CE214022 – Low-Power CapSense® Buttons

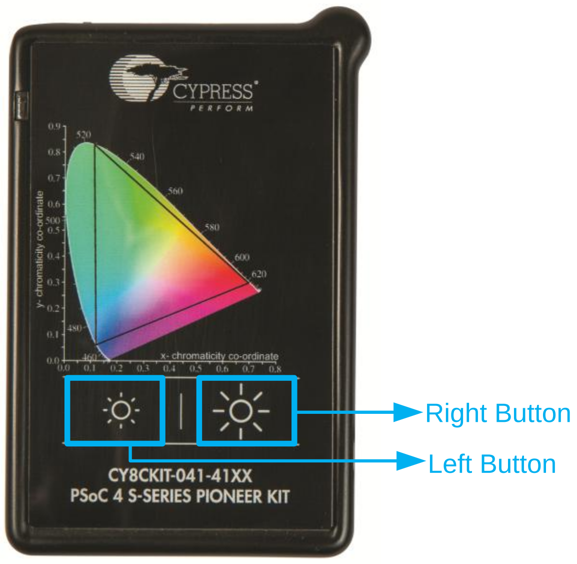
Objective

This code example demonstrates how to implement low-power CapSense® buttons with an average current consumption of 5 µA per button.

Overview

This code example implements two CapSense buttons using the CY8CKIT-041-41XX PSoC 4100S Pioneer Kit, as shown in Figure 1. The left button is used to control the onboard RGB LED color, and the right button is used to control the brightness of the RGB LED. Using the low-power modes available in the PSoC® 4100S device, an average current of 5 µA per button is achieved when the touch is not detected.

Figure 1. Left and Right Buttons on Kit



Requirements

Tool: PSoC Creator™ 4.0 or later versions

Programming Language: C (ARM® GCC 4.9.3)

Associated Parts: All PSoC 4100S parts

Related Hardware: CY8CKIT-041-41XX PSoC 4100S Pioneer Kit

Design

Figure 2 and Figure 3 show the PSoC Creator schematics of this code example. The code example uses the CapSense, PWM, Pins, Clock, and EZI2C Slave Components.

The CapSense Component is configured to scan two self-capacitance-based button widgets—left button and right button and a ganged widget. The two button sensors are combined and scanned as a ganged sensor. The EZI2C Slave Component is used to monitor the sensor data on the PC using the CapSense Tuner available in the PSoC Creator integrated design environment (IDE).

The PWM Component controls the brightness of the RGB LED by driving a pseudo-random PWM signal. A pseudo-random PWM signal spreads the energy of the PWM signal at different frequencies so that it is easy to filter the higher order harmonics, if required.

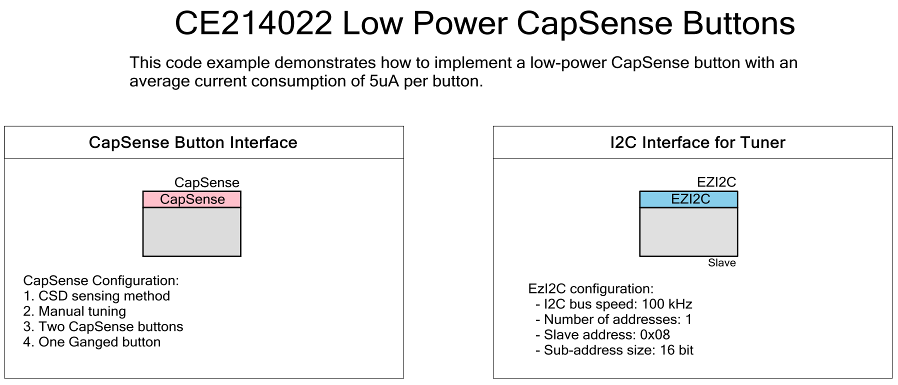
Figure 4 shows the flow chart for the code example. To reduce the power consumed by the PSoC device and to provide an optimum touch response, this code example implements two modes: Fast Scan and Slow Scan. When the user is interacting with the buttons, the PSoC device is in the Fast Scan mode, and when the user is not interacting with the buttons for a specific duration, the Slow Scan mode is used.

In the Fast Scan mode, both button sensors are scanned at a refresh rate of 50 Hz (or a scan interval of 20 ms), and the RGB LED is driven based on the button status. The PSoC device is put into the CPU Sleep mode after the CapSense data is processed. The watchdog timer is used to periodically wake up the device from the Sleep mode. This mode provides an optimum touch response, but consumes a higher power compared to the Slow Scan mode.

In the Slow Scan mode, both the button sensors are ganged and scanned at a refresh rate of 10 Hz (or a scan interval of 100 ms). The RGB LED is turned OFF, and the PSoC device is put into the Deep Sleep mode periodically. The Slow Scan mode consumes a lower average power 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 the optimum touch response at the expense of a higher power consumption.

