Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

CE210291 - PSoC® 4 CapSense® Low-Power One Button

Objective

This code example demonstrates how to implement a low-power CapSense button with an average current consumption of 6 µA per button.

Overview

This code example implements a CapSense button. Using the low-power modes available in the PSoC device, an average current of 6 µA is achieved for a button sensor when no touch is present. The example supports a CapSense tuner to read the CapSense debug data such as raw count, baseline, and difference count.

Requirements

Tool: PSoC Creator™ 4.1 and later versions

Programming Language: C (ARM® GCC 5.4.1, ARM MDK)

Associated Parts: All PSoC 4, PSoC 4 M-Series, PSoC 4 L-Series, PSoC 4 S-Series, PSoC 4 BLE, and PRoC BLE devices

Related Hardware: CY8CKIT-041-40XX, CY8CKIT-041-41XX, CY8CKIT-042, CY8CKIT-044, CY8CKIT-046, and CY8CKIT-042-BLE.

Design

The project demonstrates the core functionality of the CapSense Component. Figure 1 shows the PSoC Creator schematics of this code example. This code example uses CapSense, EZI2C Slave, and Global Signal Reference Components.

The CapSense Component is configured with a one-button widget. The project uses the CapSense Sigma Delta (CSD) manual tuning mode with Current output Digital to Analog Converter (IDAC) auto-calibration enabled. The EZI2C Slave Component is used to monitor the sensor data and slider touch position information on a PC using the CapSense tuner available in the PSoC Creator IDE via I^2C communication.

To reduce the power consumed by the PSoC device and provide an optimum touch response, this code example implements two modes: Fast Scan Mode and Slow Scan Mode. When the user is interacting with the buttons, the PSoC device is in the Fast Scan Mode and when the user is not interacting with the buttons for a specific duration, the Slow Scan Mode is used.

In the Fast Scan Mode, the button sensor is scanned at a refresh rate of 33 Hz (or scan interval of 30 ms). The PSoC device is put to Deep Sleep after the CapSense data is processed. The Global Signal Reference Component, which is configured as a watchdog timer (WDT) is used to periodically wake up the device from Deep Sleep mode. This mode provides the optimum touch response, but consumes a higher average current (< 30 µA) when compared to the Slow Scan Mode.

The Slow Scan Mode is similar to the Fast Scan Mode except that the refresh rate is 5 Hz (or scan interval is 200 ms). The Slow Scan Mode consumes a lower average current of 5 µA per button, but with a slower touch response. Once a touch is detected in the Slow Scan Mode, the PSoC device switches to the Fast Scan Mode to provide an optimum touch response at the expense of a higher current consumption.

The firmware flowchart is shown in Figure 2.

Figure 1. Top Design

CE210291 PSoC4 CapSense Low Power One Button

This code example demonstrates how to implement a low-power CapSense button with an average current consumption of 6uA per button.

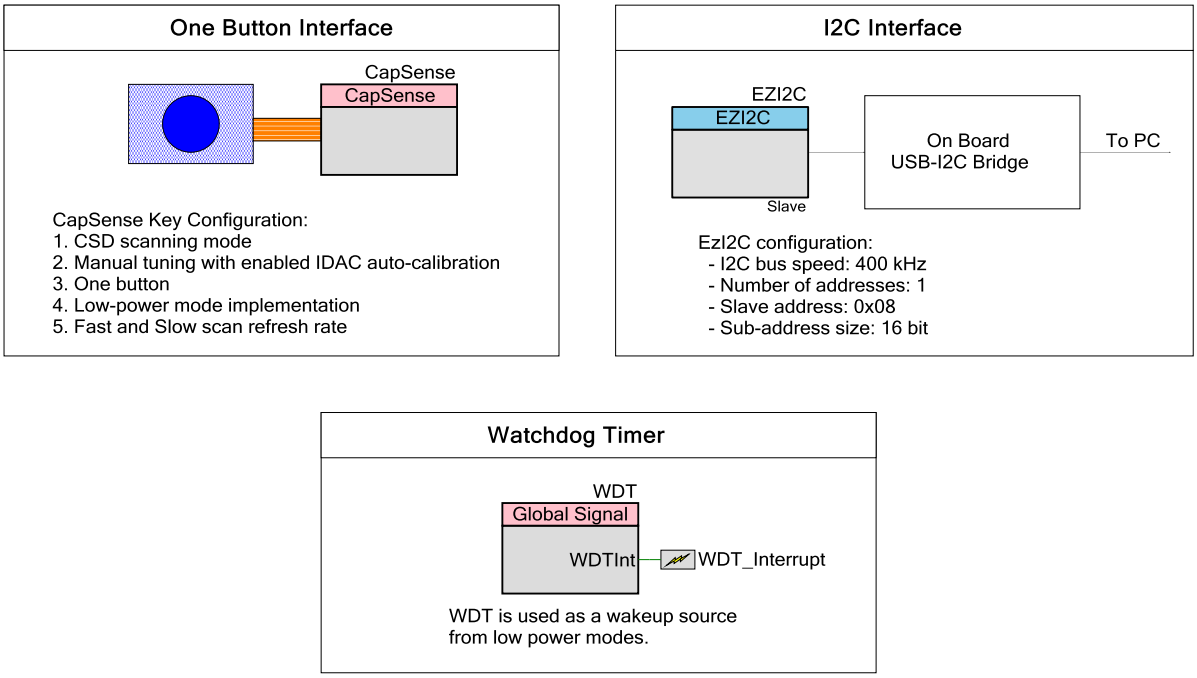
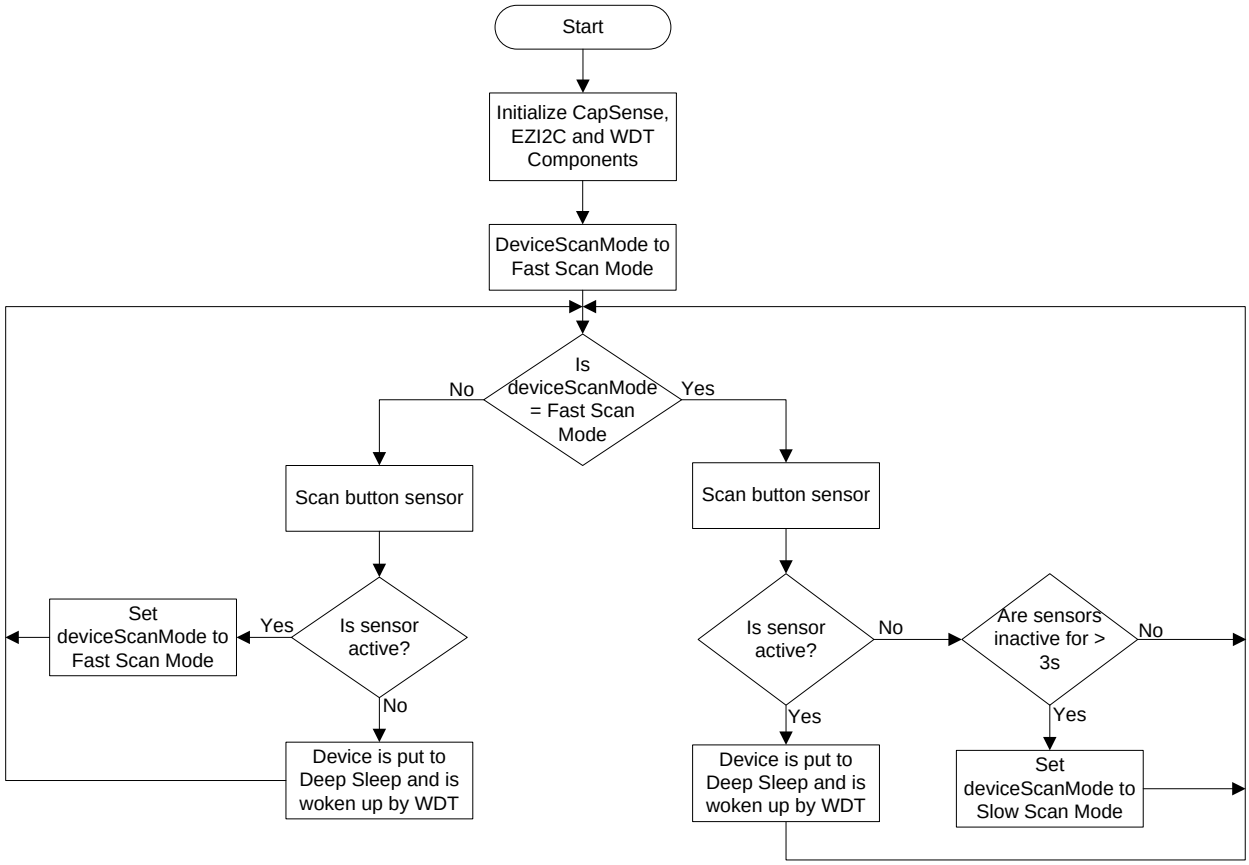


