

Objective

This code example demonstrates how to interface Cypress I²C F-RAM devices with PSoC® 4 using dedicated serial NVRAM component in PSoC Creator.

Overview

This code example provides details on Serial NVRAM Component configuration to access the I²C F-RAM using PSoC 4. This document also discusses the hardware and software setup and its configuration to execute the associated example project and validate results.

Requirements

Tool: PSoC Creator 4.1 or above

Programming Language: C (GCC 5.4), Arm® Cortex®-M3 assembler

Associated Parts:

- Cypress Serial NVRAMs: All I²C F-RAM™ and I²C nvSRAM parts including F-RAM processor companion and nvSRAM with RTC. This code example is validated with 256-Kbit I²C F-RAM, FM24W256.
- PSoC 4 family parts

Related Hardware:

- Serial NVRAM Kits: [CY15FRAMKIT-001](#) is an Arduino compatible F-RAM development board.
- PSoC Development Kits: See [Table 1](#) for details on supported PSoC 4 kits.
- This code example is validated with CY15FRAMKIT-001 and CY8CKIT-042 PSoC 4 Pioneer Kit.

Design

The key design features include, configuring the following:

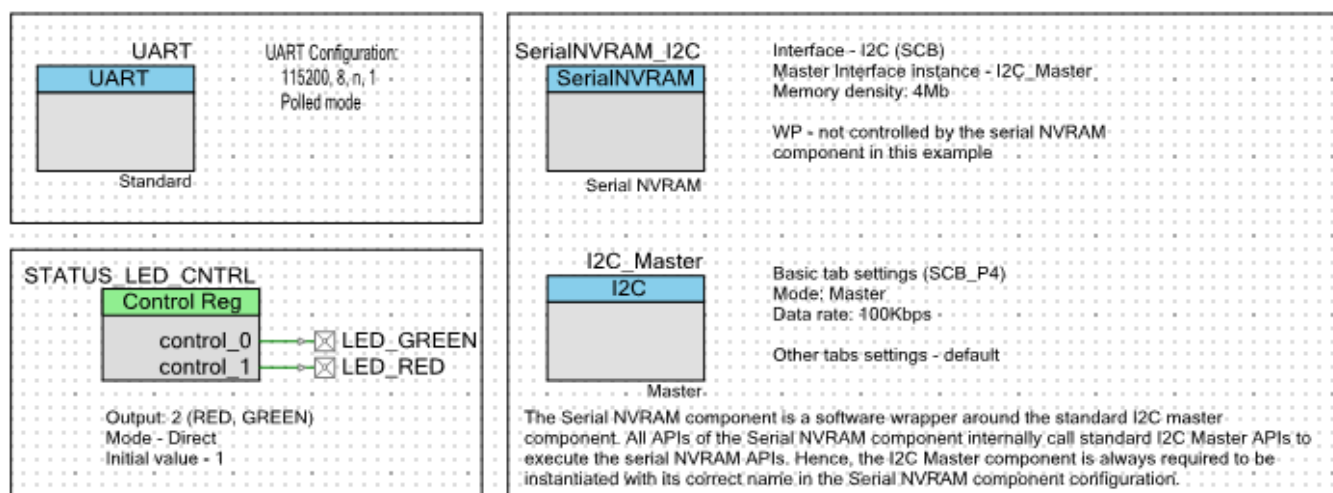
- UART to display the result on a PC
- I2C Component (SCB mode) as I²C master
- Serial NVRAM Component to enable I²C NVRAM access
- Control Register to drive status LEDs

The code example demonstrates the use of the following features:

- i. Write 1-Byte to an address of serial F-RAM
- ii. Read 1-Byte from an address of serial F-RAM
- iii. Write 16 bytes into serial F-RAM; burst write from a start address
- iv. Read 16 bytes from serial F-RAM; burst read from a start address

Design Considerations

The I²C NVRAM Component uses PSoC 4 serial communication block (SCB) resources to enable the I²C master. Therefore, the I²C host speed and data throughput are primarily determined by the PSoC 4 device. Cypress I²C NVRAMs can support I²C access up to 3.4 MHz. However, the full speed can't be demonstrated on PSoC 4 platforms due to I²C speed limitations. See the [PSoC 4 datasheet](#) for more details on PSoC 4 I²C master capability and setup requirement to achieve the best performance. [Figure 1](#) shows PSoC Creator design schematic of the code example.

