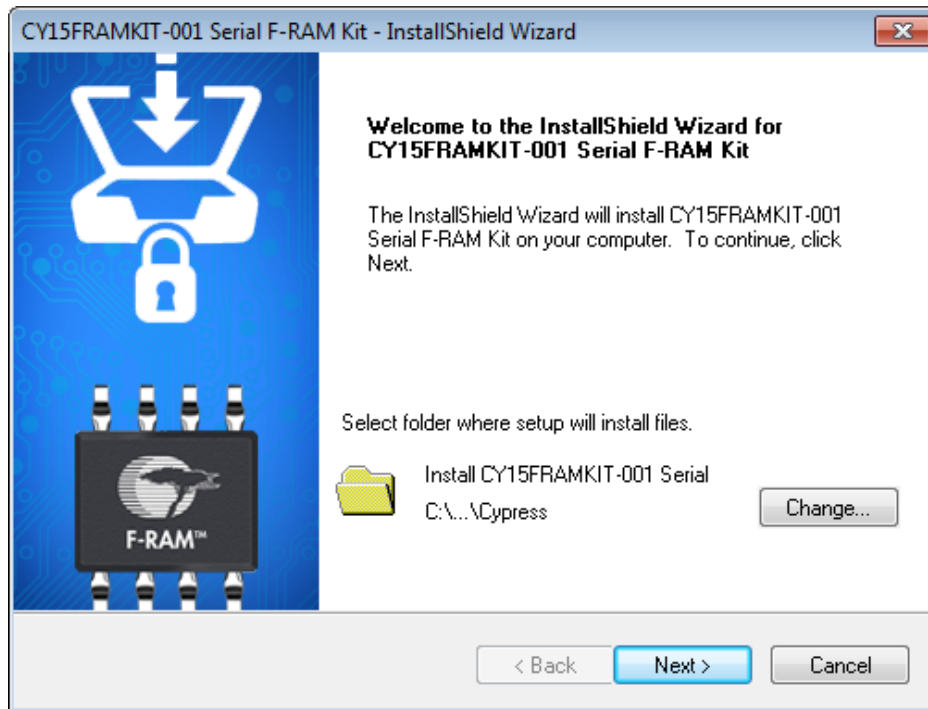


Select the folder in which you want to install the CY15FRAMKIT-001 kit-related files. Choose the directory and click **Next**.

4. Select the **Typical installation** type in the **Product Installation Overview** window, as shown in Figure 2-3. Click **Next**.

Figure 2-3. Product Installation Overview Window
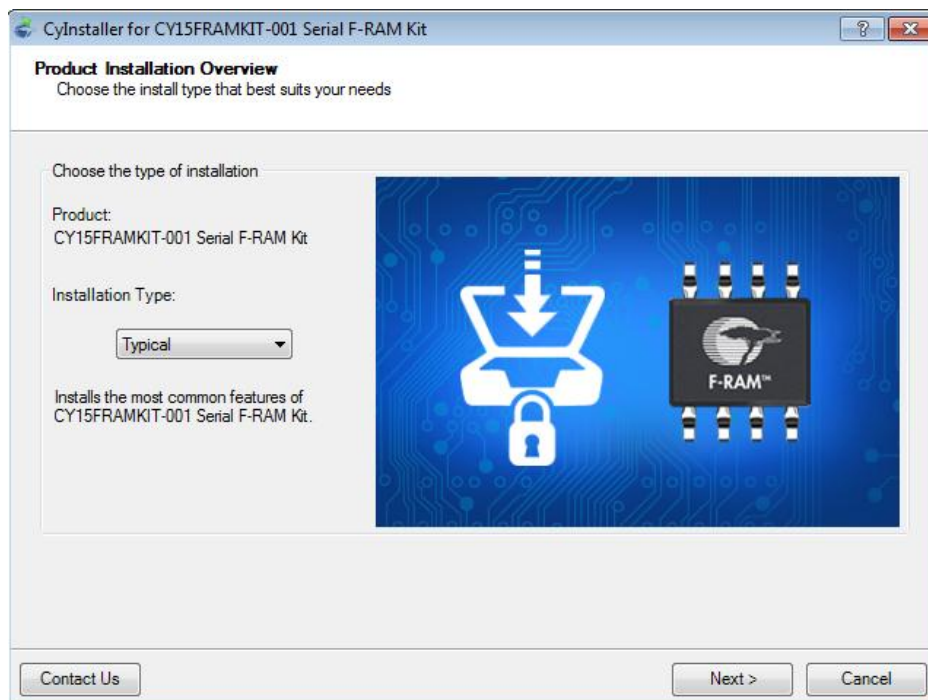


When the installation begins, a list of packages appears on the installation page. A green check mark appears next to each package after successful installation.

5. If this is the first Cypress product that you have installed, enter your contact information or select the option **Continue without Contact Information**. Click **Finish** to complete the CY15FRAMKIT-001 kit installation.

After the installation is complete, the kit contents are available at `<Install_Directory>\CY15FRAMKIT-001 Serial F-RAM Kit\1.0`. The default locations are as follows:

- Windows 7 (64-bit): C:\Program Files (x86)\Cypress\CY15FRAMKIT-001 Serial F-RAM Kit\1.0

- Windows 7 (32-bit): C:\Program Files\Cypress\CY15FRAMKIT-001 Serial F-RAM Kit\1.0

**Note:** For Windows 7/8/8.1 users, the installed files and the folder are read-only. To change the property, right-click the folder and choose **Properties** > **Attributes** and then disable the **Read-only** option. Click **Apply** and **OK** to close the window.

After the installation is complete, the following are installed on your computer:

- Kit documents
  - Quick start guide
  - Kit guide
  - Release note
- Firmware
  - PSoC 4 F-RAM SPI example project
  - PSoC 4 F-RAM I2C example project
  - Arduino UNO F-RAM SPI example project
  - Arduino UNO F-RAM I2C example project
- Hardware
  - Schematic
  - Layout
  - Gerber
  - PCB assembly drawing
  - Bill of materials (BOM)

## 2.3 Uninstall Software

The software can be uninstalled using one of the following methods:

- Go to **Start** > **Control Panel** > **Programs and Features**; select the appropriate software package and click **Uninstall**.

- Go to Start > All Programs > Cypress > Cypress Update Manager > Cypress Update Manager.
  - Select the "CY15FRAMKIT-001 Serial F-RAM Kit 1.0 Rev **" row and click **Uninstall**. In the **Product Installation Overview** window, select **Remove** from the **Installation Type** drop-down menu. Follow the instructions to uninstall.

    **Note:** This method will uninstall only the kit software and not all the other software that may have been installed along with the kit software.

# 3. Kit Overview

## 3.1 CY15FRAMKIT-001 Serial F-RAM Development Kit Overview

The CY15FRAMKIT-001 Serial F-RAM DVK can be used to understand the features of the serial F-RAMs (SPI and $I^2C$). The DVK is a shield board, which contains the 256-Kbit SPI F-RAM and 256-Kbit $I^2C$ F-RAM. It has four connectors that are Arduino UNO-compatible and connect to either Cypress PSoC 4 Pioneer Kit or Arduino UNO R3 board. These connectors are stackable and hence any other Arduino-compatible shields can be stacked on it. The kit has options to operate at either 3.3 V or 5.0 V. Figure 3-1 shows the DVK board and its general description.

Figure 3-1. General Description of DVK Board
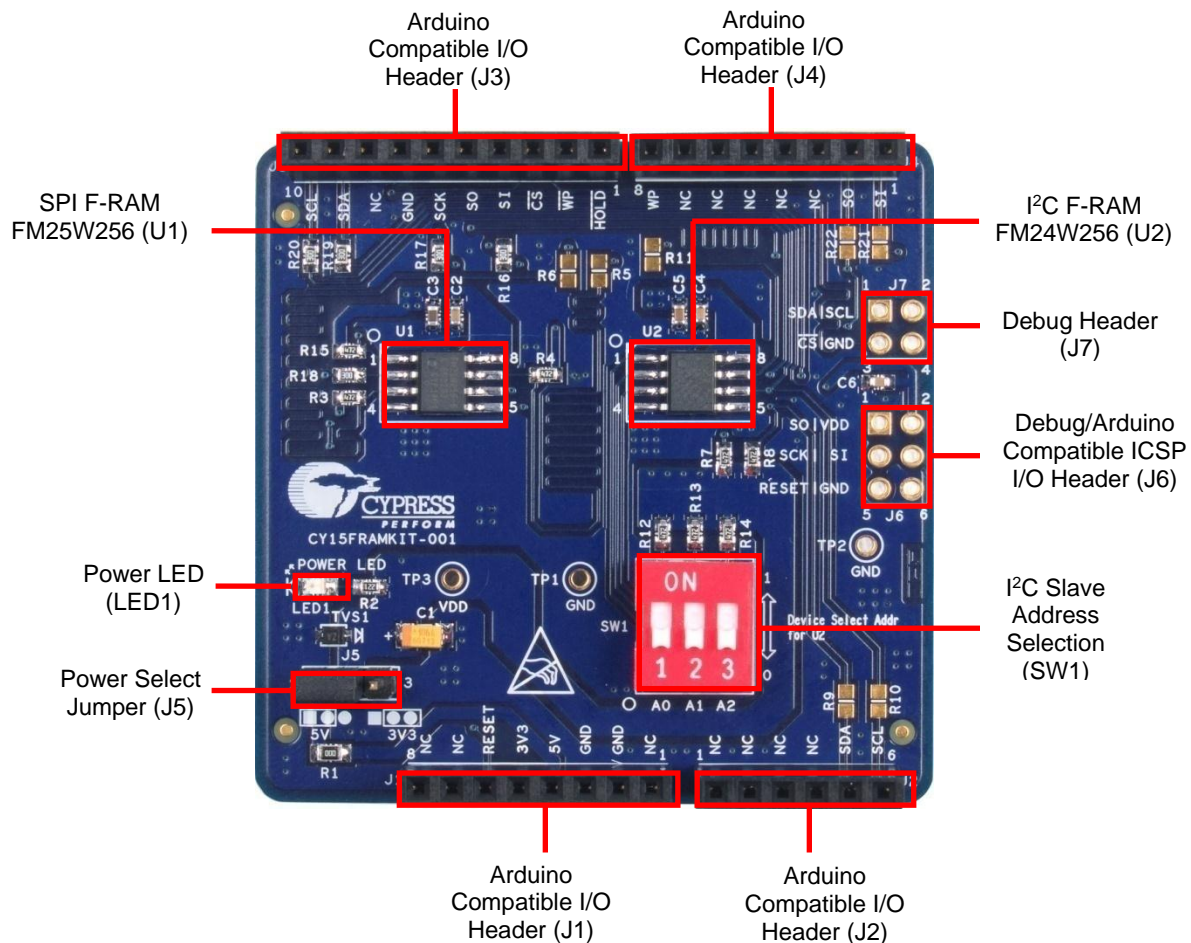


Refer to section A.5 Use of 30-ohm Resistors and No Load to modify the CY15FRAMKIT-001 hardware by mounting two 30-Ω resistors to make it Arduino UNO R2-compatible.

## 3.2 Kit Operation and Configuration Guide

### 3.2.1 Power Supply Jumper

CY15FRAMKIT-001 can operate at either 3.3 V or 5.0 V, selected through jumper J5, as shown in Figure 3-2. The factory default jumper setting is 5.0 V (short pins 1 and 2). Arduino UNO R3 boards operate at 5.0 V and use the default setting of the jumper J5. The CY8CKIT-042 Pioneer Kit operates at either 3.3 V or 5.0 V. The default setting of the Pioneer Kit is 3.3 V. Therefore, depending on the voltage setting on the mother board, set jumper J5 to either 5.0 V (short pins 1 and 2) or 3.3 V (short pins 2 and 3).

| CAUTION | 1. Do not power the CY15FRAMKIT-001 board through an external power source. The board is designed to be powered by the mother board. |
|---|---|
| | 2. The CY15FRAMKIT-001 operates from 2.7 V to 5.5 V. Exceeding the maximum voltage limit (5.5 V) can damage the board. |
| | 3. For proper operation of the board, ensure that jumper J5 is set to match the operating voltage on the motherboard. |

Figure 3-2. Power Select Jumper



Power Select Jumper (J5)
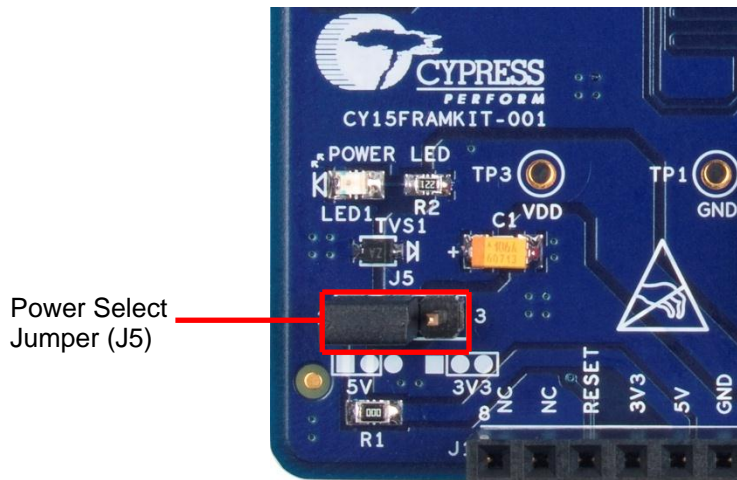
### 3.2.2 F-RAM Devices

Figure 3-3. Cypress Serial F-RAM Devices on the Board



256-Kbit SPI F-RAM (U1)

256-Kbit I$^2$C F-RAM (U2)
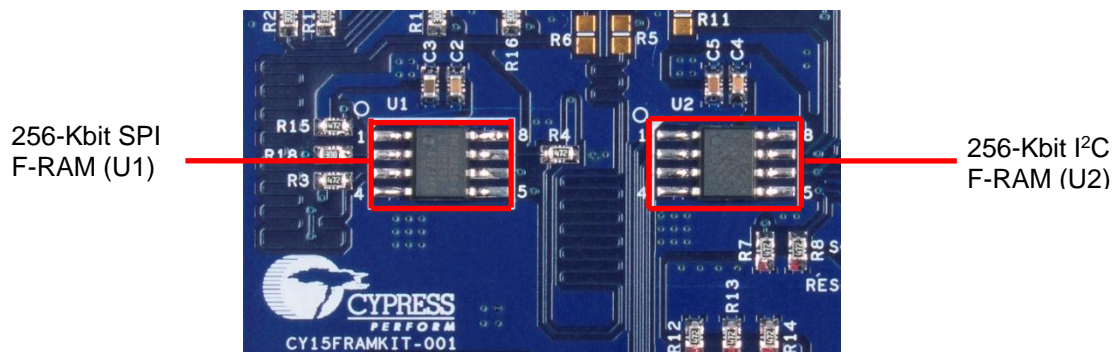
### 3.2.3 DIP Switch for I$^2$C F-RAM Device Slave Address Select

The DVK provides a DIP switch option, as shown in Figure 3-4, to configure the different slave address for the I$^2$C F-RAM by assigning its individual slave select address pins (A2-A0) either to logic LOW '0' or logic HIGH '1'. When the DIP switch is in the ON position, it keeps the respective slave select address pin to logic HIGH using a 4.7 kΩ pull-up resistor. When the DIP switch is not in the ON position, the respective slave select address pins are kept at logic LOW by F-RAM device

through an internal weak pull-down resistor. The factory default setting for the DIP switch is not in the ON position which sets the values for A2, A1, and A0 to logic LOW '0'.

Figure 3-4. DIP Switch for I²C F-RAM A0, A1, and A2 Configuration



DIP Switch (SW1)

### 3.2.4 Connectors to CY8CKIT-042/Arduino UNO R3 Board

Figure 3-5 shows the Arduino-compatible connectors to the CY8CKIT-042/Arduino UNO R3 boards: J1, J2, J3, and J4. You can plug the DVK board into the CY8CKIT-042/Arduino UNO R3 board through these connectors. For schematic details, refer to the Hardware chapter on page 17.

Figure 3-5. Connectors to CY8CKIT-042/Arduino UNO R3 Board



Arduino
Compatible I/O
Header (J3)

Arduino
Compatible I/O
Header (J4)

Arduino
Compatible I/O
Header (J1)

Arduino
Compatible I/O
Header (J2)

### 3.2.5 Debug Headers

Debug headers are provided for easy access to the SPI and I²C communication pins. As shown in Figure 3-6, jumper J6 follows the in-circuit serial programming (ICSP) header pin layout and provides access to the SPI signals. Jumper J7 provides access to the chip select of the SPI F-RAM device and I²C communication signals.