Figure 2. Firmware Flowchart



Design Considerations

The code example is designed for PSoC 4000S and the associated CY8CKIT-041-40XX kit. The design is easily portable to other PSoC 4 devices and kits, typically by just changing the device and pin assignments.

To switch from CY8CKIT-041-40XX to other PSoC 4 pioneer kits, follow the steps below:

1. Select the appropriate device with the Device Selector called from the project's context menu (Figure 3). Table 1 lists the device number for each pioneer kit.

Figure 3. Select Device Selector from Project's Context Menu

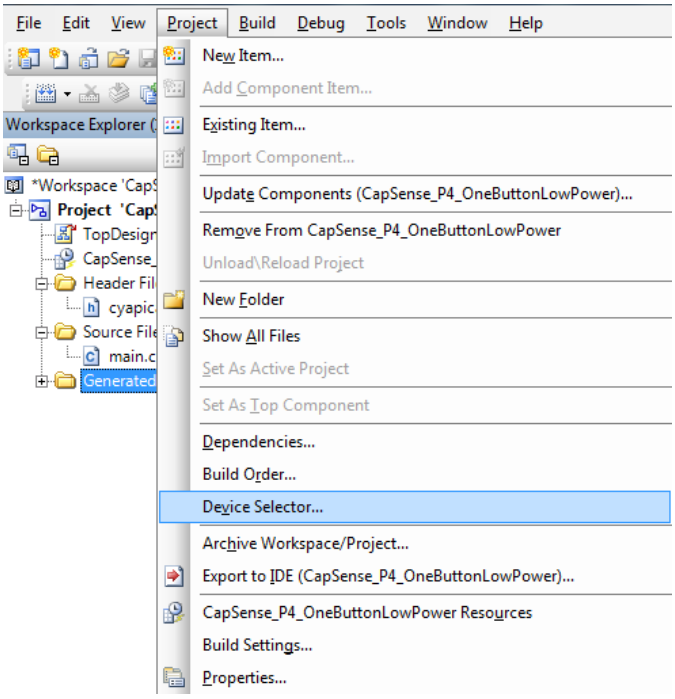


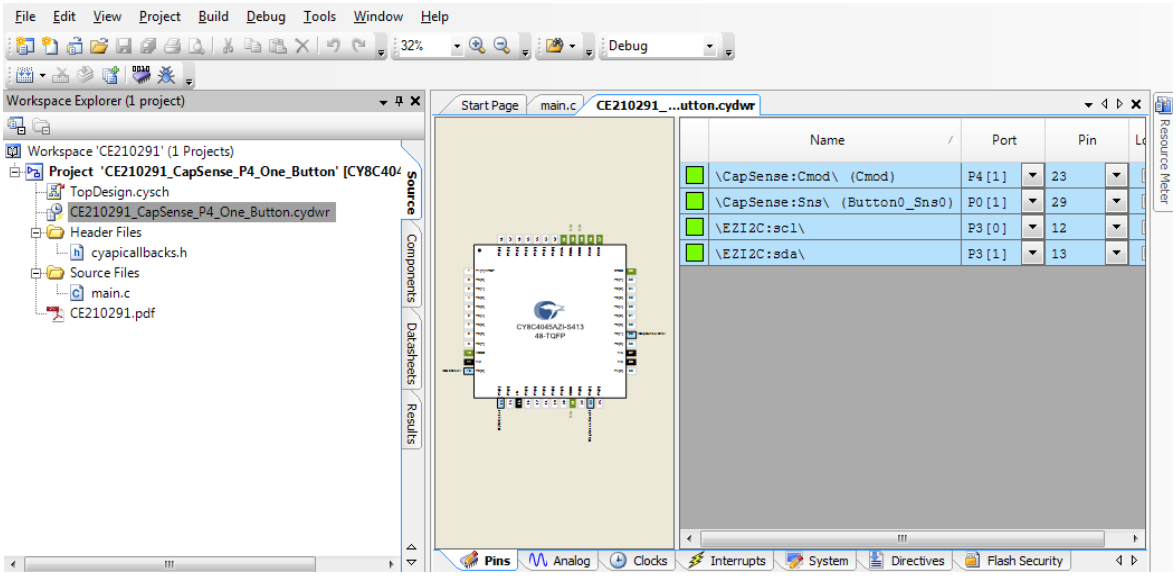
Table 1. Pin Assignment for the PSoC4 CapSense Low-Power One Button Project