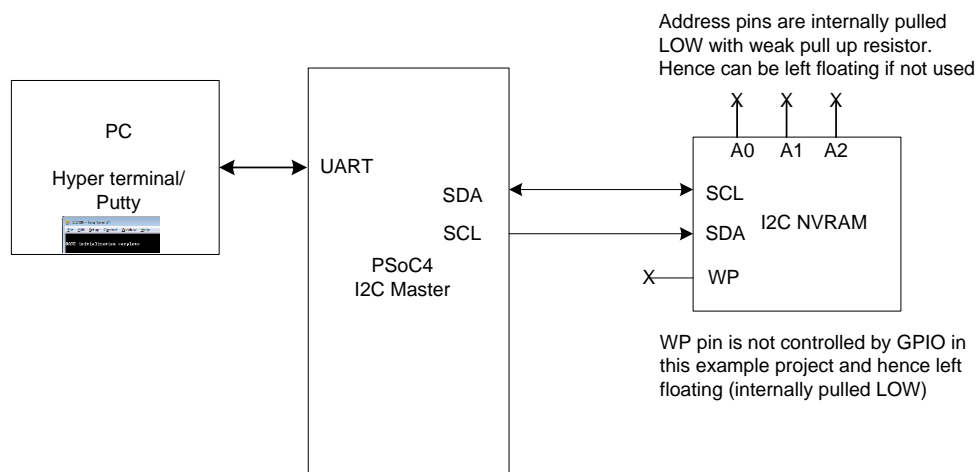
Figure 1. I²C NVRAM Access – PSoC Creator Design Schematic


Hardware Setup

The hardware setup involves connecting the F-RAM or NVSRAM I²C interface pins with PSoC 4 I²C master pins. You can use either dedicated hardware as described in the [Requirements](#) section or can connect through jumper wires.

This example project uses the default configuration of Cypress' PSoC 4 Pioneer kit (CY8CKIT-042) and serial F-RAM development kit CY15FRAMKIT-001. Refer to [CY8CKIT-042 PSoC 4 Pioneer kit](#) and [CY15FRAMKIT-001 Serial F-RAM Development Kit](#) user guide for hardware connection details.