In the Fast Scan mode, when the left button is touched, the color of the RGB LED changes in the following order: Red → Green → Blue → Red. When the right button is touched, the brightness of the RGB LED varies in the following order: Low → Medium → High → Low.

Figure 2. TopDesign – CapSense

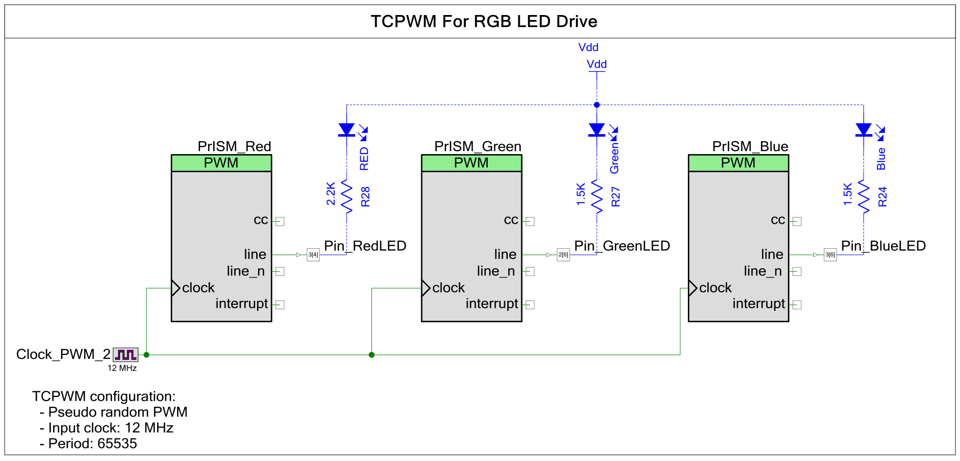

CE214022 Low Power CapSense Buttons
CapSense Button Interface
CapSense
CapSense
I2C Interface for Tuner
EZI2C
EZI2C
Slave

This code example demonstrates how to implement a low-power CapSense button with an average current consumption of 5uA per button.

CapSense Configuration:
1. CSD sensing method
2. Manual tuning
3. Two CapSense buttons
4. One Ganged button

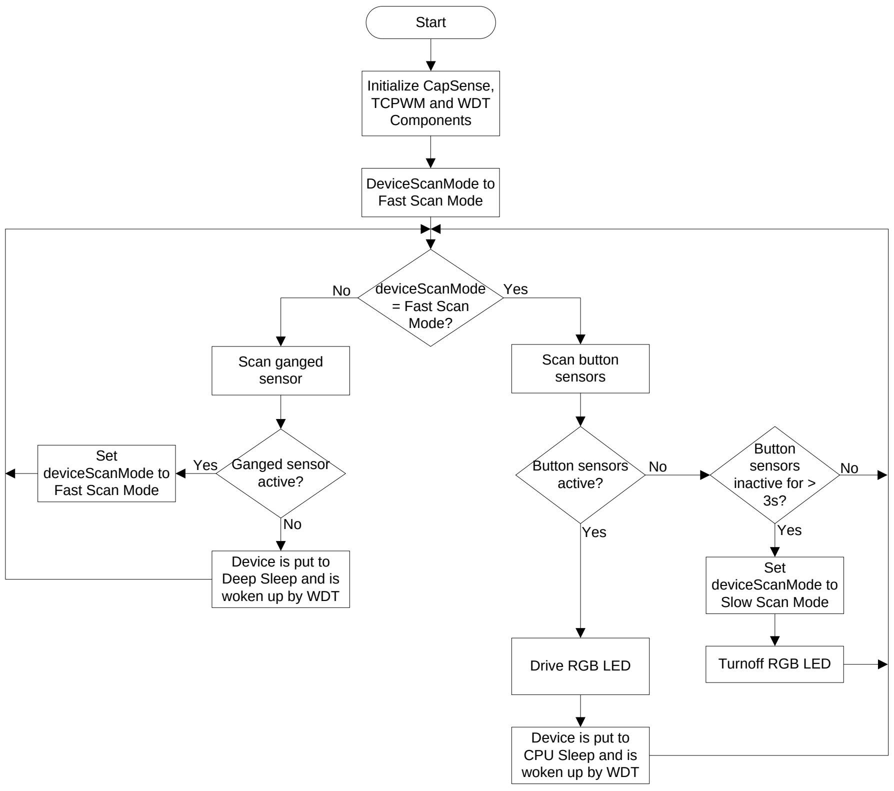
EzI2C configuration:
- I2C bus speed: 100 kHz
- Number of addresses: 1
- Slave address: 0x08
- Sub-address size: 16 bit