Figure 3-6. Debug Headers



Debug Header (J7)

Debug / Arduino Compatible ICSP I/O Header (J6)
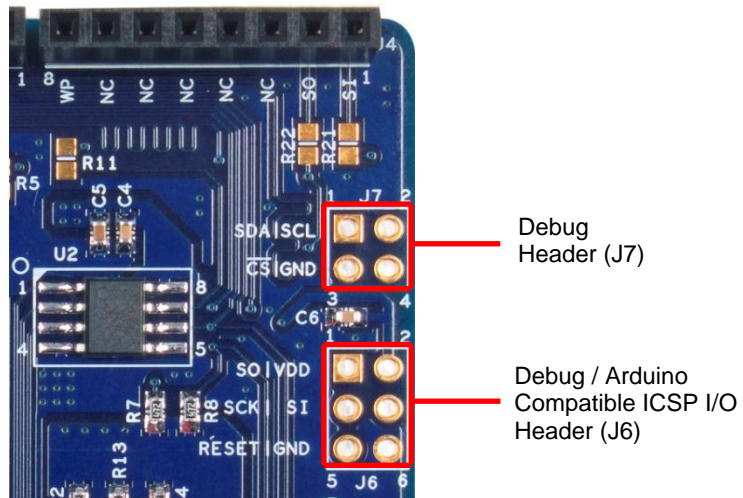
### 3.2.6 Test Points

The DVK board provides $V_{DD}$ (TP3) and GND (TP1 and TP2) test points as shown in Figure 3-7.

Figure 3-7. Test Points
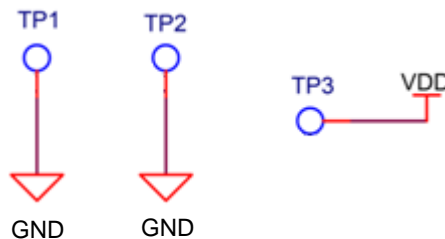


| CAUTION | These test point are only for probing and measurement purposes. Do not power the kit using these test points to avoid any damage to the board. |
|---|---|

## 3.3 CY15FRAMKIT-001 with PSoC 4 Pioneer Kit

The CY15FRAMKIT-001 is plugged into the PSoC 4 Pioneer Kit (CY8CKIT-042) through four connectors: J1, J2, J3, and J4. The default operating voltage of serial F-RAM kit is 5.0 V. To match this, change the voltage setting on the PSoC 4 pioneer kit to 5.0 V. You can also operate at 3.3 V by selecting the 3.3 V power settings on both PSoC 4 pioneer and serial F-RAM kits.

**Note**: J2 on the CY15FRAMKIT-001 is a 6-pin, single row jumper. On the PSoC 4 Pioneer Kit, J2 is a 9x2 header. Therefore, J2 of the CY15FRAMKIT-001 plugs into the six pins on the extreme left of the outer row of the J2 header on the Pioneer board. This can be easily observed because the CY15FRAMKIT-001 can be inserted only in one direction.

Figure 3-8. CY15FRAMKIT-001 Mounted on PSoC 4 Pioneer Kit
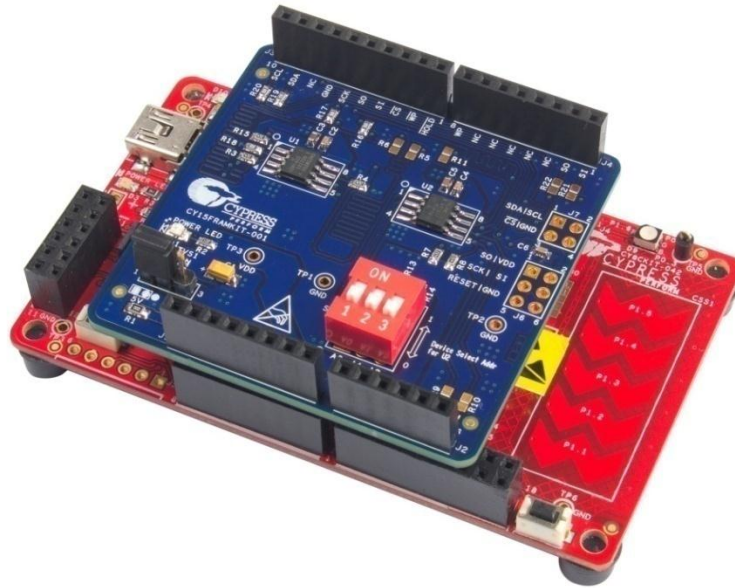


## 3.4   CY15FRAMKIT-001 with Arduino UNO R3 Kit

The CY15FRAMKIT-001 is plugged into the Arduino UNO R3 Kit through four connectors: J1, J2, J3, and J4. These connector pins are mapped one to one.

Figure 3-9. CY15FRAMKIT-001 Mounted on Arduino UNO R3 Kit
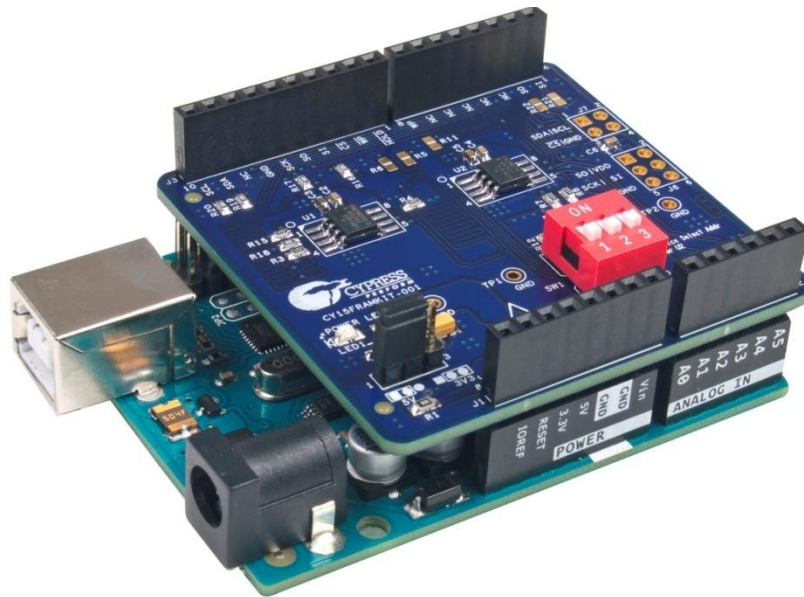
# 4. Hardware

## 4.1 Board Details

The CY15FRAMKIT-001 Serial F-RAM DVK consists of the following blocks as shown in Figure 4-1.

- 256-Kbit SPI F-RAM (U1)
- 256-Kbit I2C F-RAM (U2)
- Power select jumper (J5)
- Power LED (LED1)
- DIP switch for device address selection (SW1)
- Debug headers for SPI/I2C signals (J6, J7)
- Arduino-compatible connectors (J1, J2, J3, J4)

Figure 4-1. CY15FRAMKIT-001 Serial F-RAM DVK Details
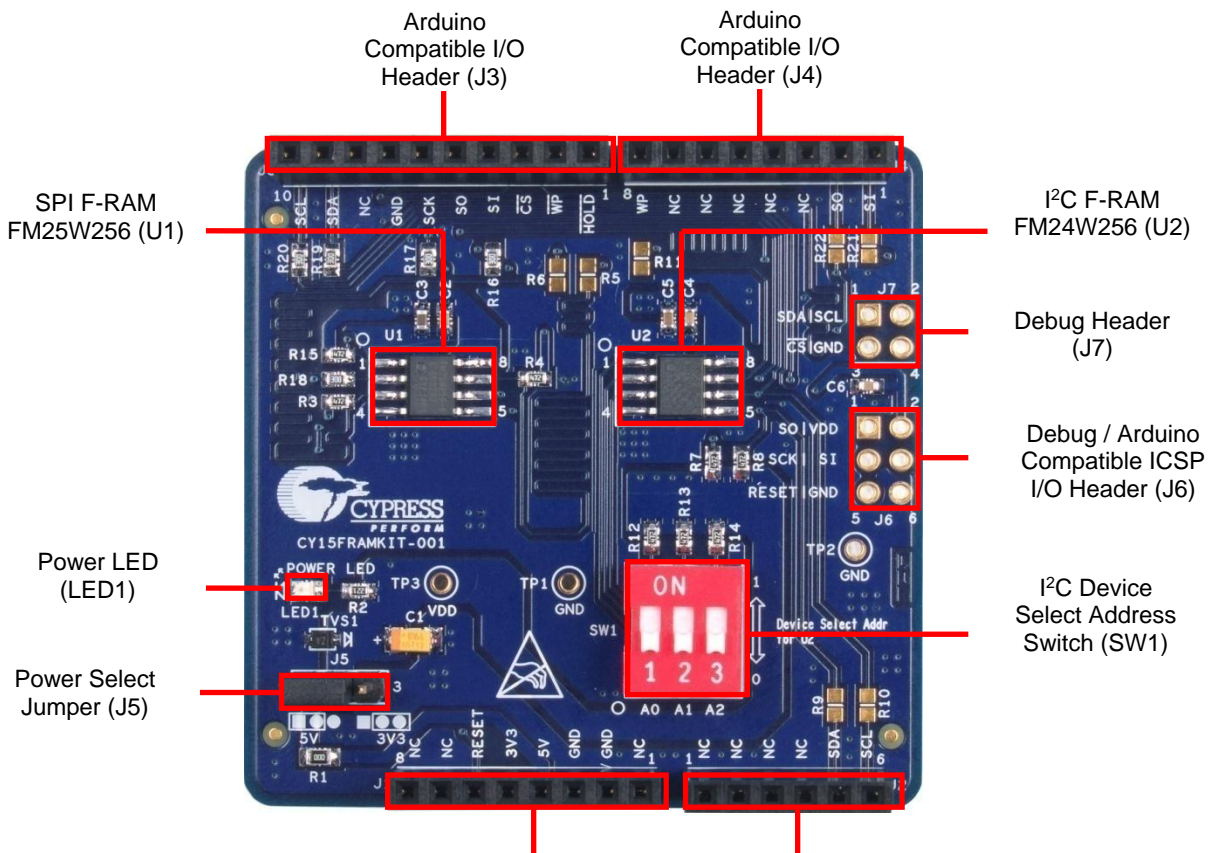


Figure 4-2. CY15FRAMK　　　DVK F　　　C 4 Pioneer/Arduino UNO R3 Kit
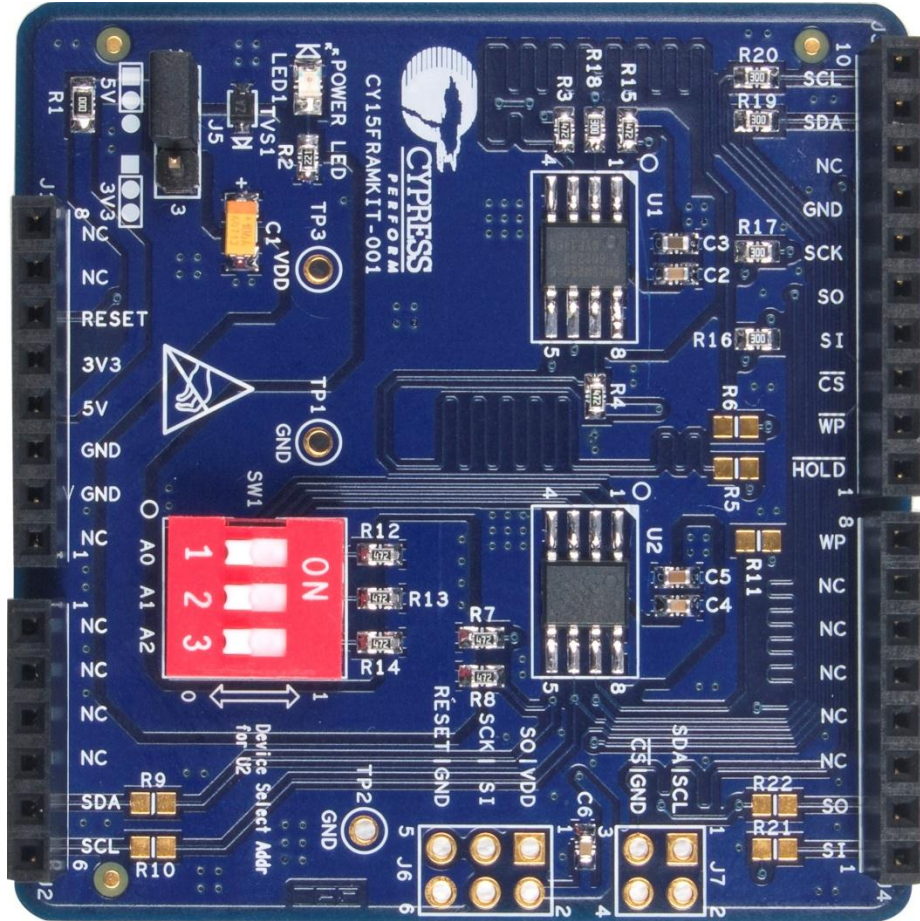
## 4.2 Theory of Operation

This section provides the block-level description of the PSoC 4 Pioneer Kit.

Figure 4-3. Block Diagram
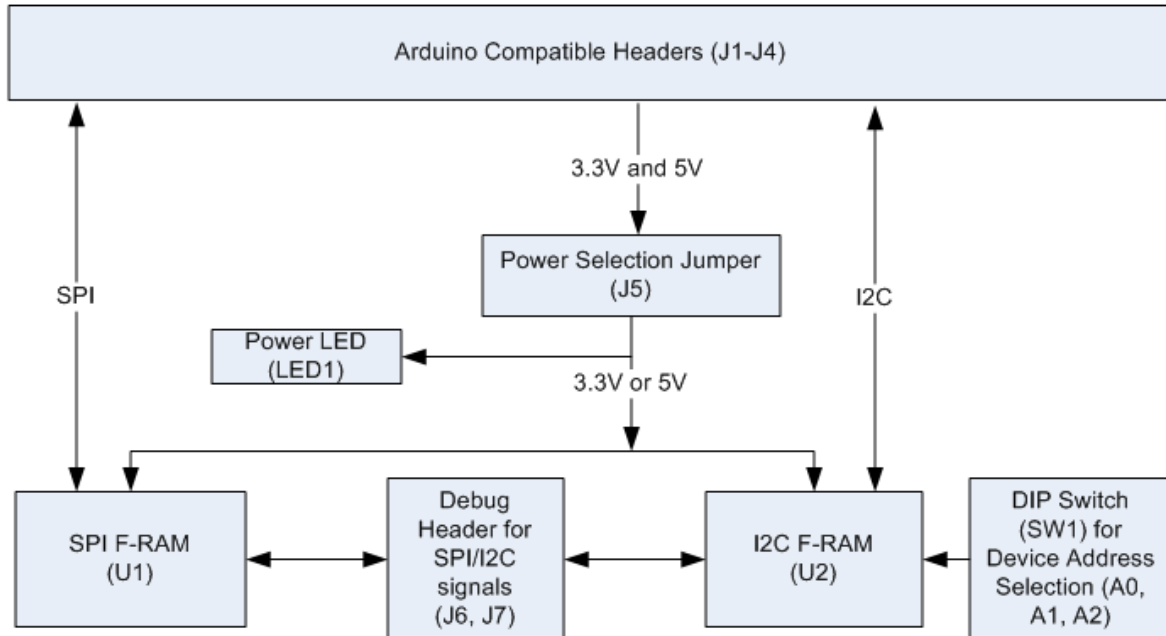


The CY15FRAMKIT-001 Serial F-RAM DVK provides 256-Kbit SPI and 256-Kbit I$^2$C F-RAM devices. The DVK is a shield board, which can be plugged into the PSoC 4 Pioneer Kit or the Arduino UNO R3 Kit.

The CY15FRAMKIT-001 Serial F-RAM DVK has a power LED (LED1) indicating the power-ON condition. The kit can operate either at 3.3 V or 5 V selected through jumper J5. The kit also has the option to change the device address for the I$^2$C F-RAM. The kit also provides debug headers for SPI / I$^2$C signals.

## 4.3 Functional Description

### 4.3.1 SPI F-RAM Device (256-Kbit FM25W256)

The CY15FRAMKIT-001 Serial F-RAM DVK uses Cypress 256-Kbit SPI F-RAM device. The device pins are accessible for reads and writes through Arduino-compatible headers.