Figure 2. Hardware Setup Block Diagram

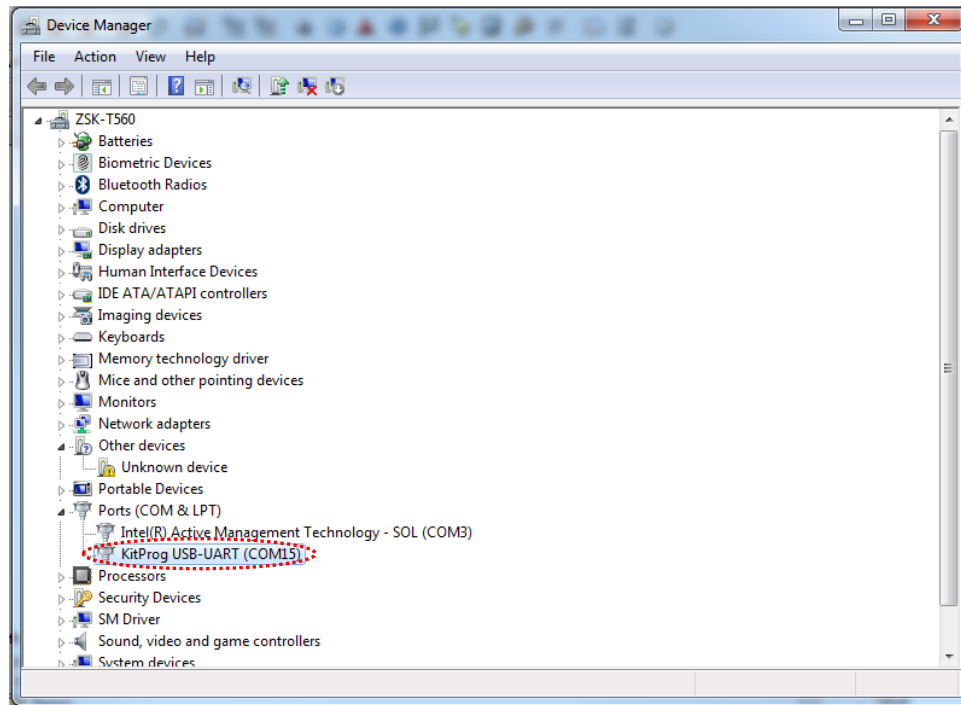


Software Setup

This section demonstrates the procedure to setup the serial (UART) connection using HyperTerminal or PuTTY on PC to communicate with the PSoC 4 Pioneer Kit. This code example is created using PSoC Creator software as described in the [Requirements](#) section. PuTTY is a free SSH and telnet client for Windows. You can download PuTTY from www.putty.org. Follow these instructions demonstrate to determine the COM port number and setup the PuTTY to monitor the code example outputs on PC.

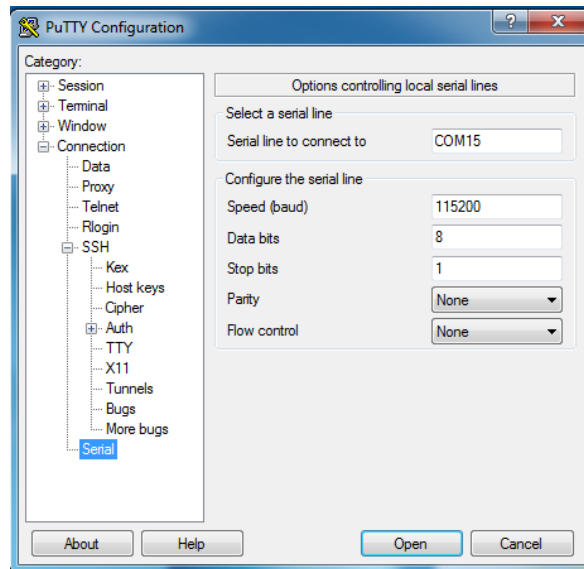
1. Connect PSoC 4 Pioneer Kit to the PC using USB Mini-B cable. The kit enumerates as KitProg USB-UART and is available under the **Device Manager > Ports (COM & LPT)**. A communication port (COMx) is assigned to KitProg USB-UART; for example, COM15 is assigned to PSoC 4 Pioneer Kit on the sample setup, shown in [Figure 3](#).

Figure 3. KitProg USB-UART in Device Manager



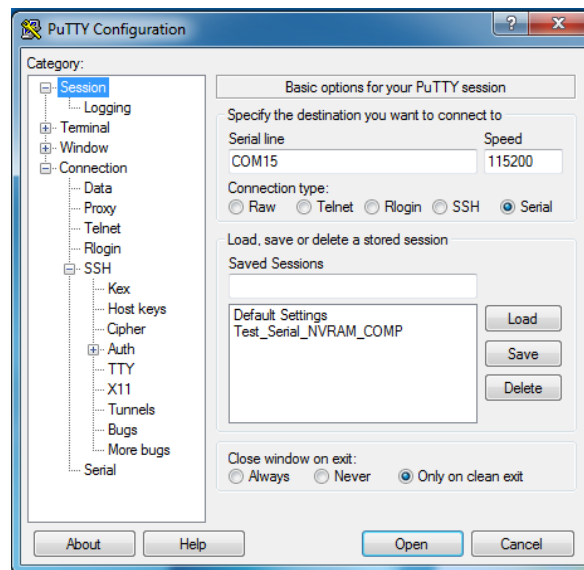
2. After you download and install PuTTY, double-click the PuTTY icon and select **Serial** under **Connection**.
3. A new window opens where the communication port can be selected. Do the following in the **Options controlling local serial lines** section:
 - Enter the PSoC 4 Port (COM & LPT), COMx, in **Serial line to connect to**. This code example uses **COM15**. Verify the COM setting for your setup and select the appropriate COMx.
 - Enter **Speed (baud)**, **Data bits**, and **Stop bits**.
 - Select **Parity** and **Flow control**.