Pin name	Development Kits					
	CY8CKIT-041-40XX (CY8C4045AZI-S423)	CY8CKIT-041-41XX (CY8C4146AZI-S433)	CY8CKIT-042 (CY8C4245AXI-483)	CY8CKIT-042-BLE (CY8C4247LQI-BL483)	CY8CKIT-044 (CY8C4247AZI-M485)	CY8CKIT-046 (CY8C4248BZI-L489)
\Capsense: Cmod\	P4[1]	P4[1]	P4[2]	P4[0]	P4[2]	P4[2]
\CapSense: Sns[0]\ (Button0_Sns0)	P0[1]	P0[1]	P1[1]	P2[1]	P4[4]	P0[6]
\EZI2C: scl\	P3[0]	P3[0]	P3[0]	P3[5]	P4[0]	P4[0]
\EZI2C: sda\	P3[1]	P3[1]	P3[1]	P3[4]	P4[1]	P4[1]

When you select the device, pins are assigned automatically in the design wide resource file according to the device selected.

Note: If the assigned pins are not as shown in Table 1 or you want to overwrite the existing pin assignments, double-click the project's design wide resource (DWR) file (Figure 4) in the workspace explorer window and assign the pins. See the device datasheet to learn about how to use other pins.

Figure 4. Pin Assignments in Project's Design Wide Resource File



2. Select 3.3 V as supply by using switch SW5. For other kits, refer to the kit user guide.
3. Build the project and ensure that there are no errors or warnings.

   Note: For a PSoC 4 device, before building the project, the modulator clock frequency should to be set to 12 MHz instead of 24 MHz because the maximum frequency supported is 12 MHz.

PSoC Creator Components

Table 2 lists the PSoC Creator Components used in this example, as well as the hardware resources used by each Component.

Table 2. List of PSoC Creator Components

Component	Instance Name	Component Version	Hardware Resources
CapSense	CapSense	v4.0	CSD, and 2 GPIO pins
EZI2C Save (SCB mode)	EZI2C	v3.20	SCB, 2 GPIO pins
Interrupt	WDT_Interrupt	v1.70	Interrupt
Global Signal Reference	WDT	v2.10	Watchdog Timer

Parameter Settings

CapSense

Figure 5, Figure 6 and Figure 7 show the settings for CapSense Component. See the CapSense component datasheet for additional information.

Figure 5. CapSense Component's Basic Tab

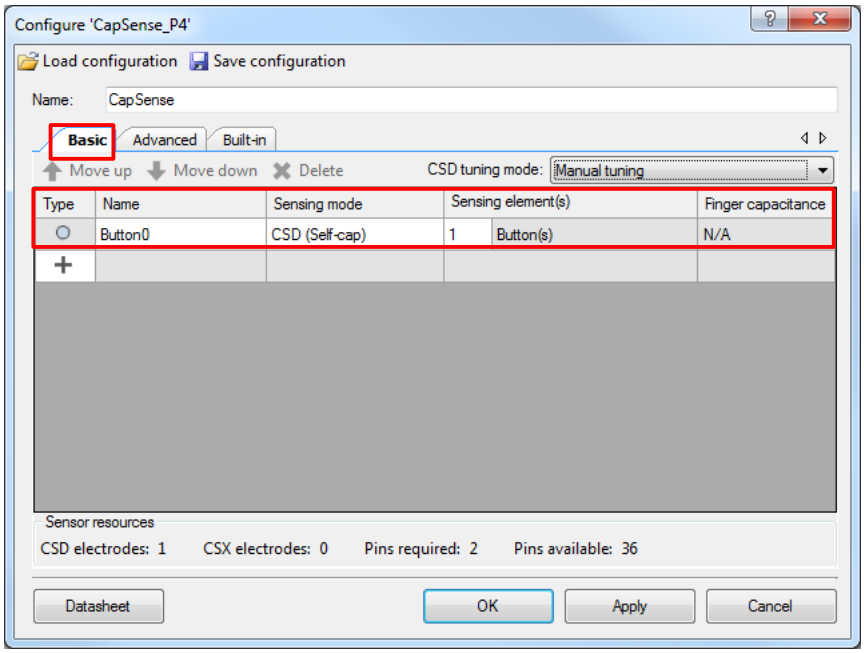


Figure 6. CapSense Component Customizer Advanced Tab (CSD Settings)