**Features**

■ 256-Kbit ferroelectric random access memory (F-RAM) logically organized as 32K × 8

- High-endurance 100 trillion ($10^{14}$) read/writes

- 151-year data retention

- NoDelay™ writes

- Advanced high-reliability ferroelectric process

■ Fast serial peripheral interface (SPI)

- Up to 20 MHz frequency

- Direct hardware replacement for serial flash and EEPROM

- Supports SPI mode 0 (0,0) and mode 3 (1,1)

■ Sophisticated write protection scheme

- Hardware protection using the Write Protect ($\overline{\text{WP}}$) pin

- Software protection using Write Disable instruction

- Software block protection for 1/4, 1/2, or the entire array

■ Low power consumption

- 250 µA active current at 1 MHz

- 15 µA (typ) standby current

■ Wide voltage operation: VDD = 2.7 V to 5.5 V

■ Industrial temperature: –40 ºC to +85 ºC

■ 8-pin small outline integrated circuit (SOIC) package

■ Restriction of hazardous substances (RoHS) compliant

**Memory Write**

All writes to the SPI F-RAM begin with a WREN (Write Enable) opcode (0x06). The WRITE opcode (0x02) is followed by a two-byte address containing the 15-bit address (A14-A0) of the first data byte to be written into the memory. The upper bit of the two-byte address is ignored. The subsequent bytes after the two address bytes are data bytes, which are written sequentially. Addresses are incremented internally as long as the bus master continues to issue clocks and keeps $\overline{\text{CS}}$ LOW. If the last address of 7FFFh is reached, the counter will roll over to 0000h. Data is written MSB first. The rising edge of $\overline{\text{CS}}$ terminates a write operation. SPI F-RAM write operation is shown in the following figure.

Figure 4-4. SPI F-RAM Memory Write
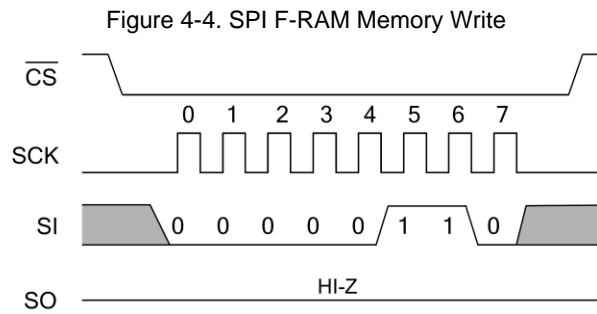
Figure 4-5. SPI F-RAM Memory Write
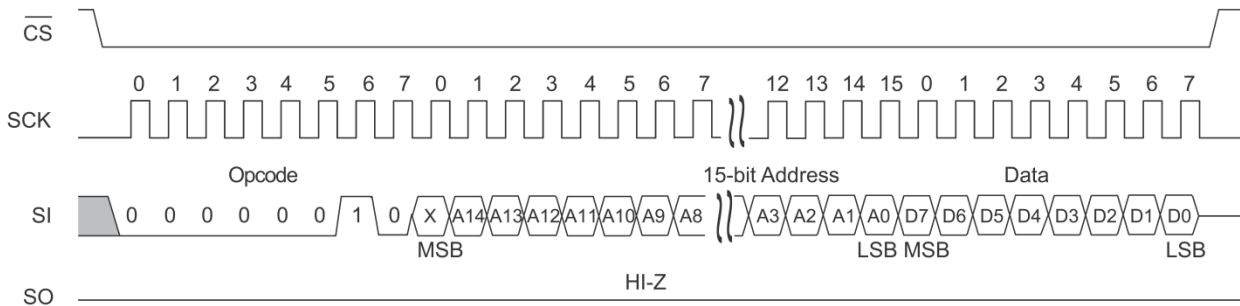
**Memory Read**

After the falling edge of $\overline{\text{CS}}$, the bus master can issue a READ opcode (0x03). Following the READ command is a two-byte address containing the 15-bit address (A14-A0) of the first byte of the read operation. The upper bit of the address is ignored. After the read opcode and two address bytes are issued, the device drives out the read data byte on the next eight clocks on SO pin. The SI input is ignored during read data bytes. Subsequent bytes are data bytes, which are read out sequentially. Addresses are incremented internally as long as the bus master continues to issue clocks and $\overline{\text{CS}}$ is LOW. If the last address of 7FFFh is reached, the counter will roll over to 0000h. Data is read MSB first. The rising edge of $\overline{\text{CS}}$ terminates a read operation and tristates the SO pin. SPI F-RAM read operation is shown in the following figure.

Figure 4-6. SPI F-RAM Memory Read
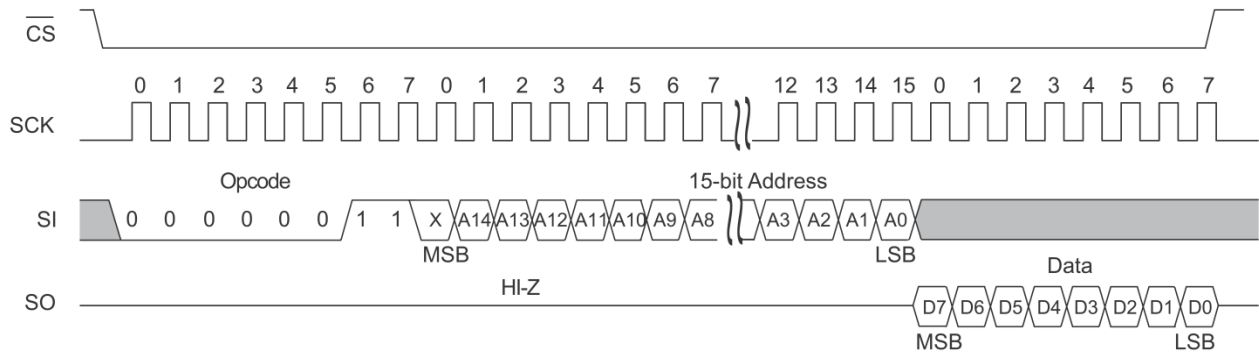


### Status Register Write

The write status register WRSR command (0x01) allows the SPI bus master to write into the status register of the SPI F-RAM.

Figure 4-7. SPI F-RAM Status Register Write
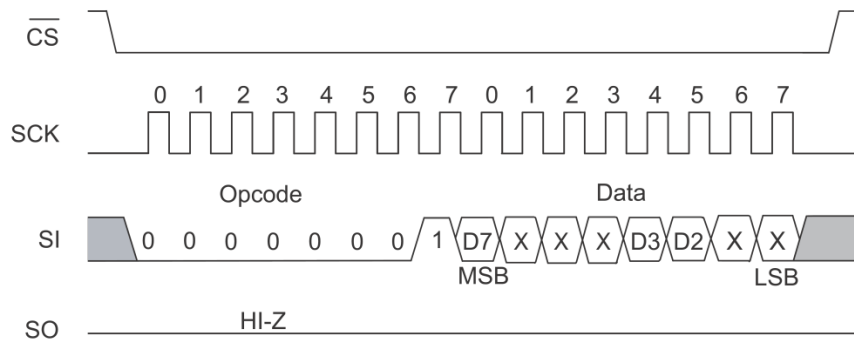


### Status Register Read

The read status register RDSR command (0x05) allows the bus master to verify the contents of the status register of the SPI F-RAM. Reading the status register provides information about the current state of the write-protection features. Following the RDSR opcode, the SPI F-RAM device will return one byte with the contents of the status register.

Figure 4-8. SPI F-RAM Status Register Read
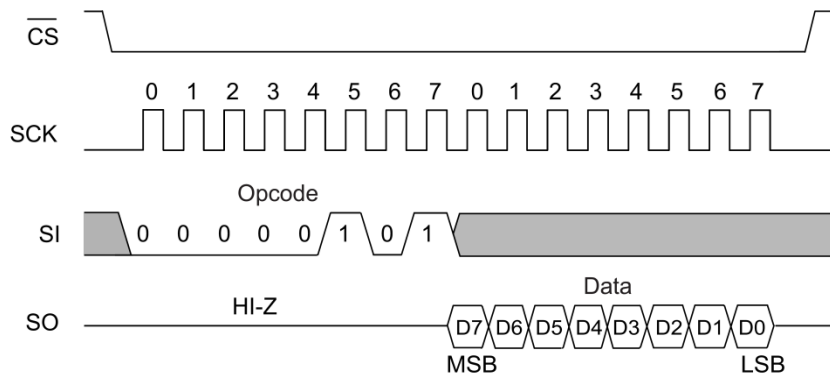


For more details on the F-RAM device, refer to the FM25W256 datasheet.

## 4.3.2  I²C F-RAM Device (256-Kbit FM24W256)

The CY15FRAMKIT-001 Serial F-RAM DVK uses the 256-Kbit I²C F-RAM device. The device is accessible for read and writes through Arduino-compatible headers.

**Features**

- 256-Kbit F-RAM is logically organized as 32K × 8

  - High-endurance 100 trillion (1014) read/writes

  - 151-year data retention

  - NoDelay™ writes

  - Advanced high-reliability ferroelectric process

- Fast 2-wire serial interface ($I^2C$)

  - Up to 1-MHz frequency

  - Direct hardware replacement for serial (I2C) EEPROM

  - Supports legacy timings for 100 kHz and 400 kHz

- Low power consumption

  - 100 µA active current at 100 kHz

  - 15 µA (typical) standby current

- Wide voltage operation: VDD = 2.7 V to 5.5 V

- Industrial temperature: –40 ºC to +85 ºC

- 8-pin small outline integrated circuit (SOIC) package

- Restriction of hazardous substances (RoHS) compliant

**Slave Device Address**

The first byte that the FM24W256 expects after a START condition is the slave address. As shown in Figure 4-9, the slave address contains the device type or slave ID, the device select address bits, and a bit that specifies if the transaction is a read or a write. Bits 7-4 are the device type (slave ID) and should be set to 1010b for the FM24W256. These bits allow other function types to reside on the $I^2C$ bus within an identical address range. Bits 3-1 are the device select address bits. They must match the corresponding value on the external address pins to select the device. Up to eight FM24W256 devices can reside on the same $I^2C$ bus by assigning a different address to each. Bit 0 is the read/write bit ($R/\overline{W}$). $R/\overline{W}$ = '1' indicates a read operation and $R/\overline{W}$ = '0' indicates a write operation.

Figure 4-9. $I^2C$ F-RAM Slave Device Address Register
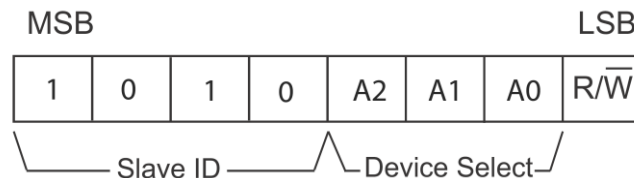


**Memory Write**

All writes begin with a slave address, then a memory address. The bus master indicates a write operation by setting the LSB of the slave address ($R/\overline{W}$ bit) to a '0'. After addressing, the bus master sends each byte of data to the memory and the memory generates an acknowledge condition. Any number of sequential bytes may be written. If the end of the address range is reached internally, the address counter will wrap from 7FFFh to 0000h.

Single-Byte Write:

Figure 4-10. I$^2$C F-RAM Single Byte Memory Write
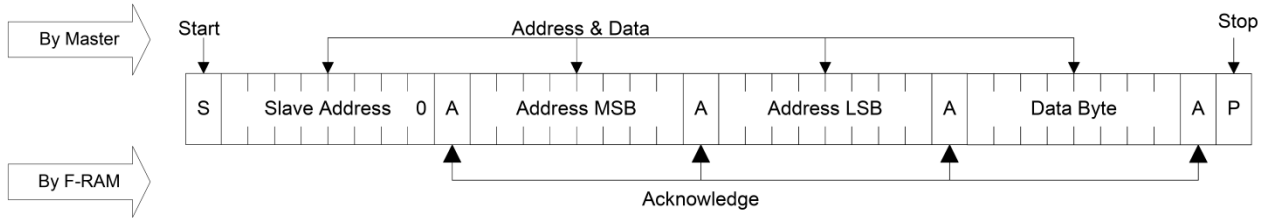


Multi-Byte Write:

Figure 4-11. I$^2$C F-RAM Multisystem Memory Write
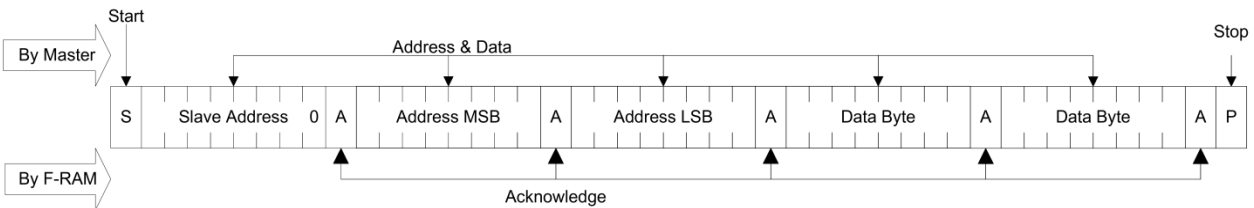


**Memory Read**

There are two basic types of read operations: current address read and selective (random) address read. In a current address read, the FM24W256 uses the internal address latch to supply the address. In a selective read, the user performs a procedure to set the address to a specific value.

*Current Address and Sequential Read*

The FM24W256 uses an internal latch to supply the address for a read operation. A current address read uses the existing value in the address latch as a starting place for the read operation. The system reads from the address immediately following that of the last operation.

To perform a current address read, the bus master supplies a slave address with the LSB set to a '1'. This indicates that a read operation is requested. After receiving the complete slave address, the FM24W256 will begin shifting out data from the current address on the next clock. The current address is the value held in the internal address latch.

Beginning with the current address, the bus master can read any number of bytes. Thus, a sequential read is simply a current address read with multiple byte transfers. After each byte the internal address counter will be incremented.

Figure 4-12. I$^2$C F-RAM Single Byte Current Address Read
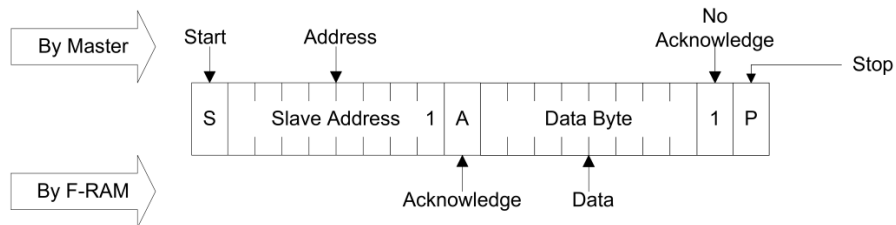


Figure 4-13. I$^2$C F-RAM Multisystem Sequential Read
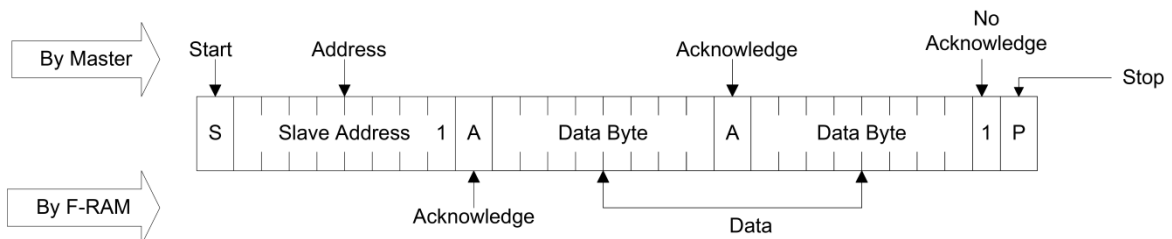
**Selective (Random) Read**

A simple technique allows a user to select a random address location as the starting point for a read operation. This involves using the first three bytes of a write operation to set the internal address followed by subsequent read operations.

To perform a selective read, the bus master sends out the slave address with the LSB (R/$\overline{W}$) set to 0. This specifies a write operation. According to the write protocol, the bus master then sends the address bytes that are loaded into the internal address latch. After the FM24W256 acknowledges the address, the bus master issues a START condition. This simultaneously aborts the write operation and allows the read command to be issued with the slave address LSB set to a '1'. The operation is now a current address read.