Figure 4. Open New Connection



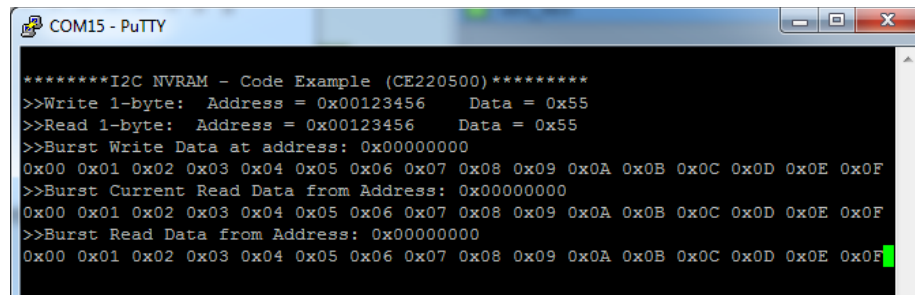
4. Select **Session** under **Category**. Select **Serial** under **Connection type** as shown in Figure 5. You can save this current session and **load** the settings when required. Enter a name in **Saved Sessions** and click **Save**. Click **Open** to proceed.

Figure 5. Select Communication Type in PuTTY



5. The COM terminal window then displays the code example results as shown in Figure 6. You may have to reprogram PSoC 4 with the code example hex file or reset PSoC 4 (already programmed) to restart the code execution and monitor the result.

Figure 6. Result Displayed on PuTTY



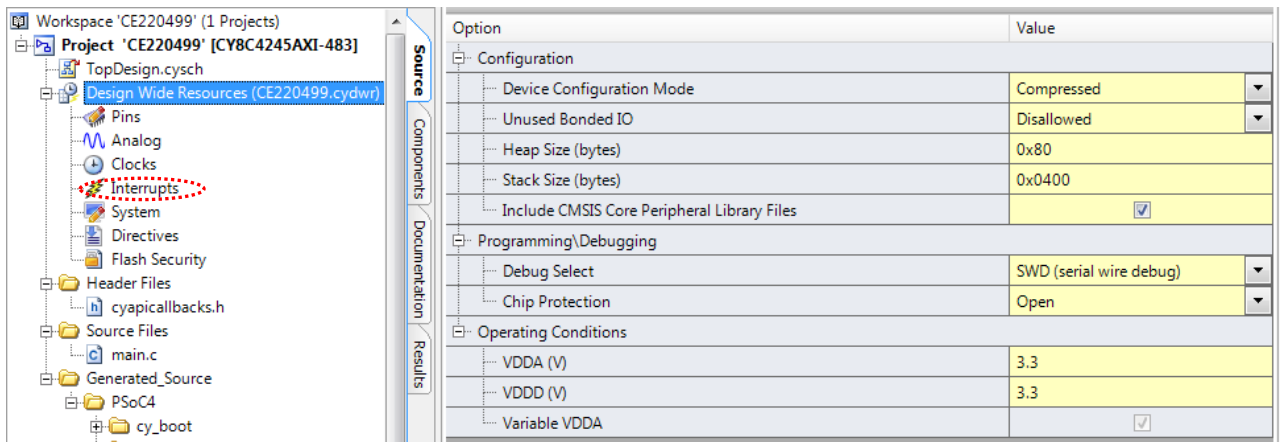
```

*****I2C NVRAM - Code Example (CE220500)*****
>>Write 1-byte: Address = 0x00123456    Data = 0x55
>>Read 1-byte: Address = 0x00123456    Data = 0x55
>>Burst Write Data at address: 0x00000000
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
>>Burst Current Read Data from Address: 0x00000000
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
>>Burst Read Data from Address: 0x00000000
0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07 0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F
  
```

Alternatively, you can run the HyperTerminal if supported on your PC to monitor the above result.

Design-Wide Resources

Make sure that V_{DD} (**PSoc Creator > Design Wide Resources > System** tab) is set to 2.7 V or above, as shown in [Figure 7](#), to drive the STATUS_LED. Also, make sure that PSoC 4 I/O voltage is set correctly to match the I²C NVRAM operating range (V_{DD}/V_{CC}).

 Figure 7. V_{DD} Setting using Design Wide Resources


[Figure 8](#) shows the pin assignment for the code example.