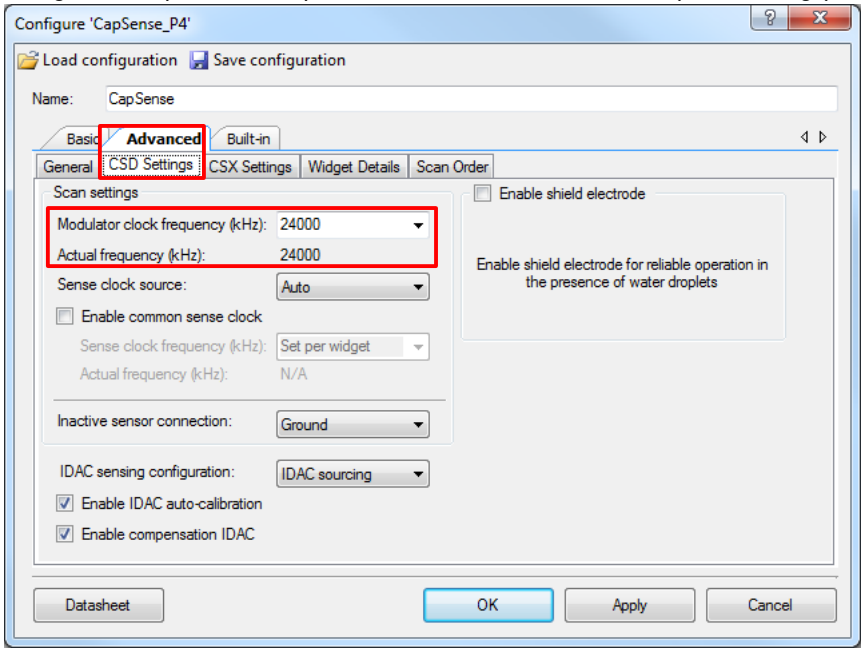
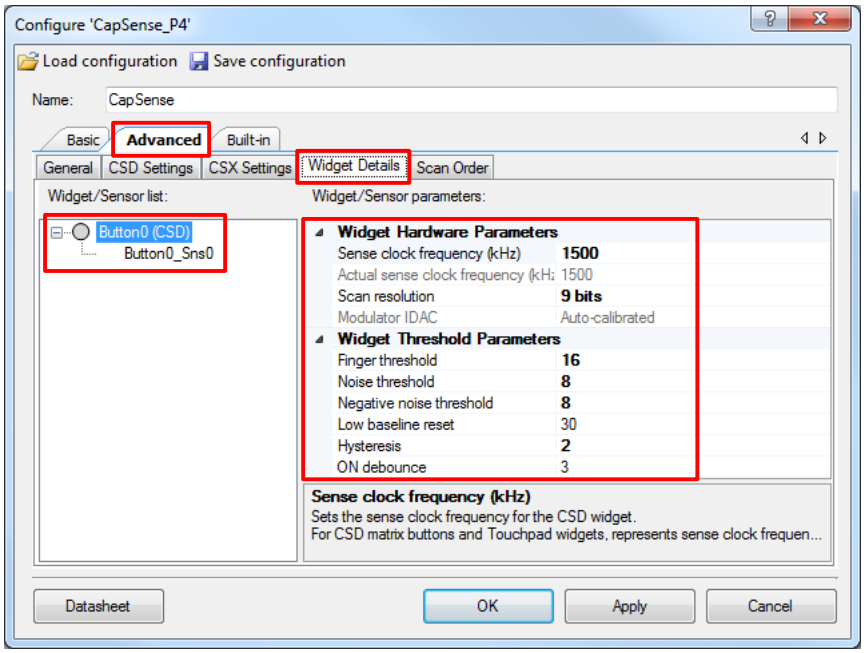


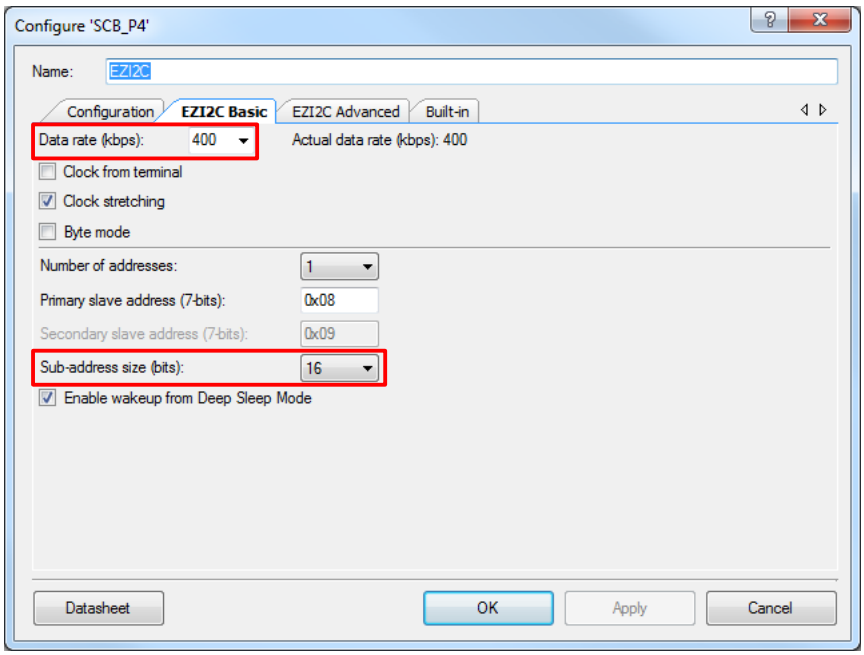
Figure 7. CapSense Component Widget Details Settings



EZI2C Slave Component

Figure 8 shows the settings for EZI2C Slave Component. See the SCB component datasheet for additional information.

Figure 8. EZI2C Component's Basic Tab



Design-Wide

Figure 9 and Figure 10 show the non-default cydwr settings for the project.