Figure 4-14. F-RAM I$^2$C Selective (Random) Read
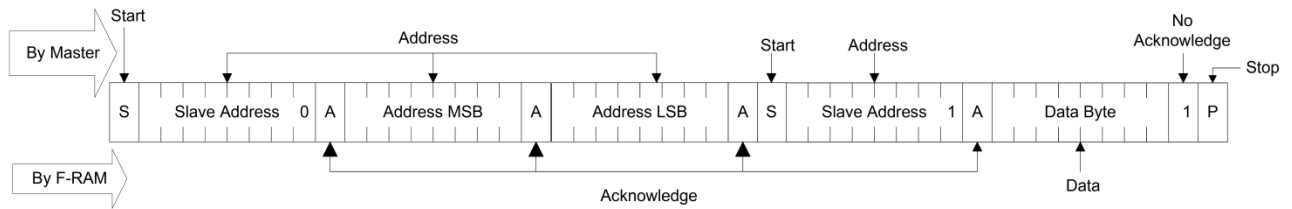
# 5. Example Projects

Cypress provides four example projects with the CY15FRAMKIT-001 Serial F-RAM DVK package. These example projects help you to understand the serial F-RAM interface with a microcontroller. The example projects are based on PSoC 4 Pioneer Kit and Arduino UNO R3 Kit.

Download the example projects from www.cypress.com/go/CY15FRAMKIT-001. The installer will place the example projects under `<Install_Directory>\CY15FRAMKIT-001 Serial F-RAM Kit\1.0\Firmware\` folder

The following example projects are PSoC 4-based and need the PSoC Creator software and the PSoC 4 Pioneer Kit.

■   PSoC4_FRAM_SPI

■   PSoC4_FRAM_I2C

The following example projects are Arduino-based and need Arduino IDE and Arduino UNO R3 Kit.

■   Arduino_FRAM_SPI

■   Arduino_FRAM_I2C

## 5.1   Programming PSoC 4 Pioneer Kit

Follow these steps to program the PSoC 4 example projects (PSoC4_FRAM_SPI project is shown here):

1.   Launch PSoC Creator from **Start > All Programs > Cypress > PSoC Creator 3.1 > PSoC Creator 3.1**. If PSoC Creator is not installed, download from www.cypress.com/psoccreator.

2.   On the Start Page, under **Examples and Kits** section, choose **Kits > CY15FRAMKIT-001**. A list of example projects appears, as shown in Figure 5-1. Click on the desired example project.

Figure 5-1. Opening Example Project from PSoC Creator

3. Select the folder where you want to save the project and click **OK**.

4. Build the code example by clicking **Build > Build <Project name>** to generate the hex file.

Figure 5-2. Build Project from PSoC Creator
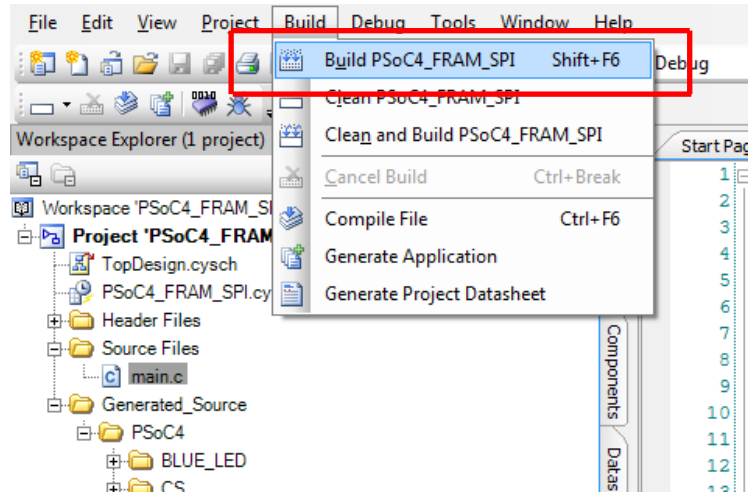


5. Select the operating voltage. CY15FRAMKIT-001 can operate at either 3.3 V or 5.0 V, selected through jumper J5. The factory default jumper setting is 5.0 V (short pins 1 and 2). Arduino UNO R3 boards operate at 5.0 V and use the default setting of the jumper J5. The CY8CKIT-042 Pioneer Kit operates at either 3.3 V or 5.0 V. The default setting of the Pioneer Kit is 3.3 V. Therefore, depending on the voltage setting on the mother board, set jumper J5 to either 5.0 V (short pins 1 and 2) or 3.3 V (short pins 2 and 3). To program PSoC 4, connect the Pioneer board to a PC using the USB cable connected to USB Mini-B connector J10 as shown in Figure 5-3.

Figure 5-3. Connect PSoC 4 Pioneer Kit to PC through USB
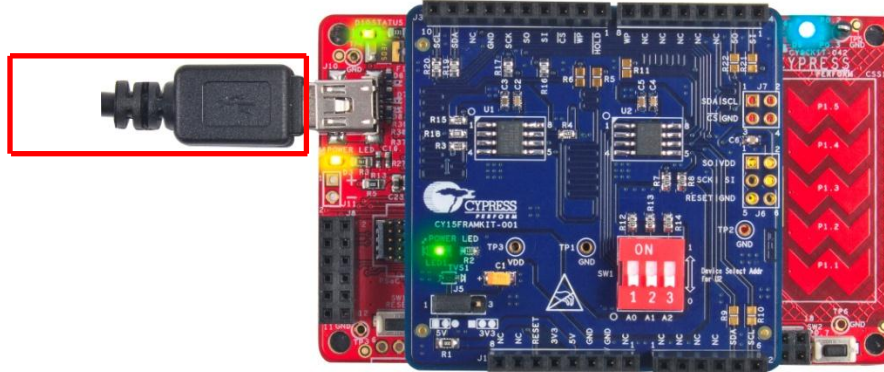


6. Click **Debug > Program** in PSoC Creator.

Figure 5-4. Program Device from PSoC Creator
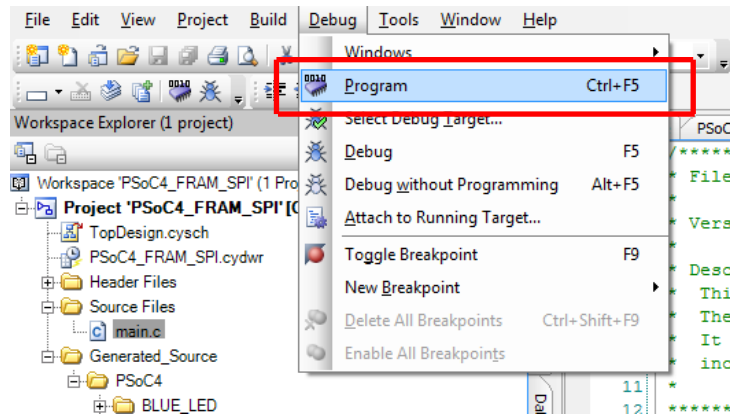


7. If the device is not yet acquired, PSoC Creator will open the programming window. Select **KitProg** and click the **Port Acquire** button.

Figure 5-5. Acquire Device from PSoC Creator
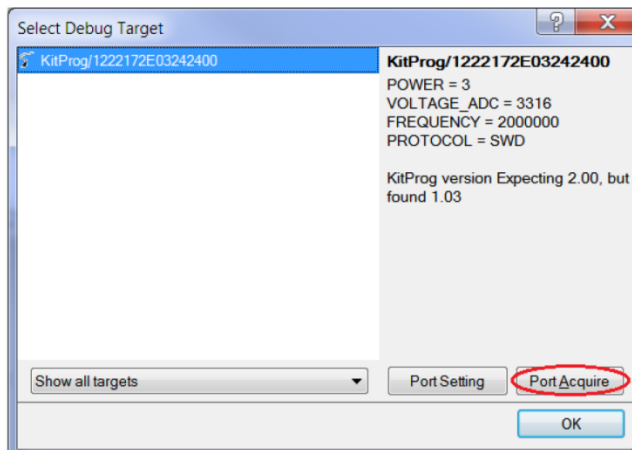


8. After the device is acquired, it is shown in a tree structure below the **KitProg**. Now, click the **Connect** button.

Figure 5-6. Connect Device from PSoC Creator
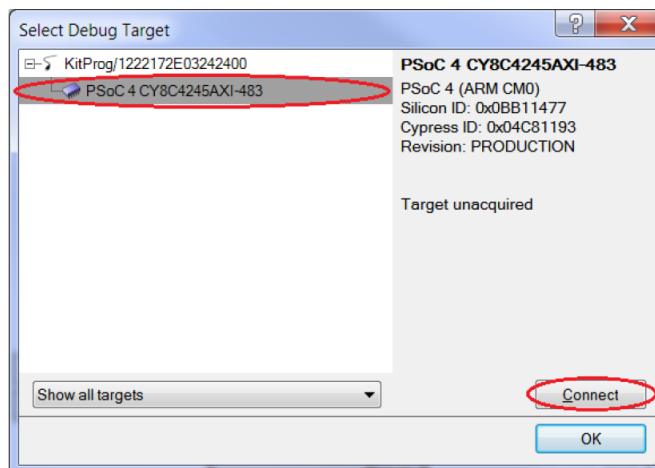


9. Click **OK** to exit the window and start programming.

Figure 5-7. Program Device from PSoC Creator
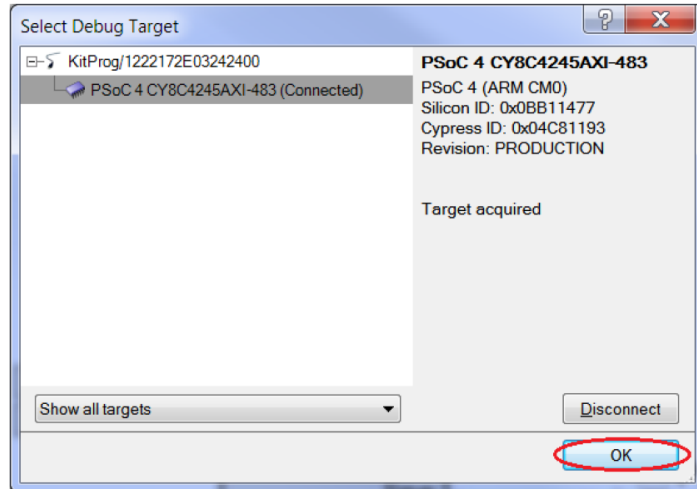


## 5.2  UART Setup

This section describes the UART interface setup to display the PSoC 4 example project output on the PC either using HyperTerminal or PuTTY as an alternative to HyperTerminal for the serial COM connection.

The PSoC 5LP device on PSoC 4 Pioneer Kit can be configured as USB to UART Bridge as shown in Figure 5-8. Both the PSoC 4 example projects include UART communication to transmit the result to PC via USB port. For more details on the hardware setup and the architecture, refer to the PSoC 4 Pioneer Kit web page.

Figure 5-8. Block Diagram of UART Connection between PSoC 4 and PSoC 5LP
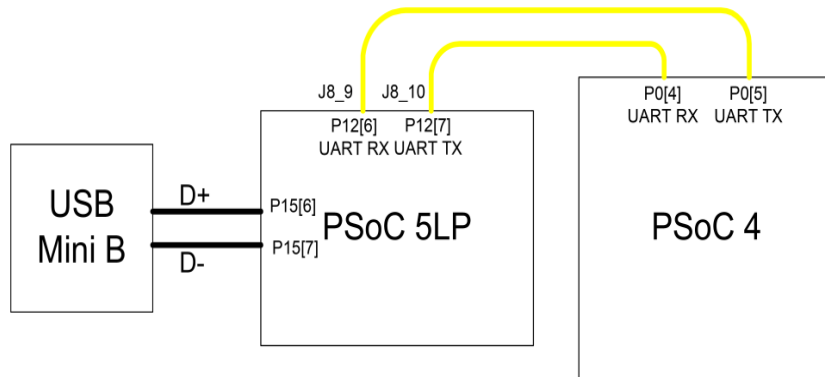


Connect the RX port of the PSoC 4, accessible through J4_1 (SI) on the CY15FRAMKIT-001 kit to J8_10 (TX). Similarly connect the TX port of PSoC 4, accessible through J4_2 (SO) on the CY15FRAMKIT-001 kit to J8_9 as shown in Figure 5-9.

Figure 5-9. UART Connection on PSoC 4 Pioneer Kit
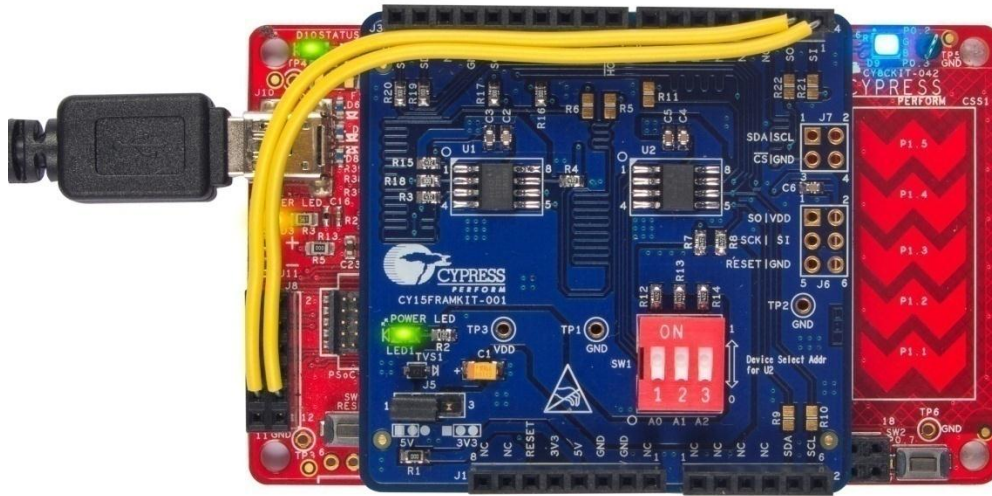


Connect USB Mini-B to J10. The kit enumerates as a **KitProg USB-UART** and is available under the **Device Manager > Ports (COM & LPT)**. A communication port is assigned to the KitProg USB-UART.

Figure 5-10. KitProg USB-UART in Device Manager
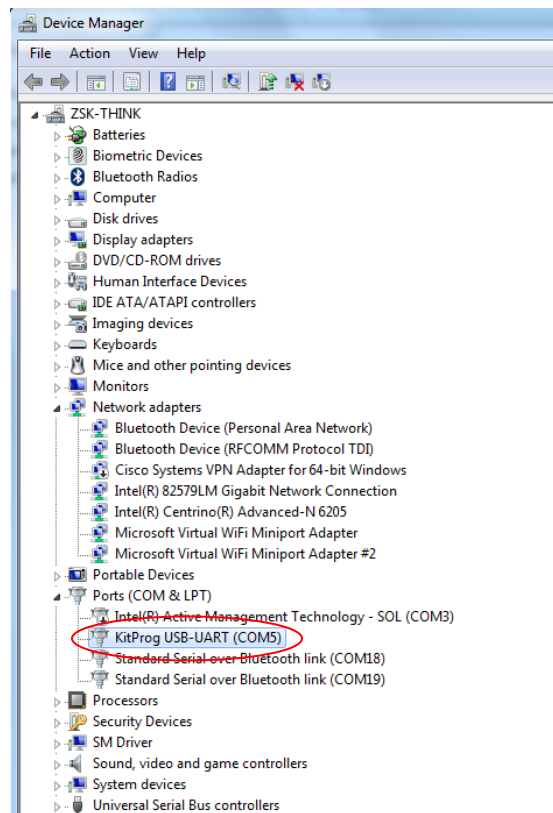


The following sections demonstrate the procedure to setup the serial connection using HyperTerminal or PuTTY on PC to communicate with the PSoC 4 Pioneer Kit.

### 5.2.1 HyperTerminal Setup

Open HyperTerminal and select **File > New Connection** and enter a "Name" for the new connection; click **OK**.

