# CapSense® Controller Code Examples

Doc. No. 001-74590 Rev. *F

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone (USA): 800.858.1810
Phone (Intnl): +1.408.943.2600
http://www.cypress.com

**Copyrights**

# Contents

# Introduction

Cypress CapSense® solutions bring elegant, reliable, and easy-to-use capacitive touch sensing functionality to your design. Our capacitive touch sensing solutions have replaced more than 4 billion mechanical buttons. CapSense also changed the face of industrial design in products such as cell phones, PCs, consumer electronics, and white goods. Cypress's robust CapSense solutions leverage our flexible Programmable System-on-Chip (PSoC®) architecture to accelerate time-to-market, integrate critical system functions, and reduce bill of materials costs.

## Overview

This document provides a detailed overview of 10 CapSense code examples with an intention to help customers experience and evaluate features of Cypress CapSense Controllers, CapSense Kits, and CapSense User Modules. This document also explains how to read back and plot sensor data through $I^2C$ and UART interfaces. You should be familiar with CapSense sensing technology before you read this document. If you would like more information about general CapSense theory and operation, see Getting Started with CapSense.

**Note** The code examples in this application note are particular to the kits referenced in this document. Refer to Getting Started with CapSense for the key design considerations and layout best practices to ensure successful porting of these code examples on custom boards. If this document does not address your needs or answer all of your questions or if you wish to provide feedback please contact capsense@cypress.com.

To facilitate your evaluation, we present the code examples in order of increasing complexity:

- **Introduction**: For beginners, we recommend that you start with a demonstration of how CapSense touch sensors function. *Code Example 1: Buttons and Sliders with SmartSense with a CY8C20xx6A CapSense Controller* showcases how easy and quick it is to implement touch buttons and sliders using the SmartSense Auto-Tuning capacitive sensing algorithm.
- **Communicate**: After you understand how CapSense functions, the next important step is to learn how to communicate with the CapSense controller to send and receive data. This communication is necessary to tune CapSense sensors as well as to interface the CapSense controller with the host processor. *Code Example 2: Host Communication Through I2C with a CY8C20xx6A CapSense Controller* and *Code Example 3: Data Transmission Through UART with a CY8C20xx6A CapSense Controller* describe in detail how to communicate with the CapSense controller.
- **Tune**: The next step is to tune CapSense sensors for robust operation. To complete the tuning procedure, you must first know the parasitic capacitance of all the capacitive sensors on the PCB. *Code Example 4: Measuring Absolute Sensor Capacitance with a CY8C20xx6A CapSense Controller*, *Code Example 5: Measuring Absolute Sensor Capacitance with a CY8C21x34/B CapSense Controller*, and *Code Example 6: Measuring Absolute Sensor Capacitance with a CY8C20x34 CapSense Controller* demonstrate how to measure the parasitic capacitance of each capacitive sensor for different CapSense devices. *Code Example 7: Tuning CSD with a CY8C20xx6A CapSense Controller*, *Code Example 8: Tuning CSD with Feedback Resistor with a CY8C21x34/B CapSense Controller*, and *Code Example 9: Tuning a CSA_EMC with a CY8C20x34 CapSense Controller* describe the steps to tune CapSense controllers.
- **Optimize for Power Consumption**: Finally, *Code Example 10: Power Consumption Optimization for One CapSense Button Using SmartSense on CY8C20xx6A* explains how to optimize the power consumed by the CapSense controller to ensure longer battery life in the end application.

**Note** Only four CapSense devices are under the scope of this design guide: CY8C20xx6A, CY8C20x34, CY8C21x34/B, and CY8C24x94. Refer to the Appendix of this document for instructions on how to port the code examples to other CapSense devices. The Appendix also provides instructions on Configuring the CSD User Module, which is required to create or modify a code example.

Table 1. List of Code Examples

| Code Example | Code Example | Target Kit | Description |
|---|---|---|---|
| 1 | Buttons and Sliders with SmartSense with a CY8C20xx6A CapSense Controller | CY3280-20x66 | Demonstrates how to implement CapSense buttons and sliders with SmartSense auto-tuning user module. |
| 2 | Host Communication Through I2C with a CY8C20xx6A CapSense Controller | CY3280-20x66 | Demonstrates how to transfer CapSense data to and from a CapSense controller through $I^2C$. It also explains how to plot read back data using Bridge Control Panel tool. |
| 3 | Data Transmission Through UART with a CY8C20xx6A CapSense Controller | CY3280-20x66 | Demonstrates how to transmit CapSense data in RS232 data packet format. It also explains how to plot read back data using MultiChart tool. |
| 4 | Measuring Absolute Sensor Capacitance with a CY8C20xx6A CapSense Controller | CY3280-20x66 | Measures the absolute capacitance of sensors with parasitic capacitance ($C_P$) up to 60 pF to within 1 pF accuracy and then display the results on a PC using Windows HyperTerminal or Bridge Control Panel. |
| 5 | Measuring Absolute Sensor Capacitance with a CY8C21x34/B CapSense Controller | CY3280-21x34 | Measures the absolute capacitance of sensors with parasitic capacitance ($C_P$) up to 60 pF to within 1 pF accuracy and then display the results on a PC using Windows HyperTerminal or Bridge Control Panel. |
| 6 | Measuring Absolute Sensor Capacitance with a CY8C20x34 CapSense Controller | CY3280-20x34 | Measures the absolute capacitance of sensors with parasitic capacitance ($C_P$) up to 60 pF to within 1 pF accuracy. |
| 7 | Tuning CSD with a CY8C20xx6A CapSense Controller | CY3280-20x66 | Tunes the CSD user module in the CY8C20xx6A. |
| 8 | Tuning CSD with Feedback Resistor with a CY8C21x34/B CapSense Controller | CY3280-21x34 | Tunes the CSD user module in the CY8C21x34/B. |
| 9 | Tuning a CSA_EMC with a CY8C20x34 CapSense Controller | CY3280-20x34 | Tunes the CSA_EMC user module in the CY8C20x34. |
| 10 | Power Consumption Optimization for One CapSense Button Using SmartSense on CY8C20xx6A | CY3280-20x66 | Provides a method to achieve power consumption less than 50 µA for one sensor using SmartSense user module. |

Table 2. Code Examples by Product Family

| | Product Family | | | |
|---|---|---|---|---|
| **Intended Functionality** | **CY8C20xx6A** | **CY8C20x34** | **CY8C21x34/B** | **CY8C24x94** |
| Buttons and sliders with SmartSense | Code Example 1 | N/A[1] | After Port[2] | N/A[1] |
| Host communication with $I^2C$ | Code Example 2 | After Port[2] | After Port[2] | After Port[2] |
| Data Transmission with UART | Code Example 3 | After Port[2] | After Port[2] | After Port[2] |
| Absolute Sensor Capacitance Measurement | Code Example 4 | Code Example 5 | Code Example 6 | After Port[2] |
| Tuning CSD | Code Example 7 | N/A[3] | No | N/A[3] |
| Tuning CSD with feedback resistor | N/A3 | N/A[3] | Code Example 8 | N/A[3] |
| Tuning CSA_EMC | N/A3 | Code Example 9 | N/A[3] | N/A[3] |
| Power Consumption optimization with SmartSense | Code Example 10 | No | No | No |

[1] These devices do not support SmartSense UM.
[2] Code example works after porting the project provided with this guide to the respective devices as explained in the Appendix section *Migrating Code Examples to Other CapSense Devices*

[3] The tuning method is unique to the device and UM.

# Code Example 1 Buttons and Sliders with SmartSense with a CY8C20xx6A CapSense Controller

## 1.1 Project Name

CE_1_ButtonsandSliders_withSmartSense_withCY8C20xx6A

## 1.2 Overview

The proper operation of capacitive sensors requires you to manually tune several parameters depending on the parasitic capacitance ($C_P$). The SmartSense™ User Module improves and expands the basic architecture of the CSD UM. A key advance is SmartSense Auto-Tuning. SmartSense UM optimizes the parasitic capacitance operational parameters during runtime based on the $C_P$ of each sensor; you do not need to manually tune when you use SmartSense. This code example demonstrates how to use the SmartSense User Module. We use the EzI2Cs User Module to send CapSense® parameters, such as raw counts, difference counts, baselines of each button, and centroid position of slider, to the $I^2C$ master.

## 1.3 Hardware Setup

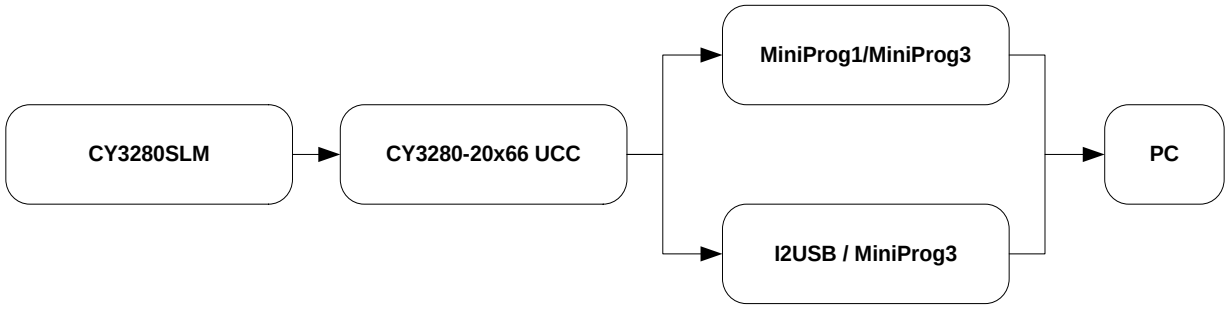
### 1.3.1 Requirements

- CY3280-20x66 Universal CapSense Controller board
- CY3280-SLM Universal CapSense Linear Slider Module
- CY3240-I2USB Bridge or CY8CKIT-002 Miniprog3
- CY3217-MiniProg1 Programmer Kit or CY8CKIT-002 Miniprog3
- USB A to Mini B Cable
- PC running Windows XP or later

### 1.3.2 Assembly

Figure 1-1 represents the hardware setup. The CY3280-20x66 UCC kit connects to the CY3280-SLM module through a 22x2_RA_Receptacle. Use MiniProg1/MiniProg3 or the I2USB bridge/MiniProg3 to connect to the ISSP header of the kit. The setup uses the MiniProg1/MiniProg3 for programming and the I2USB bridge/Miniprog3 to send data to a PC. They connect to the PC through a USB cable.

Figure 1-1. Hardware Setup Block Diagram

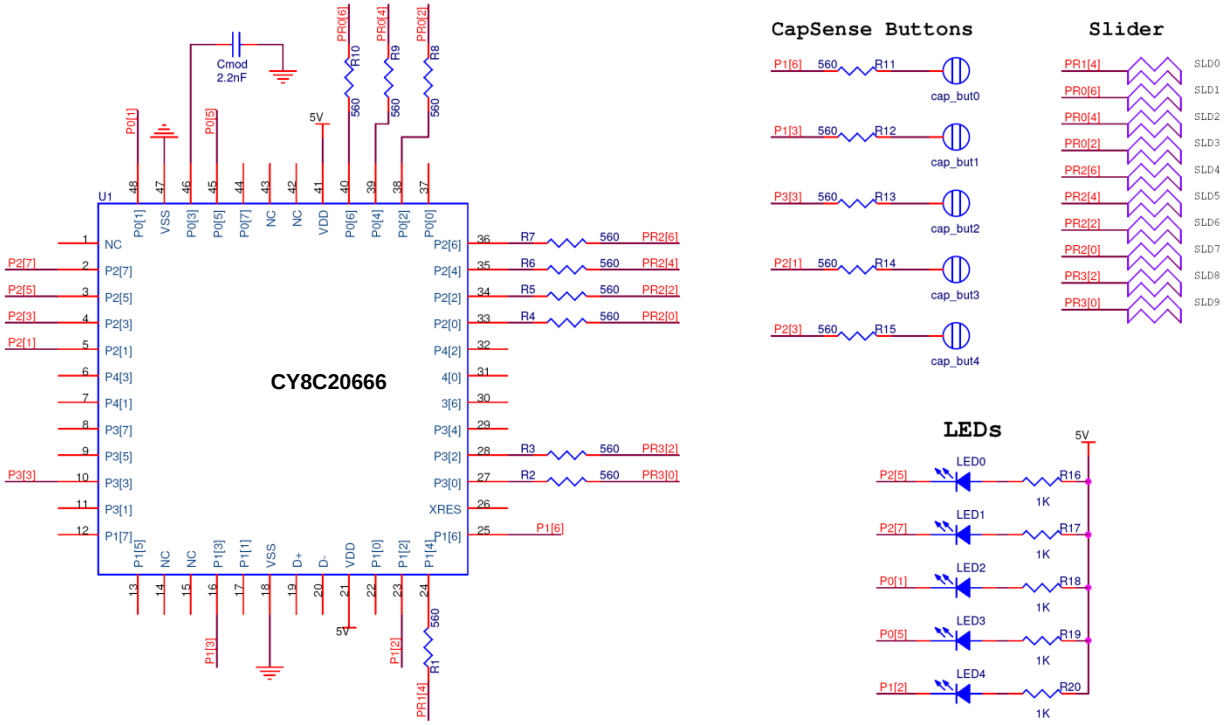



### 1.3.3 Setting Up the Board

Make following hardware connections:

- Connect header J1 of the CY3280-SLM daughter card to the 22x2_RA_Receptacle (connector P2) on the CY3280-20x66 UCC board.
- Place a jumper on header J7 to short the pins $V_{CC}$ and $V_{CC\_PROG}$ of the UCC board. This setting allows you to power the CapSense controller from the ISSP connector.
- Place a jumper on header J4 to short the pins XRES and XRES/INT (pins 1 and 2) of the UCC board. This setting routes the XRES pin of the CapSense controller to pin 3 of ISSP connector J3.
- Place a jumper on header J2 to short the pins GND and SHIELD (pins 2 and 3) of the CY3280-SLM board. This setting allows the board hatch pattern on the CY3280-SLM board to connect to ground.
- Connect the MiniProg1/MiniProg3 to the ISSP header on J3 of the UCC board. This connection is required only when the UCC is to be programmed with the code generated hex file. This connection has to be replaced with the I2USB bridge/MiniProg3 when the CapSense data is to be read in the Bridge Control Panel software.
- Use the USB A to Mini B cable to connect the other end of the MiniProg1 (or the I2USB bridge or the MiniProg3) to the PC.

## 1.4 Schematic




The modulator capacitor ($C_{MOD}$) is a 2.2 nF capacitor connected to P0[3]. A 560 Ω resistor is connected in series with each CapSense button to reduce RF interference. LEDs are connected in active low configuration with 1 kΩ series resistors. In total there are five LEDs, five buttons, and one slider with 10 segments available.

Table 1-1. Pin Assignments for LEDs, Buttons, and Slider Segments

| LED | Button | Slider Segment |
|---|---|---|
| LED0 - P2[5] | BTN0 - P1[6] | SLD0 - P1[4], SLD1 - P0[6] |
| LED1 - P2[7] | BTN1 - P1[3] | SLD2 - P0[4], SLD3 - P0[2] |
| LED2 - P0[1] | BTN2 - P3[3] | SLD4 - P2[6], SLD5 - P2[4] |
| LED3 - P0[5] | BTN3 - P2[1] | SLD6 - P2[2], SLD7 - P2[0] |
| LED4 - P1[2] | BTN4 - P2[3] | SLD8 - P3[2], SLD9 - P3[0] |

## 1.5 Software Setup

### 1.5.1 Tools Required

- PSoC Designer™ (version 5.2 or higher)
- PSoC Programmer (version 3.13 or higher)
- Bridge Control Panel

### 1.5.2 User Module List

The following table lists the user modules (UM) used in this code example and the hardware resources occupied by each UM.

| User Module | Hardware Resources |
|---|---|
| SmartSense | CapSense and Comparator, Timer1 |
| EzI2Cs | $I^2C$/SPI Block |

### 1.5.3 User Module Parameters, Global Resources

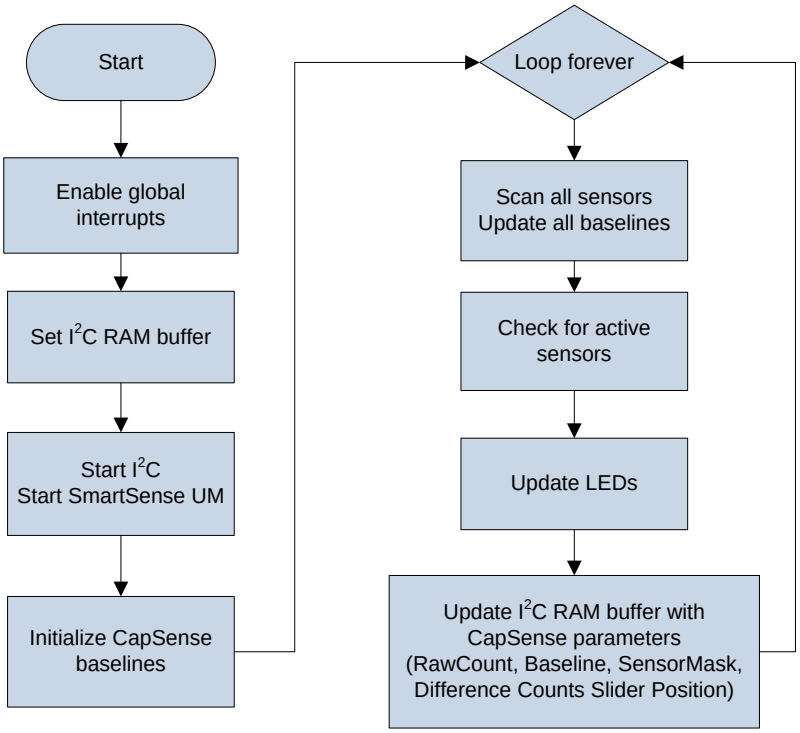
The `ReadMe.txt` file in the project describes the parameter settings for the user modules used in this code example. It also includes a list of the Global Resources.

## 1.6 Operation

On reset, the firmware performs the following operations:

- Defines a structure MyI2C_Regs to store the button number, raw count, difference count, baseline, the centroid position, and the status of the CapSense button corresponding to the given button number.
- Enables global interrupt, and then starts the SmartSense user module.
- Starts the EzI2Cs UM, and then sets the structure MyI2C_Regs as the $I^2C$ RAM buffer.
- Performs the following operations in an infinite loop:
  - Scans all sensors continuously and updates the structure MyI2C_Regs with the raw count, difference count, baseline, slider centroid position, and the status of the requested CapSense button. The $I^2C$ master can request CapSense data of a particular button by writing the button number into the first byte of the $I^2C$ buffer of the EzI2Cs slave.
  - When the firmware detects a button press, it switches the corresponding LED to ON. It switches the LED to OFF when the button is released.
  - When the slider is touched, the firmware turns the LED to ON indicating the touch position.

Figure 1-2. Functional Flow for Code Example 1




## 1.7 Running the Code Example

Program the board with the project, and then use this procedure to run the code example. For details on how to program the UCC board, refer to the Chapter 5 of CY3280-20x66 kit guide.

1. Use MiniProg1/MiniProg3 or any of the sources mentioned in the CY3280-20x66 UCC kit guide to power the board at 5 V.
2. Touch the linear slider on the CY3280-SLM module board. The corresponding LEDs on the CY3280-SLM board light up.
3. Touch a button. The corresponding LED on the CY3280-SLM module board lights up. Multiple buttons can be touched at the same time. The linear slider and buttons can also be used simultaneously.

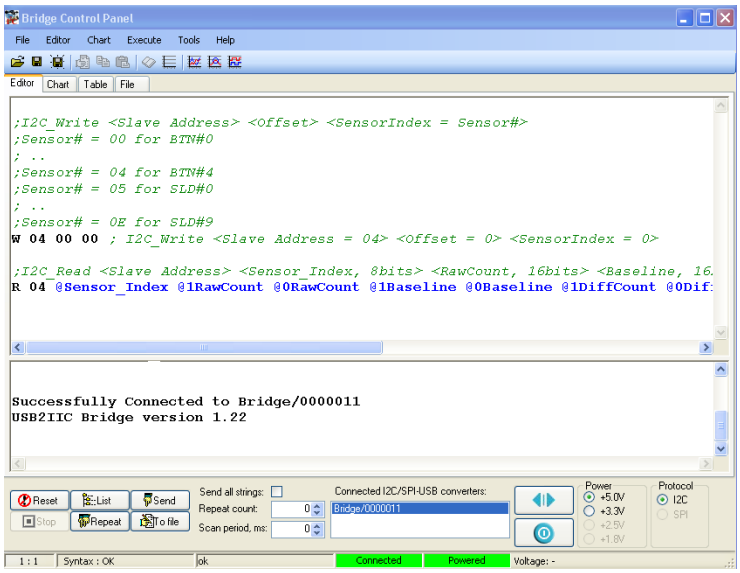
## 1.8 Reading CapSense Data over I²C

Use this procedure to run the code example and to read back the CapSense data on to Bridge Control Panel tool. For additional details on the Bridge Control Panel, see AN2397, “CapSense® Data Viewing Tools”.