Figure 9. CYWDR Pins Tab Settings

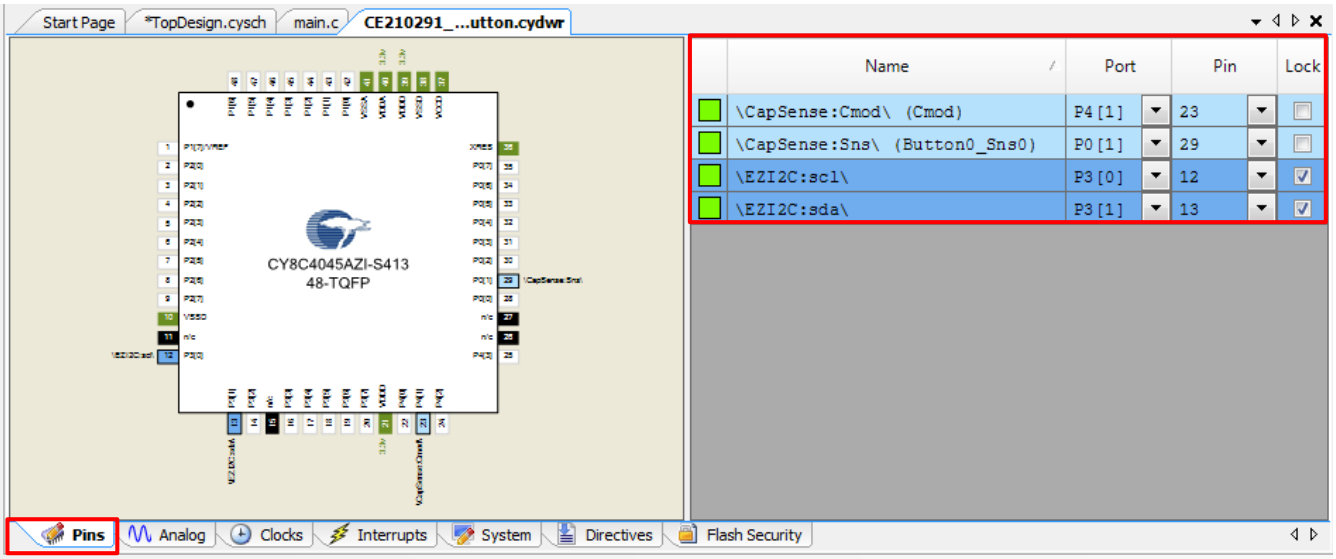
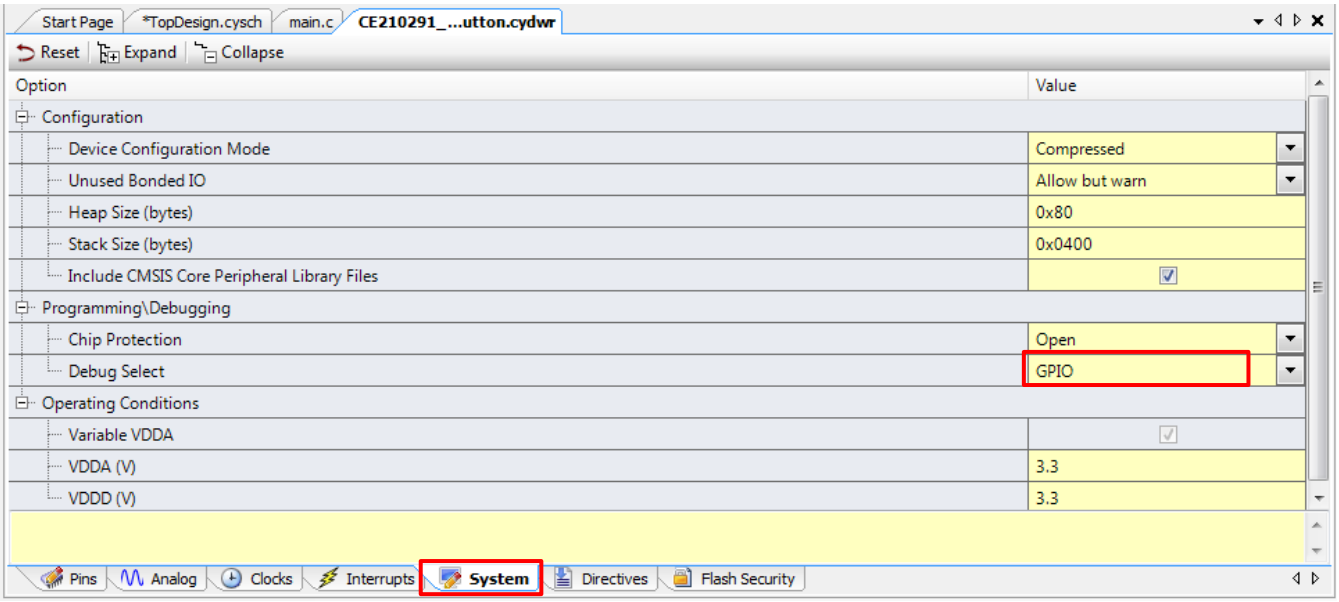


Figure 10. CYDWR System Tab Settings



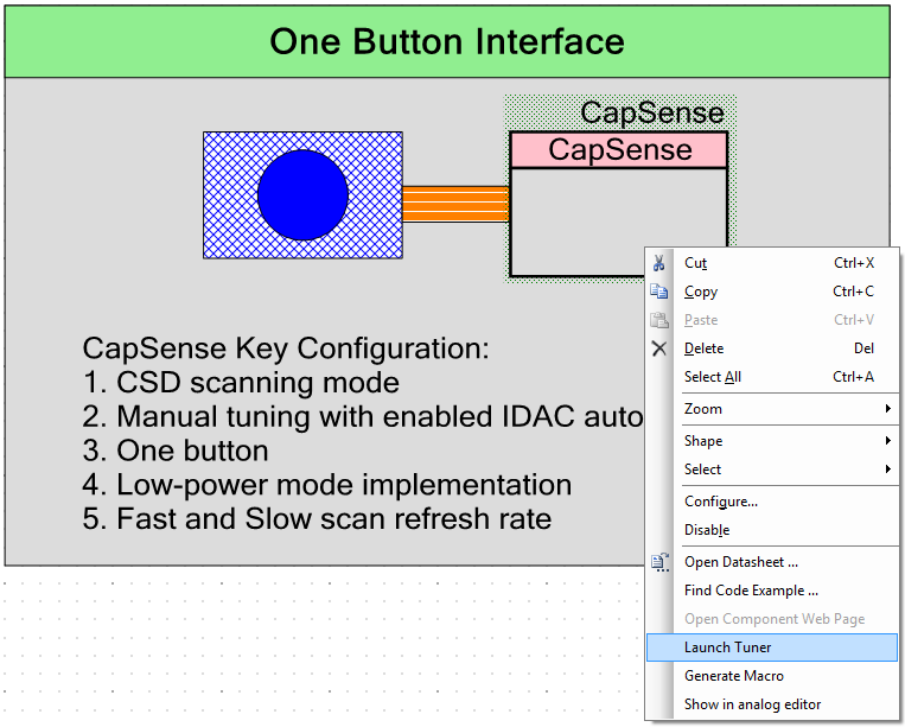
Operation

Follow the steps below to test the project on CY8CKIT-041:

1. Plug the CY8CKIT-041-40XX board into your computer's USB port and program the project into the kit. Refer to the kit user guide for details on steps for programming.
2. Connect an ammeter between test points P4.VDD and VDD on the CY8CKIT-041-40XX board. Refer to the kit user guide "Current Measurement Switch" section for complete details on power measurement steps.
3. Touch the left button on the kit and confirm that the current displayed on the ammeter in Fast Scan Mode is ~30 µA.
4. Release the finger from the button and wait for 3 seconds. Note that the current displayed on the ammeter in Slow Scan Mode is ~6 µA.