Figure 3. TopDesign – RGB LED Drive



TCPWM configuration:
- Pseudo random PWM
- Input clock: 12 MHz
- Period: 65535

Figure 4. Flow Chart


Start
Initialize CapSense, TCPWM and WDT Components
DeviceScanMode to Fast Scan Mode
deviceScanMode = Fast Scan Mode?
No
Yes
Scan ganged sensor
Scan button sensors
Ganged sensor active?
Yes
No
Set deviceScanMode to Fast Scan Mode
Device is put to Deep Sleep and is woken up by WDT
Button sensors active?
No
Yes
Button sensors inactive for > 3s?
No
Yes
Set deviceScanMode to Slow Scan Mode
Turnoff RGB LED
Drive RGB LED
Device is put to CPU Sleep and is woken up by WDT

Design Considerations

This code example is designed to run on the CY8CKIT-041-41XX PSoC 4100S Pioneer Kit with the PSoC 4100S device. To port the design to other PSoC 4 devices and kits, you must change the target device in the Device Selector, change the pin assignments in the *.cydwr* settings, and retune the CapSense sensors. For the tuning procedure, see AN85951 – PSoC 4 and PSoC Analog Coprocessor CapSense Design Guide.

The response time for the first touch after the sensor is inactive is about 100 ms because the refresh rate is set to100 ms in the Slow Scan mode to achieve an average current of 5 µA per button. You can configure the refresh rate by changing the macro `LOOP_TIME_SLOWSCANMODE` in the *main.c* file.

Hardware Setup

The code example works with the default settings on the CY8CKIT-041-41XX PSoC 4100S Pioneer Kit. If the settings are different from the default values, see the “Switches Default Position” table in the kit guide to reset to the default settings.

Software Setup

The code example does not require any special software considerations.

PSoC Creator Components

Table 1 lists the PSoC Creator Components used in this example, as well as the hardware resources used by each.

Table 1. PSoC Creator Components

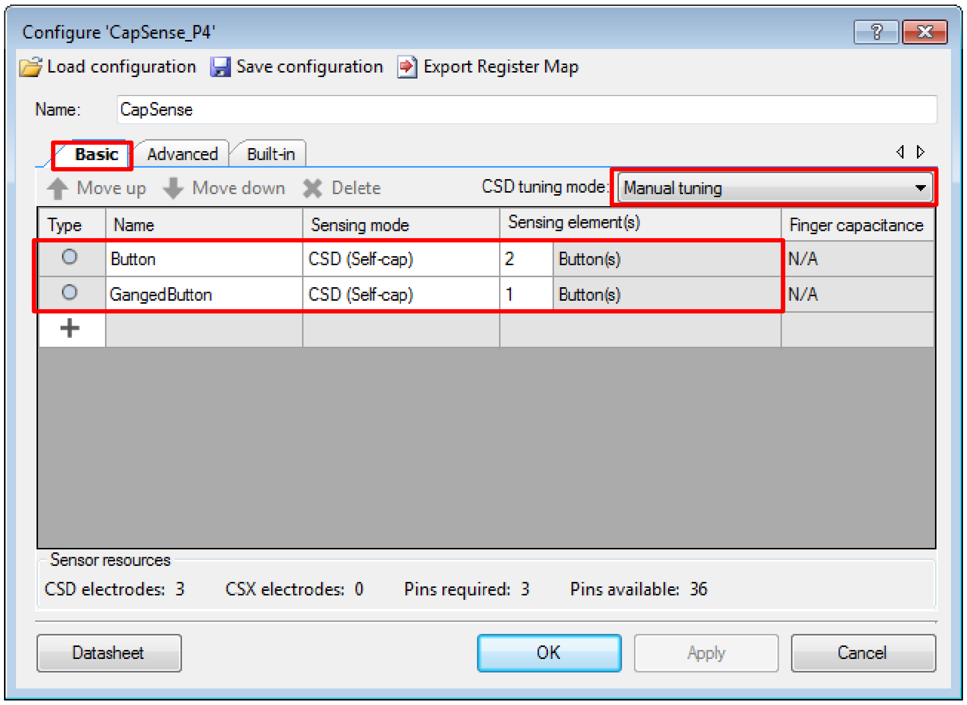
Component	Instance Name	Version	Hardware Resources
CapSense	CapSense	v3.10	CSD, 3 GPIO pins
EZI2C Slave (SCB mode)	EZI2C	v3.20	SCB, 2 GPIO pins
Clock	Clock_PMW_2	v2.20	1 Clock Divider
PWM (TCPWM mode)	PrISM_Red, PrISM_Green, PrISM_Blue	v2.10	1 TCPWM each
Digital Output Pin	Pin_BlueLED, Pin_GreenLED, Pin_RedLED	v2.20	1 GPIO pin each

Parameter Settings

CapSense

Figure 5, Figure 6, and Figure 7 show the CapSense Component settings that are changed from the default values. See the CapSense Component datasheet for additional information.