### 1.8.1 Loading the Bridge Control Panel

1. Use I2USB bridge/ MiniProg3 and a USB A to Mini B cable to connect your computer to the ISSP connector J3 of the CY3280-20x66 Universal CapSense Controller board.
2. On the computer desktop, select **Start** > **All Programs** > **Cypress** > **Bridge Control Panel** *(version)* > **Bridge Control Panel** *(version)*.
   The Bridge Control Panel is a component installed during the PSoC Programmer installation.
3. Select the device from the port selection window.
4. Power the CY3280-20x66 CapSense Controller board at 5 V.
5. From the Bridge Control Panel, select **File** > **Open**. Load the `BCP.iic` file from *BCP Configuration Files* folder contained in the code example project folder.

6. Select **Charts** > **Variable Settings**. Load the `BCP.ini` file from *BCP Configuration Files* folder contained in the code example project folder.




7. Click **OK** to return to the main window.

## 1.8.2 Reading Raw Count, Baseline, and Difference Count of BTN0

1. Send the I$^2$C write instruction W 04 00 00 once.
2. Click the **Repeat** button to send the following I$^2$C read instruction continuously:

```
R 04 @Sensor_Index @1RawCount @0RawCount @1Baseline @0Baseline @1Diffcount
@0DiffCount @ButtonStatus @SliderPosition
```

3. Click the **Chart** tab to view raw count, baseline, and difference count of BTN0.

## 1.8.3 Reading raw count, baseline, difference count, and slider position of SLD0

1. Send the I$^2$C write instruction W 04 00 05 once.
2. Click the **Repeat** button to send the following I$^2$C read instruction continuously:

```
R 04 @Sensor_Index @1RawCount @0RawCount @1Baseline @0Baseline @1Diffcount
@0DiffCount @ButtonStatus @SliderPosition
```

3. Click the **Chart** tab to view raw count, baseline, difference count, and slider position of SLD0.

# Code Example 2 Host Communication Through I²C with a CY8C20xx6A CapSense Controller

## 2.1 Project Name

CE_2_HostCommunication_viaI2C_withCY8C20xx6A

## 2.2 Overview

This code example uses the I²C communication protocol to send the data (raw counts, difference counts, baseline, and status) of five CapSense® buttons and a ten-segment slider to the Bridge Control Panel tool where we can view it. The CSD algorithm implements the buttons on the CY8C20xx6A CapSense controller. By specifying the button number, you can obtain the data of that particular button.

## 2.3 Hardware Setup

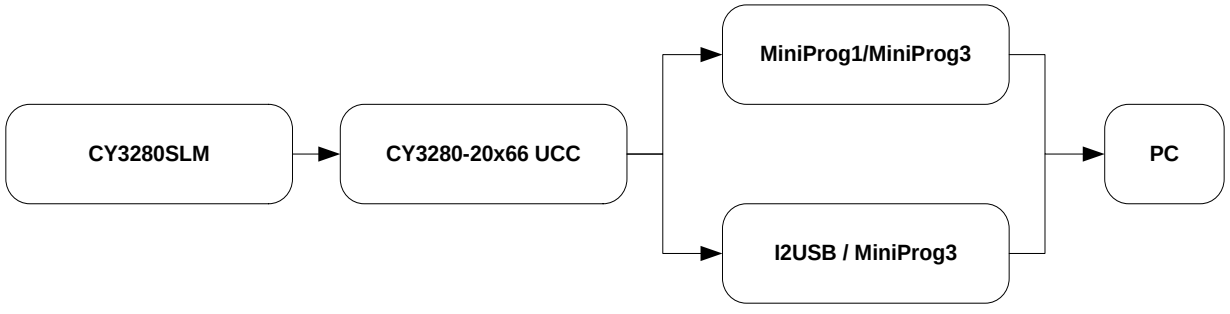
### 2.3.1 Requirements

- CY3280-20x66 Universal CapSense Controller board
- CY3280-SLM Universal CapSense Linear Slider Module
- CY3240-I2USB Bridge or CY8CKIT-002 Miniprog3
- CY3217-MiniProg1 Programmer Kit or CY8CKIT-002 Miniprog3
- USB A to Mini B Cable

### 2.3.2 Assembly

Figure 2-1 represents the hardware setup. The CY3280-20x66 UCC kit connects to the CY3280-SLM module through a 22x2_RA_Receptacle. Connect either the MiniProg1/MiniProg3 or the I2USB bridge/MiniProg3 to the ISSP header of the kit. The setup uses the MiniProg1/MiniProg3 for programming and the I2USB bridge/MiniProg3 to send data to PC. They connect to the PC through a USB cable.

Figure 2-1. Hardware Setup Block Diagram

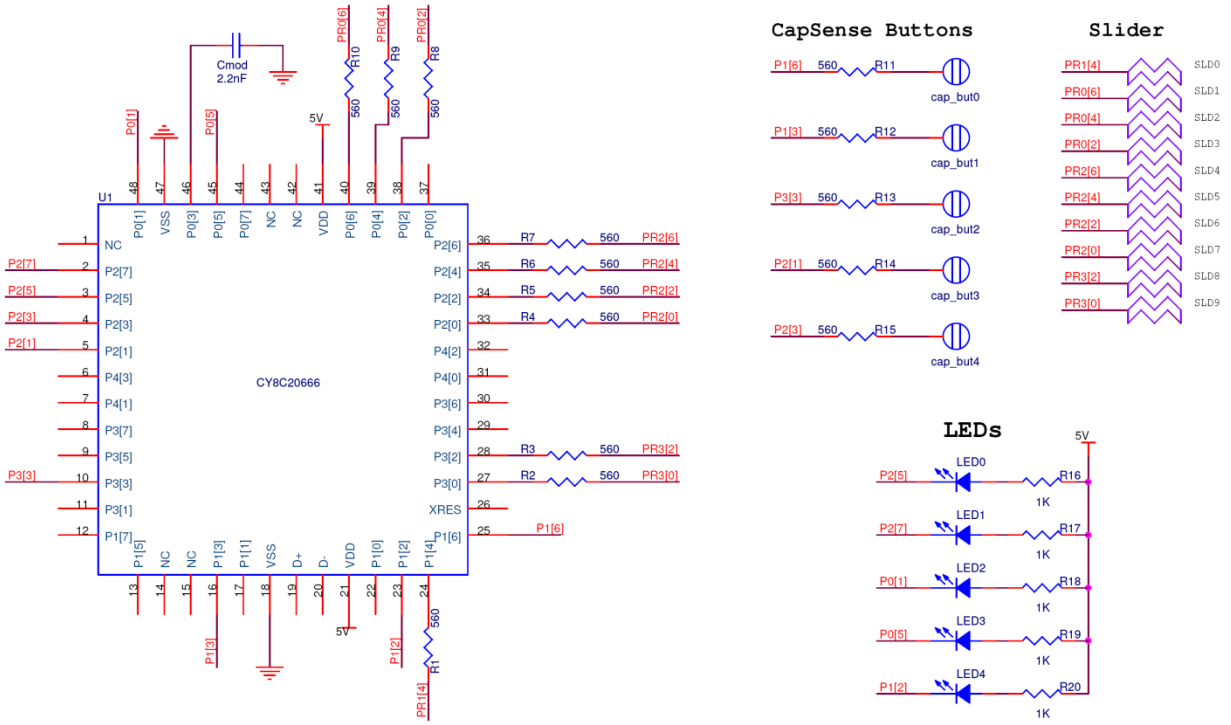


### 2.3.3 Setting Up the Board

Make the following hardware connections:

- Connect header J1 of the CY3280-SLM daughter card to the 22x2_RA_Receptacle (connector P2) on the CY3280-20x66 UCC board.
- Place a jumper on header J7 to short the pins $V_{CC}$ and $V_{CC\_PROG}$ of the UCC board. This setting allows you to power the CapSense controller from the ISSP connector.
- Place a jumper on header J4 to short the pins XRES and XRES/INT (pins 1 and 2) of the UCC board. This setting routes the XRES pin of the CapSense controller to pin 3 of ISSP connector J3.
- Place jumper on header J2 to short the pins GND and SHIELD (pins 2 and 3) of the CY3280-SLM board. This setting allows the board hatch pattern on the CY3280-SLM board to connect to ground.
- Connect the MiniProg1/MiniProg3 to the ISSP header on J3 of the UCC board. This connection is required only when the UCC is to be programmed with the code generated hex file. This connection has to be replaced with the I2USB bridge/MiniProg3 when the CapSense data is to be read in the Bridge Control Panel software
- Use the USB A to Mini B cable to connect the other end of the MiniProg1 (or the I2USB bridge or the MiniProg3) to the PC.

## 2.4 Schematic




The modulator capacitor ($C_{MOD}$) is a 2.2 nF capacitor connected to P0[3]. A 560-Ω resistor is connected in series with each CapSense button to reduce RF interference. LEDs are connected in active low configuration with 1-kΩ series resistors. In total there are five LEDs, five buttons, and one slider with 10 segments available.

Table 2-1. Pin Assignments for LEDs, Buttons, and Slider Segments

| LED | Button | Slider Segment |
|---|---|---|
| LED0 - P2[5] | BTN0 - P1[6] | SLD0 - P1[4], SLD1 - P0[6] |
| LED1 - P2[7] | BTN1 - P1[3] | SLD2 - P0[4], SLD3 - P0[2] |
| LED2 - P0[1] | BTN2 - P3[3] | SLD4 - P2[6], SLD5 - P2[4] |
| LED3 - P0[5] | BTN3 - P2[1] | SLD6 - P2[2], SLD7 - P2[0] |
| LED4 - P1[2] | BTN4 - P2[3] | SLD8 - P3[2], SLD9 - P3[0] |

## 2.5 Software Setup

### 2.5.1 Tools Required

- PSoC Designer (version 5.2 or higher)
- PSoC Programmer (version 3.13 or higher)
- Bridge Control Panel

### 2.5.2 User Modules List

The following table lists the user modules (UM) used in this code example and the hardware resources occupied by each UM.

| User Module | Hardware Resources |
|---|---|
| CSD | CapSense Block, Timer1 (default) |
| EzI2C | $I^2C$/SPI Block |

### 2.5.3 User Module Parameters, Global Resources

The `ReadMe.txt` file in the project describes the parameter settings for the user modules used in this code example. It also includes a list of the Global Resources.

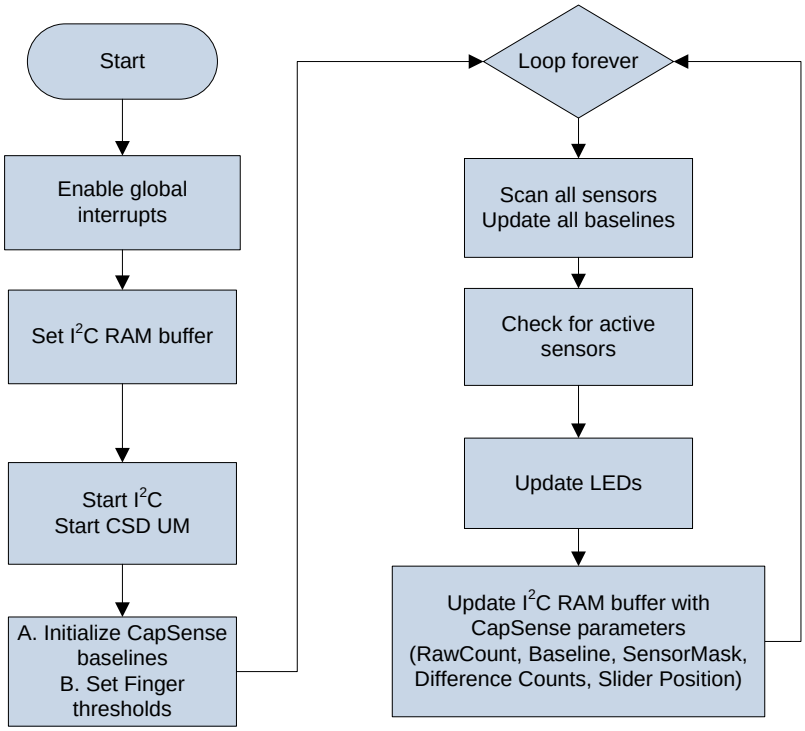
**Note** We tuned the code example for the 1.5 mm acrylic overlay that comes with the CY3280-SLM kit. To check the example with a thicker overlay, tune the project as described in *Code Example 7: Tuning CSD with a CY8C20xx6A CapSense Controller*.

## 2.6 Operation

On reset, the firmware performs the following operations:

- Defines a structure MyI2C_Regs to store the button number, raw count, difference count, baseline, the centroid position, and the status of the CapSense button corresponding to the given button number.
- Enables global interrupt, and then starts the CSD UM.
- Starts the EzI2Cs UM, and then sets the structure MyI2C_Regs as the $I^2C$ RAM buffer.
- Performs the following operations in an infinite loop:
  - Scans all sensors continuously and updates the structure MyI2C_Regs with the raw count, difference count, baseline, slider centroid position, and the status of the requested CapSense button. The $I^2C$ master can request CapSense data of a particular button by writing the button number into first byte of the $I^2C$ buffer of EzI2Cs slave.
  - When the firmware detects a button press, it switches the corresponding LED to ON. It switches the LED to OFF when the button is released.
  - When slider is touched, the firmware turns the LED to ON indicating the touch position.

Figure 2-2. Functional Flow for Code Example 2




## 2.7 Running the Code Example

Program the board with the project, and then use this procedure to run the code example. For details on how to program the UCC board, refer to chapter 5 of CY3280-20x66 kit guide.

1. Use MiniProg1/Miniprog3 or any of the sources mentioned in the CY3280-20x66 UCC kit guide to power the board at 5 V.
2. Touch the linear slider on the CY3280-SLM module board.
   The corresponding LEDs on the CY3280-SLM board light up.
3. Touch a button.
   The corresponding LED on the CY3280-SLM module board lights up. Multiple buttons can be touched at the same time. The linear slider and buttons can also be used simultaneously.

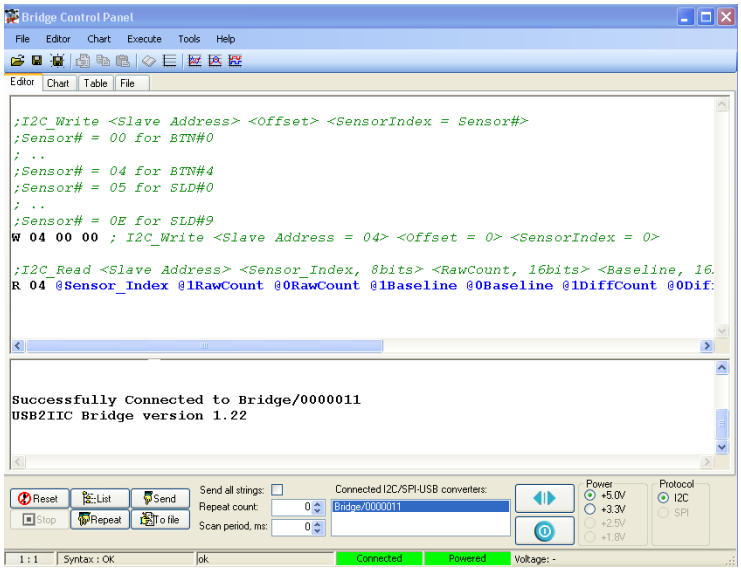
## 2.8 Reading CapSense Data Over I²C

Use this procedure to run the code example and to read back the CapSense data on to Bridge Control Panel tool. For additional details on the Bridge Control Panel tool, see AN2397, “CapSense® Data Viewing Tools”.

### 2.8.1 Loading the Bridge Control Panel

1. Use the I2USB bridge/MiniProg3 and a USB A to Mini B cable to connect your computer to the ISSP connector J3 of the CY3280-20x66 Universal CapSense Controller board.
2. On the computer desktop, select **Start** > **All Programs** > **Cypress** > **Bridge Control Panel** *(version)* > **Bridge Control Panel** *(version)*.
   The Bridge Control Panel is a component installed during the PSoC Programmer installation.
3. Select the device from the port selection window.
4. Power the CY3280-20x66 CapSense Controller board at 5 V.
5. From the Bridge Control Panel, select **File** > **Open.** Load the `BCP.iic` file from *BCP Configuration Files* folder contained in the code example project folder.

6. Select **Charts** > **Variable Settings**. Load the `BCP.ini` file from *BCP Configuration Files* folder contained in the code example project folder.




7. Click **OK** to return to the main window.

### 2.8.2 Reading raw count, baseline, and difference count of BTN0

1. Send the I$^2$C write instruction W 04 00 00 once.
2. Click the **Repeat** button to send the following I$^2$C read instruction continuously:

```
R 04 @Sensor_Index @1RawCount @0RawCount @1Baseline @0Baseline @1DiffCount
@0DiffCount @ButtonStatus @SliderPosition
```

3. Click the **Chart** tab to view raw count, baseline, and difference count of BTN0.

### 2.8.3 Reading raw count, baseline, difference count, and slider position of SLD0

1. Send the I$^2$C write instruction W 04 00 05 once.
2. Click the **Repeat** button to send the following I$^2$C read instruction continuously:

```
R 04 @Sensor_Index @1RawCount @0RawCount @1Baseline @0Baseline @1DiffCount
@0DiffCount @ButtonStatus @SliderPosition
```

3. Click the **Chart** tab to view raw count, baseline, difference count, and slider position of SLD0.

# Code Example 3 Data Transmission Through UART with a CY8C20xx6A CapSense Controller

## 3.1 Project Name

CE_3_DataTransmission_viaUART_withCY8C20xx6A

## 3.2 Overview

This code example uses the UART communication protocol to send the CapSense® data to a PC in a format that the MultiChart tool can read. We use the MultiChart tool to view the following CapSense parameters: raw counts, difference counts (signal), baseline, and the slider position in a graphical form. The SmartSense™ user module (UM) continuously scans all the buttons and slider segments on the CY3280-SLM, and the UART UM sends the CapSense parameters to a TX8 output pin. An external level translator converts the TX8 data to RS232 levels then sends it over the standard RS232 cable to the PC. On the PC, we use the MultiChart tool (available with AN2397) to monitor the data. The code example also turns the LEDs on the CY3280-SLM ON and OFF to indicate the button status and slider position.

## 3.3 Hardware Setup

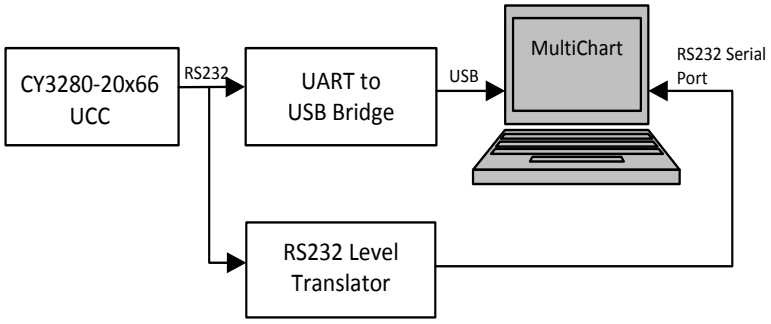
### 3.3.1 Requirements

- CY3280-20x66 Universal CapSense Controller board
- CY3280-SLM Universal CapSense Linear Slider Module
- CY3217-MiniProg1 Programmer Kit or CY8CKIT-002 Miniprog3
- USB A to Mini-B Cable
- An external RS232 level translator
- A standard 9-pin RS232 serial cable
- PC with serial (9-pin RS232) port

### 3.3.2 Assembly

Figure 3-1 represents the hardware setup. The CY3280-20x66 UCC kit connects to the CY3280-SLM module through a 22x2_RA_Receptacle. A MiniProg1/MiniProg3 connects to header J3 to program and power the kit. P3[5] on connector P3 connects to an external level translator which in turn connects to the PC through a UART cable.

Figure 3-1. Hardware Setup Block Diagram




**Note** The CY3280-20x66 UCC can send data to the PC via one of two methods: a UART to USB Bridge or an RS232 level translator (requires a PC with an RS232 serial port).

### 3.3.3 Setting Up the Board

Make the following hardware connections:

- Connect header J1 of the CY3280-SLM daughter card to the 22x2_RA_Receptacle (connector P2) on the CY3280-20x66 UCC board.
- Place a jumper on header J7 to short the pins $V_{CC}$ and $V_{CC\_PROG}$ of the UCC board. This setting allows you to power the CapSense controller from the ISSP connector.
- Place a jumper on header J4 to short the pins XRES and XRES/INT (pins 1 and 2) of the UCC board. This setting routes the XRES pin of the CapSense controller to pin 3 of ISSP connector J3.
- Place a jumper on header J2 to short the pins GND and SHIELD (pins 2 and 3) of the CY3280-SLM board. This setting allows the board hatch pattern on the CY3280-SLM board to connect to ground.
- Connect the MiniProg1/ MiniProg3 to the CY3280-20x66 UCC on the J3 header.
- Connect P3[5] on connector P3 on the CY3280-20x66 UCC to the external level translator. See Figure 3-2 for an example schematic.
- Connect GND from connector P3 on the CY3280-20x66 UCC to the external level translator.
- Power the external level translator. The $V_{CC}$ from connector P3 on the CY3280-20x66 UCC may be used to power the external level translator with 5 V.
- Connect the RS232 cable from the external level translator output to the PC.