Figure 8. PSoC 4 Pin Assignments for Code Example

| | Name | / | Port | Pin | Lock |
|-------------------------------------|------------------|---|--------|-----|-------------------------------------|
| <input checked="" type="checkbox"/> | \I2C_Master:scl\ | | P4 [0] | 20 | <input checked="" type="checkbox"/> |
| <input checked="" type="checkbox"/> | \I2C_Master:sda\ | | P4 [1] | 21 | <input checked="" type="checkbox"/> |
| <input checked="" type="checkbox"/> | \UART:rx\ | | P0 [4] | 28 | <input checked="" type="checkbox"/> |
| <input checked="" type="checkbox"/> | \UART:tx\ | | P0 [5] | 29 | <input checked="" type="checkbox"/> |
| <input checked="" type="checkbox"/> | LED_GREEN | | P0 [2] | 26 | <input checked="" type="checkbox"/> |
| <input checked="" type="checkbox"/> | LED_RED | | P1 [6] | 43 | <input checked="" type="checkbox"/> |

To switch the code example from CY8CKIT-042 to any other PSoC 4 kit, change the project's device with the help of Device Selector called from the project's context menu as shown in [Table 1](#).

Table 1. Supported PSoC 4 Development Kits Versus PSoC 4 Parts

| Development Kit | Device |
|-----------------|-------------------|
| CY8CKIT-040 | CY8C4014LQS-422 |
| CY8CKIT-042 | CY8C4245AXI-483 |
| CY8CKIT-042-BLE | CY8C4248LQI-BL583 |
| CY8CKIT-044 | CY8C4247AZI-M485 |

The port (pin) assignments for the supported kits are shown in [Table 2](#).

Table 2. Pin Assignment on Supported Development Kits

| Code Example - Pin Name | PSoC 4 Development Kit - Port Name | | | |
|-------------------------|------------------------------------|-----------------|-------------|---|
| | CY8CKIT-042 | CY8CKIT-042-BLE | CY8CKIT-044 | CY8CKIT-040 |
| I2C_Master:scl\ | P4[0] | P3[5] | P4[0] | P1[2] |
| I2C_Master:sda\ | P4[1] | P3[4] | P4[1] | P1[3] |
| UART:rx\ | P0[4] | P3[0] | P7[0] | UART is not supported through SCB ^[Note 1] |
| UART:tx\ | P0[5] | P3[1] | P7[1] | |
| LED_GREEN | P0[2] | P3[6] | P2[6] | P1[1] ^[Note 2] |
| LED_RED | P1[6] | P2[6] | P0[6] | P3[2] ^[Note 2] |

Note 1: The device can chose Software Transmit UART component and configure it accordingly to monitor results.

Note 2: The code example uses the Control Register Component to control the status LED drive. The PSoC 4000 device on CY8CKIT-040 kit doesn't support the Control Register Component; therefore, the status LEDs should be driven through individual I/O controls and code example should be modified accordingly.

Components

[Table 3](#) lists the PSoC Creator components used in this example and the hardware resources used by each Component.

Table 3. PSoC Creator Components

| Component | Instance name | Hardware resources |
|-------------|------------------|---|
| SerialNVRAM | SerialNVRAM_I2C | Serial NVRAM Block in System > External Memory Interface component category |
| I2C | I2C_Master | I ² C (SCB Mode) in Communication > I2C component category |
| UART | UART | UART (SCB Mode) in Communication component category |
| Control Reg | STATUS_LED_CNTRL | Control Register in Digital > Registers component category |

[Figure 9](#) to [Figure 12](#) show the component configuration for the code example project.