Figure 5. CapSense Component – Basic Tab Configuration



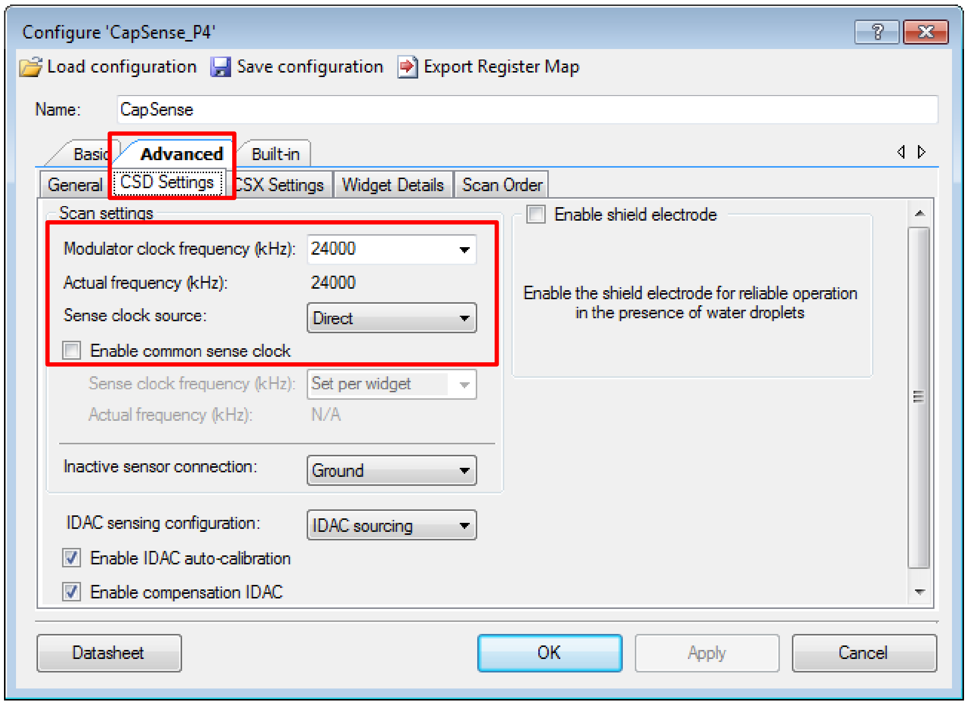
Name: CapSense

CSD tuning mode: Manual tuning

Type	Name	Sensing mode	Sensing element(s)		Finger capacitance
○	Button	CSD (Self-cap)	2	Button(s)	N/A
○	GangedButton	CSD (Self-cap)	1	Button(s)	N/A

Sensor resources
CSD electrodes: 3 CSX electrodes: 0 Pins required: 3 Pins available: 36

Figure 6. CapSense Component – Advanced Tab CSD Settings



Name: CapSense

Scan settings

Modulator clock frequency (kHz): 24000

Actual frequency (kHz): 24000

Sense clock source: Direct

☐ Enable common sense clock

Sense clock frequency (kHz): Set per widget

Actual frequency (kHz): N/A

Inactive sensor connection: Ground

IDAC sensing configuration: IDAC sourcing

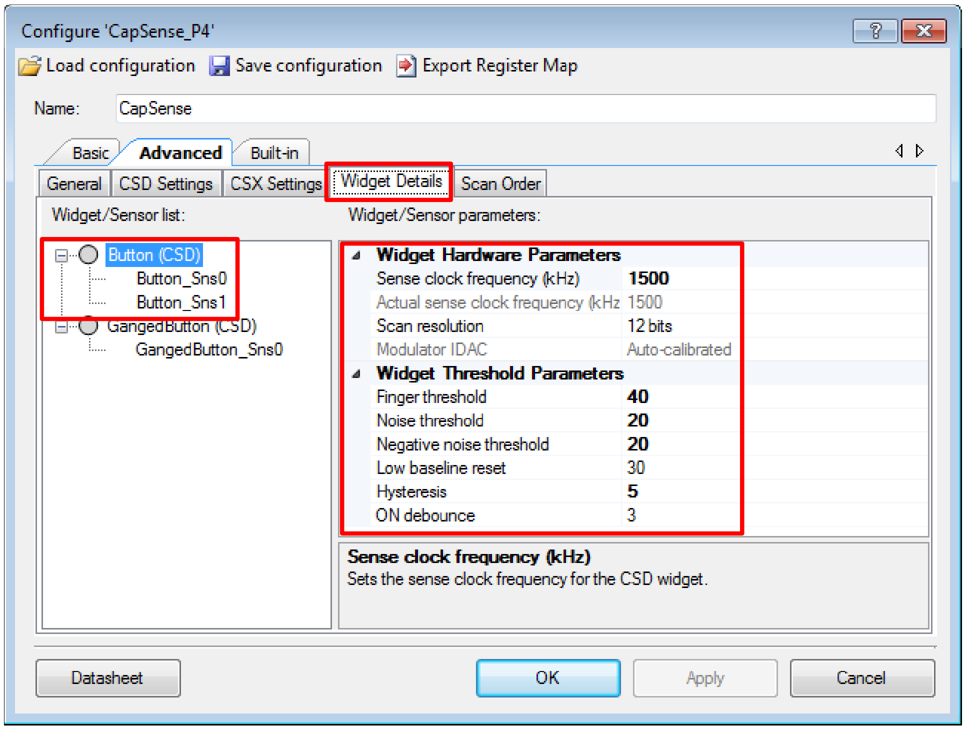
☑ Enable IDAC auto-calibration

☑ Enable compensation IDAC

☐ Enable shield electrode

Enable the shield electrode for reliable operation in the presence of water droplets