## 3.4 Schematic

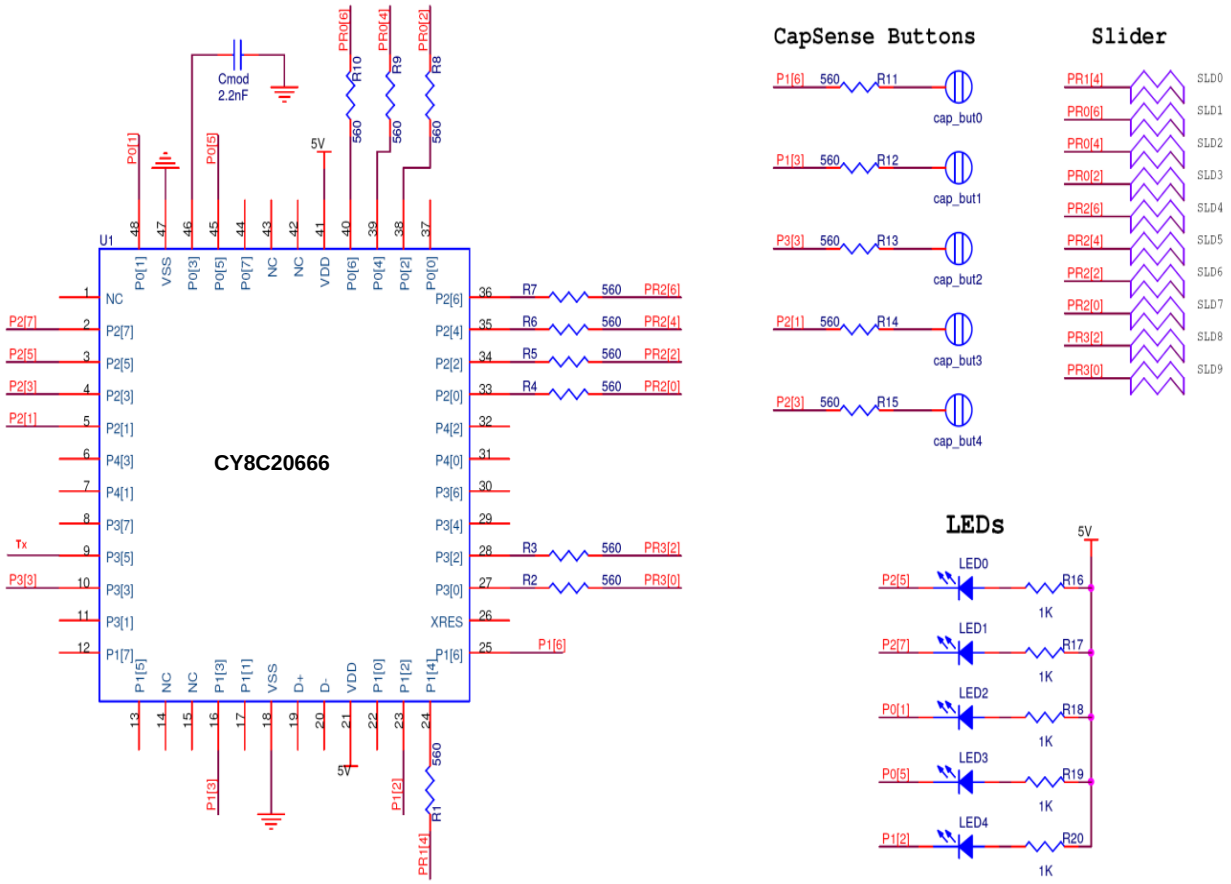
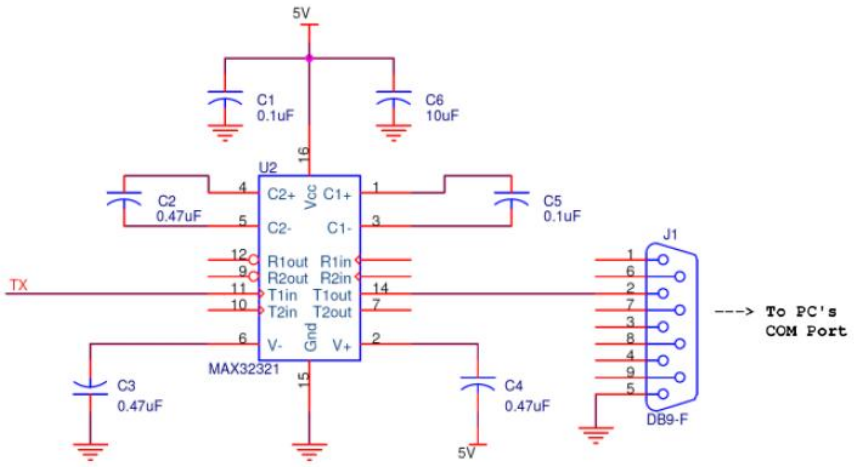



Figure 3-2. External Level Translator



The modulator capacitor ($C_{MOD}$) is a 2.2 nF capacitor connected to P0[3]. A 560 Ω resistor is connected in series with each CapSense button to reduce RF interference. LEDs are connected in active low configuration with 1 kΩ series resistors. In total there are five LEDs, five buttons, and one slider with 10 segments available.

Table 3-1. Pin Assignments for LEDs, Buttons, and Slider Segments

| LED | Button | Slider Segment |
|---|---|---|
| LED0 - P2[5] | BTN0 - P1[6] | SLD0 - P1[4], SLD1 - P0[6] |
| LED1 - P2[7] | BTN1 - P1[3] | SLD2 - P0[4], SLD3 - P0[2] |
| LED2 - P0[1] | BTN2 - P3[3] | SLD4 - P2[6], SLD5 - P2[4] |
| LED3 - P0[5] | BTN3 - P2[1] | SLD6 - P2[2], SLD7 - P2[0] |
| LED4 - P1[2] | BTN4 - P2[3] | SLD8 - P3[2], SLD9 - P3[0] |

## 3.5 Software Setup

### 3.5.1 Tools Required

- PSoC Designer (version 5.2 or higher)
- PSoC Programmer (version 3.13 or higher)
- MultiChart tool (available with AN2397)

### 3.5.2 User Module List

The following table lists the user modules used in this project and the hardware resources occupied by each user module.

| User Module | Hardware Resources |
|---|---|
| SmartSense | CapSense Block, Timer1 (default) |
| UART | No blocks occupied (Software implementation) |

### 3.5.3 User Module Parameters, Global Resource

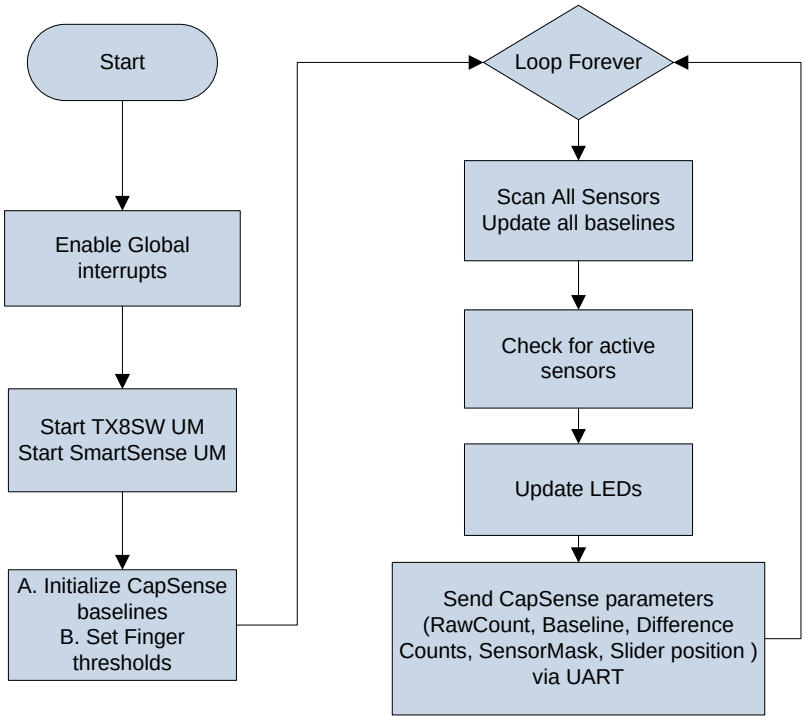
The `ReadMe.txt` file in the project describes the parameter settings for the user modules used in this code example. It also includes a list of the Global Resources.

## 3.6 Operation

On reset, the program loads all hardware settings from the device configuration into the device and executes *main.c*. The firmware then performs the following operations:

- Enables global interrupt, and then starts the SmartSense and UART user modules.
- Initializes the sensor baselines.
- Performs the following operations in an infinite loop:
  - Scans all sensors continuously and uses the UART user module to send the raw counts, difference counts, baseline, and the slider position to the TX8 pin.
  - When the firmware detects a button press, it switches the corresponding LED to ON. It switches the LED to OFF when the button is released.
  - When slider is touched, the firmware turns the LED to ON indicating the touch position.

Figure 3-3. Functional Flow for Code Example 3




## 3.7 Running the Code Example

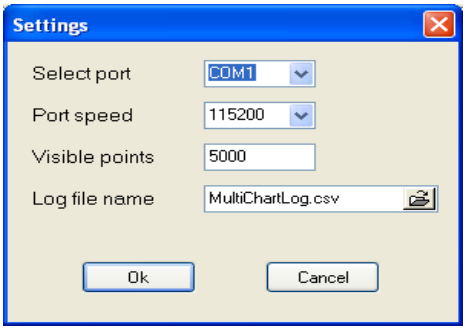
Program the board with the project, and then use this procedure to run the code example. For details on how to program the UCC board, refer to chapter5 of CY3280-20x66 kit guide.

1. Use MiniProg1/ MiniProg3 or any of the sources mentioned in the CY3280-20x66 UCC kit guide to power the board at 5 V.
2. Touch the linear slider on the CY3280-SLM module board.
   The corresponding LEDs on the CY3280-SLM board light up.
3. Touch a button.
   The corresponding LED on the CY3280-SLM module board lights up. Multiple buttons can be touched at the same time. The linear slider and buttons can also be used simultaneously.

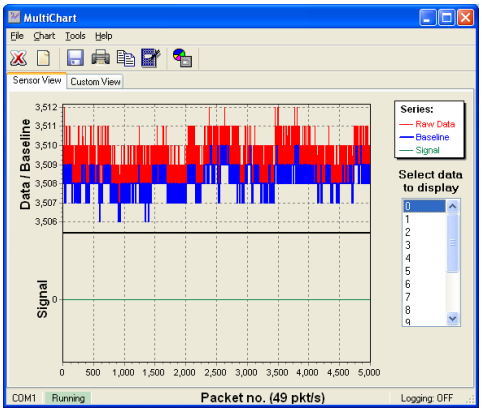
## 3.8 Plotting CapSense Data with MultiChart

Use this procedure to run the code example and view the plotted data with the MultiChart tool. For more information and to download the MultiChart tool, see AN2397, “CapSense® Data Viewing Tools”

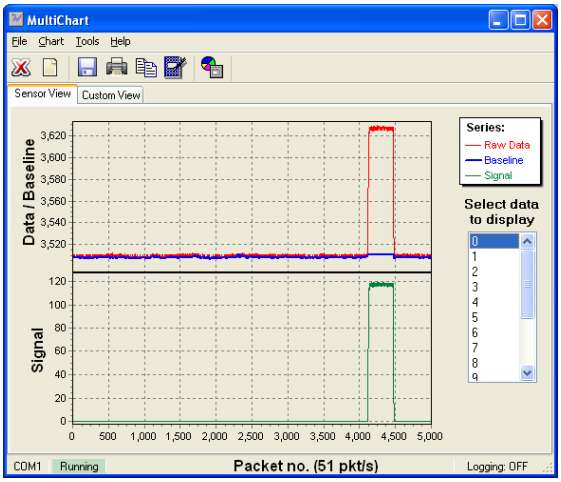
1. Open the MultiChart tool.
2. From the MultiChart tool, select **Tools** > **Settings.**
   The Settings dialog box appears.



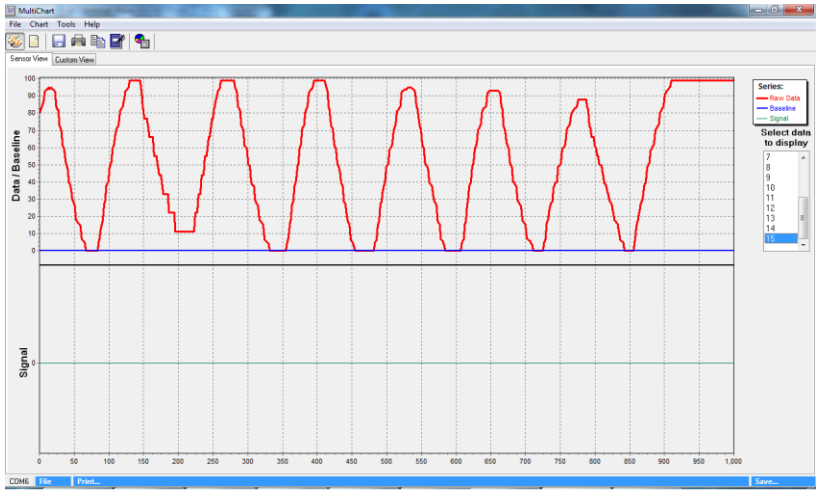

3. Use the **Select port** drop list to select the COM port used to connect the board to computer.
   Most users connect via COM1. However, your system may be different.
4. Use the **Port speed** drop list to select 115200, and then click **OK**.
   The raw counts appear in red and the baseline in blue.



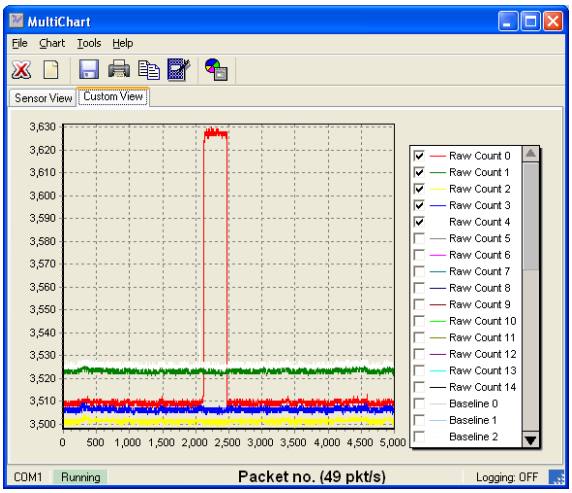

5. Touch button 0 to observe the signal (difference counts).



6. The ‘raw count 15’ variable of series 15 represents the slider position (varies between 0 – 100), shown below. If you are not able see it, zoom in or set the axes to auto in Chart > Edit...>Axis.




7. The sensor view shows only one sensor at a time. In this view, ‘Raw Count 15’ represents the slider position. To observe multiple sensors together, click the Custom View tab.



# Code Example 4 Measuring Absolute Sensor Capacitance with a CY8C20xx6A CapSense Controller

## 4.1 Project Name

CE_4_MeasuringAbsSensorCap_withCY8C20xx6A

## 4.2 Overview

This code example demonstrates how to use a CY8C20xx6A CapSense® controller to calculate the absolute capacitance of five sensors and to display the results on a computer with Windows HyperTerminal software or Bridge Control Panel. The measured absolute capacitance is accurate to 1 pF.

## 4.3 Hardware Setup

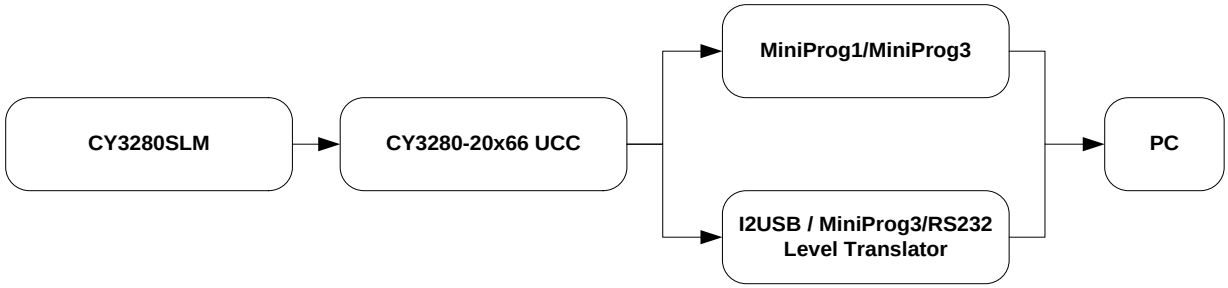
### 4.3.1 Requirements

- CY3280-20x66 Universal CapSense Controller
- CY3280-SLM Universal CapSense Linear Slider Module
- RS232 Level Translator Module
- CY3240-I2USB Bridge or CY8CKIT-002 Miniprog3
- CY3217-MiniProg1 Programmer Kit or CY8CKIT-002 Miniprog3
- USB A to Mini B Cable
- PC running Windows XP or later

### 4.3.2 Assembly

The following figure represents the hardware setup. The CY3280-20x66 UCC kit connects to the CY3280-SLM module through a 22x2_RA_Receptacle. Connect a MiniProg1/MiniProg3 to header J3 to program and power the kit.

Figure 4-1. Hardware Setup Block Diagram



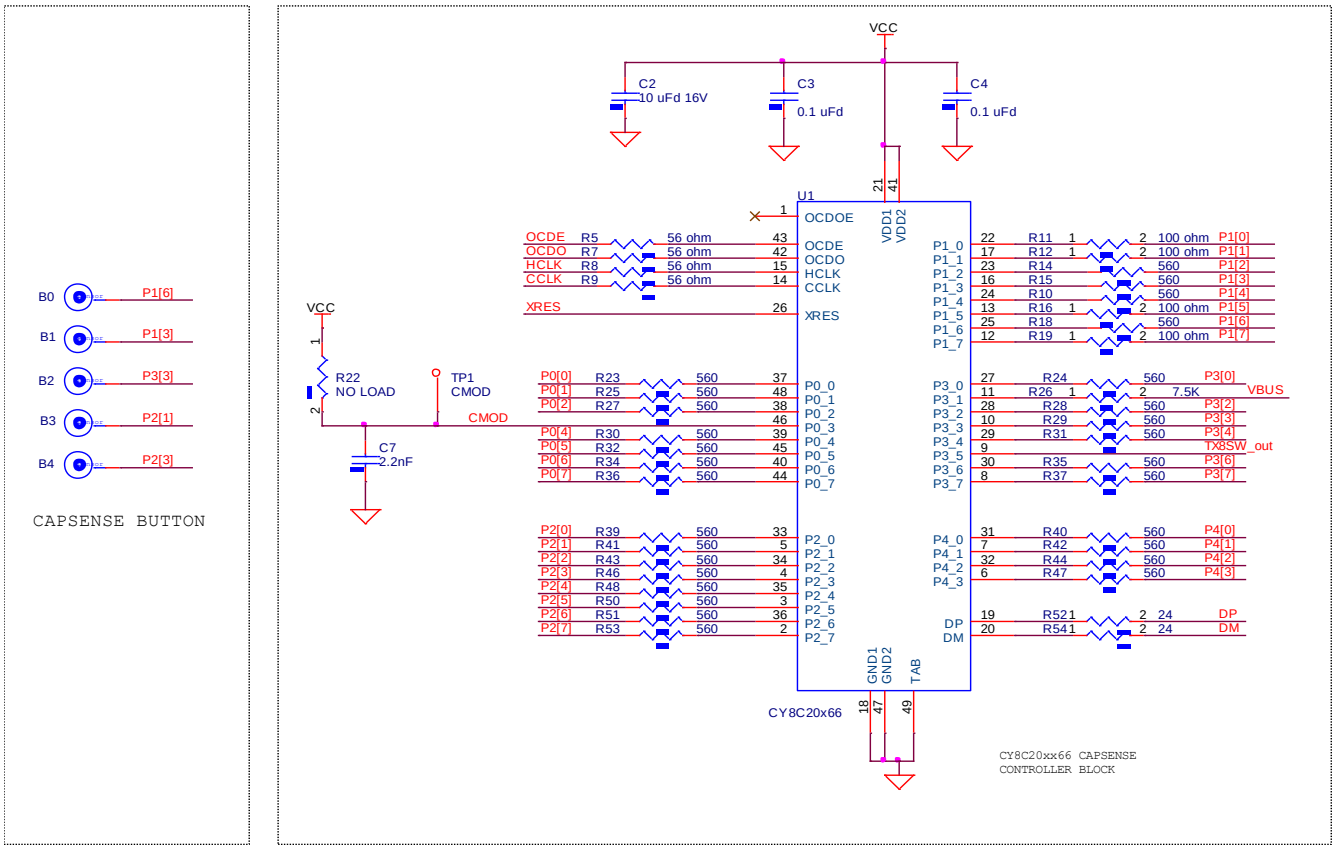

**Note** This code example uses the external level translator. However, you can also use a UART-to-USB bridge to view the data on the PC. You can convert the CY3240-I2USB bridge included in the UCC kit to a UART-to-USB bridge. For more details, see AN2397, "CapSense® Data Viewing Tools". Bridge Control Panel software can be used to view sensor Cp on the PC using either I2C-USB bridge or MiniProg3. HyperTerminal can be used to view the data on the PC using RS232 communication protocol.