Figure 9. Serial NVRAM Component Configuration and Parameter Settings

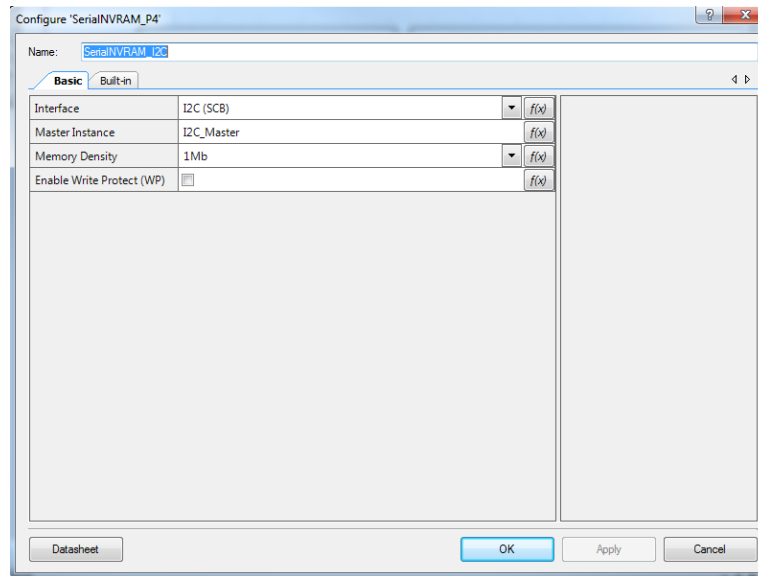
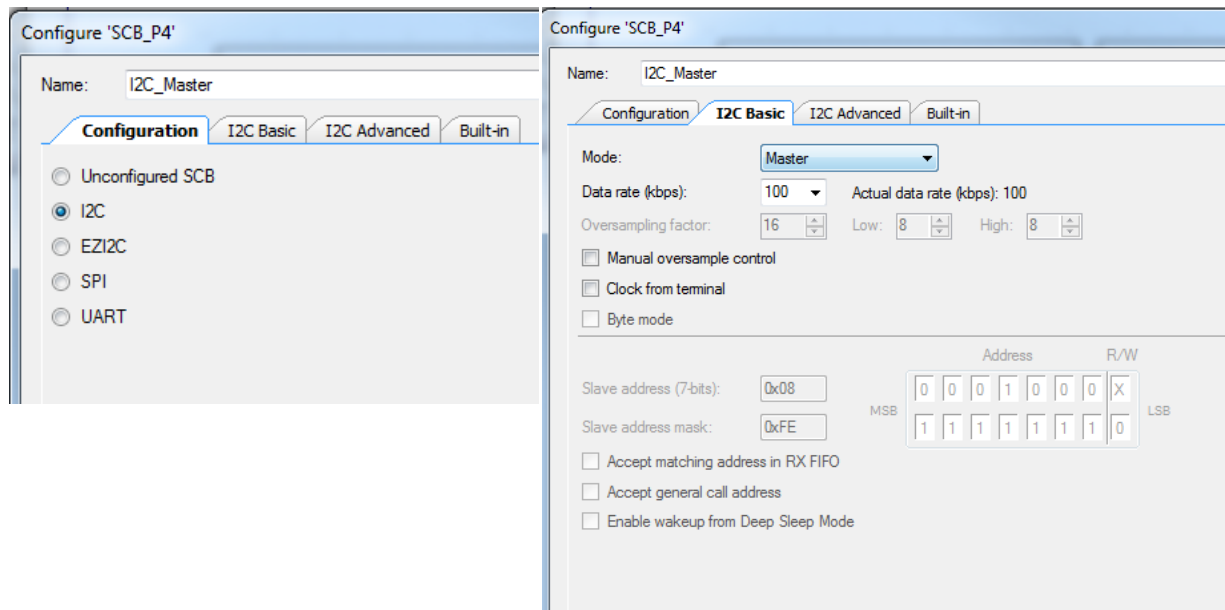