Figure 7. CapSense Component – Advanced Tab Widget Details



Name: CapSense

Widget/Sensor list:

- Button (CSD)
  - Button_Sns0
  - Button_Sns1
- GangedButton (CSD)
  - GangedButton_Sns0

Widget/Sensor parameters:

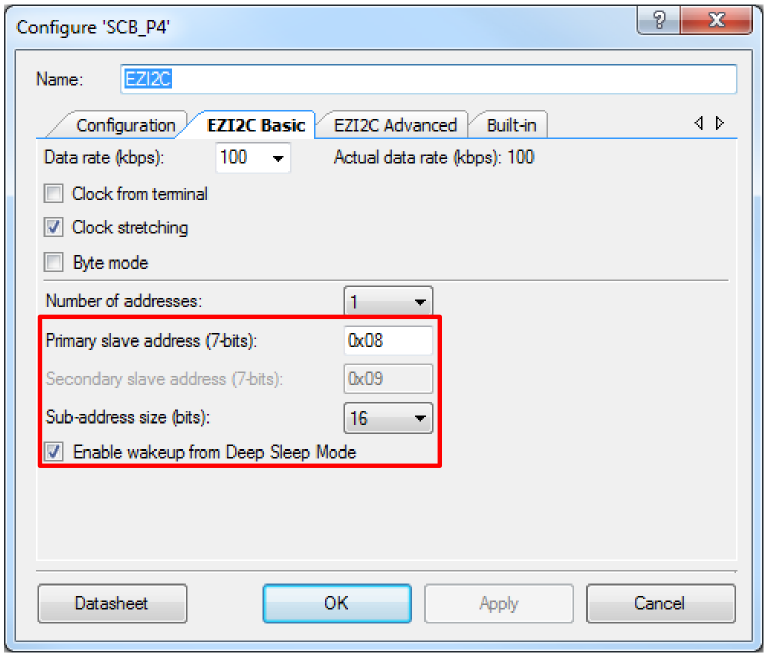
Widget Hardware Parameters	
Sense clock frequency (kHz)	1500
Actual sense clock frequency (kHz	1500
Scan resolution	12 bits
Modulator IDAC	Auto-calibrated
Widget Threshold Parameters	
Finger threshold	40
Noise threshold	20
Negative noise threshold	20
Low baseline reset	30
Hysteresis	5
ON debounce	3

Sense clock frequency (kHz)
Sets the sense clock frequency for the CSD widget.

EZI2C Slave

Figure 8 shows the non-default EZI2C Slave Component settings. See the SCB Component datasheet for additional information.

Figure 8. EZI2C Slave Component Basic Settings



Configure 'SCB_P4'

Name: EZI2C

Configuration | **EZI2C Basic** | EZI2C Advanced | Built-in

Data rate (kbps): 100 — Actual data rate (kbps): 100

☐ Clock from terminal

☑ Clock stretching

☐ Byte mode

Number of addresses: 1

Primary slave address (7-bits): 0x08

Secondary slave address (7-bits): 0x09

Sub-address size (bits): 16

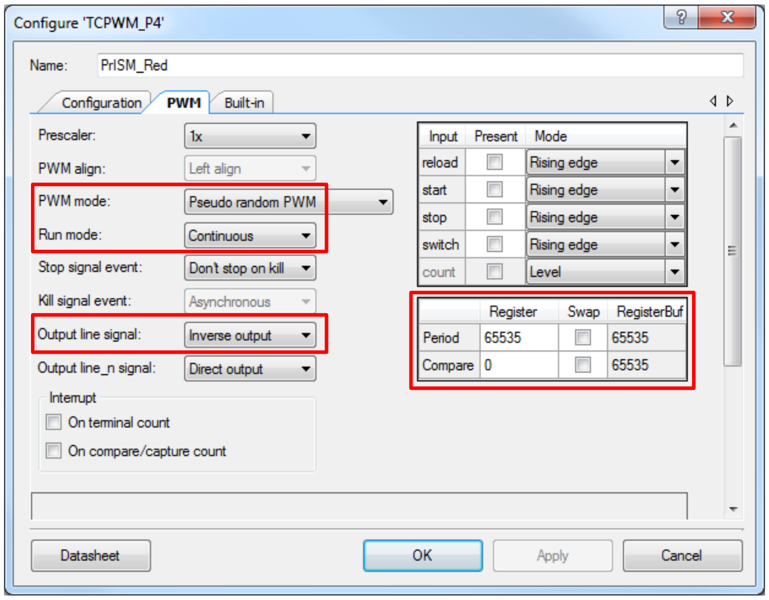
☑ Enable wakeup from Deep Sleep Mode

Datasheet | OK | Apply | Cancel

PWM

Figure 9 shows the non-default PWM Component settings. See the TCPWM Component datasheet for additional information.