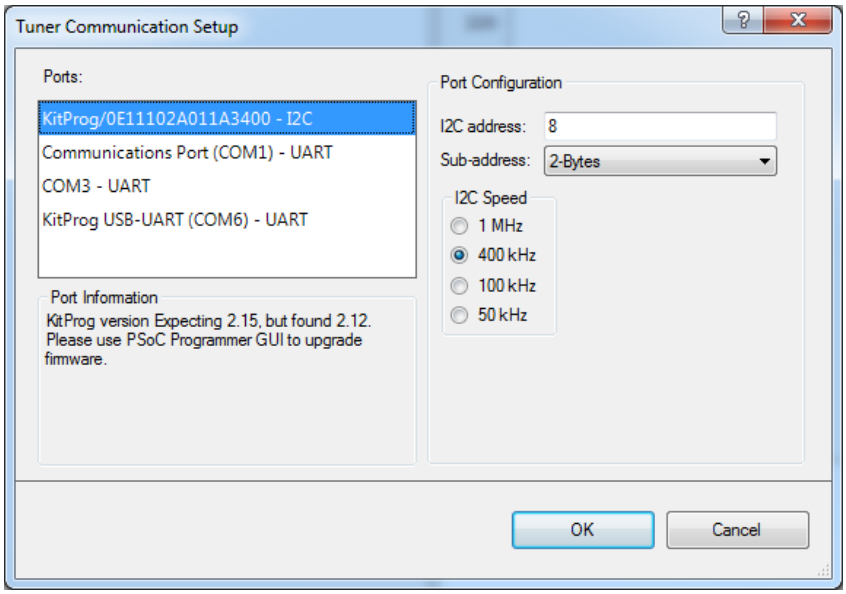
5. By default, the parameters updated in tuner are not written to the firmware. To enable this, set the macro TUNER_UPDATE_ENABLE to '1' in the *main.c* file. Then build the project and program the device. Note that additional current will be consumed if the conditional code is enabled.
6. To launch the Tuner GUI, right-click the CapSense Component as shown in Figure 11. Then select Launch Tuner in the menu.

Figure 11. Launch Tuner GUI



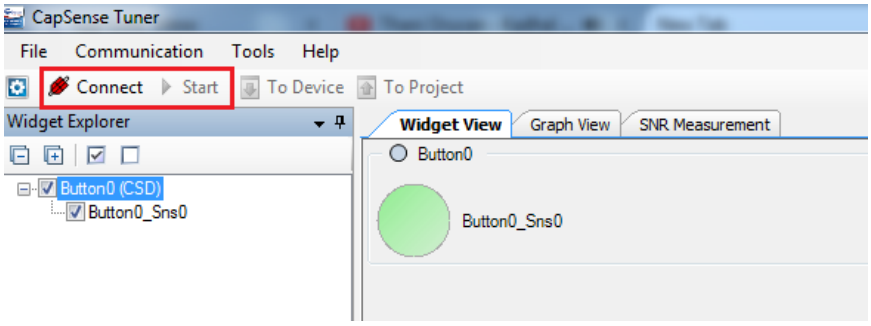
7. Navigate to Tools/Protocol Settings in the Tuner GUI menu to set up the I2C communication (Figure 12).
   - Select the I2C port of the kit in the Ports menu.
   - Choose the I2C address, sub-address size, and speed as shown in Figure 12.

Figure 12. Setting up I2C Communication



8. In the tuner GUI, click the **Connect** button followed by **Start** button (Figure 13).

Figure 13. Starting Communication



9. After establishing the I2C communication between the device and Tuner GUI, you can observe the sensor data and the button touch status.