Figure 5-11. Open New Connection
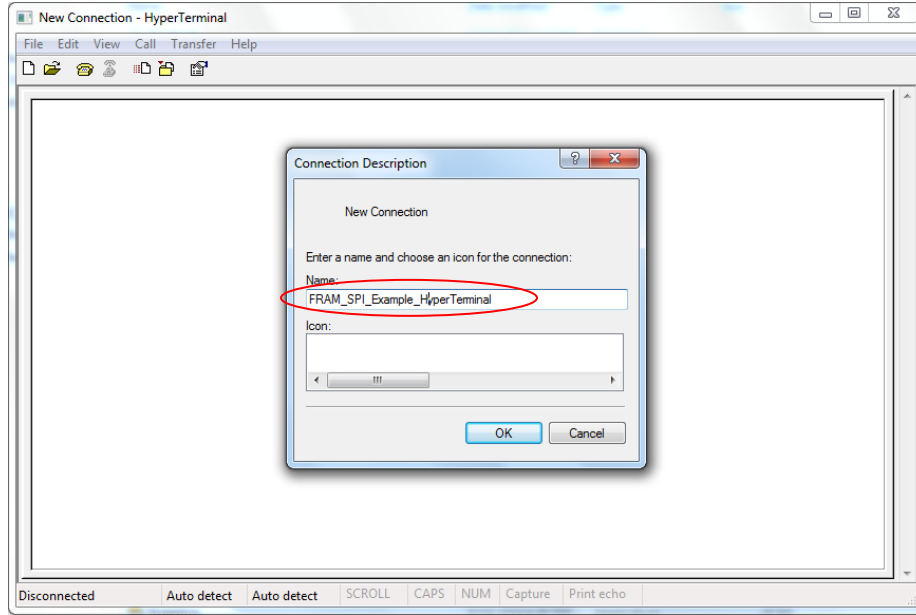
1.  A new window opens where the communication port can be selected.

    Select the appropriate communication port "COMX" (or the specific communication port that is assigned to KitProg USB-UART) in **Connect using** and click **OK** to go to the next setting for HyperTerminal.

    This code example uses **COM5**. Verify the COM setting for your setup and select the appropriate COMX.

Figure 5-12. Select Communication Port



2.  The **COMX Properties** window opens with the **Port Settings** tab. Click **Restore Defaults** and then configure '*Bits per second*', '*Data bits*', '*Parity*', '*Stop bits*', and '*Flow control*' as shown in Figure 5-13 and click **OK**. If you change the PSoC 4 UART port setting in the example project, make sure the same setting is applied on HyperTerminal. Click **OK** to start the communication.

Figure 5-13. Configure Communication Port
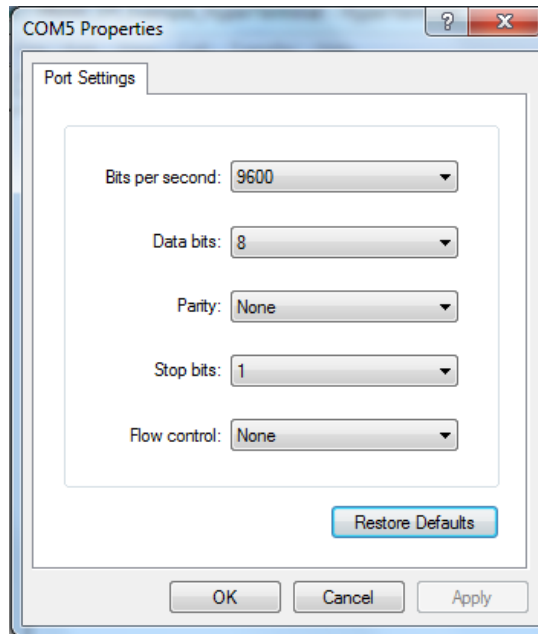


3. HyperTerminal window opens to display the PSoC 4 example projects output.

Figure 5-14. Configure Communication Port
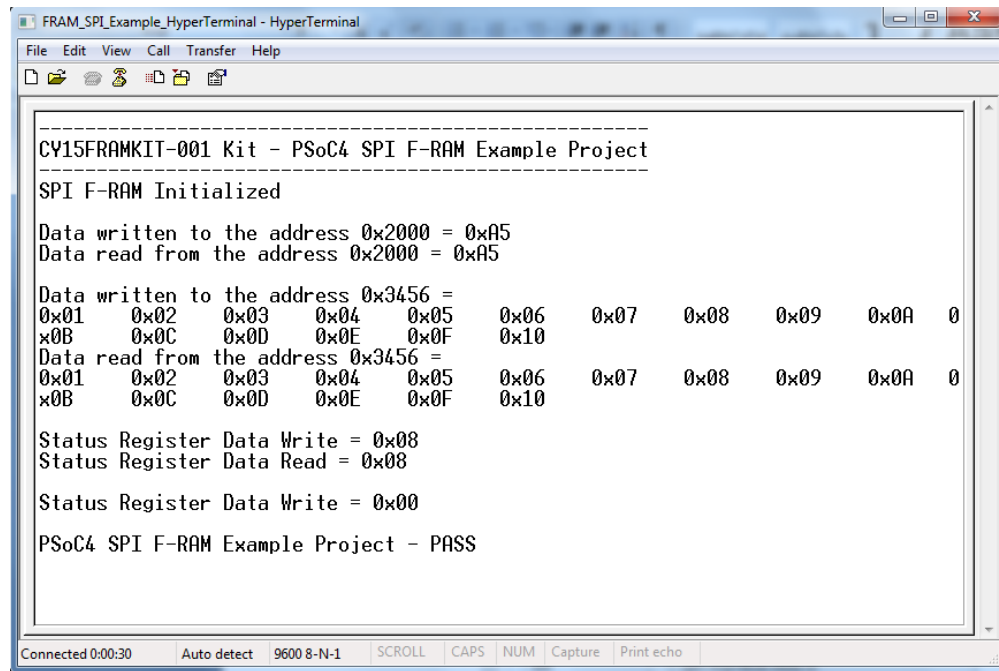


## 5.2.2 PuTTY Setup

PuTTY is a free SSH and telnet client for Windows. You can download PuTTY from www.putty.org.

1. Double-click the putty icon and select **Serial** under **Connection**.

Figure 5-15. Open New Connection
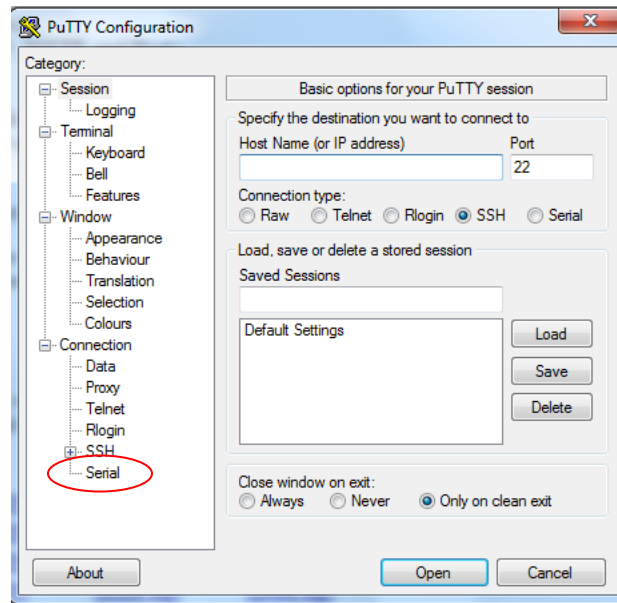


2. A new window opens where the communication port can be selected. Enter the COMX in **Serial line to connect to**. This code example uses **COM5**. Verify the COM setting for your setup and select the appropriate COMX.

   Select *Speed (baud)*, *Data bits*, *Stop bits*, *Parity,* and *Flow control* under **Configure the serial line**. Click **Session** and select **Serial** under **Connection type**.

Figure 5-16. Select and Configure Communication Port
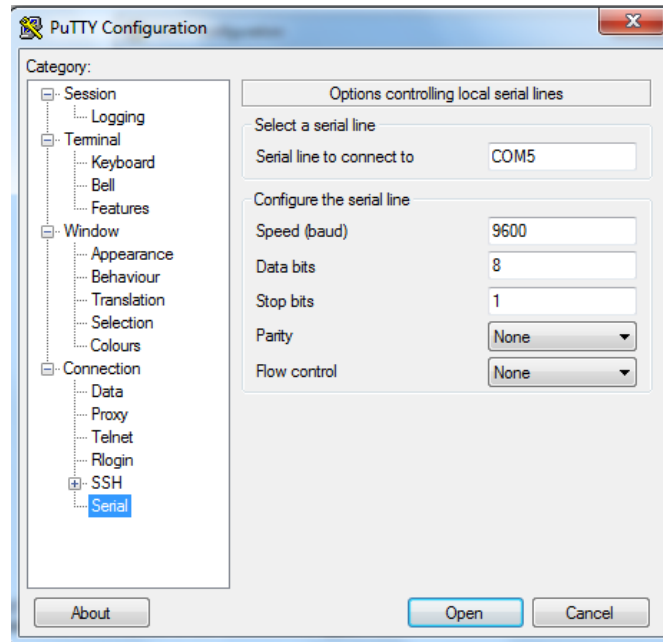


3. Select the **Session** under **Category** and set the **Communication type:** to Serial**.** Click **Open.**

Figure 5-17. Select Communication Type in PuTTY
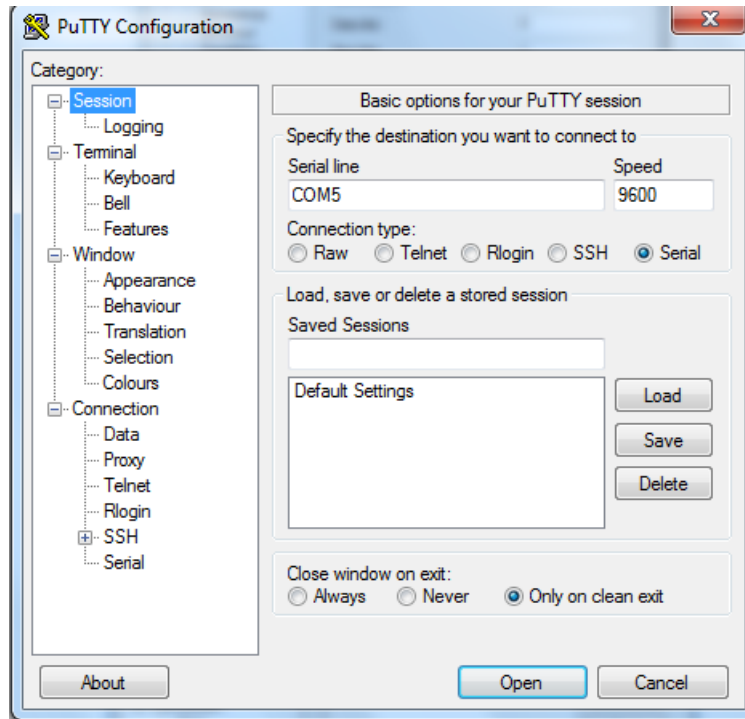


4. The COM terminal software displays the data from PSoC 4. For example, the initial screen of the SPI example project is shown in Figure 5-18.

Figure 5-18. Data Displayed on PuTTY
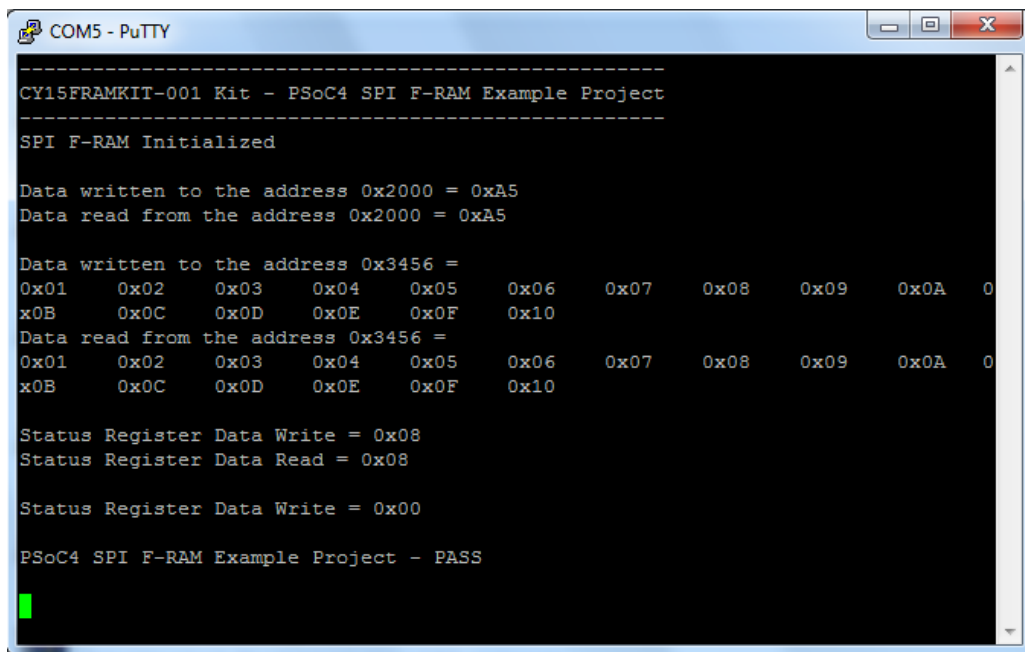
## 5.3  Project: PSoC 4 F-RAM SPI

### 5.3.1  Project Description

This example project runs on the CY8CKIT-042 PSoC 4 Pioneer Kit and provides the SPI F-RAM component and APIs to write and read F-RAM memory/status register. The example project will write to F-RAM, read from F-RAM, and indicate PASS by illuminating the RGB LED (D9) on PSoC 4 in blue. The output is also displayed on HyperTerminal through UART.

### 5.3.2  Hardware Connections

Refer to section 5.2  for UART setup. Apart from this, all other connections are hardwired on the board. Open *PSoC4_FRAM_SPI.cydwr* in the Workspace Explorer and select the suitable pin.

Table 5-1. PSoC 4 F-RAM SPI I/O Pin Assignment

| F-RAM I/O | PSoC 4 Pin |
|-----------|------------|
| CS# | P3_4 |
| SI | P3_0 |
| SO | P3_1 |
| SCK | P0_6 |

Figure 5-19. PSoC 4 F-RAM SPI Example Project Pin Configuration
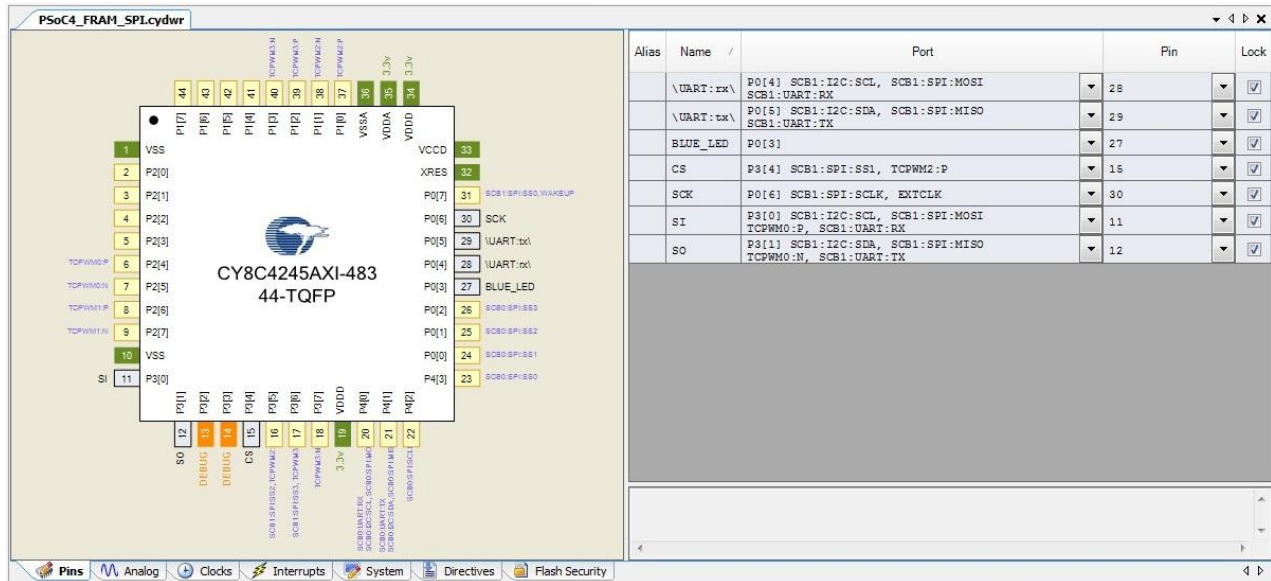


### 5.3.3  Firmware Flow

Following steps provide the example project firmware flow.

a.   Initialize the F-RAM SPI and USB-UART block.

b.   Wait for "Enter" key from the display (HyperTerminal / PuTTY)

c.   Write EXAMPLE_DATA_BYTE (0xA5) to EXAMPLE_ADDR_1 (0x2000) address.

d.   Read a byte from address EXAMPLE_ADDR_1 (0x2000) and store in the fram_rd_byte.

e.   Write EXAMPLE_DATA_BYTE (0xA5) to EXAMPLE_ADDR_1 (0x2000) address.

f.   Read a byte from address EXAMPLE_ADDR_1 (0x2000) and store in the fram_rd_byte variable.

g.    Write 16 bytes of data from data_bytes array (1, 2, 3, …,16) to EXAMPLE_ADDR_2 (0x3456) address.

h.   Read 16 bytes of data from EXAMPLE_ADDR_2 (0x3456) address through the read function.

i.   Write EXAMPLE_STS_REG_VALUE (0x08) to the status register.

j.   Read the status register and store in the 'statug_reg' variable.

k.   Compare the read data with the written data. If successful, glow the blue LED.

l.   Clear the status register.