 Figure 10. I²C (SCB Mode) Configuration and Parameter Settings


Figure 11. UART (SCB Mode) Configuration and Parameter Settings

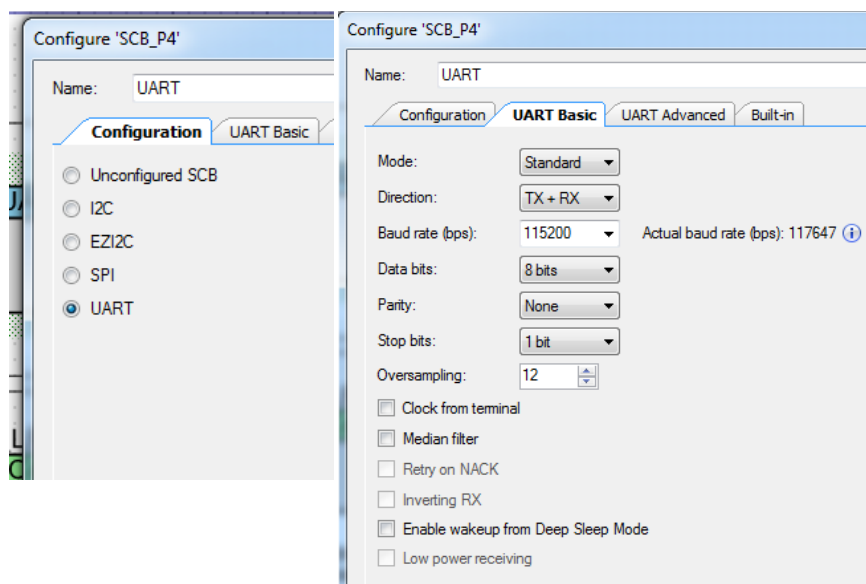
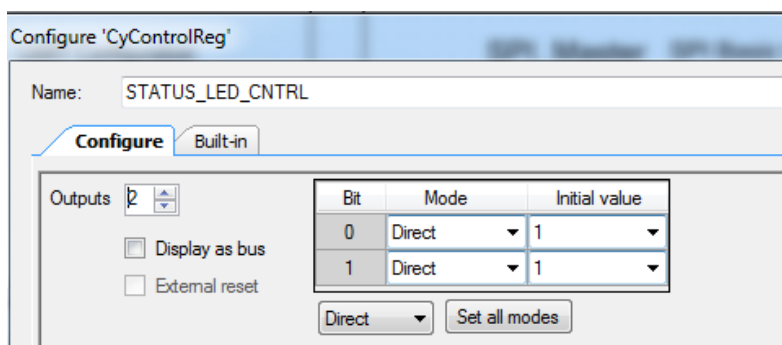


Figure 12. Control Register Configuration and Parameter Settings



Related Documents

| Application Notes | |
|--|--|
| AN96578 – Designing with I2C F-RAM | Describes I ² C F-RAM design in a system |
| Component Datasheets (Right click on the component in PSoC Creator schematic to access the component datasheet) | |
| SerialNVRAM | Serial NVRAM component configuration for different serial interfaces (SPI and I2C) and API details |
| I2C | I2C interface SCB mode |
| UART | UART configuration SCB mode |
| Control Reg | Allows firmware to set values to use for digital signals |
| Device Documentation | |
| FM24W256 | 256Kb, I2C F-RAM datasheet |
| All Serial NVRAM Datasheets | Serial (SPI and I2C) NVRAM (F-RAM and nvSRAM) |

| Development Kit Documentation | |
|--|---|
| CY8CKIT-042 PSoC 4 Pioneer Kit | Development kit for the PSoC 4200 device family |
| CY15FRAMKIT-001 Serial F-RAM Development Kit | Development kit for Cypress's serial F-RAMs. This kit has a wide operating range (2.7V to 5.5V), 256-Kbit I2C F-RAM in 8-pin SOIC package |

Document History

Document Title: CE220500 - Interfacing the I²C F-RAM with PSoC 4

Document Number: 002-20500

| Revision | ECN | Orig. of Change | Submission Date | Description of Change |
|----------|---------|-----------------|-----------------|-----------------------|
| ** | 5955430 | ZSK | 11/07/2017 | New code example |

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at [Cypress Locations](#).

Products

| | |
|-------------------------------|--|
| ARM® Cortex® Microcontrollers | cypress.com/arm |
| Automotive | cypress.com/automotive |
| Clocks & Buffers | cypress.com/clocks |
| Interface | cypress.com/interface |
| Internet of Things | cypress.com/iot |
| Memory | cypress.com/memory |
| Microcontrollers | cypress.com/mcu |
| PSoC | cypress.com/psoc |
| Power Management ICs | cypress.com/pmic |
| Touch Sensing | cypress.com/touch |
| USB Controllers | cypress.com/usb |
| Wireless Connectivity | cypress.com/wireless |

PSoC® Solutions

[PSoC 1](#) | [PSoC 3](#) | [PSoC 4](#) | [PSoC 5LP](#) | [PSoC 6](#)

Cypress Developer Community

[Forums](#) | [WICED IOT Forums](#) | [Projects](#) | [Videos](#) | [Blogs](#) | [Training](#) | [Components](#)

Technical Support

cypress.com/support

All other trademarks or registered trademarks referenced herein are the property of their respective owners.



Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

© Cypress Semiconductor Corporation, 2017. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("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 hardware products. Any other use, reproduction, 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. 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. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, 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 or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

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.