10. Figure 14 and Figure 15 show sensor debug data such as difference count, raw count, and baseline.

Figure 14. Tuner: Widget View

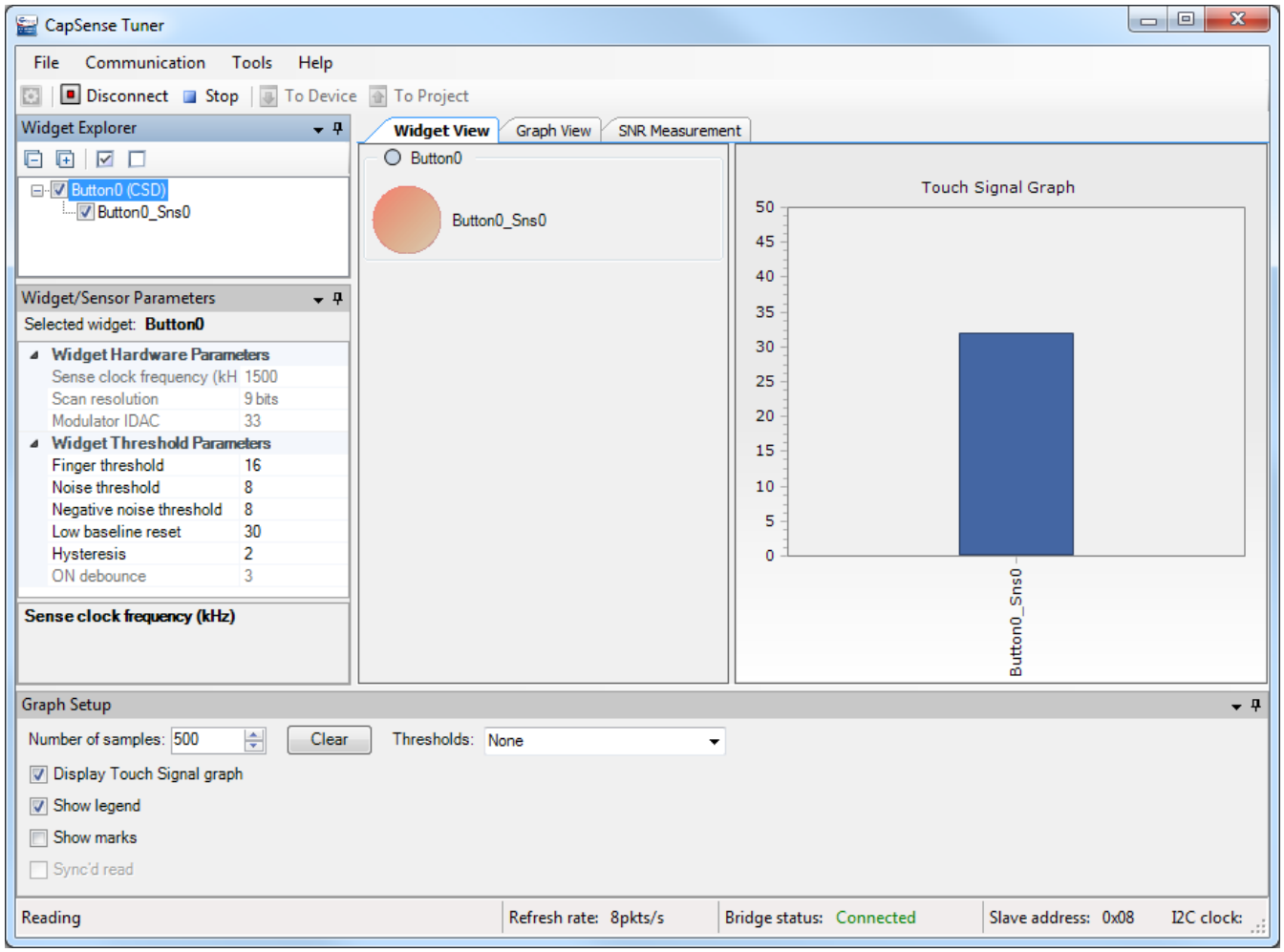
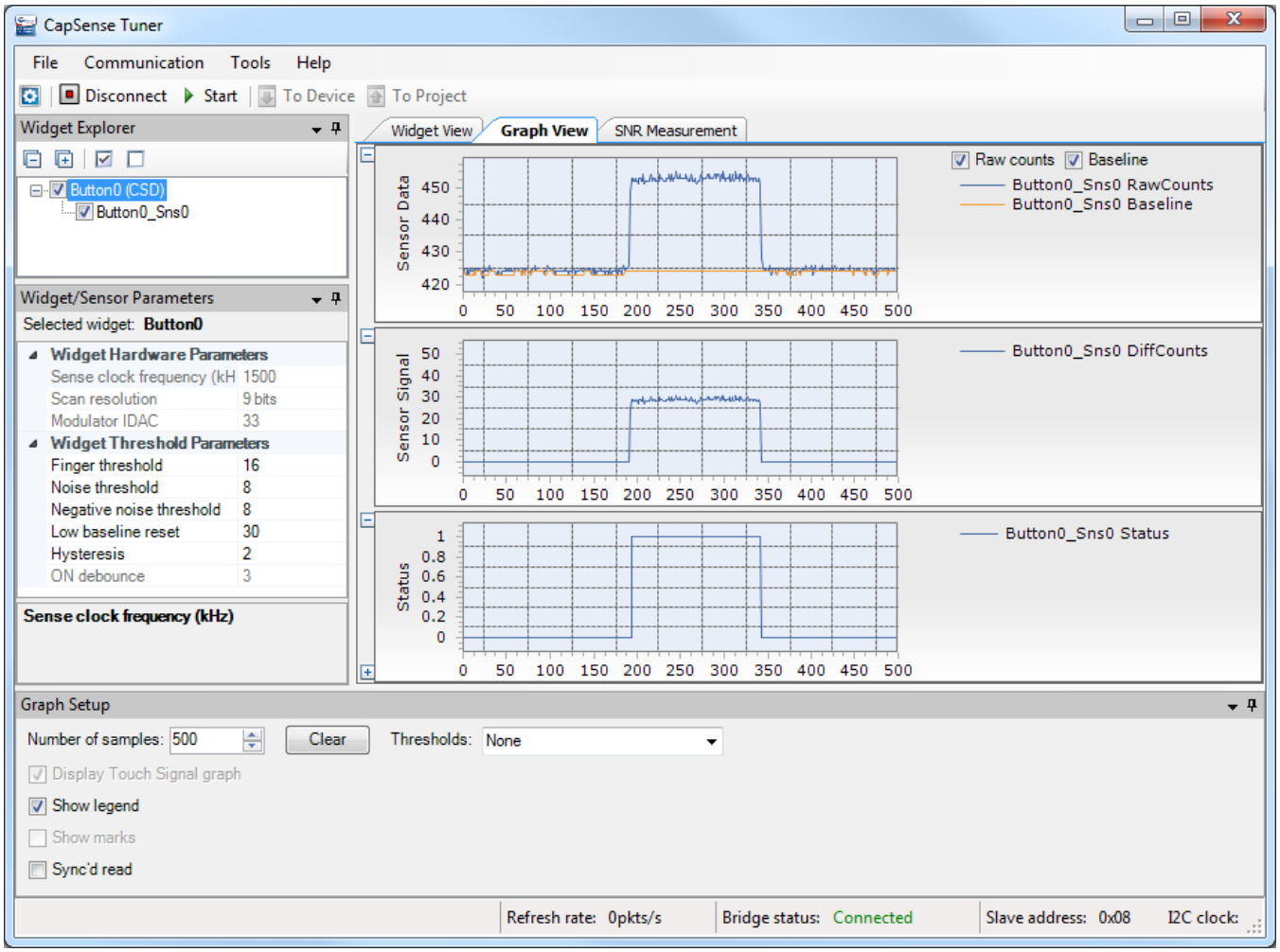


Figure 15. Tuner: Graph View



Related Documents

Table 3 lists relevant application notes, code examples, device datasheets, and PSoC Creator Component datasheets.

Table 3. Related Documents

Application Notes		
AN85951	AN85951 - PSoC® 4 CapSense® Design Guide	Design Guide shows how to design capacitive touch sensing applications with PSoC 4.
AN79953	Getting Started with PSoC 4	Describes the PSoC 4, and how to build a first PSoC Creator project
Code Examples		
CE210290	PSoC 4 CapSense Low-Power Ganged Sensor	
PSoC Creator Component Datasheets		
CapSense	Supports capacitive touch sensing	
EZI2C Slave	Supports I2C slave operation	
Global Signal Reference	Supports Watchdog timer	
Device Documentation		
PSoC 4 Datasheets	PSoC 4 Technical Reference Manuals	
Development Kit (DVK) Documentation		
CY8CKIT-041-40xx PSoC 4 S-Series Pioneer Kit		
CY8CKIT-041-41XX PSoC 4 S-Series Pioneer Kit		
CY8CKIT-042 PSoC® 4 Pioneer Kit		
CY8CKIT-042-BLE Bluetooth® Low Energy (BLE) Pioneer Kit		
CY8CKIT-044 PSoC® 4 M-Series Pioneer Kit		
CY8CKIT-046 PSoC® 4 L-Series Pioneer Kit		