### 4.3.3 Setting Up the Board

Make the following hardware connections:

- Connect header J1 of the CY3280-SLM daughter card to the 22x2_RA_Receptacle (connector P2) on the CY3280-20x66 UCC board.
- Place a jumper on header J7 to short the pins $V_{CC}$ and $V_{CC\_PROG}$ of the UCC Board. This setting allows you to power the CapSense controller from ISSP connector.
- Place a jumper on header J4 to short the pins XRES and XRES/INT (pins 1 and 2) of the UCC Board. This setting routes the XRES pin of the CapSense controller to pin 3 of ISSP connector J3.
- Place a jumper on header J2 to short the pins GND and SHIELD (pins 2 and 3) of the CY3280-SLM board. This setting allows the board hatch pattern on the CY3280-SLM board to connect to ground.
- Connect the MiniProg1/MiniProg3 to the CY3280-20x66 UCC on the J3 header to program the device.
- If sensor Cp is viewed on Bridge control panel through I2C communication protocol, connect the jumper J3 of CY3280-20x66A to I2C-USB Bridge/MiniProg3 and the other end of bridge/MiniProg3 to the computer through USB A to mini B cable. Power the device by clicking on Toggle power button in Bridge control panel.
- If sensor Cp is viewed on HyperTerminal through UART, connect the TX pin of RS232 level translator board to P1[7] of UCC. Connect Vcc and GND pins of level translator to Vcc and GND pins of UCC respectively. A good example of RS232 level translator is MAX 232 serial level converter
- Use the serial cable to connect RS232 port of level translator board to the computer.

## 4.4 Schematic




The modulator capacitor ($C_{MOD}$) is a 2.2 nF capacitor connected to P0[3]. A 560 Ω resistor is connected in series with each CapSense button to reduce RF interference.

Table 4-1. Pin Assignment for CapSense Buttons

| Button | Pin |
|---|---|
| 0 | P1[6] |
| 1 | P1[3] |
| 2 | P3[3] |
| 3 | P2[1] |
| 4 | P2[3] |

## 4.5 Software Setup

### 4.5.1 Tools Required

- PSoC® Designer™ (version 5.2 or higher)
- PSoC Programmer (version 3.13 or higher)
- Bridge Control Panel
- Windows HyperTerminal

### 4.5.2 User Modules List

The following table lists the user modules (UM) used in this code example and the hardware resources occupied by each UM.

| User Module | Hardware Resources |
|---|---|
| CSD | CapSense block,Timer1(default) |
| EzI2Cs | I2C block |
| TX8SW | No blocks occupied ( Software implementation) |

### 4.5.3 User Module Parameters, Global Resources

The `ReadMe.txt` file in the project describes the parameter settings for the user modules used in this code example. It also includes a list of the Global Resources.

## 4.6 Operation

The program uses the following formula to calculate raw counts for a sensor:

$$raw\_count = \left( \frac{(2^n - 1) \cdot V_{ref} \cdot \overline{f_s}}{i_{DAC}} \right) \cdot C_s$$

Where,
n = resolution
Vref = Reference voltage
fs = switching voltage of the pre-charge clocks
Cs = sensor Capacitance
iDAC = iDAC current

n, f and Vref can be determined from CSD UM parameters.

iDAC can be determined by the below equation:

$$i_{DAC} = IDAC\_D \cdot iDAC_{gain} \cdot iDAC_{range}$$

IDAC_D is the control register setting, and iDACrange is the “Idac Range” in CSD UM parameter setting. Both of these are known at design time. The only missing piece needed to calculate IDAC is iDACgain. iDACgain is the iDAC strength in A/bit which varies over PVT from part to part. In the 4X range, the test limits for iDACgain are 230nA/bit < iDACgain < 270nA/bit.

The Krypton ATE program (at production) measures iDAC current in the 4x range with IDAC_D = 0xC0 = 192. With a nominal 247nA/bit iDAC, this current would be 247nA/bit; 192 * 4 = 190uA. The ATE measures the actual current and records the difference in uA from 190uA in a sign-and-magnitude format and stores it as bIDACComp. The MSB (mask = 0x80) is interpreted as the sign with 1 = negative and 0 = positive. For example, a part with 0x8D stored has iDACgain = (190uA – 13uA) / (4 * 192) = 230nA/bit.

Note that:

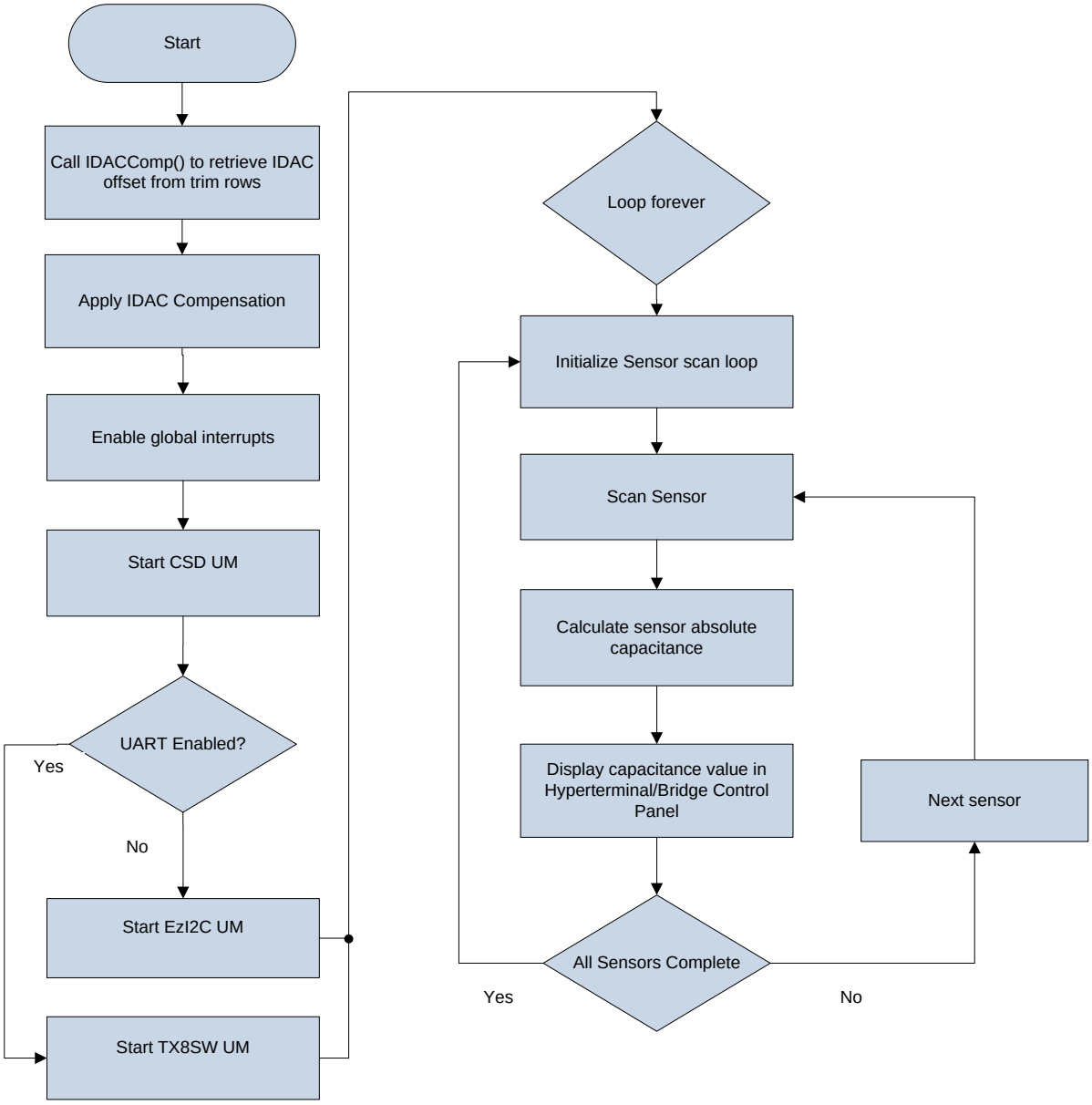
$$\text{uSensitivity} = \text{Cs/RawCount} = \frac{(2^n - 1) * V_{ref} * f_s}{i_{DAC}}$$

The program uses the following procedure to display the absolute capacitance on the screen:

1. Calculates the sensitivity for the sensors and stores the results in the variable uSensitivity. GetIDACComp() is a custom function defined in the file *GetIDACComp.asm* to retrieve the IDAC compensation for part-to-part variations in IDAC.
2. Note: Since IDAC varies with temperature variation, this code example is only valid for room temperature.

3. Enables the global interrupts, and starts the user modules.
4. Scans each sensor sequentially, and gets the raw counts from the CSD_waSnsResult[] array.
5. Calculates the absolute capacitance by taking the ratio of the raw counts and the sensitivity.
6. Transmits the result to a computer for display in the Bridge Control Panel or HyperTerminal, depending on user choice.

Figure 4-2. Functional Flow for Code Example 4




## 4.7 Running the Code Example

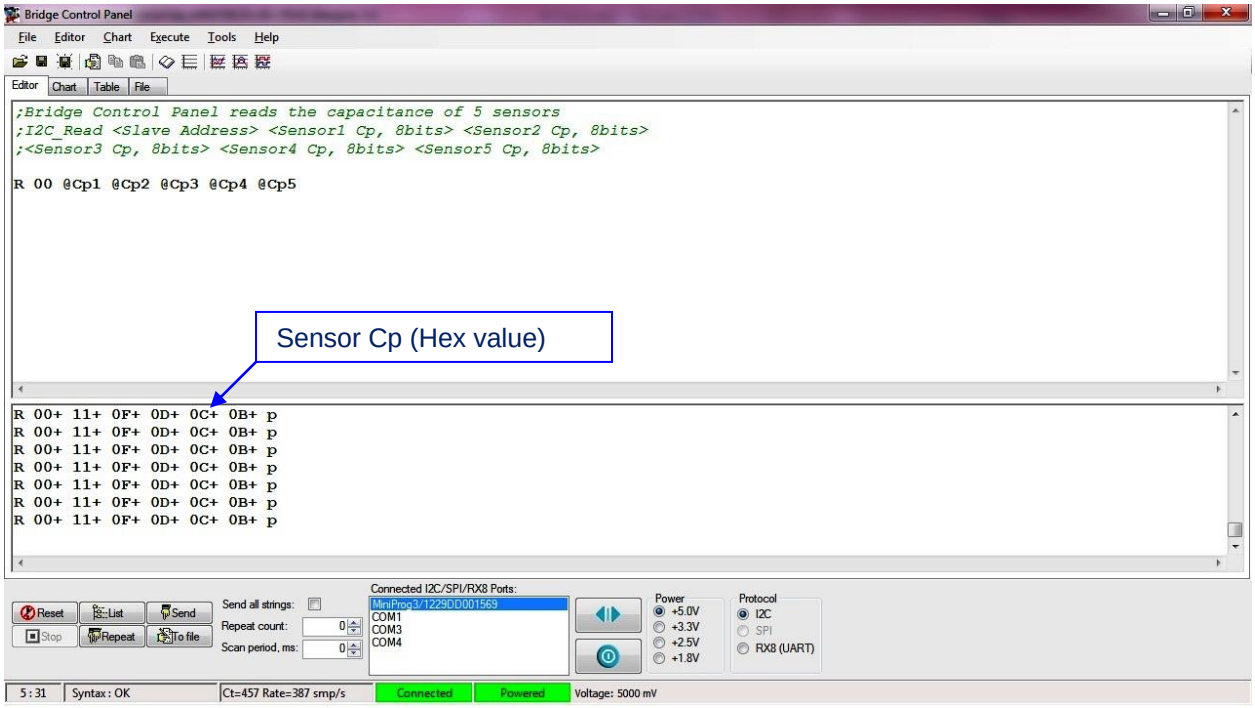
### Programming the UCC Board

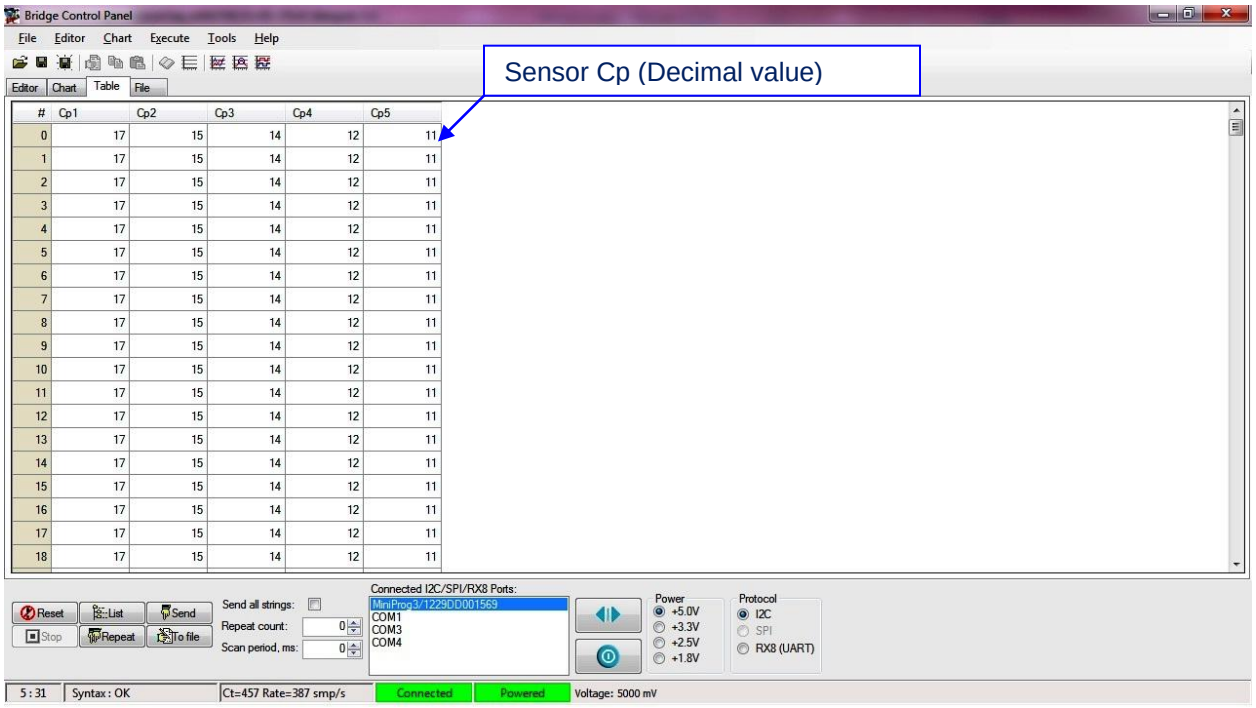
Ensure that the #define UART statement is commented out in *main.c*, as shown in the following screenshot. Next, generate the project (press F6), and program the board. For details on how to program the UCC board, refer to chapter 5 of the CY3280-20x66 kit guide.

```
74
75 /* Comment this while using I2C;Uncomment this while using UART */
76  //#define UART
77
```

### Monitoring sensor data using BCP

1. Connect jumper J3 of CY3280-20x66A to I2C-USB Bridge or MiniProg3 and the other end of I2C-USB bridge/MiniProg3 to the computer through USB A to mini B cable.
2. On the computer desktop, select **Start** > **All Programs** > **Cypress** > **Bridge Control Panel (version)** > **Bridge Control Panel (version).**
3. Select the device (I2C-USB Bridge or MiniProg3) from the port selection window.
4. Power the device by clicking the Toggle power button in Bridge Control Panel.
5. From the Bridge Control Panel, select **File** > **Open.** Load the *BCP.iic* file from BCP Configuration Files folder contained in the project folder.
6. Select **Charts** > **Variable Settings**. Load the *BCP.ini* file from BCP Configuration Files folder contained in the project folder.
7. Click on the send button to send the $I^2C$ command to the CY8C20xx6 controller to get the sensor Cp values in the bottom window in the hex format as shown in the following figure.






## Programming the UCC board

Uncomment #define UART statement in main.c as shown in the following screenshot, generate the project, and program the UCC board.

```
74
75  /* Comment this while using I2C;Uncomment this while using UART */
76    #define UART
77
```

## Monitoring sensor data using HyperTerminal

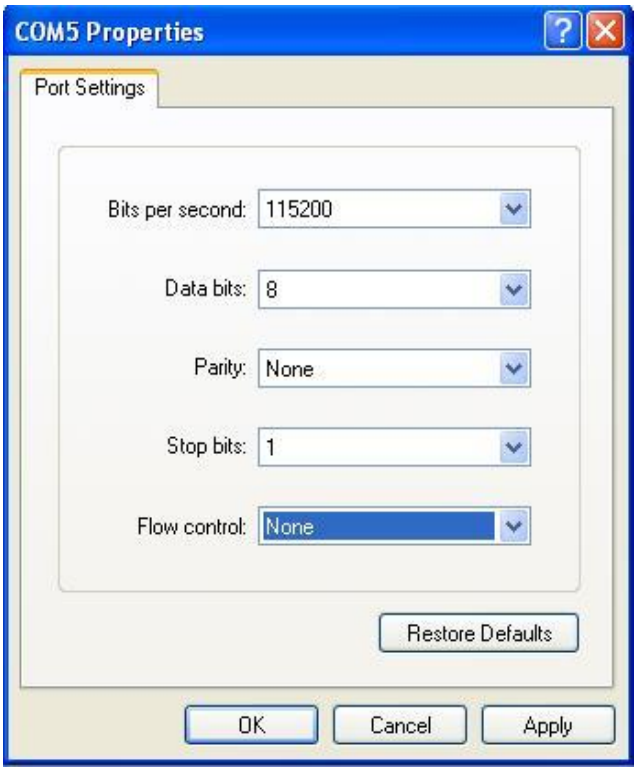
1. Power the board using either Miniprog1 or Miniprog3.
2. On the computer desktop, select **Start** > **All Programs** > **Accessories > Communication > HyperTerminal.**
3. Type a name for the connection.
4. Click **OK**.


5. Use the **Connect** using drop list to select the serial port used to receive data from the PSoC.
6. Click **OK.**


7. In the **COM Properties** dialog box, configure the following parameters:

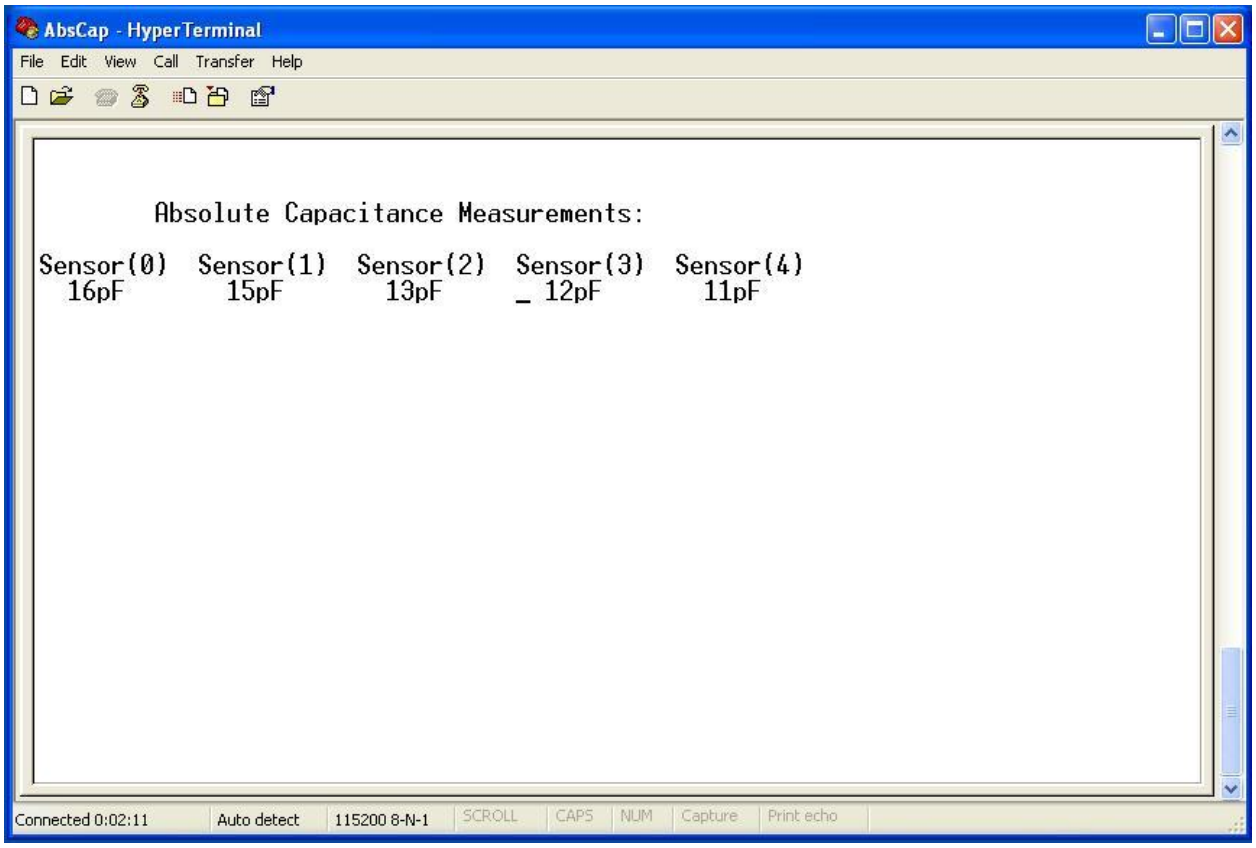
- Bits per second = 115200
- Data bits = 8
- Parity = None
- Stop bits = 1
- Flow control = None

8. Click **OK**. The HyperTerminal connects to the specified COM port and is ready for use.




9. Power on and reset the UCC.

The results appear in the HyperTerminal window.