Figure 9. PWM Component Configuration



Configure 'TCPWM_P4'

Name: PrISM_Red

Configuration | **PWM** | Built-in

Prescaler: 1x

PWM align: Left align

PWM mode: Pseudo random PWM

Run mode: Continuous

Stop signal event: Don't stop on kill

Kill signal event: Asynchronous

Output line signal: Inverse output

Output line_n signal: Direct output

Interrupt

☐ On terminal count

☐ On compare/capture count

Input	Present	Mode
reload	☐	Rising edge
start	☐	Rising edge
stop	☐	Rising edge
switch	☐	Rising edge
count	☐	Level

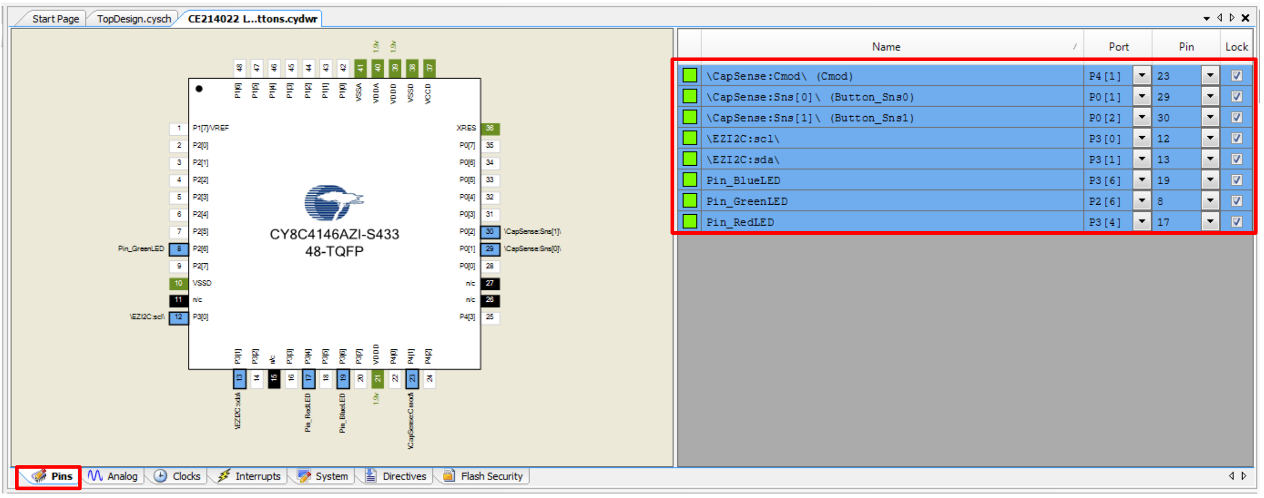
	Register	Swap	RegisterBuf
Period	65535	☐	65535
Compare	0	☐	65535

Datasheet | OK | Apply | Cancel

Design-Wide Resources

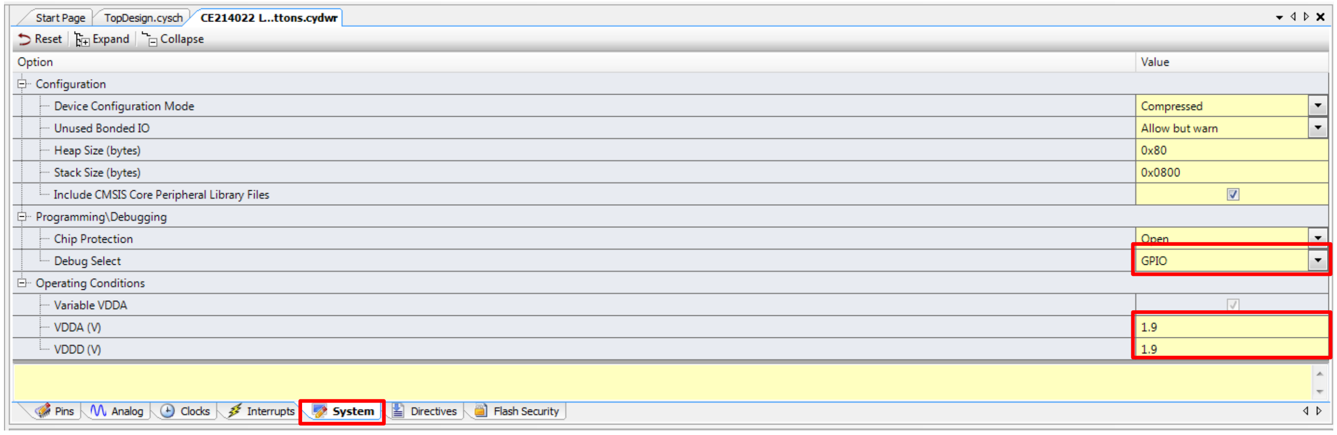
Figure 10 and Figure 11 show the non-default .cydwr settings for the project.