PSoC Resources

Cypress provides a wealth of data at www.cypress.com to help you select the right PSoC device, and quickly and effectively integrate the device into your design. For a comprehensive list of resources, see KBA86521, How to Design with PSoC 3, PSoC 4, and PSoC 5LP. The following is an abbreviated list for PSoC 4:

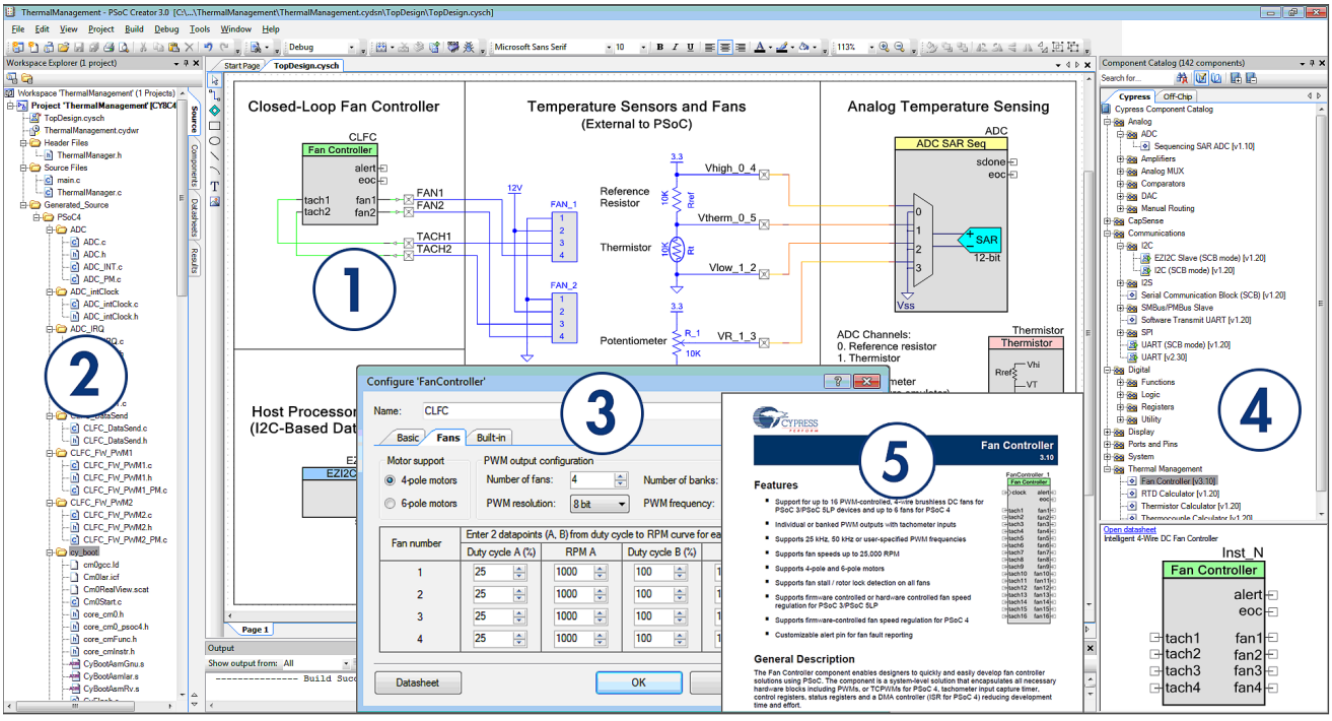
- **Overview:** PSoC Portfolio, PSoC Roadmap
- **Product Selectors:** PSoC 1, PSoC 3, PSoC 4, or PSoC 5LP. In addition, PSoC Creator includes a device selection tool.
- **Datasheets**: Describe and provide electrical specifications for the PSoC 4000, PSoC 4000S, PSoC 4100S, PSoC 4100, and PSoC 4200, PSoC 4xx7 BLE, PSoC 4200-M and PSoC Analog Coprocessor device families
- **CapSense® Design Guide:** Learn how to design capacitive touch-sensing applications with the PSoC 4 family of devices.
- **Application Notes and Code Examples:** Cover a broad range of topics, from basic to advanced level. Many of the application notes include code examples. PSoC Creator provides additional code examples.
- **Technical Reference Manuals (TRM):** Provide detailed descriptions of the architecture and registers in each PSoC 4 device family.
- **PSoC Training Videos**: These videos provide step-by-step instructions on getting started building complex designs with PSoC.
- **Development Kits:**
  - CY8CKIT-040, CY8CKIT-041-40xx, CY8CKIT-041-41xx, CY8CKIT-042, CY8CKIT-042-BLE, CY8CKIT-044 and CY8CKIT-048 PSoC Pioneer Kits are easy-to-use and inexpensive development platforms. These kits include connectors for Arduino™ compatible shields and Digilent® Pmod™ daughter cards.
  - CY8CKIT-049 is a very low-cost prototyping platform for sampling PSoC 4 devices.
  - CY8CKIT-001 is a common development platform for all PSoC family devices.
- The MiniProg3 device provides an interface for flash programming and debug.

PSoC Creator

PSoC Creator is a free Windows-based Integrated Design Environment (IDE). It enables concurrent hardware and firmware design of systems based on PSoC 3, PSoC 4, and PSoC 5LP. See Figure 16– with PSoC Creator, you can:

1. Drag and drop Components to build your hardware system design in the main design workspace
2. Codesign your application firmware with the PSoC hardware
3. Configure Components using configuration tools
4. Explore the library of 100+ Components
5. Review Component datasheets

Figure 16. PSoC Creator Features



Document History

Document Title: CE210291 – PSoC® 4 CapSense® Low-Power One Button

Document Number: 002-10291

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5144063	AKSM	10/08/2015	New code example
*A	5442733	VMED	09/20/2016	Updated components to the latest versions available in PSoC Creator 4.0
*B	5638792	VMED	03/02/2017	Updated components to the latest versions available in PSoC Creator 4.1 Minor edits throughout the document Updated template
*C	5740270	AESATP12	05/26/2017	Updated logo and copyright.

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, 2016-2017. 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.