AbsCap - HyperTerminal
File Edit View Call Transfer Help
Absolute Capacitance Measurements:
Sensor(0) Sensor(1) Sensor(2) Sensor(3) Sensor(4)
16pF 15pF 13pF _ 12pF 11pF
Connected 0:02:11
Auto detect
115200 8-N-1
SCROLL
CAPS
NUM
Capture
Print echo

# Code Example 5 Measuring Absolute Sensor Capacitance with a CY8C21x34/B CapSense Controller

## 5.1 Project Name

CE_5_MeasuringAbsSensorCap_withCY8C21x34

## 5.2 Overview

This code example uses a CY8C21x34/B CapSense® controller to calculate the absolute capacitance of five sensors and display it on a computer with Windows HyperTerminal software or Bridge control panel. The measured absolute capacitance is accurate to 1 pF.

## 5.3 Hardware Setup

### 5.3.1 Requirements

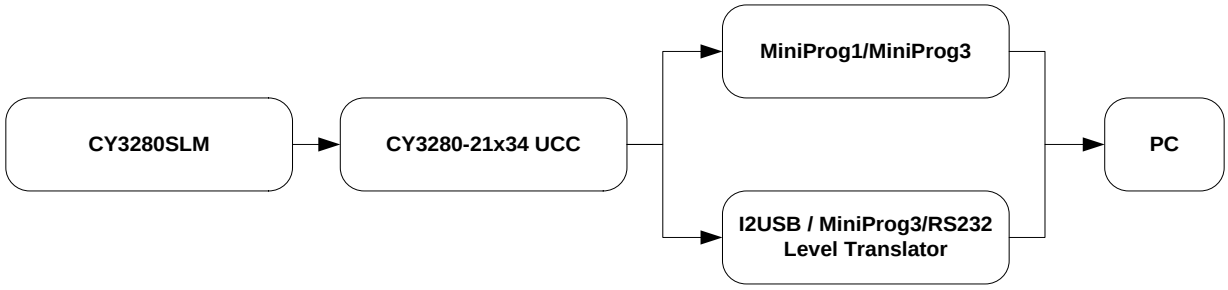
- CY3280-21x34 Universal CapSense Controller
- CY3280-SLM Universal CapSense Linear Slider Module
- RS232 Level Translator Module
- CY3240-I2USB Bridge or CY8CKIT-002 Miniprog3
- CY3217-MiniProg1 Programmer or CY8CKIT-002 Miniprog3
- USB A to Mini B Cable
- PC running Windows XP or later

### 5.3.2 Assembly

Figure 5-1 represents the hardware setup. The CY3280-21x34 UCC kit connects to the CY3280-SLM module through a 22x2_RA_Receptacle. A MiniProg1/MiniProg3 connects to header J3 to program and power the kit.

The Bridge Control Panel software can be used to view the sensor Cp on the PC using either the I2C-USB bridge or MiniProg3.
The HyperTerminal can be used to view the data on the PC using the RS232 communication protocol.

Figure 5-1. Hardware Setup Block Diagram




### 5.3.3 Setting Up the Board

Make the following hardware connections:

- Connect header J1 of the CY3280-SLM daughter card to the 22x2_RA_Receptacle (connector P2) on the CY3280-21x34 UCC board.
- Place a jumper on header J1 to short the pins $V_{CC}$ and 5 V of the UCC board. This setting allows you to power the CapSense controller from the ISSP connector.
- Place a jumper on header J4 to short the pins XRES and XRES/INT (pins 1 and 2) of the UCC board. This setting routes the XRES pin of the CapSense controller to pin 3 of ISSP connector J3.
- Place jumper on header J2 to short the pins GND and SHIELD (pins 2 and 3) of the CY3280-SLM board. This setting allows the board hatch pattern on the CY3280-SLM board to connect to ground.
- Connect the MiniProg1/MiniProg3 to the CY3280-21x34 UCC on the J3 header to program the device
- If sensor Cp is viewed on Bridge control panel through I2C communication protocol, connect the jumper J3 of CY3280-21x34 to I2C-USB Bridge/MiniProg3 and the other end of bridge to the computer through USB A to mini B cable. Power the device by clicking on Toggle power button in Bridge control panel.
- If sensor Cp is viewed on the HyperTerminal through UART, connect the TX pin of the RS232 level translator board to P1[7] of UCC. Connect the Vcc and GND pins of the level translator to the Vcc and GND pins of UCC respectively. A good example of an RS232 level translator is the MAX 232 serial level converter
- Use the serial cable to connect the RS232 port of the level translator board to the computer.

## 5.4 Schematic

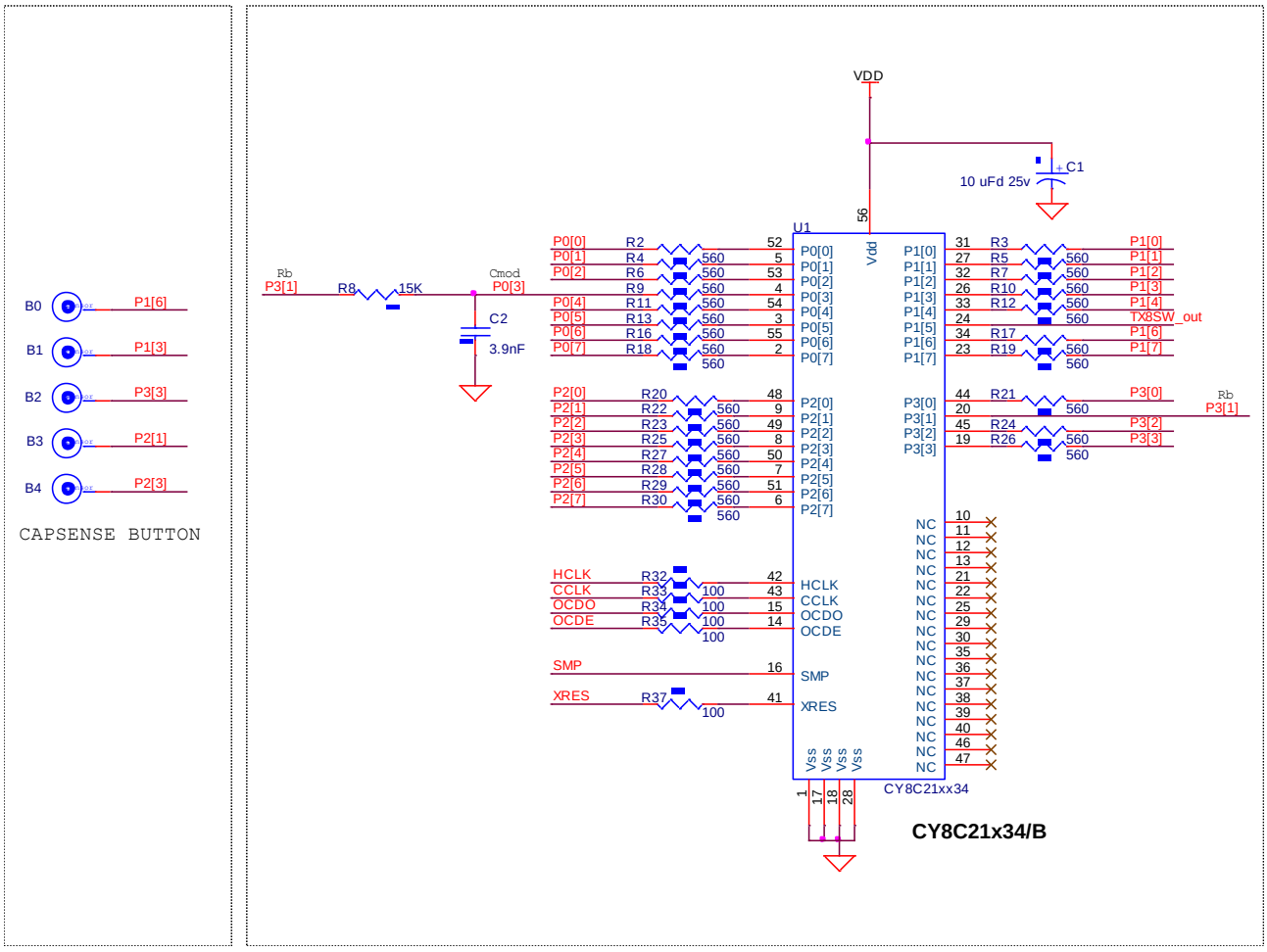



Table 5-1. Pin Assignments for Code Example 5

| Object | Pin |
|---|---|
| Button 0 | P1[6] |
| Button 1 | P1[3] |
| Button 2 | P3[3] |
| Button 3 | P2[1] |
| $C_{MOD}$ | P0[3] |
| Rb | P3[1] |

## 5.5 Software Setup

### 5.5.1 Tools Required

- PSoC® Designer (version 5.2 or higher)
- PSoC Programmer (version 3.13 or higher)
- Bridge Control Panel
- Windows HyperTerminal

### 5.5.2 User Modules List

The following table lists the user modules (UM) used in this code example and the hardware resources occupied by each UM.

| User Module | Hardware Resources |
|---|---|
| CSD | CapSense block,Timer1(default) |
| EzI2Cs | I2C block |
| TX8SW | No blocks occupied (Software implementation) |

### 5.5.3 User Module Parameters, Global Resources

The `ReadMe.txt` file in the project describes the parameter settings for the user modules used in this code example. It also includes a list of the Global Resources.

## 5.6 Operation

The program uses the following formula to calculate raw counts for a sensor:

$$\text{Raw Counts} = C_P \,.\, \text{Sensitivity}$$

Where: $C_P$ is the sensor capacitance

Thus, we calculate the absolute capacitance of each sensor by taking the ratio of the raw counts and the sensitivity.

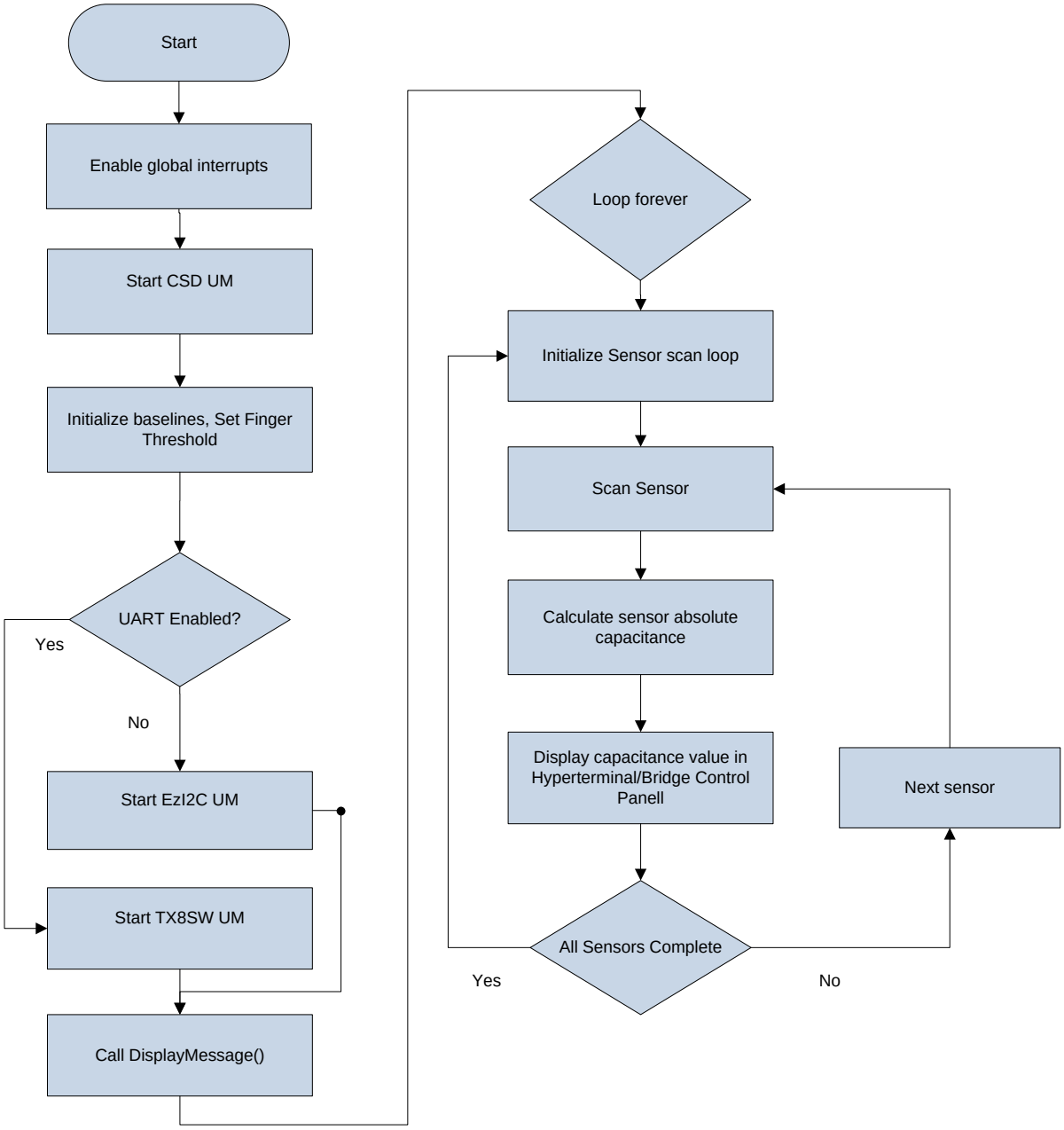
This project includes a file called `CY8C21x34/B_CalculSensitivity.xls`. It contains the calculation of the sensitivity value. The settings for the Ref Value and Prescaler parameters that produced the most accurate results are:

- Ref Value = 5
- Prescaler = 5

The program uses the following procedure to display the absolute capacitance on the screen:

1. With this configuration, the calculated sensitivity equals 465 counts/pF. Refer to *CY8C21x34/B*_CalculSensitivity.*xls*.
   The .xls file is part of the project and stored in the project folder.
2. Enables global interrupts, and starts the user modules.
3. Scans each sensor sequentially, and gets the raw counts from the CSD_waSnsResult[] array.
4. Calculates the absolute capacitance by taking the ratio of the raw counts and the sensitivity.
5. Transmits the result to a computer for display in Bridge Control Panel or HyperTerminal depending on user choice.

Figure 5-2. Functional Flow for Code Example 5




## 5.7 Running the Code Example

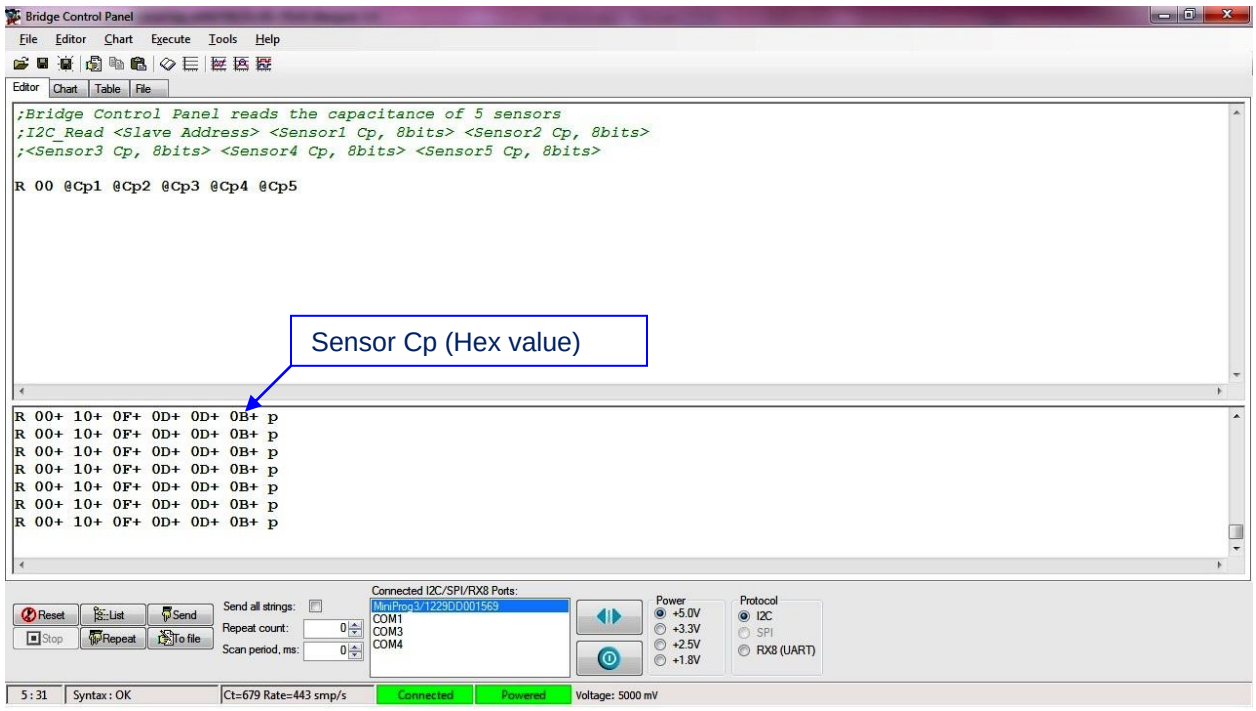
### Programming the Board

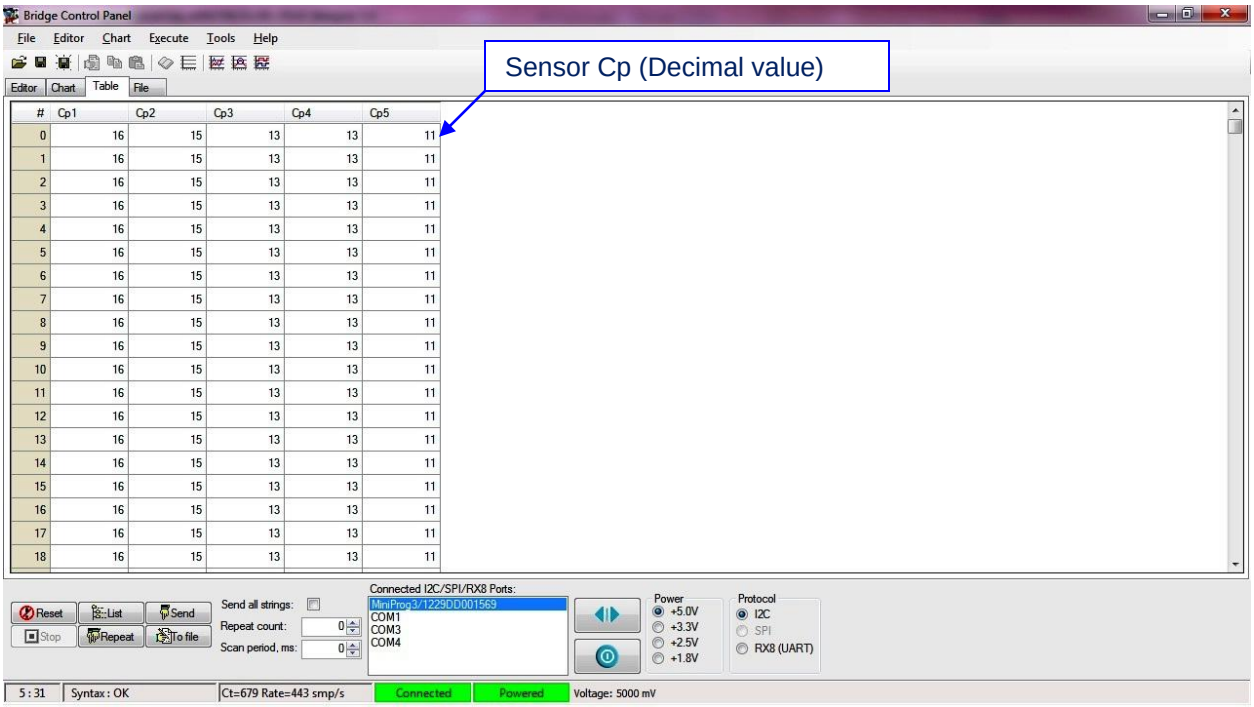
Ensure that the #define UART statement in commented out in *main.c* as shown in the following screenshot, generate the project (press F6) and program the board. For details on how to program the UCC board, refer to Chapter 5 of the CY3280-21x34 kit guide.

```
76 /* comment this while using I2C;Uncomment this while using UART */
77
78  //#define UART
79
```

### Monitoring sensor data using BCP

1) Connect jumper J3 of CY3280-21x34 to I2C-USB Bridge or MiniProg3 and the other end of I2C-USB Bridge/MiniProg3 to the computer through USB A to mini B cable.
2) On the computer desktop, select **Start** > **All Programs** > **Cypress** > **Bridge Control Panel (version)** > **Bridge Control Panel (version).**
3) Select the device from the port selection window.
4) Power the CY3280-21x34 UCC kit board at 5 V.
5) From the Bridge Control Panel, select **File** > **Open**. Load the BCP.iic file from BCP Configuration Files folder contained in the project folder.
6) Select **Charts** > **Variable Settings**. Load the *BCP.ini* file from BCP Configuration Files folder contained in the project folder.
7) Click on the send button to send the $I^2C$ command to the CY8C21x34 controller to get sensor $C_P$ values in the bottom window in hex format as shown in the following screenshot.