### 5.3.4  Verify Output

The data output is shown in Figure 5-20 and Figure 5-22.

Figure 5-20. PSoC 4 F-RAM SPI Example Project Output - Waiting for **Enter Key** pressed by the User
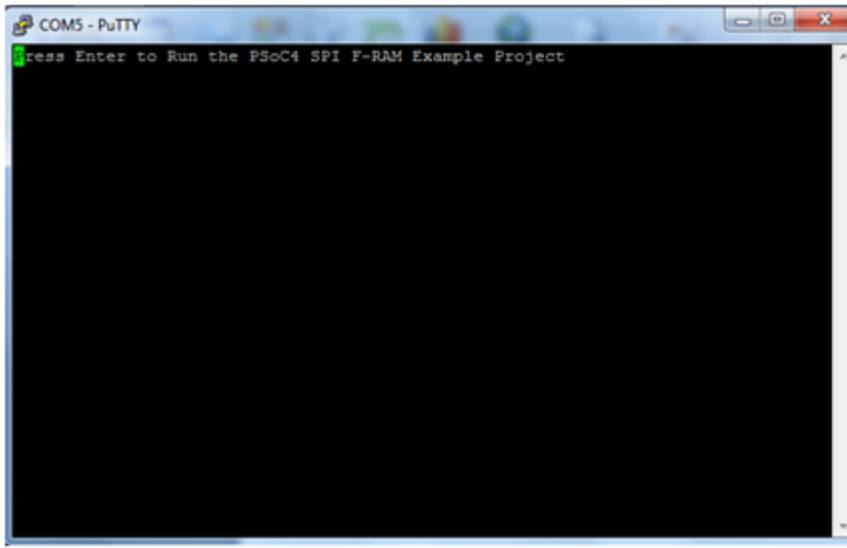


Figure 5-21. PSoC 4 F-RAM SPI Example Project Output Data
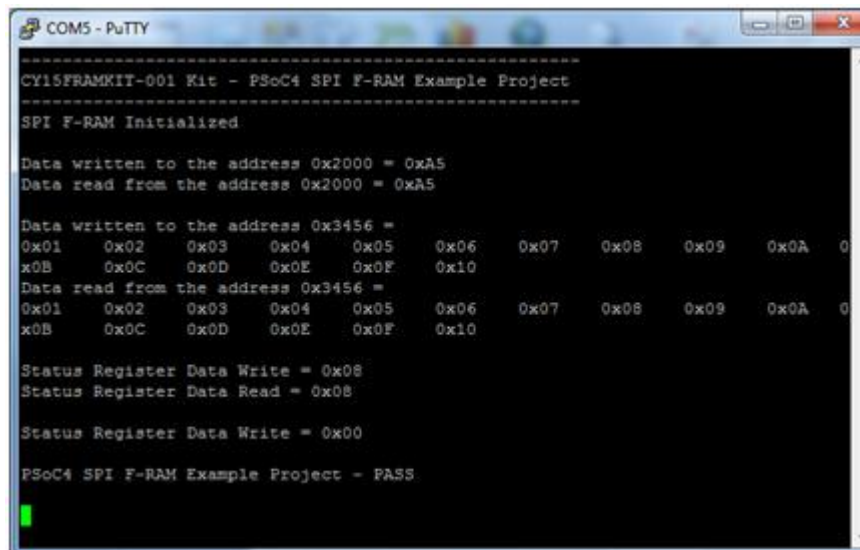


## 5.3.5  Modifying the Project

Open file **main.c** in the project under **Source Files** and change the address and data byte values by setting the variables EXAMPLE_ADDR_1, EXAMPLE_ADDR_2and EXAMPLE_DATA_BYTE shown in Figure 5-22. For example change: **0x2000** to **0x1200, 0x3456** to **0x1600, 0xa5** to **0x5a.**

Figure 5-22. Snapshot of **main.c** Source File



Also the user can change the burst write data shown in Figure 5-23. For example, change **i+1** to **i+3** at line 83 will initialize the input data from 0x03 – 0x13.

Figure 5-23. Snapshot of **main.c** source file for initializing input data buffer



Build the project with the above changes and program the Pioneer kit. The output will show the data **0x5a** written to address **0x1200** and data **0x03 – 0x13** written from address **0x1600** as shown in Figure 5-24. APIs provided in the next section can be used to build different applications.

Figure 5-24. Modified SPI F-RAM Project Output



## 5.3.6  APIs

### 5.3.6.1  void FRAM_SPI_Init(void)

Description:          Initializes the SPI block.

Parameters:       None

Return Value:    None

Side Effects:     None

### 5.3.6.2 uint8 FRAM_SPI_Write(uint32 addr, uint8 data_write)

Description:     Write a single byte of data to F-RAM.

Parameters:      uint32 addr: 32 bit F-RAM address for write.

                 uint8   data_write: data byte to be written.

Return Value:    uint8: 0 - Communication Error

                   1 – Success

Side Effects:     None

### 5.3.6.3 uint8 FRAM_SPI_Read(uint32 addr, uint8 *data_read)

Description:     Read a byte of data from F-RAM.

Parameters:      uint32 addr : 32 bit F-RAM address for read

                 uint8 *data_read. : 8 bit pointer variable to hold data

Return Value:    uint8: 0 – Communication Error

                   1 – Success

Side Effects:     None

### 5.3.6.4 uint8 FRAM_SPI_BurstWrite(uint32 addr, uint8 *data_write_ptr, uint32 total_data_count)

Description:     Write total_data_count number of data into F-RAM.

Parameters:      uint32 addr: 32 bit F-RAM address for write.

                 uint8   *data_write_ptr: Pointer to an array of data bytes to be written.

                 uint32 total_data_count: Number of data bytes to be written.

Return Value:    uint8: 0 – Communication Error

                   1 – Success

Side Effects:     None

### 5.3.6.5 uint8 FRAM_SPI_BurstRead(uint32 addr, uint8 *data_read_ptr, uint32 total_data_count )

Description:     Read total_data_count number of data from F-RAM.

Parameters:      uint32 addr : 32 bit F-RAM address for read.

                 uint8 *data_read_ptr: Pointer to an array for storing data bytes.

                 uint32 total_data_count: Number of data bytes to be read.

Return Value:    uint8: 0 – Communication Error

                   1 – Success

Side Effects:     None

### 5.3.6.6 uint8 FRAM_SPI_ Status_Reg_Write (uint8 data_byte)

Description:      F-RAM Status Register Write

Parameters:            uint8 data_byte -> 1 byte status register data to be written

Return Value:       uint8: 0 – Communication Error

                               1 – Success

Side Effects:        None

### 5.3.6.7 uint8 FRAM_SPI_ Status_Reg_Read (uint8 * status)

Description:          F-RAM Status Register Read

Parameters:            uint8 * status : pointer to hold status register value

Return Value:       uint8: 0 – Communication Error

                               1 – Success

Side Effects:        None

## 5.4 Project: PSoC 4 F-RAM I²C

### 5.4.1 Project Description

This example project runs on the CY8CKIT-042 PSoC 4 Pioneer Kit and provides the I²C F-RAM component and APIs to write and read F-RAM memory. The example project will write to F-RAM, read from F-RAM, and indicate PASS by illuminating the RGB LED (D9) on PSoC 4 in blue. The output is also displayed on HyperTerminal through UART.

### 5.4.2 Hardware Connections

Refer to section 5.2 for UART setup. Apart from this, all other connections are hardwired on the board. Open *PSoC4_FRAM_I2C.cydwr* in the Workspace Explorer and select the suitable pin.

Table 5-2. PSoC 4 F-RAM I²C I/O Pin Assignment

| F-RAM I/O | PSoC 4 Pin |
|-----------|-----------|
| SDA | P4_1 |
| SCL | P4_0 |

Figure 5-25. PSoC 4 F-RAM I²C Example Project Pin Configuration
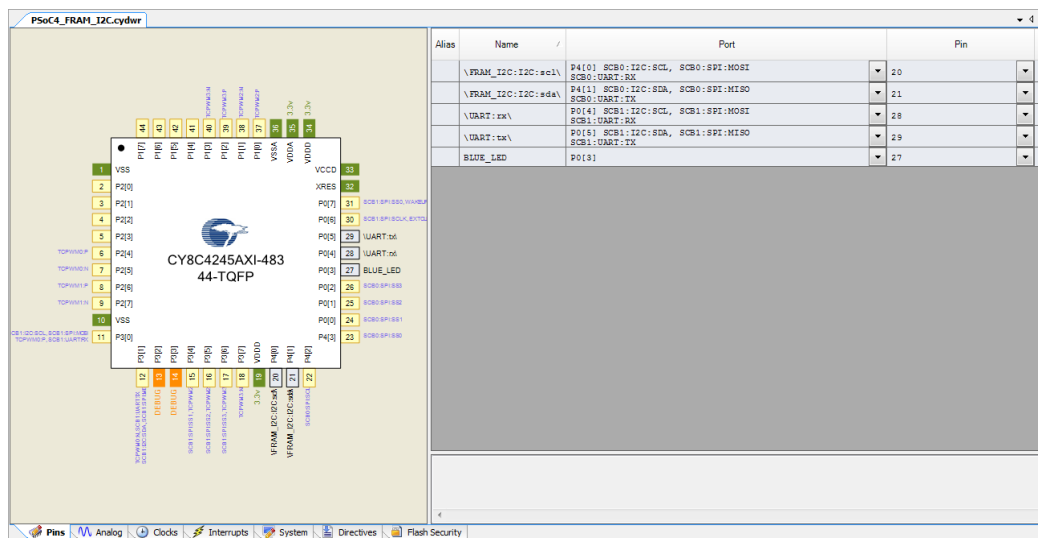
### 5.4.3  Firmware Flow

Following steps provide the example project firmware flow.

    a.   Init the F-RAM I$^2$C and USB-UART blocks.

    b.   Write EXAMPLE_DATA_BYTE (0xA5) to EXAMPLE_ADDR_1 (0x2000) address.

    c.   Read a byte from address EXAMPLE_ADDR_1 (0x2000) and store in the fram_rd_byte variable.

    d.   Write 16 bytes of data from data_bytes array (1, 2, 3, …,16) to EXAMPLE_ADDR_2 (0x3456) address.

    e.   Read 15 bytes of data from EXAMPLE_ADDR_2 (0x3456) address through the random read function.

    f.   Read the sixteenth byte through the current read function.

    g.   Compare the read data with the written data and illuminate the blue LED if read is PASS.

### 5.4.4  Verify Output

The RGB LED illuminates in blue if the read passes. If read fails, the RGB LED will not glow. The data output is shown in Figure 5-26 and Figure 5-27.

Figure 5-26. PSoC 4 F-RAM I$^2$C Example Project Output - Waiting for **Enter Key** pressed by the User
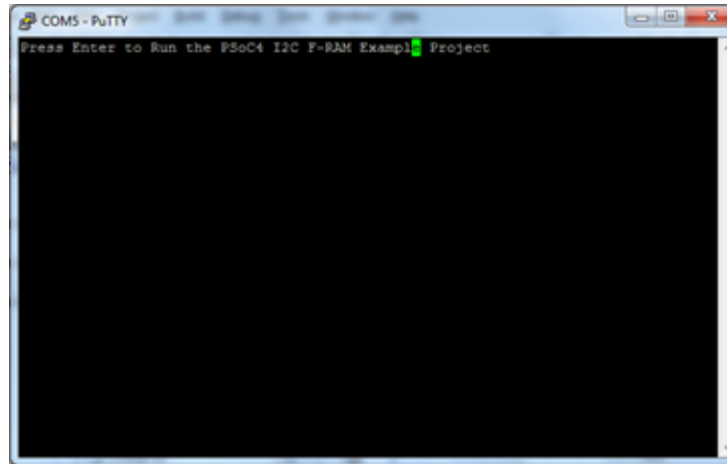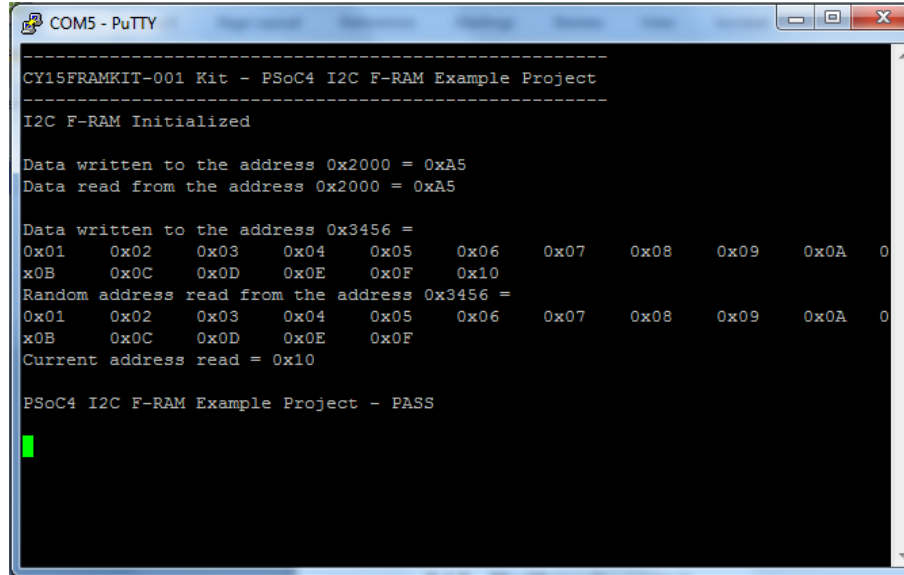


Figure 5-27. PSoC 4 F-RAM I$^2$C Example Project Output Data
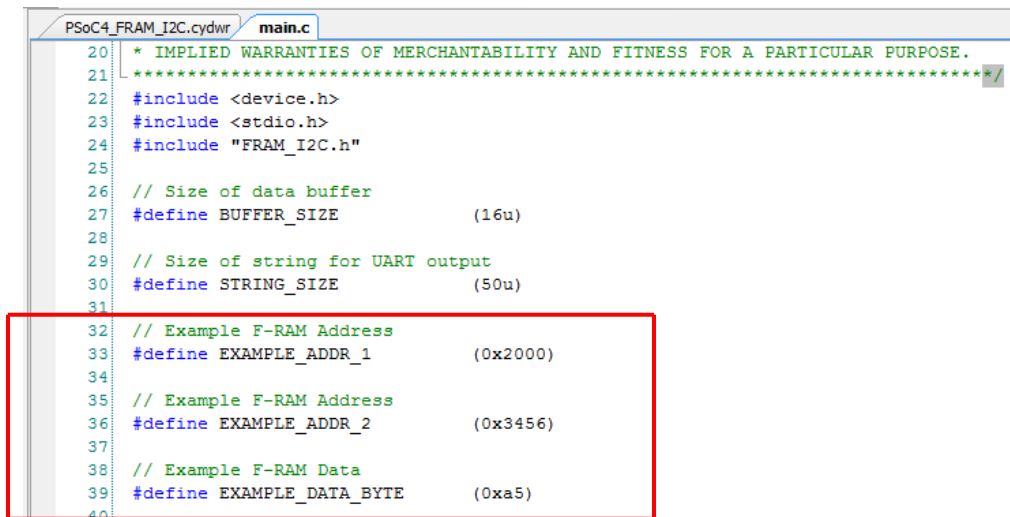
### 5.4.5  Modifying the Project

Open **main.c** in the project under **Source Files** and change the values of EXAMPLE_ADDR_1, EXAMPLE_ADDR_2and EXAMPLE_DATA_BYTE shown in Figure 5-28. For example change **0x2000 to 0x1200, 0x3456 to 0x1600, 0xa5 to 0x5a**

Figure 5-28. Snapshot of **main.c** source file
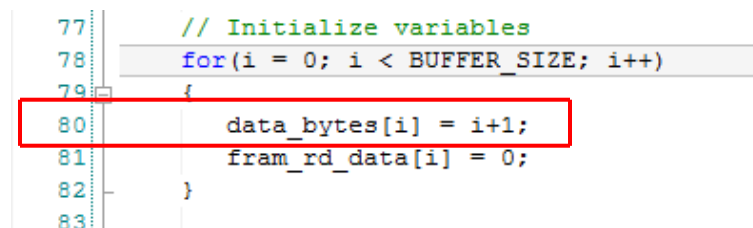


Also the user can change the burst write data shown in Figure 5-29. For example, change **i+1** to **i+3 at** line 80 will initialize the input data from 0x03-0x13.