Figure 10. .cydwr Pins Tab Settings



	Name	Port	Pin	Lock
🟩	\CapSense:Cmod\ (Cmod)	P4[1]	23	✓
🟩	\CapSense:Sns[0]\ (Button_Sns0)	P0[1]	29	✓
🟩	\CapSense:Sns[1]\ (Button_Sns1)	P0[2]	30	✓
🟩	\EZI2C:scl\	P3[0]	12	✓
🟩	\EZI2C:sda\	P3[1]	13	✓
🟩	Pin_BlueLED	P3[6]	19	✓
🟩	Pin_GreenLED	P2[6]	8	✓
🟩	Pin_RedLED	P3[4]	17	✓

Figure 11. .cydwr System Tab Settings



Option	Value
Configuration	
Device Configuration Mode	Compressed
Unused Bonded IO	Allow but warn
Heap Size (bytes)	0x80
Stack Size (bytes)	0x0800
Include CMSIS Core Peripheral Library Files	✓
Programming\Debugging	
Chip Protection	Open
Debug Select	GPIO
Operating Conditions	
Variable VDDA	✓
VDDA (V)	1.9
VDDD (V)	1.9

Note: For the PSoC 4100S device, the CapSense V_{REF} voltage is set based on the VDDA setting in the .cydwr tab per the following table.

Table 2. CapSense V_{REF} Values Based on VDDA Setting

VDDA (V)	VREF (V)
< 2.7	1.2
2.7 to 4.8	2.1
≥ 4.8	4.2

If VDDA is set to 1.9 V in the .cydwr tab, V_{REF} is set to 1.2 V. This V_{REF} voltage ensures that the CapSense tuning parameters do not vary with respect to VDDA, thereby avoiding retuning of the sensors.

Operation

Follow these steps to test the project:

1. Select the *CE214022 LP CapSense Buttons.cywrk* file on the PSoC Creator Start Page at **Examples and Kits** > **Kits** > **CY8CKIT-041-41XX**. Select a location to save the code example.
2. Build the project (**Build** > **Build CE214022 LP CapSense Buttons**).
3. Connect the PSoC 4100S Pioneer Kit to your computer using the USB cable provided.
4. Program the PSoC 4100S device (**Debug** > **Program**). See the kit guide for details on programming the kit.
5. Touch the left button and observe that the Red LED is turned ON. Upon repeated touch, the RGB LED turns ON in the sequence Red → Green → Blue → Red.
6. Touch the right button and observe that the RGB LED color brightness changes. Upon repeated touches, the brightness switches among three levels: Min → Mid → Max.
7. Connect an ammeter between the P4.VDD and VDD test points on the main board to measure the PSoC 4100S device current consumption. See the “Current Measurement Switch” section in the kit guide for complete details on power measurement steps.
8. Release your finger from the buttons and wait for three seconds. Notice that the average current is about 5 µA per button.

   Note: At 5 V, the average current consumption is much higher than 5 uA. This is because the VDDA value in the *.cydwr* settings is set to 1.9 V instead of the actual operating voltage. See the “Low Voltage Analog Boost Clocks” section in the PSoC 4 System Reference Guide for more information.
9. Touch the buttons and notice that the PSoC 4100S current consumption increases to 3 mA because the LEDs are turned ON.

Upgrade Information

The code example is updated to the latest version of PSoC Creator and hence does not require an upgrade.

Related Documents

Table 3 lists the relevant application notes, code examples, PSoC Creator Component datasheets, device documentation, and development kit (DVK) documentation.

Table 3. Related Documents

Application Notes		
AN79953	Getting Started with PSoC 4	Describes PSoC 4 and how to build your first PSoC Creator project
AN85951	PSoC 4 and PSoC Analog Coprocessor CapSense Design Guide	Describes PSoC 4 and PSoC Analog Coprocessor CapSense Component tuning
Code Examples		
CE210291	PSoC 4 CapSense One Button	
CE210290	PSoC 4 CapSense Low-Power Ganged Sensor	
PSoC Creator Component Datasheets		
CapSense	Supports capacitive touch sensing	
PWM	Supports 16-bit fixed-function pseudo-random PWM implementation	
EZI2C Slave	Supports I^{2}C slave operation	
Pins	Supports connection of hardware resources to physical pins	
Clock	Supports local clock generation	
Device Documentation		
PSoC 4100S Family Datasheet		
PSoC 4100S Family PSoC 4 Architecture Technical Reference Manual		
Development Kit (DVK) Documentation		
CY8CKIT-041-41XX PSoC 4100S Pioneer Kit		

PSoC Resources

Cypress provides a wealth of data at www.cypress.com to help you to select the right PSoC device for your design 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:

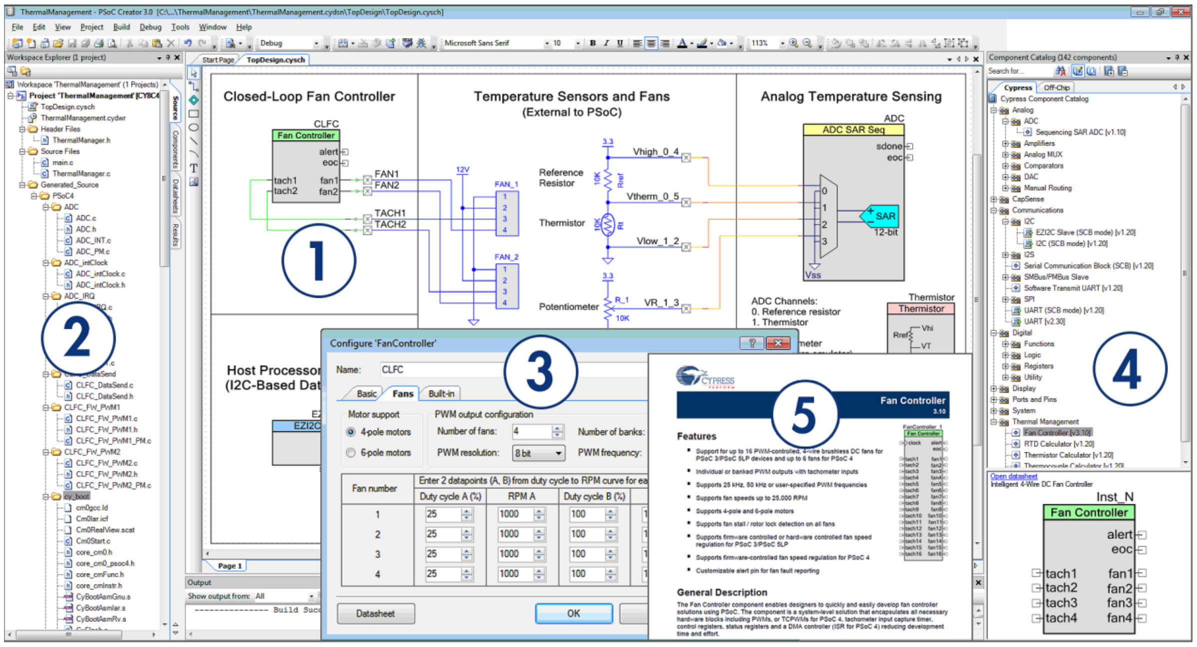
- **Overview:** PSoC Portfolio, PSoC Roadmap
- **Product Selectors:** PSoC 1, PSoC 3, PSoC 4, or PSoC 5LP. In addition, PSoC Creator includes a Device Selector tool.
- **Datasheets** describe and provide electrical specifications for the PSoC 3, PSoC 4, and PSoC 5LP device families.
- **CapSense Design Guides:** Learn how to design capacitive touch-sensing applications with the PSoC 3, PSoC 4, and PSoC 5LP families 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.
- **Technical Reference Manuals (TRM)** provide detailed descriptions of the architecture and registers in the PSoC 3, PSoC 4, and PSoC 5LP device families.
- **PSoC Training Videos**: These videos provide step-by-step instructions on getting started building complex designs with PSoC.
- **Development Kits**:
  - CY8CKIT-041-41XX PSoC 4100S Pioneer Kit is an easy-to-use and inexpensive development platform. This kit includes connectors for Arduino™ compatible shields and Digilent® Pmod™ daughter cards.
  - CY8CKIT-145 is a very low-cost prototyping platform for evaluating PSoC 4 S-Series devices.
- The MiniProg3 device provides an interface for flash programming and debugging.

PSoC Creator

PSoC Creator is a free, Windows-based IDE. It enables concurrent hardware and firmware design of systems based on PSoC 3, PSoC 4, and PSoC 5LP. See Figure 12. With PSoC Creator, you can:

1. Drag and drop Components to build your hardware system design in the main design workspace
2. Co-design 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 12. PSoC Creator Features



1
2
3
4
5
Closed-Loop Fan Controller
Temperature Sensors and Fans (External to PSoC)
Analog Temperature Sensing
Host Processor (I2C-Based Data)
Configure 'FanController'
Fan Controller
Features
General Description
Component Catalog (142 components)
Workspace Explorer (1 project)

Document History

Document Title: CE214022 – LP CapSense Buttons

Document Number: 002-14022

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5401352	SRDS	11/18/2016	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
Lighting & Power Control	cypress.com/powerpsoc
Memory	cypress.com/memory
PSoC	cypress.com/psoc
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless/RF	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

Cypress Developer Community

Forums | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

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

Phone	: 408-943-2600
Fax	: 408-943-4730
Website	: www.cypress.com