## Programming the board

Uncomment the #define UART statement in main.c as shown below and program the board.

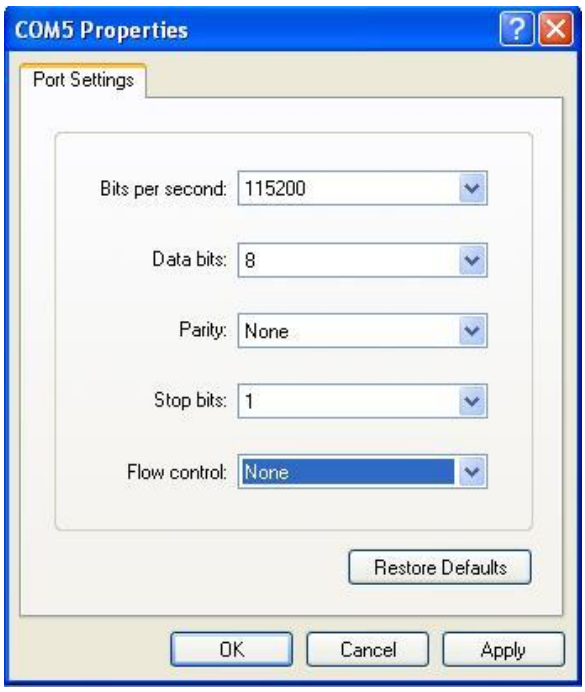
```
76 /* comment this while using I2C;Uncomment this while using UART */
77
78  #define UART
79
```

## Monitoring sensor data using HyperTerminal.

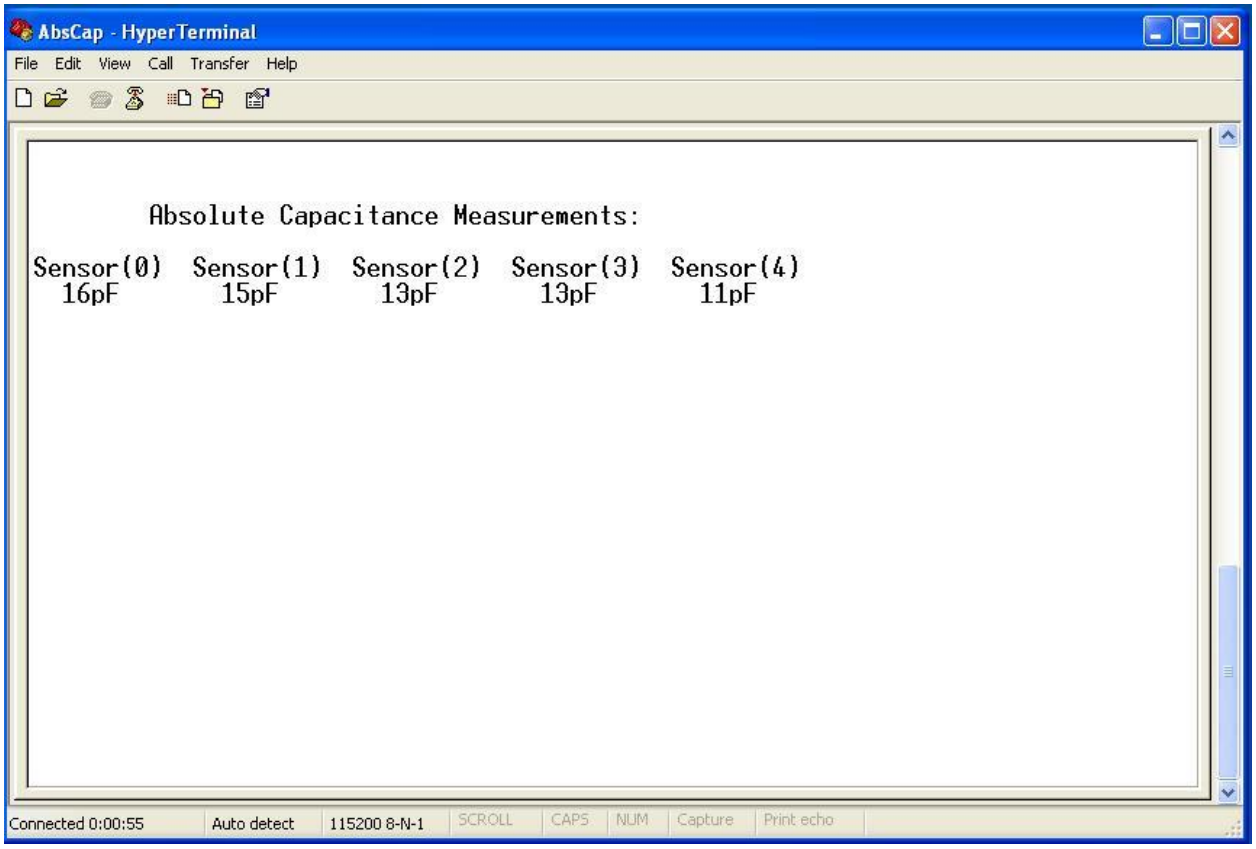
1. Power the board using either Miniprog1 or Miniprog3.
2. On the computer desktop, select **Start** > **All Programs** > **Accessories** > **Communication** > **HyperTerminal**.
3. Type a name for the connection.
4. Click the **OK** button.

5. Use the Connect using drop list to select the serial port used to receive data from the PSoC. This setting varies based on the PC configuration. In our example, we use COM5.
6. Click OK.


6. In the COM Properties dialog box, configure the following parameters:
   - Bits per second = 115200
   - Data bits = 8
   - Parity = None
   - Stop bits = 1
   - Flow control = None
7. Click **OK**.
   The HyperTerminal connects to the specified COM port and is ready for use.



8. Power on and reset the UCC.
   The results appear in the HyperTerminal window.



# Code Example 6 Measuring Absolute Sensor Capacitance with a CY8C20x34 CapSense Controller

## 6.1 Project Name

CE_6_MeasuringAbsSensorCap_withCY8C20x34

## 6.2 Overview

This code example uses a CY3280-20x34 CapSense® controller to calculate the absolute capacitance of five sensors. The measured absolute capacitance is accurate to 1 pF.

## 6.3 Hardware Setup

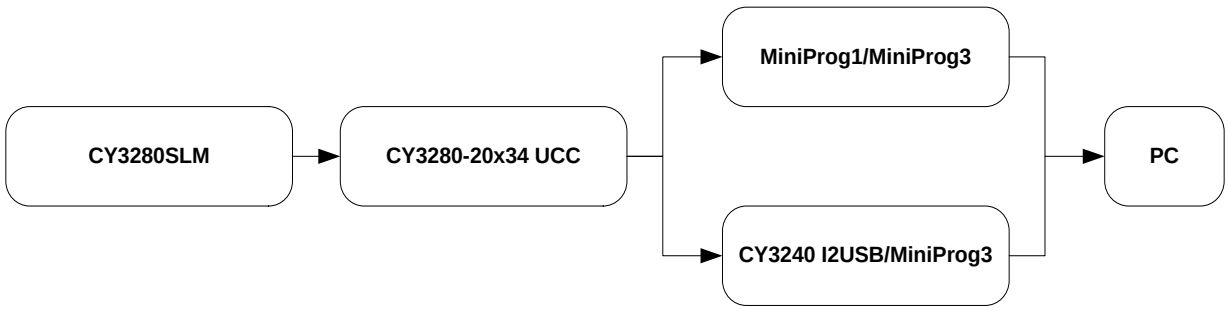
### 6.3.1 Requirements

- CY3280-20x34 Universal CapSense Controller
- CY3280-SLM Universal CapSense Linear Slider Module
- CY3217 – MiniProg1 Programmer or CY8CKIT-002 Miniprog3
- CY3240 – I2USB Bridge or CY8CKIT-002 Miniprog3
- USB A to Mini B Cable
- PC running Windows XP or later
- Multimeter

### 6.3.2 Assembly

Figure 6-1 represents the hardware setup. The CY3280-20x34 UCC kit connects to the CY3280-SLM module through a 22x2_RA_Receptacle. A MiniProg1/Miniprog3 connects to header J3 to program and power the kit.

Figure 6-1. Hardware Setup Block Diagram

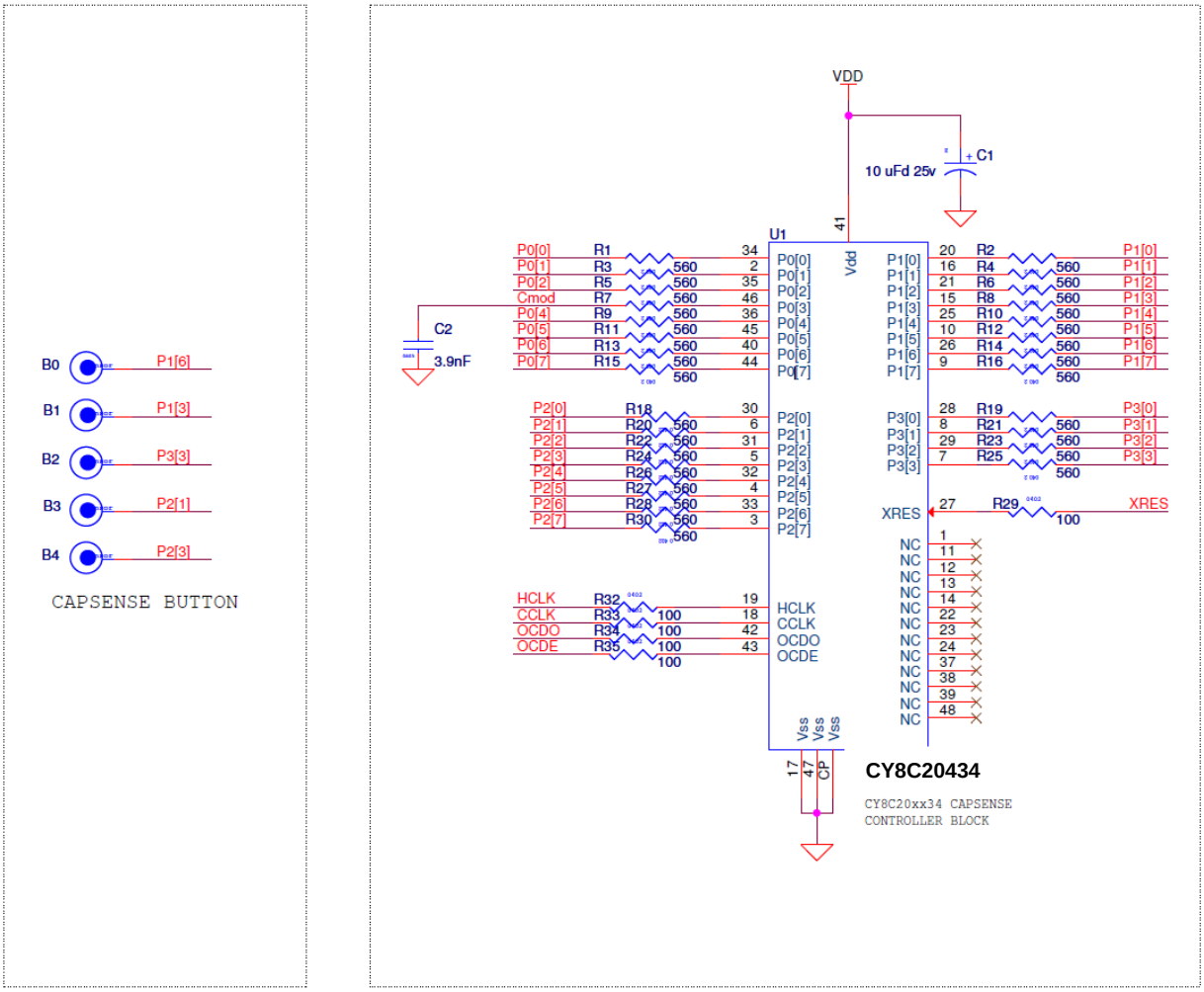


### 6.3.3 Setting Up the Board

Make the following hardware connections:

- Connect header J1 of the CY3280-SLM daughter card to the 22x2_RA_Receptacle (connector P2) on the CY3280-20x34 UCC board.
- Place a jumper on header J1 to short the pins $V_{CC}$ and 5 V of the UCC board. This setting allows you to power the CapSense controller from the ISSP connector.
- Place a jumper on header J4 to short the pins XRES and XRES/INT (pins 1 and 2) of the UCC board. This setting routes the XRES pin of the CapSense controller to pin 3 of ISSP connector J3.
- Place a jumper on header J2 to short the pins GND and SHIELD (pins 2 and 3) of the CY3280-SLM board. This setting allows the board hatch pattern on the CY3280-SLM board to connect to ground.
- Connect the MiniProg1/Minirpog3 to the ISSP header on J3 of the UCC board. This connection is required only when the UCC is to be programmed with the code generated hex file. This connection has to be replaced with the I2USB bridge when the CapSense data is to be read in the Bridge Control Panel software
- Use the USB A to Mini B cable to connect the other end of the MiniProg1/Miniprog3 (or the I2USB bridge) to the PC.

## 6.4 Schematic



The modulator capacitor ($C_{MOD}$) is a 3.9-nF capacitor connected to P0[3]. A 560 Ω resistor is connected in series with each CapSense button to reduce RF interference.

Table 6-1. Pin Assignments for Code Example 6

| Button | Pin |
|---|---|
| 0 | P1[6] |
| 1 | P1[3] |
| 2 | P3[3] |
| 3 | P2[1] |
| 4 | P2[3] |

## 6.5 Software Setup

### 6.5.1 Tools Required

- PSoC® Designer (version 5.2 or higher)
- PSoC Programmer (version 3.13 or above)
- Bridge Control Panel

### 6.5.2 User Modules List

The following table lists the user modules (UM) used in this code example and the hardware resources occupied by each UM.

| User Module | Hardware Resources |
|---|---|
| CSA_EMC | CapSense Block, Comparator (default) |
| EzI2Cs | $I^2C$/SPI |

### 6.5.3 User Module Parameters, Global Resources

The `ReadMe.txt` file in the project describes the parameter settings for the user modules used in this code example. It also includes a list of the Global Resources.

## 6.6 Operation

The *main.c* file has two functions. It calculates the IDAC code of each sensor. It then directs a proportional amount of current to a port pin from which a user can physically measure the current. We calculate the absolute capacitance of each sensor with the following equation:

$$C_P = \frac{I_{measured}}{(IMO/8) \times 1.3}$$

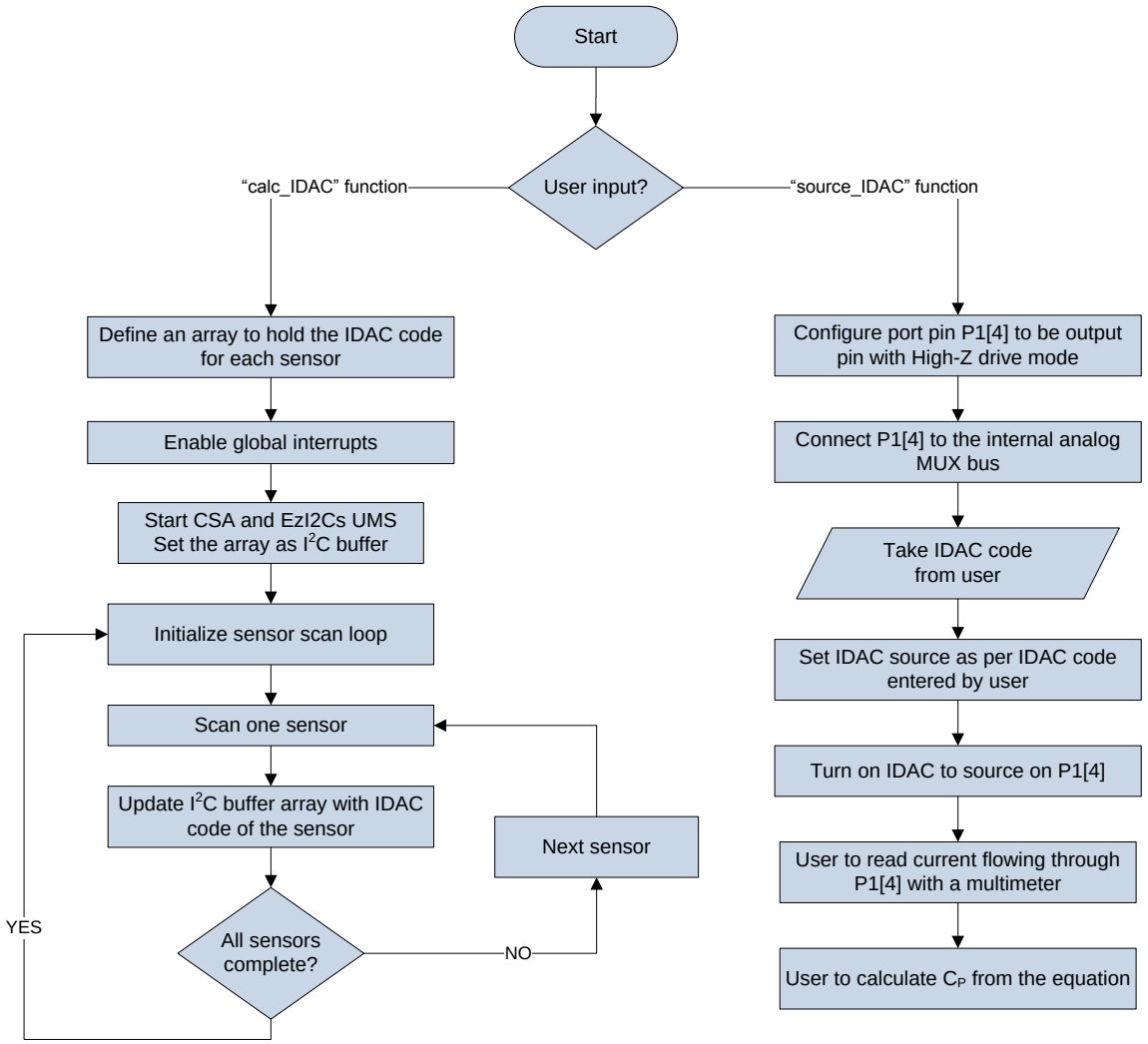
Where: $C_P$ is the sensor capacitance.

The program completes the following procedure:

- Defines the two functions calc_IDAC and source_IDAC, and then it calls one of them, depending on the user input.
- The first function calc_IDAC performs the following operations:
  - Defines an array baIDAC to store the IDAC code of each CapSense sensor as per the sensor number.
  - Sets the array as the $I^2C$ read buffer.
  - Enables the global interrupts, and then starts the user modules.
  - In an infinite loop, scans each sensor sequentially and gets the IDAC code stored in the array. It also updates the $I^2C$ buffer.

- The second function source_IDAC performs the following operations:
  - Configures the port pin P1[4] as an output pin with a High-Z drive.
  - Connects the pin P1[4] to the internal analog MUX bus.
  - Reads the IDAC code entered by the user.
  - Sources the corresponding current on the pin P1[4].

Figure 6-2. Functional Flow for Code Example 6



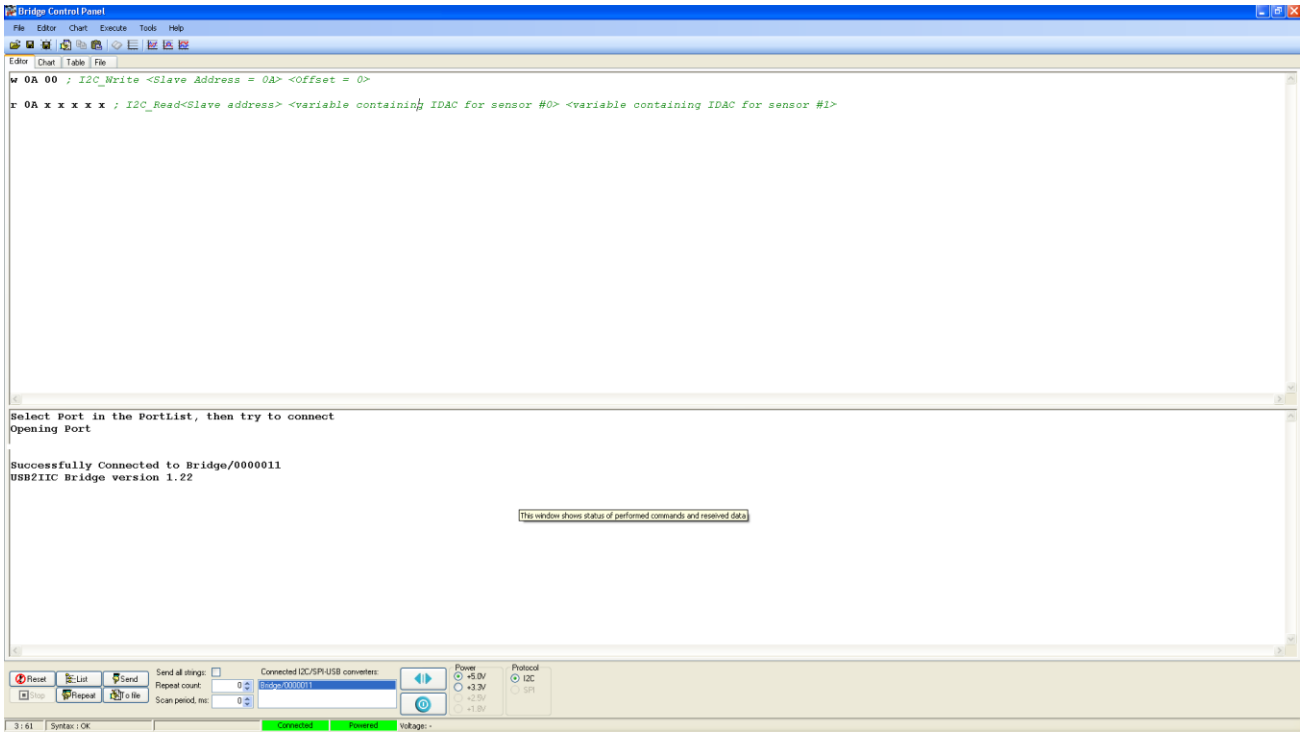

## 6.7 Running the Code Example

Program the board with the project, and then use this procedure to run the code example. For details on how to program the UCC board, refer to Chapter 5 of CY3280-20x34 kit guide.