Figure 5-29. Snapshot of **main.c** source file for initializing input data buffer

With these changes, build the project and program the kit. Output will show the data **0x5a** written to address **0x1200** and data **0x03 – 0x13** written from address **0x1600** as shown in Figure 5-30. APIs provided in the next section can be used to build different applications.

Figure 5-30. Modified I²C F-RAM Project Output
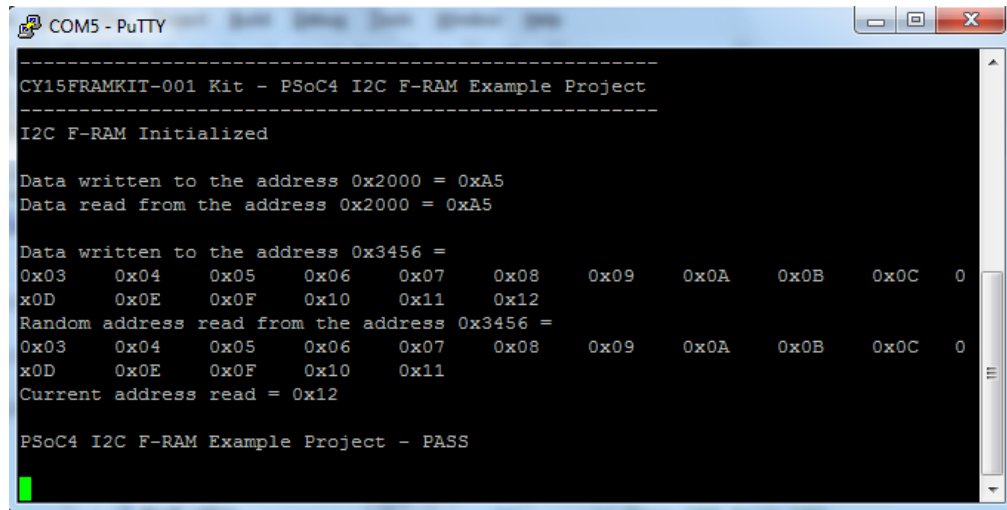


## 5.4.6  APIs

The following APIs are defined for the PSoC 4 F-RAM I²C. These APIs are the same for Arduino as well.

### 5.4.6.1  void FRAM_I2C_Init(void)

| | |
|---|---|
| Description: | Initializes the I2C block. |
| Parameters: | None |
| Return Value: | None |
| Side Effects: | None |

### 5.4.6.2  uint32 FRAM_I2C_Write (uint8 slave_id, uint32 addr, uint8 *data_write_ptr, uint32 total_data_count)

| | |
|---|---|
| Description: | Write total_data_count number of data into F-RAM specified by slave_id. |
| Parameters: | uint8 : 7 bit Slave ID |
| | uint32 addr: 32 bit F-RAM address for write. |
| | uint8 *data_write_ptr: Pointer to an array of data bytes to be written. |
| | uint32 total_data_count: Number of data bytes to be written. |
| Return Value: | For PSoC 4, |
| | uint32 : Error Status |
| | FRAM_I2C_MSTR_NO_ERROR          -- (0x00u) |
| | FRAM_I2C_MSTR_ERR_ARB_LOST  -- (0x01u) |
| | FRAM_I2C_MSTR_ERR_LB_NAK       -- (0x02u) |
| | FRAM_I2C_MSTR_NOT_READY        --  (0x04u) |
| | FRAM_I2C_MSTR_BUS_BUSY          -- (0x08u) |
| | FRAM_I2C_MSTR_ ERR_ABORT_START    -- (0x10u) |

FRAM_I2C_MSTR_ ERR_BUS_ERR          -- (0x100u)

FRAM_I2C_MSTR_ERR                   -- (0xFFu)

For Arduino UNO,

uint32 : Error Status

    Success                                 -- (0x00u)

Data too long to fit in transmit buffer    -- (0x01u)

Received NACK on transmit of address    -- (0x02u)

Received NACK on transmit of data        -- (0x03u)

Other error                              --  (0x04u)

Side Effects:    None

### 5.4.6.3 uint32 FRAM_I2C_Current_Read (uint8 slave_id, uint8 *data_read_ptr, uint32 total_data_count)

Description:    Read total_data_count number of data from the current address of the F-RAM specified by slave_id.

Parameters:    uint8 : 7 bit Slave ID

    uint8 *data_read_ptr: Pointer to an array for storing data bytes.

    uint32 total_data_count: Number of data bytes to be read.

Return Value:    For PSoC 4

    uint32 : Error Status

        FRAM_I2C_MSTR_NO_ERROR          -- (0x00u)

        FRAM_I2C_MSTR_ERR_ARB_LOST  -- (0x01u)

        FRAM_I2C_MSTR_ERR_LB_NAK          -- (0x02u)

        FRAM_I2C_MSTR_NOT_READY          -- (0x04u)

        FRAM_I2C_MSTR_BUS_BUSY          -- (0x08u)

        FRAM_I2C_MSTR_ ERR_ABORT_START    -- (0x10u)

        FRAM_I2C_MSTR_ ERR_BUS_ERR          -- (0x100u)

        FRAM_I2C_MSTR_ERR                   -- (0xFFu)

For Arduino UNO, returns the number of bytes read

Side Effects:    None

### 5.4.6.4 uint32 FRAM_ I2C_Random_Read (uint8 slave_id, uint32 addr, uint8 *data_read_ptr, uint32 total_data_count)

Description:    Read total_data_count number of data from F-RAM specified by slave_id.

Parameters:    uint8 : 7 bit Slave ID

    uint32 addr: 32 bit F-RAM address for read.

    uint8 *data_read_ptr: Pointer to an array for storing data bytes.

    uint32 total_data_count: Number of data bytes to be read.

Return Value:    For PSoC 4

    uint32 : Error Status

        FRAM_I2C_MSTR_NO_ERROR          -- (0x00u)

FRAM_I2C_MSTR_ERR_ARB_LOST  -- (0x01u)

FRAM_I2C_MSTR_ERR_LB_NAK       -- (0x02u)

FRAM_I2C_MSTR_NOT_READY        --  (0x04u)

FRAM_I2C_MSTR_BUS_BUSY          -- (0x08u)

FRAM_I2C_MSTR_ ERR_ABORT_START    -- (0x10u)

FRAM_I2C_MSTR_ ERR_BUS_ERR          -- (0x100u)

FRAM_I2C_MSTR_ERR                      -- (0xFFu)

For Arduino UNO, returns the number of bytes read

Side Effects:        None

## 5.5  Programming Arduino UNO Kit

Follow these steps to program the Arduino example projects (Arduino_FRAM_SPI project is shown here):

1.  Install Arduino IDE. You can download Arduino IDE software from the Arduino webpage: http://arduino.cc/en/Main/Software.

2.  Open the "Arduino_FRAM_SPI" project folder  and  double-click the **FRAM_SPI_Example.ino** file.

Figure 5-31. Arduino Project File Open
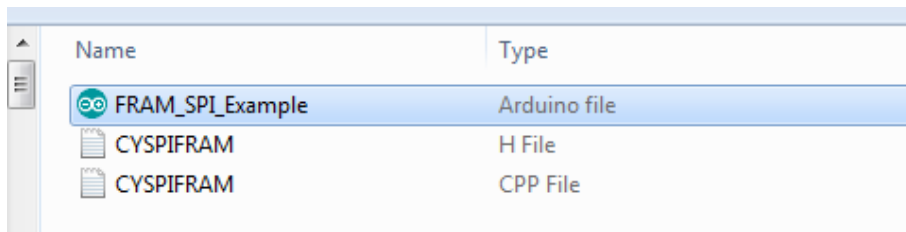


3.  Compile the code example by clicking **Sketch** > **Verify / Compile**. At the end of successful compiling, the IDE displays the status "Done compiling." and generates a binary sketch.

Figure 5-32. Compile the Arduino Project
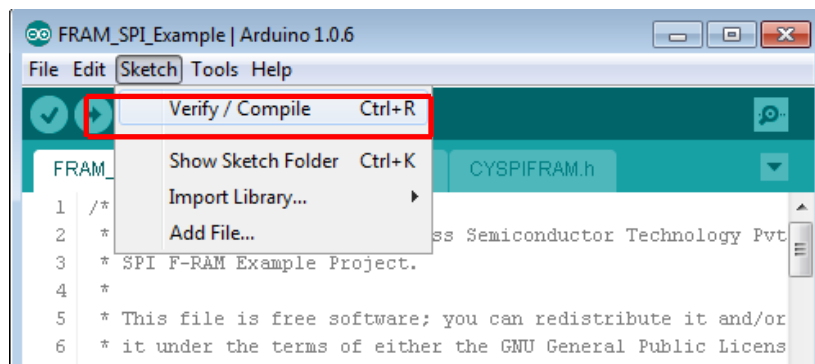


4.  To program the Arduino UNO kit, connect the PC to the Arduino board via USB port using an Arduino UNO compatible USB cable as shown in Figure 5-33. Because Arduino UNO kits operate at 5 V, ensure that the power select jumper (J5) on the CY15FRAMKIT-001 is placed between pin 1 and pin 2 for 5 V operations.

Figure 5-33. CY15FRAMKIT-001 mounted on Arduino UNO R3 Kit
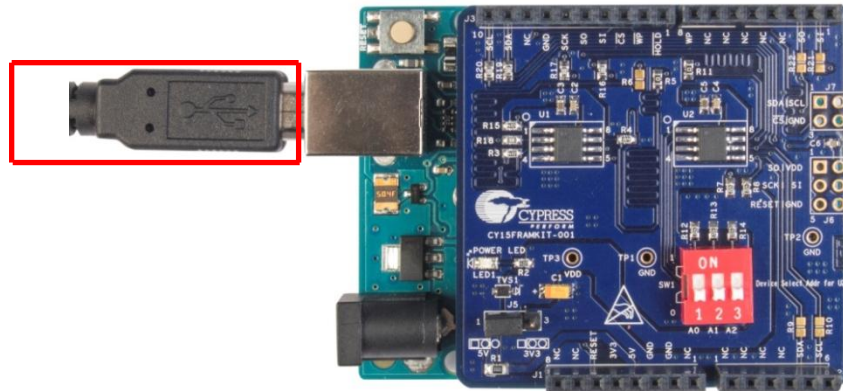


5.  Load the binary sketch file by clicking the **Upload** button.

Figure 5-34. Program the Arduino UNO R3 Kit
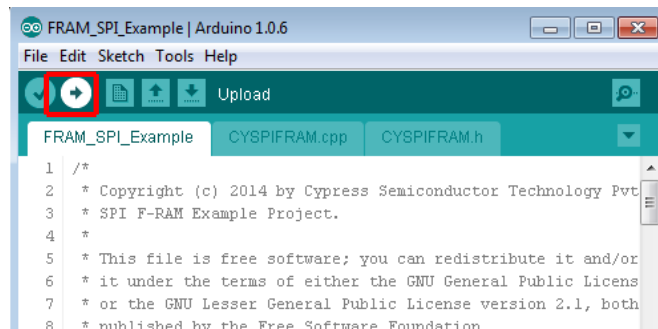


# 5.6    Project: Arduino F-RAM SPI

## 5.6.1  Project Description

This example project runs on the Arduino UNO board and provides SPI APIs to write and read F-RAM memory/status register. The example project will write to F-RAM, read from F-RAM, and display the results in serial output window.

The CY15FRAMKIT-001 Serial F-RAM DVK uses two-byte addressable, 256 Kbit, SPI F-RAM. Two-byte address can support a maximum of 512 Kbit (64K × 8) density access. To access SPI F-RAMs with a density of 1-Mbit or higher, three-byte address access needs to be enabled in this example project. Refer to A.7 Enable Three-Byte Address in SPI F-RAM for details.

## 5.6.2  Hardware Connections

No additional hardware connection is required for this project because all connections are hardwired on the board. The Arduino SPI library uses the following pins by default.

Table 5-3. Arduino F-RAM SPI I/O Pin Assignment

| F-RAM I/O | Arduino Pin |
|-----------|-------------|
| CS#       | D10         |
| SI        | D11         |
| SO        | D12         |
| SCK       | D13         |

### 5.6.3 Firmware Flow

Following steps provide the example project firmware flow.

a. Initialize the F-RAM SPI.

b. Write EXAMPLE_DATA_BYTE (0xA5) to EXAMPLE_ADDR_1 (0x2000) address. Display the written data on serial output.

c. Read a byte from address EXAMPLE_ADDInitialize the F-RAM SPI.

d. Write EXAMPLE_DATA_BYTE (0xA5) to EXAMPLE_ADDR_1 (0x2000) address. Display the written data on serial output.

e. Read a byte from address EXAMPLE_ADDR_1 (0x2000) and store in the fram_rd_byte variable. Display the read data on serial output.

f. Compare the written and read data and display the result (PASS/FAIL) on serial output.

g.  Write 16 bytes of data from data_bytes array (1, 2, 3, …,16) to EXAMPLE_ADDR_2 (0x3456) address.

h. Read 16 bytes of data from EXAMPLE_ADDR_2 (0x3456) address through the read function.

i. Compare the written and read data and display the result (PASS/FAIL) on serial output.

j. Write EXAMPLE_STS_REG_VALUE (0x08) to the status register.

k. Read the status register and store in the 'statug_reg' variable.

l. Compare the written and read data and display the result (PASS/FAIL) on serial output.

m. Clear the status register.

Refer to A.7 Enable Three-Byte Address in SPI F-RAM to enable three-byte addressing in the example project.

### 5.6.4 Verify Output

Open **Tools >Serial Port** and select the appropriate COM port for communication as shown in Figure 5-35.

Figure 5-35. Serial Port in Arduino IDE
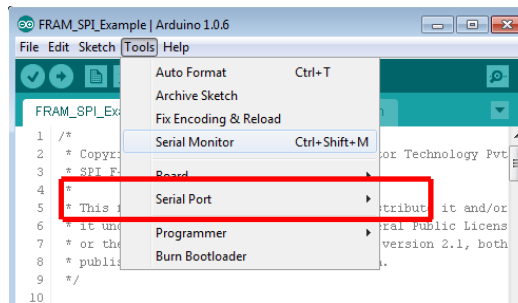


Open **Tools > Serial Monitor** to view the serial output as shown in Figure 5-36 and Figure 5-37.

Figure 5-36. Serial Monitor in Arduino IDE
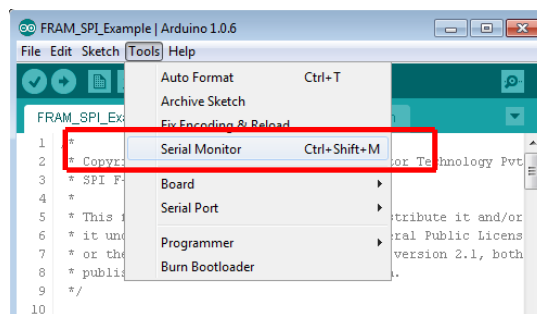
Figure 5-37. Serial Monitor Output for SPI F-RAM Project



### 5.6.5 APIs

#### 5.6.5.1 void FRAM_SPI_Init()

Description:    Initializes the SPI library

Parameters:    None

Return Value:  None

Side Effects:   None

#### 5.6.5.2 void FRAM_SPI_Write(uint32 addr, uint8 data_write)

Description:        Write a single byte of data to F-RAM.

Parameters:        uint32 addr: 32 bit F-RAM address for write.

                   uint8   data_write: data byte to be written.

Return Value:      None

Side Effects:      None

#### 5.6.5.3 uint8 FRAM_SPI_Read(uint32 addr)

Description:        Read a byte of data from F-RAM.

Parameters:        uint32 addr : 32 bit F-RAM address for read.

Return Value:      uint8 : data read

Side Effects:      None

#### 5.6.5.4 void    FRAM_SPI_BurstWrite(uint32    addr,    uint8    *data_write_ptr,    uint32 total_data_count)

Description:        Write total_data_count number of data into F-RAM.