1. Use I2USB bridge and a USB A to Mini B cable to connect your computer to the ISSP connector J3 of the CY3280-20x34 Universal CapSense Controller board.
2. On the computer desktop, select **Start** > **All Programs** > **Cypress** > **Bridge Control Panel** *(version)* > **Bridge Control Panel** *(version)*.
   The Bridge Control Panel is a component of the PSoC Programmer installation.
3. Select the device from the port selection window.

4. Power the CY3280-20x34 CapSense Controller board at 5 V.
5. From the Bridge Control Panel, select **Tools** > **Protocol Configuration**. In the **I2C** tab, select **I2C Speed** as 100 kHz.
6. Select **File** > **Open**. Load the `BCP.iic` file from *BCP Configuration files* folder contained in the code example project folder.

Figure 6-3. Display of Slave Address in Status Window




7. Press [**Enter**] to execute the second instruction in the Bridge Control Panel.
   The status window displays the IDAC code for all the sensors.
8. After you have the IDAC code for the five sensors, open the project, uncomment the statement *source_IDAC(),* and comment the statement *calc_IDAC()* to source the corresponding current on the port pin P1[4].
9. In the function *source_IDAC(),* set the IDAC code to the one obtained in the Bridge Control Panel for any one sensor, by assigning the appropriate hexadecimal value to the IDAC_D variable.
10. Use PSoC Designer to build the project. Select **Build** > **Generate** or **CE_6_MeasuringAbsSensorCap_withCY8C20x34** Project. Verify that there are no errors or warnings during the build.
11. To program the UCC with the generated hex file, select **Program** > **Program Part**.
12. In the Program Part window, verify that the MiniProg1/Miniprog3 is connected. If it is not connected, click the **Connect** button.
13. Click the **Program** button.
14. Disconnect the CY3280-SLM board from the UCC.
15. Click the power button to power on the UCC with the MiniProg1/Miniprog3.
16. Measure the current flowing through P1[4] ($I_{MEASURED}$)by placing a current meter between the pin and ground.
17. Calculate $C_P$ using the equation:

$$C_P = \frac{I_{measured}}{\left(IMO/_8\right) \times 1.3}$$

    Where: IMO is the system clock source. For this project, it is set to 12 MHz.
18. Repeat steps 9 to 17 to obtain the $C_P$ for other sensors.

Table 6-2 lists the $C_P$ values of CapSense sensors, along with the IDAC code obtained through the Bridge Control Panel and the IDAC current sourced on the pin.

Table 6-2. Measured $C_P$

| CapSense Sensor | IDAC Code | IDAC Current Sourced on Port Pin (µA) | Measured $C_P$ Value (pF) |
|---|---|---|---|
| P1[6] | 0A | 28.0 | 14.40 |
| P1[3] | 09 | 25.3 | 12.98 |
| P3[3] | 08 | 22.6 | 11.58 |
| P2[1] | 07 | 19.3 | 9.90 |
| P2[3] | 07 | 19.3 | 9.90 |

# Code Example 7 Tuning CSD with a CY8C20xx6A CapSense Controller

## 7.1 Project Name

CE_7_TuningCSD_withCY8C20xx6A

## 7.2 Overview

This code example demonstrates how to tune the CapSense® UM for a CY8C20xx6A device. To tune a CapSense UM, you must be able to view the CapSense data in a graphical format. This code example uses the EzI2C UM to transmit the required CapSense parameters over an $I^2C$ bus and an I2USB bridge. It uses the Bridge Control Panel software on a PC to view the data.

## 7.3 Hardware Setup

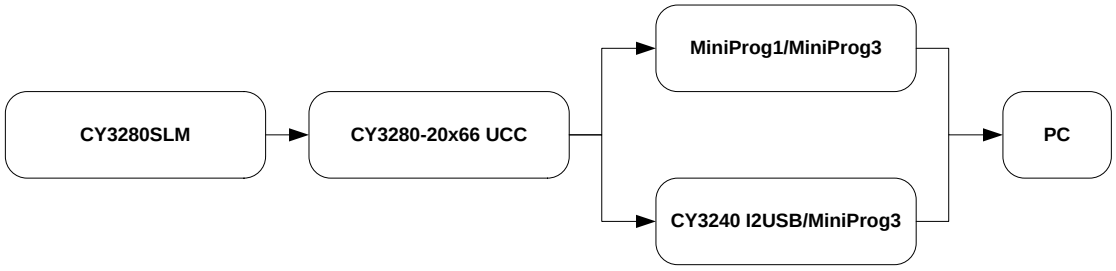
### 7.3.1 Requirements

- CY3280-20x66 Universal CapSense Controller board
- CY3280-SLM Universal CapSense Linear Slider Module
- CY3240-I2USB Bridge or CY8CKIT-002 Miniprog3
- CY3217-MiniProg1 Programmer Kit or CY8CKIT-002 Miniprog3
- USB A to Mini-B Cable
- PC running Windows XP or later

### 7.3.2 Assembly

Figure 7-1 represents the hardware setup. The CY3280-20x66 UCC kit connects to the CY3280-SLM module through a 22x2_RA_Receptacle. Connect either MiniProg1/MiniProg3 or I2USB bridge/MiniProg3 to the ISSP header of the kit. The setup uses the MiniProg1/MiniProg3 for programming and I2USB bridge/MiniProg3 to send data to PC. They connect to the PC through a USB cable.

Figure 7-1. Hardware Setup Block Diagram

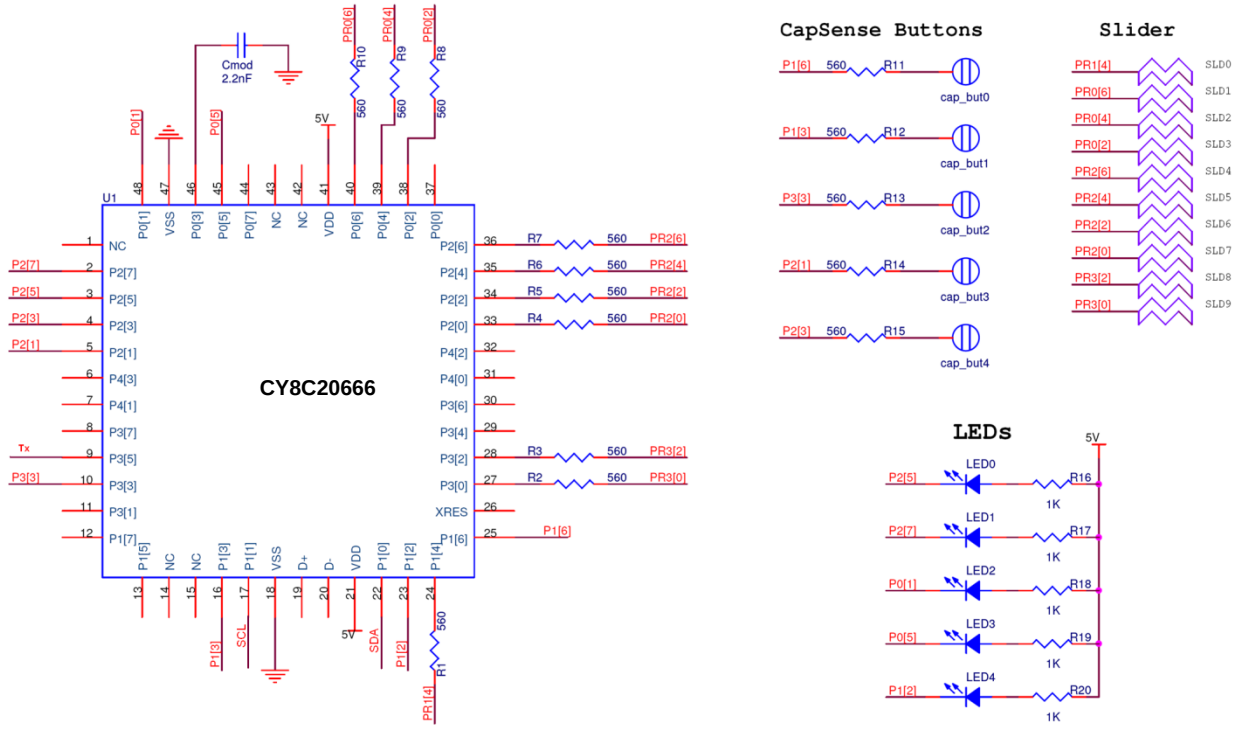


### 7.3.3 Setting Up the Board

Make the following hardware connections:

- Connect header J1 of the CY3280-SLM daughter card to the 22x2_RA_Receptacle (connector P2) on the CY3280-20x66 UCC board.
- Place a jumper on header J7 to short the pins VCC and VCC_PROG of the UCC board. This setting allows you to power the CapSense controller from the ISSP connector.
- Place a jumper on header J4 to short the pins XRES and XRES/INT (pins 1 and 2) of the UCC board. This setting routes the XRES pin of the CapSense controller to pin 3 of ISSP connector J3.
- Place a jumper on header J2 to short the pins GND and SHIELD (pins 2 and 3) of the CY3280-SLM board. This setting allows the board hatch pattern on the CY3280-SLM board to connect to ground.
- Connect the MiniProg1/Miniprog3 to the ISSP header on J3 of the UCC board. This connection is required only when the UCC is to be programmed with the code generated hex file. This connection has to be replaced with the I2USB bridge when the CapSense data is to be read in the Bridge Control Panel software.
- Use the USB A to Mini B cable to connect the other end of the MiniProg1/Miniprog3 (or the I2USB Bridge).

## 7.4 Schematic




The modulator capacitor ($C_{MOD}$) is a 2.2 nF capacitor connected to P0[3]. A 560 Ω resistor is connected in series with each CapSense button to reduce RF interference. LEDs are connected in active low configuration with 1 kΩ series resistors. In total there are five LEDs, five buttons, and one slider with 10 segments available.

Table 7-1. Pin Assignments for LEDs, Buttons, and Slider Segments

| LED | Button | Slider Segment |
|---|---|---|
| LED0 - P2[5] | BTN0 - P1[6] | SLD0 - P1[4], SLD1 - P0[6] |
| LED1 - P2[7] | BTN1 - P1[3] | SLD2 - P0[4], SLD3 - P0[2] |
| LED2 - P0[1] | BTN2 - P3[3] | SLD4 - P2[6], SLD5 - P2[4] |
| LED3 - P0[5] | BTN3 - P2[1] | SLD6 - P2[2], SLD7 - P2[0] |
| LED4 - P1[2] | BTN4 - P2[3] | SLD8 - P3[2], SLD9 - P3[0] |

# 7.5 Software Setup

## 7.5.1 Tools Required

- PSoC® Designer (version 5.2 or higher)
- PSoC Programmer (version 3.14 or higher)
- Bridge Control Panel

## 7.5.2 User Module List and Placement

The following table lists the user modules used in this code example and the hardware resources occupied by each user module.

| User Module | Placement |
|---|---|
| CSD | CapSense Block, Timer1 (default) |
| EzI2Cs | $I^2C$/SPI Block |

## 7.5.3 User Module Parameters, Global Resources

The `ReadMe.txt` file in the project describes the parameter settings for the user modules used in this code example. It also includes a list of the Global Resources.

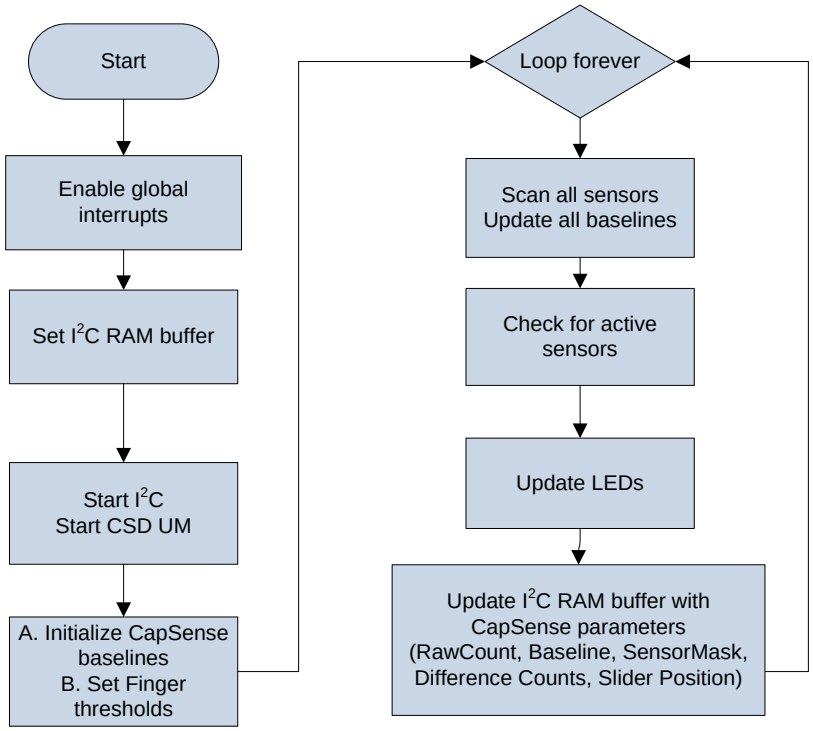
**Note** We tuned the code example for the 1.5 mm acrylic overlay that comes with the CY3280-SLM kit. To check the example with a thicker overlay, tune the project as described in this code example.

# 7.6 Operation

On reset, the firmware performs the following operations:

- Defines a structure MyI2C_Regs to store the button number, raw count, difference count, baseline, the centroid position, and the status of the CapSense button corresponding to the given button number.
- Enables global interrupt, and then starts the CSD UM.
- Starts the EzI2Cs UM, and then sets the structure MyI2C_Regs as the $I^2C$ RAM buffer.
- Performs the following operations in an infinite loop:
  - Scans all sensors continuously and updates the structure MyI2C_Regs with the raw count, difference count, baseline, slider centroid position, and the status of the requested CapSense button. The $I^2C$ master can request CapSense data of a particular button by writing the button number into first byte of the $I^2C$ buffer of EzI2Cs slave.
  - When the firmware detects a button press, it switches the corresponding LED to ON. It switches the LED to OFF when the button is released.
  - When slider is touched, the firmware turns the LED to ON indicating the touch position.

Figure 7-2. Functional Flow for Code Example 7




## 7.7 Running the Code Example

Program the board with the project, and then use this procedure to run the code example. For details on how to program the UCC board, refer to the Chapter 5 of CY3280-20x66 kit guide.

1. Use MiniProg1/Miniprog3 or any of the sources mentioned in the CY3280-20x66 UCC kit guide to power the board at 5 V.
2. Touch the linear slider on the CY3280-SLM module board.
   The corresponding LEDs on the CY3280-SLM board light up.
3. Touch a button.
   The corresponding LED on the CY3280-SLM module board lights up. Multiple buttons can be touched at the same time. The linear slider and buttons can also be used simultaneously.

## 7.8 Reading CapSense Data over I²C

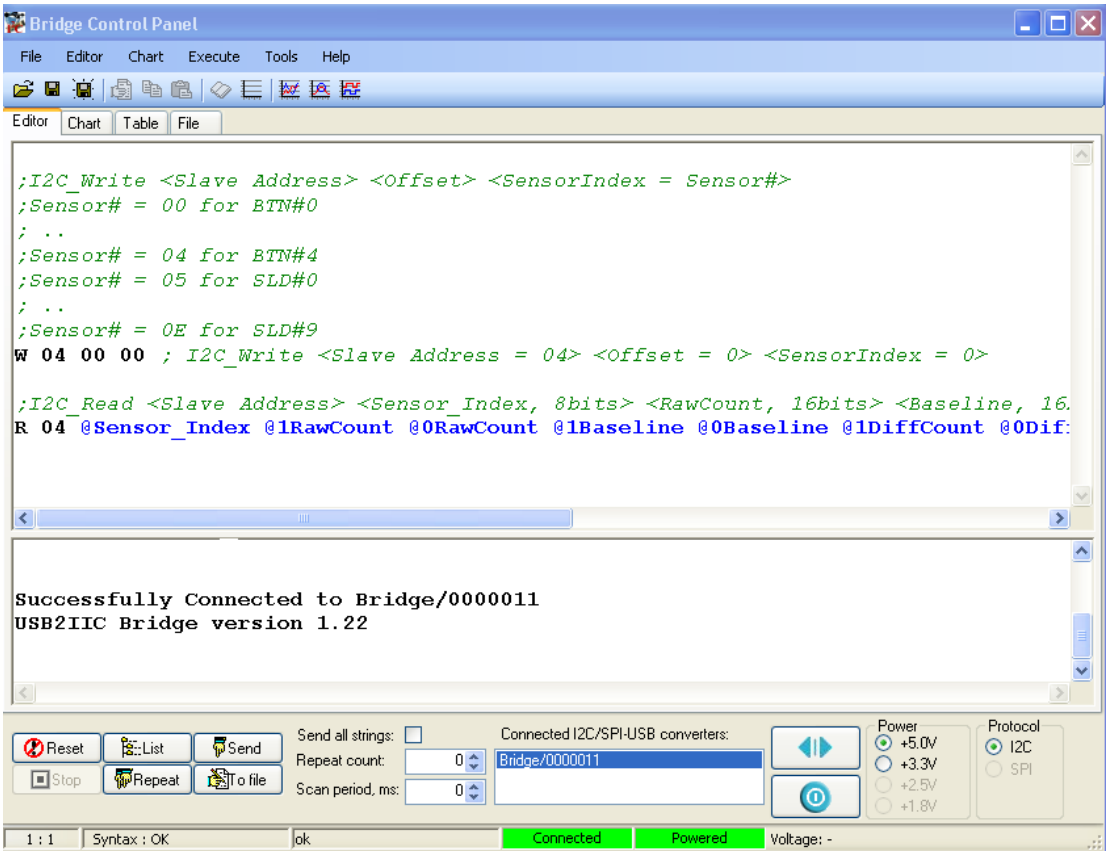
Use this procedure to run the code example and to read back the CapSense data on to Bridge Control Panel tool. For additional details on the Bridge Control Panel tool, see AN2397, “CapSense® Data Viewing Tools”.

### 7.8.1 Loading the Bridge Control Panel

1. Use the I2USB bridge/MiniProg3 and a USB A to Mini B cable to connect your computer to the ISSP connector J3 of the CY3280-20x66 UCC board.
2. On the computer desktop, select **Start** > **All Programs** > **Cypress** > **Bridge Control Panel** *(version)* > **Bridge Control Panel** *(version)*.
   The Bridge Control Panel is a component installed during the PSoC Programmer installation.
3. Select the device from the port selection window.
4. Power the CY3280-20x66 CapSense Controller board at 5 V.
5. From the Bridge Control Panel, select **File** > **Open**. Load the `BCP.iic` file from *BCP Configuration files* folder contained in the code example project folder.

6. Select **Charts** > **Variable Settings**. Load the `BCP.ini` file from *BCP Configuration files* folder contained in the code example project folder. Click **OK** to return to the main window.