Parameters:        uint32 addr: 32 bit F-RAM address for write.

                   uint8   *data_write_ptr: Pointer to an array of data bytes to be written.

                   uint32 total_data_count: Number of data bytes to be written.

Return Value:      None

Side Effects:      None

### 5.6.5.5 void FRAM_SPI_BurstRead(uint32 addr, uint8 *data_read_ptr, uint32 total_data_count )

| | |
|---|---|
| Description: | Read total_data_count number of data from F-RAM. |
| Parameters: | uint32 addr : 32 bit F-RAM address for read. |
| | uint8 *data_read_ptr: Pointer to an array for storing data bytes. |
| | uint32 total_data_count: Number of data bytes to be read. |
| Return Value: | None |
| Side Effects: | None |

### 5.6.5.6 void FRAM_SPI_ Status_Reg_Write (uint8 data_byte)

| | |
|---|---|
| Description: | F-RAM Status Register Write |
| Parameters: | uint8 data_byte -> 1 byte Status register data to be written |
| Return Value: | None |
| Side Effects: | None |

### 5.6.5.7 uint8 FRAM_SPI_ Status_Reg_Read (void)

| | |
|---|---|
| Description: | F-RAM Status Register Read |
| Parameters: | None |
| Return Value: | uint8: 1 byte status register data |
| Side Effects: | None |

## 5.7 Project: Arduino F-RAM I$^2$C

### 5.7.1 Project Description

This example project runs on Arduino UNO board and provides I$^2$C APIs to write and read from F-RAM memory. The example project will write to F-RAM, read from F-RAM, and display the results in serial output window.

### 5.7.2 Hardware Connections

No additional hardware connection is required for this project because all connections are hardwired on the board. The Arduino WIRE library (I$^2$C) uses the following pins by default.

Table 5-4. Arduino F-RAM I$^2$C I/O Pin Assignment

| F-RAM I/O | Arduino Pin |
|---|---|
| SDA | A4 |
| SCL | A5 |

### 5.7.3 Firmware Flow

Following steps provide the example project firmware flow.

a.  Initialize F-RAM I$^2$C and provide startup delay.

b.  Write EXAMPLE_DATA_BYTE (0xA5) to EXAMPLE_ADDR_1 (0x2000) address.

c.  Read a byte from address EXAMPLE_ADDR_1 (0x2000) and store in the fram_rd_byte variable. Display the read data on serial output.

d.  Compare the written and read data and display the result (PASS/FAIL) on serial output.

e.  Write 16 bytes of data from data_bytes array (1, 2, 3, …,16) to EXAMPLE_ADDR_2 (0x3456) address.

f.  Read 15 bytes of data from EXAMPLE_ADDR_2 (0x3456) address through the random read function.

g.  Read the sixteenth byte through the current read function.

h.  Compare the written and read data and display the result (PASS/FAIL) on serial output.

### 5.7.4  Verify Output

Open **Tools →Serial Monitor** to view serial output as shown in Figure 5-38.

Figure 5-38. Serial Monitor Output for I²C F-RAM Project



### 5.7.5  APIs

The API description is the same as for the PSoC 4 I2C F-RAM. Refer to section 5.4.6

# Appendix

![Cypress logo - EMBEDDED IN TOMORROW]

## A.1 CY15FRAMKIT-001 Schematics

**Power selection**

C1 10 uF
TVS1 5V 350W
VDD 5V
R1 0E
3V3
J5
1
2
3
1
2
3
3 Pin Header

VDD
R2 220 Ohm
LED1
Power LED Green

**SPI F-RAM**

VDD
R3 4.7k
R15 4.7k

U1
CS# 1 — CS
SO R18 30E 2 — SO
WP# 3 — WP
4 — VSS
VDD 8
HOLD 7
SCK 6
SI 5
HOLD#
SCK
SI
SPI F-RAM

VDD
R4 4.7k
C2 0.1 uF
C3 1 uF

## I2C F-RAM



## I2C F-RAM Device Select Address



By default, A0, A1, A2 are pulled down internally

(J1-J4) Arduino Compatible Headers

J1

3V3   5V

1
2
3
4
5
RESET   6
7
8

1
2
3
4
5
6
7
8

8x1 RECP

HOLD#   R5      0E
NO LOAD
WP#   R6      0E
NO LOAD
CS#
SI      R16      30E
SO
SCK      R17      30E

SDA      R19      30E
SCL      R20      30E

J3

1
2
3
4
5
6
7
8
9
10

1
2
3
4
5
6
7
8
9
10

10x1 RECP

J2

1
2
3
SDA   R9      30E   4
NO LOAD      5
6
SCL   R10      30E
NO LOAD

1
2
3
4
5
6

6x1 RECP

SI      R21      30E
NO LOAD
SO      R22      30E
NO LOAD

WP      R11      0E
NO LOAD

J4

1
2
3
4
5
6
7
8

1
2
3
4
5
6
7
8

8x1 RECP

Debug Header

J6

SO      1
SCK      3
RESET      5

1   2
3   4
5   6

2
4   SI
6

VDD

C6

0.1 uF

3X2 CONN FEMALE
NO LOAD

J7

SDA      1
CS#      3

1   2
3   4

2   SCL
4

2X2 CONN FEMALE
NO LOAD

NO LOAD
TP1
BLACK

NO LOAD
TP2
BLACK

NO LOAD   VDD
TP3
BLACK

# A.2 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 | PSoC 4 Pioneer Kit Signals | CY15FRAMKIT-001 Kit Signals |
| J1_1 | VIN | VIN | NC |
| J1_2 | GND | GND | GND |
| J1_3 | GND | GND | GND |
| J1_4 | 5V | V5.0 | 5V |
| J1_5 | 3.3V | V3.3 | 3V3 |
| J1_6 | RESET# | RESET# | RESET# |
| J1_7 | IOREF | P4_VDD | NC |
| J1_8 | NC | NC | NC |

| J2 | | | |
|---|---|---|---|
| Pin | Arduino Signals | PSoC 4 Pioneer Kit Signals | CY15FRAMKIT-001 Kit Signals |
| J2_1 | A0 | P2[0] | NC |
| J2_2 | A1 | P2[1] | NC |
| J2_3 | A2 | P2[2] | NC |
| J2_4 | A3 | P2[3] | NC |
| J2_5 | A4 /SDA (I$^2$C) | P2[4] | SDA |
| J2_6 | A5/SCL (I$^2$C) | P2[5] | SCL |

| J3 | | | |
|---|---|---|---|
| Pin | Arduino Signals | PSoC 4 Pioneer Kit Signals | CY15FRAMKIT-001 Kit Signals |
| J3_1 | D8 | P2[6] | HOLD# (SPI F-RAM) |
| J3_2 | D9 (PWM) | P3[6] | WP# (SPI F-RAM) |
| J3_3 | D10 (PWM/SS) | P3[4] / CS (SCB1) | CS# (SPI F-RAM) |
| J3_4 | D11 (PWM / MOSI) | P3[0] / MOSI (SCB1) | SI |
| J3_5 | D12 (MISO) | P3[1] / MISO (SCB1) | SO |
| J3_6 | D13 (SCK) | P0[6] / SCK (SCB1) | SCK |

| J3 | | | |
|---|---|---|---|
| **Pin** | **Arduino Signals** | **PSoC 4 Pioneer Kit Signals** | **CY15FRAMKIT-001 Kit Signals** |
| J3_7 | GND | GND | GND |
| J3_8 | AREF | P1[7] | NC |
| J3_9 | SDA | P4[1] / SDA (SCB0) | SDA |
| J3_10 | SCL | P4[0] / SCL (SCB0) | SCL |

| J4 | | | |
|---|---|---|---|
| **Pin** | **Arduino Signals** | **PSoC 4 Pioneer Kit Signals** | **CY15FRAMKIT-001 Kit Signals** |
| J4_1 | D0 (UART RX) | P0[4] / SCL (SCB1) | NC |
| J4_2 | D1 (UART TX) | P0[5] / SDA (SCB1) | NC |
| J4_3 | D2 | P0[7] | NC |
| J4_4 | D3 (PWM) | P3[7] | NC |
| J4_5 | D4 | P0[0] | NC |
| J4_6 | D5 (PWM) | P3[5] | NC |
| J4_7 | D6 (PWM) | P1[0] | NC |
| J4_8 | D7 | P2[7] | WP ($I^2C$ F-RAM) |

## A.3 Debug Header I/Os

| J6 | |
|---|---|
| **Pin** | **Debug Signals** |
| J6_1 | SO |
| J6_2 | VDD |
| J6_3 | SCK |
| J6_4 | SI |
| J6_5 | RESET# |
| J6_6 | GND |

| J7 | |
|---|---|
| **Pin** | **Debug Signals** |
| J7_1 | SDA |

| J7 | |
|---|---|
| **Pin** | **Debug Signals** |
| J7_2 | SCL |
| J7_3 | CS# |
| J7_4 | GND |

## A.4 Use of Zero-ohm Resistors and No Load

| Unit | Resistor | Usage |
|---|---|---|
| SPI F-RAM Device | R5 | Solder zero-ohm resistor to control the SPI HOLD# pin from the controller |
| SPI F-RAM Device | R6 | Solder zero-ohm resistor to control the SPI WP# pin from the controller |
| I$^2$C F-RAM Device | R11 | Solder zero-ohm resistor to control the I$^2$C WP pin from the controller |

## A.5 Use of 30-ohm Resistors and No Load

| Unit | Resistor | Usage |
|---|---|---|
| I$^2$C Communication | R9 | Solder a 30-ohm resistor to access the SDA for the I$^2$C communication in Arduino UNO R2 boards |
| I$^2$C Communication | R10 | Solder a 30-ohm resistor to access the SCL for the I$^2$C communication in Arduino UNO R2 boards |
| SPI Communication | R21 | Solder a 30-ohm resistor to use J4_1 as SPI SI pin. This will free the SI on J3_4, which can be used for the I$^2$C-USB Bridge communication on the CY8CKIT-042 Pioneer Kit. |
| SPI Communication | R22 | Solder a 30-ohm resistor to use J4_2 as SPI SO pin. This will free the SO on J3_5, which can be used for the I$^2$C-USB Bridge communication on the CY8CKIT-042 Pioneer Kit. |

## A.6 Bill of Materials (BOM)

| No. | Qty | Reference | Value | Description | Manufacturer | Mfr Part Number |
|---|---|---|---|---|---|---|
| 1 | | | | PCB, 52.07mm X 53.59mm, 2 Layer, Enig Finish, High Tg, Blue color Solder mask, white color silkscreen | Cypress | |
| 2 | 1 | C1 | 10 uF | CAP TANT 10UF 10V 20% 1206 | AVX Corporation | TPSA106M010R1800 |
| 3 | 3 | C2,C4,C6 | 0.1 uF | CAP CER 0.1UF 50V Y5V 0603 | Murata Electronics | GRM188R71C104KA01D |
| 4 | 2 | C3,C5 | 1 uF | CAP CERAMIC 1.0UF 25V X5R 0603 10% | Taiyo Yuden | TMK107BJ105KA-T |
| 5 | 2 | J1,J4 | 8x1 RECP | CONN RCPT .100" 8PS R/A SGL TIN | Samtec Inc | SSQ-108-03-T-S |
| 6 | 1 | J2 | 6x1 RECP | CONN RCPT .100" 6POS SNGL TIN | Samtec Inc | SSQ-106-03-T-S |
| 7 | 1 | J3 | 10x1 RECP | CONN RCPT .100" 10POS SNGL TIN | Samtec Inc | SSQ-110-03-T-S |
| 8 | 1 | J5 | 3 Pin Header | CONN HEADER VERT SGL 3POS GOLD | 3M | 961103-6404-AR |
| 9 | 1 | LED1 | Power LED Green | LED GREEN CLEAR 0805 SMD | Chicago Miniature | CMD17-21VGC/TR8 |
| 10 | 1 | R1 | 0E | RES 0.0 OHM 1/8W JUMP 0805 SMD | Panasonic - ECG | ERJ-6GEY0R00V |
| 11 | 1 | R2 | 220 Ohm | RES 220 OHM 1/10W 5% 0603 SMD | Panasonic-ECG | ERJ-3GEYJ221V |
| 12 | 8 | R3,R4,R7,R8,R12,R13,R14, R15 | 4.7k | RES 4.7K OHM 1/10W 5% 0603 SMD | Panasonic-ECG | ERJ-3GEYJ472V |
| 13 | 5 | R16,R17,R18,R19,R20 | 30 ohm | RES SMD 30 OHM 5% 1/10W 0603 | Panasonic - ECG | ERJ-3GEYJ300V |
| 14 | 1 | SW1 | 3-POS DIP | SW DIP SLIDE SPST 3 POS UNSEALD | E-Switch | KAS1103E |
| 15 | 1 | TVS1 | 5V 350W | TVS DIODE 5VWM 14.5VC SOD323 | Diodes Incorporated | SD05-7 |
| 16 | 1 | U1 | SPI F-RAM | IC FRAM 256-KBIT 40MHZ 8SOIC | Cypress | FM25W256-G |
| 17 | 1 | U2 | I2C F-RAM | IC FRAM 256-KBIT 3.4MHZ 8SOIC | Cypress | FM24W256-G |
| 18 | 1 | N/A | Jumper | Jumper | 3M | 969102-0000-DA |
| **No Load Components** | | | | | | |
| 19 | 1 | J6 | 3X2 CONN FEMALE | CONN HEADER VERT DUAL 6POS GOLD | 3M | 961206-6404-AR |
| 20 | 1 | J7 | 2X2 CONN FEMALE | CONN HEADER VERT DUAL 4POS GOLD | 3M | 961204-6404-AR |
| 21 | 3 | R5,R6,R11 | 0E | RES 0.0 OHM 1/10W 0603 SMD | Panasonic - ECG | ERJ-3GEYJ302V |
| 22 | 4 | R9,R10,R21,R22 | 30E | RES SMD 30 OHM 5% 1/10W 0603 | Panasonic - ECG | ERJ-3GEYJ300V |
| 23 | 3 | TP1,TP2,TP3 | BLACK | TEST POINT PC MINI .040"D Black | Keystone Electronics | 5001 |

| No. | Qty | Reference | Value | Description | Manufacturer | Mfr Part Number |
|---|---|---|---|---|---|---|
| **Special Jumper Installation Instructions** | | | | | | |
| 24 | 1 | J5 | Install jumper across pins 1 and 2 | Headers & Wire Housings 2.54MM SHUNT | 3M | 969102-0000-DA |
| **Label** | | | | | | |
| 25 | 1 | N/A | N/A | LBL, PCA Label, Vendor Code, Datecode, Serial Number 121-60201-01 REV 01 (YYWWVVXXXXX) | Cypress Semiconductor | |
| 26 | 1 | N/A | N/A | LBL, CY15FRAMKIT-001 QR Code, 12mm X 12mm | Cypress Semiconductor | |
| 27 | 1 | N/A | N/A | LBL, Anti-Static Warning, ULINE PN S-6516, 5/8" x 2", "Attention Observe Precautions" | Cypress Semiconductor | |

# A.7 Enable Three-Byte Address in SPI F-RAM

This section describes how to configure the FRAM_SPI example project to access three-byte address SPI F-RAMs (1-Mbit and higher density parts).

1) Open the CYSPIFRAM.h file in the "Arduino_FRAM_SPI" project folder.

2) Set THREEBYTEADDRESS to '1' as shown below. This enables the third address byte transmitted over the SPI line to access three-byte addressable SPI F-RAMs.

 #define THREEBYTEADDRESS 1

```
#define THREEBYTEADDRESS 0   //Disable 3-byte address access in SPI F-RAMs
//#define THREEBYTEADDRESS 1 //Enable 3-byte address in 1Mb and above densities SPI F-RAMs
```

# Revision History

## Document History

**Document Title:** CY15FRAMKIT-001 Serial F-RAM™ Development Kit Guide

**Document Number:** 001-95689

| Revision | ECN | Orig. of Change | Submission Date | Description of Change |
|---|---|---|---|---|
| ** | 4660868 | MEDU | 02/13/2015 | Initial version of kit guide |
| *A | 4666640 | MEDU | 02/20/2015 | Updated Figure 2-1 |
| *B | 5171339 | ZSK | 03/11/2016 | Updated section 5.6.1  and 5.6.3  to refer to Appendix 7 to enable three-byte address option for 1-Mbit and higher densities F-RAMs<br><br>Added A.7 Enable Three-Byte Address in SPI F-RAM |
| *C | 5713151 | SHEA | 04/26/2017 | Updated logo and copyright |