Figure 7-3. Bridge Control Panel View




## 7.8.2 Reading Raw Count, Baseline, and Difference Count of BTN0

7. Send the I²C write instruction W 04 00 00 once.
8. Press the **Repeat** button to send the following I²C read instruction continuously:

```
R 04 @Sensor_Index @1RawCount @0RawCount @1Baseline @0Baseline @1DiffCount
@0DiffCount @ButtonStatus @SliderPosition
```

9. Go to the **Chart** tab to view raw count, baseline, and difference count of BTN0.

## 7.8.3 Reading Raw Count, Baseline, Difference Count, and Slider Position of SLD0

1. Send the I²C write instruction W 04 00 05 once.
2. Press the **Repeat** button to send the following I²C read instruction continuously:

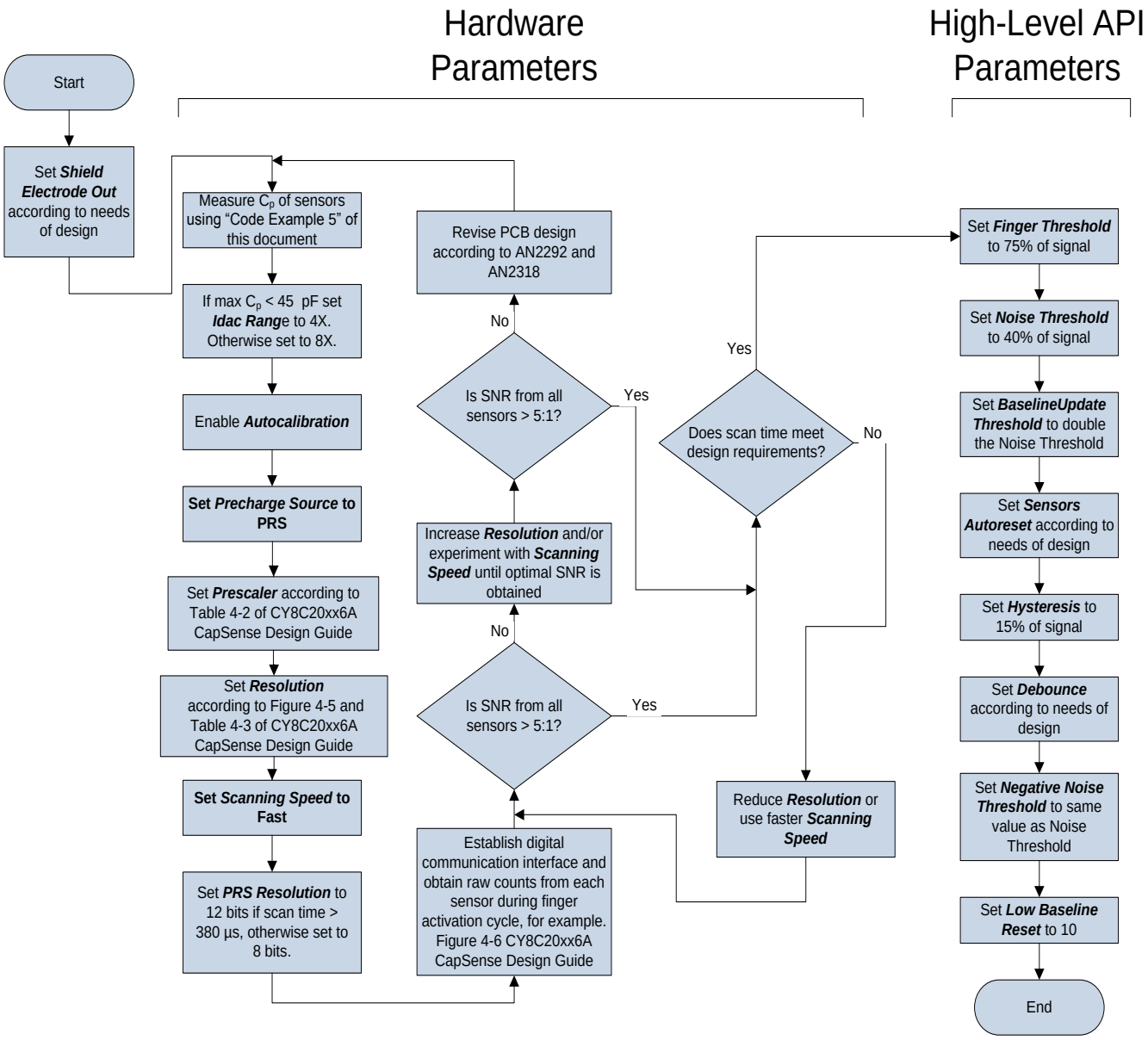
```
R 04 @Sensor_Index @1RawCount @0RawCount @1Baseline @0Baseline @1DiffCount
@0DiffCount @ButtonStatus @SliderPosition
```

3. Go to the **Chart** tab to view raw count, baseline, difference count, and slider position of SLD0.

## 7.9 Tuning the CSD User Module

In this section, we tune the user module parameters using the procedure outlined by the following flowchart. For more details, see the CY8C20xx6A CapSense Design Guide.

Figure 7-4. Flow Chart to Tune a CSD User Module




### 7.9.1 Configure CSD UM Settings

Use the PSoC Designer tool to input the CSD UM Parameter Settings as described below:

1. A shield electrode is not required. Therefore, set the **ShieldElectrodeOut** parameter to **None**.
2. Table 7-2 lists the sensor $C_P$ using the method described in *Code Example 3: Measuring Absolute Sensor Capacitance with a CY8C20xx6A CapSense Controller*. Since the $C_P$ of all sensors is less than 45 pF, set the IDAC range to 4X.

Table 7-2. $C_P$ of All the Sensors

| Sensor No. | Pin | $C_P$ |
|---|---|---|
| BTN0 | P1[6] | 16 pF |
| BTN1 | P1[3] | 14 pF |
| BTN2 | P3[3] | 13 pF |
| BTN3 | P2[1] | 12 pF |
| BTN4 | P2[3] | 11 pF |
| SLD0 | P1[4] | 14 pF |
| SLD1 | P0[6] | 15 pF |
| SLD2 | P0[4] | 14 pF |
| SLD3 | P0[2] | 14 pF |
| SLD4 | P2[6] | 13 pF |
| SLD5 | P2[4] | 12 pF |
| SLD6 | P2[2] | 12 pF |
| SLD7 | P2[0] | 12 pF |
| SLD8 | P3[2] | 11 pF |
| SLD9 | P3[0] | 11 pF |

3. Enable the **AutoCalibration** parameter.
4. Set the **Precharge** source to **PRS**.
   The PRS source provides spread-spectrum operation and ensures good immunity from external noise sources.
5. Set the **Prescaler** according to the highest $C_P$ amongst all the sensors. Since the maximum $C_P$ in our example is 16 pF, set the prescaler to 4.
   For more information, refer to the table “Prescaler Setting Based on Precharge Source, IMO, and CP” found in the CSD UM datasheet.
6. If a project has multiple sensors with different $C_P$, set the **Resolution** according to the sensor with highest $C_P$. In our example, the highest $C_P$ is 16 pF and Cf is greater than 0.2 pF. Referring to the figure, “Finger Capacitance (CF) Based on Overlay Thickness and Circular Sensor Diameter” and the table “Resolution Setting Based on Finger Capacitance and CP” both found in the CSD UM datasheet, set the resolution to 13.
7. Set the **Scan** speed to **Fast**.
8. The scan time is greater than 380 µs. Referring to the table “Scan Time for a Single Sensor in µs Based on Resolution and Scanning Speed” found in the CSD UM datasheet, the value is 720 µs so set the **PRS Resolution** to 8 bits.
9. Table 7-3 lists the noise (variation in raw counts without finger press) and signal (finger response) values for different sensors as observed on the **Bridge Control Panel**.

Table 7-3. Signals on All Sensors

| Sensor No. | Avg. Raw Counts without Finger | Noise | Signal | SNR (Approximated to the Nearest Integer) |
|---|---|---|---|---|
| BTN0 | 7072 | 24 | 260 | 11 |
| BTN1 | 7017 | 22 | 280 | 13 |
| BTN2 | 7103 | 24 | 280 | 12 |
| BTN3 | 7275 | 22 | 340 | 15 |
| BTN4 | 6953 | 20 | 360 | 18 |
| SLD0 | 6627 | 26 | 165 | 6 |
| SLD1 | 7071 | 24 | 150 | 6 |

| Sensor No. | Avg. Raw Counts without Finger | Noise | Signal | SNR (Approximated to the Nearest Integer) |
|---|---|---|---|---|
| SLD2 | 6697 | 24 | 150 | 6 |
| SLD3 | 6505 | 24 | 145 | 6 |
| SLD4 | 6028 | 22 | 160 | 7 |
| SLD5 | 5840 | 28 | 155 | 5 |
| SLD6 | 5687 | 26 | 160 | 6 |
| SLD7 | 5598 | 24 | 150 | 6 |
| SLD8 | 5362 | 24 | 150 | 6 |
| SLD9 | 5110 | 30 | 160 | 5 |

10. Since the SNR for all sensors is greater than 5:1 and the scan time is low enough that the finger presses glow the **LEDs instantaneously, you can set the threshold parameters as described in** Table 7-4**.**
    We use the lowest signal value amongst all sensors as a basis to configure the thresholds. Thus, in our example the signal value is 145.

    Table 7-4. High Level Parameters of the CSD UM

| Threshold Parameters | Recommended Values | Value | UM Parameter Setting |
|---|---|---|---|
| Finger threshold | 75% of signal | 108.75 | 108 |
| Noise threshold | 40% of signal | 58 | 58 |
| Baseline Update threshold | 2* noise threshold | 116 | 116 |
| Negative noise threshold | noise threshold | 58 | 58 |
| Hysteresis | 15% of signal | 21.75 | 21 |

11. Since scan time requirements are met and no false positives reported, set the **Low Baseline Reset** parameter to 10. Set the **Debounce** to 3.

# Code Example 8 Tuning CSD with Feedback Resistor with a CY8C21x34/B CapSense Controller

## 8.1 Project Name

CE_8_TuningCSD_withCY8C21x34/B

## 8.2 Overview

This code example demonstrates how to tune the CSD User Module (UM) for CY8C21x34/B devices. To tune a CapSense UM, you must be able to view the CapSense data in a graphical format. This code example uses the EzI2Cs UM to transmit the required CapSense parameters over an $I^2C$ bus and an I2USB bridge. It uses the Bridge Control Panel software on a PC to view the data.

## 8.3 Hardware Setup

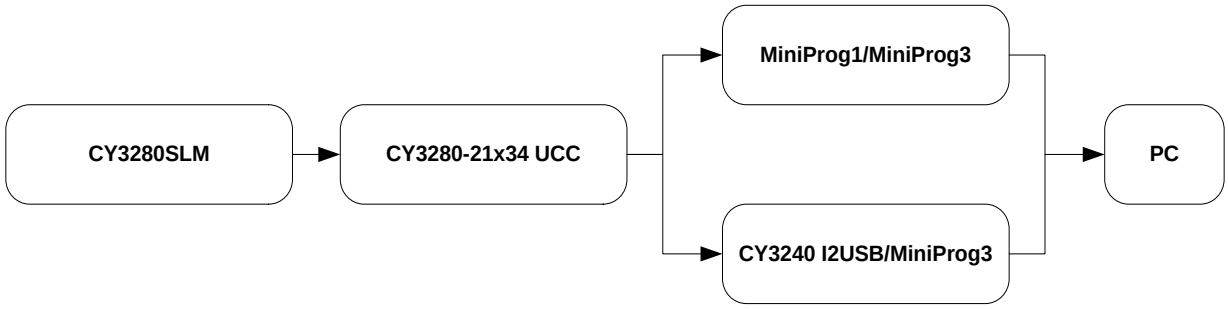
### 8.3.1 Requirements

- CY3280-21x34 Universal CapSense Controller board
- CY3280-SLM Universal CapSense Linear Slider Module
- CY3240-I2USB Bridge or CY8CKIT-002 Miniprog3
- CY3217-MiniProg1 Programmer Kit or CY8CKIT-002 Miniprog3
- USB A to Mini-B cable
- PC running Windows XP or later

### 8.3.2 Assembly

Figure 8-1 represents the hardware setup. The CY3280-21x34 UCC kit connects to the CY3280-SLM module through a 22x2_RA_Receptacle. Connect either MiniProg1/Miniprog3 or I2USB bridge to the ISSP header of the kit. The setup uses the MiniProg1/Miniprog3 for programming and the I2USB bridge/Miniprog3 to send data to the PC. They connect to the PC through a USB cable.

Figure 8-1. Hardware Setup Block Diagram

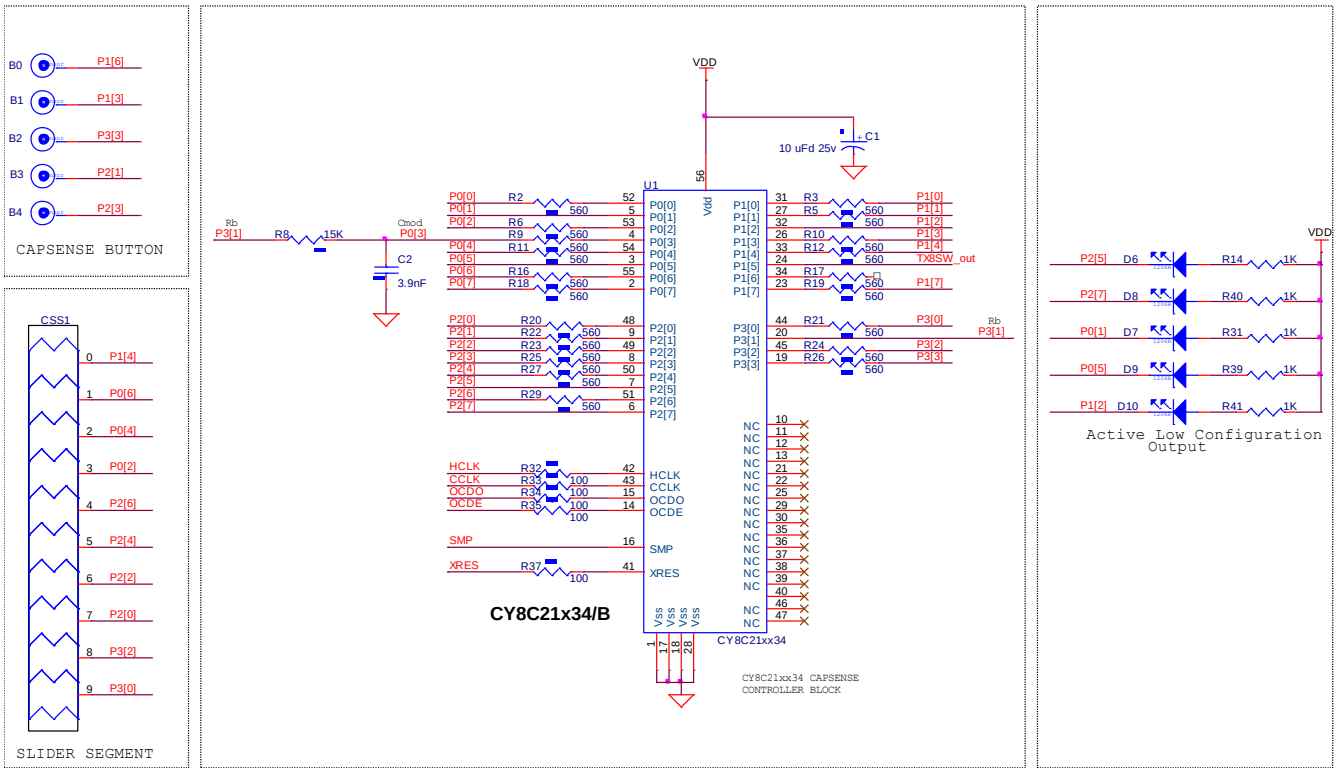



### 8.3.3 Setting Up the Board

Make the following hardware connections:

- Connect header J1 of the CY3280-SLM daughter card to the 22x2_RA_Receptacle (connector P2) on the CY3280-21x34 UCC board.
- Place a jumper on header J7 to short the pins 5 V and VCC_PROG of the UCC board. This setting allows you to power the CapSense controller from the ISSP connector.
- Place a jumper on header J4 to short the pins XRES and XRES/INT (pins 1 and 2) of the UCC board. This setting routes the XRES pin of the CapSense controller to pin 3 of ISSP connector J3.
- Place jumper on header J2 to short the pins GND and SHIELD (pins 2 and 3) of the CY3280-SLM board. This setting allows the board hatch pattern on the CY3280-SLM board to connect to ground.
- Connect the MiniProg1/MiniProg3 to the ISSP header on J3 of the UCC board. This connection is required only when the UCC is to be programmed with the code generated hex file. This connection has to be replaced with the I2USB bridge// Miniprog3 when the CapSense data is to be read in the Bridge Control Panel software
- Use the USB A and Mini B cable to connect the other end of the MiniProg1 (or the I2USB bridge or MiniProg3).

## 8.4 Schematic




The modulator capacitor ($C_{MOD}$) is a 2.2 nF capacitor connected to P0[3]. A 560 Ω resistor is connected in series with each CapSense button to reduce RF interference. LEDs are connected in active low configuration with 1 kΩ series resistors. In total there are five LEDs, five buttons, and one slider with 10 segments available.

Table 8-1. Pin Assignments for LEDs, Buttons, and Slider Segments

| LED | Button | Slider Segment |
|---|---|---|
| LED0 - P2[5] | BTN0 - P1[6] | SLD0 - P1[4], SLD1 - P0[6] |
| LED1 - P2[7] | BTN1 - P1[3] | SLD2 - P0[4], SLD3 - P0[2] |
| LED2 - P0[1] | BTN2 - P3[3] | SLD4 - P2[6], SLD5 - P2[4] |
| LED3 - P0[5] | BTN3 - P2[1] | SLD6 - P2[2], SLD7 - P2[0] |
| LED4 - P1[2] | BTN4 - P2[3] | SLD8 - P3[2], SLD9 - P3[0] |

## 8.5 Software Setup

### 8.5.1 Tools Required

- PSoC® Designer (version 5.2 or higher)
- PSoC Programmer (version 3.13 or higher)
- Bridge Control Panel

### 8.5.2 User Module List and Placement

The following table lists the user modules used in this code example and the hardware resources occupied by each user module.

| User Module | Placement |
|---|---|
| CSD | CapSense Block, Timer1 (default) |
| EzI2Cs | $I^2C$/SPI Block |

### 8.5.3 User Module Parameter Parameters, Global Resources

The `ReadMe.txt` file in the project describes the parameter settings for the user modules used in this code example. It also includes a list of the Global Resources.

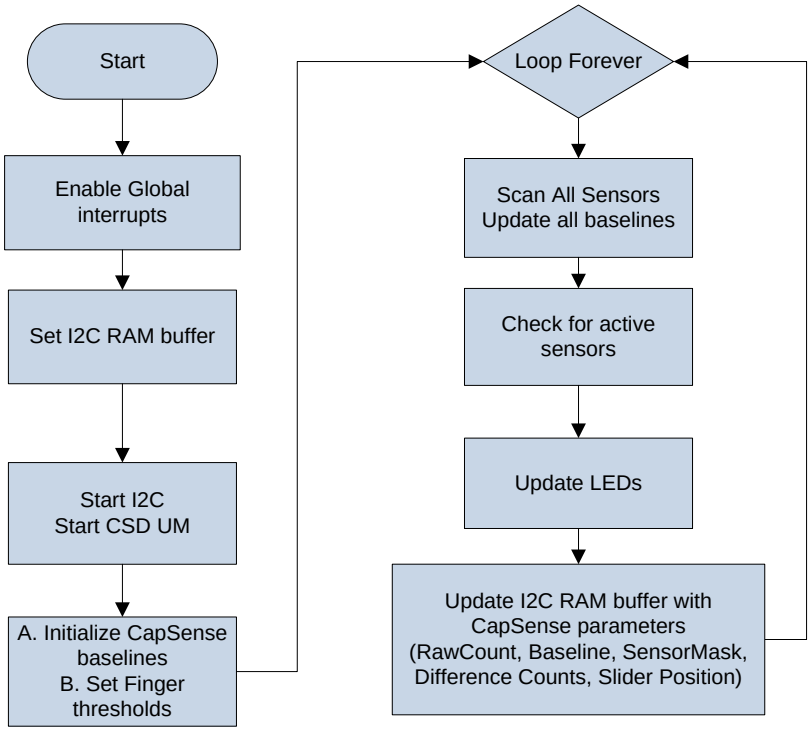
**Note** We tuned the code example for the 1.5 mm acrylic overlay that comes with the CY3280-SLM kit. To check the example with a thicker overlay, tune the project as described in this code example.

## 8.6 Operation

On reset, the firmware performs the following operations:

- Define a structure MyI2C_Regs to store the button number, raw count, difference count, baseline, the centroid position, and the status of the CapSense button corresponding to the given button number.
- Enables global interrupt, and starts the CSD UM.
- Stars the EzI2Cs UM, and sets the structure MyI2C_Regs as the $I^2C$ RAM buffer.
- Performs the following operations in an infinite loop:
  - Scans all sensors continuously and updates the structure MyI2C_Regs with the raw count, difference count, baseline, slider centroid position, and the status of the requested CapSense button. The $I^2C$ master can request CapSense data of a particular button by writing the button number into first byte of the $I^2C$ buffer of EzI2Cs slave.
  - When the firmware detects a button press, it switches the corresponding LED to ON. It switches the LED to OFF when the button is released.
  - When slider is touched, the firmware turns the LED to ON indicating the touch position.

Figure 8-2. Functional Flow for Code Example 8




## 8.7 Running the Code Example

Program the board with the project, and then use this procedure to run the code example. For details on how to program the UCC board, refer to chapter 5 of CY3280-21x34 kit guide.

1. Use MiniProg1/Miniprog3 or any of the sources mentioned in the CY3280-21x34 UCC kit guide to power the board at 5 V.
2. Touch the linear slider on the CY3280-SLM module board.
   The corresponding LEDs on the CY3280-SLM board light up.
3. Touch a button.
   The corresponding LED on the CY3280-SLM module board lights up. Multiple buttons can be touched at the same time. The linear slider and buttons can also be used simultaneously.

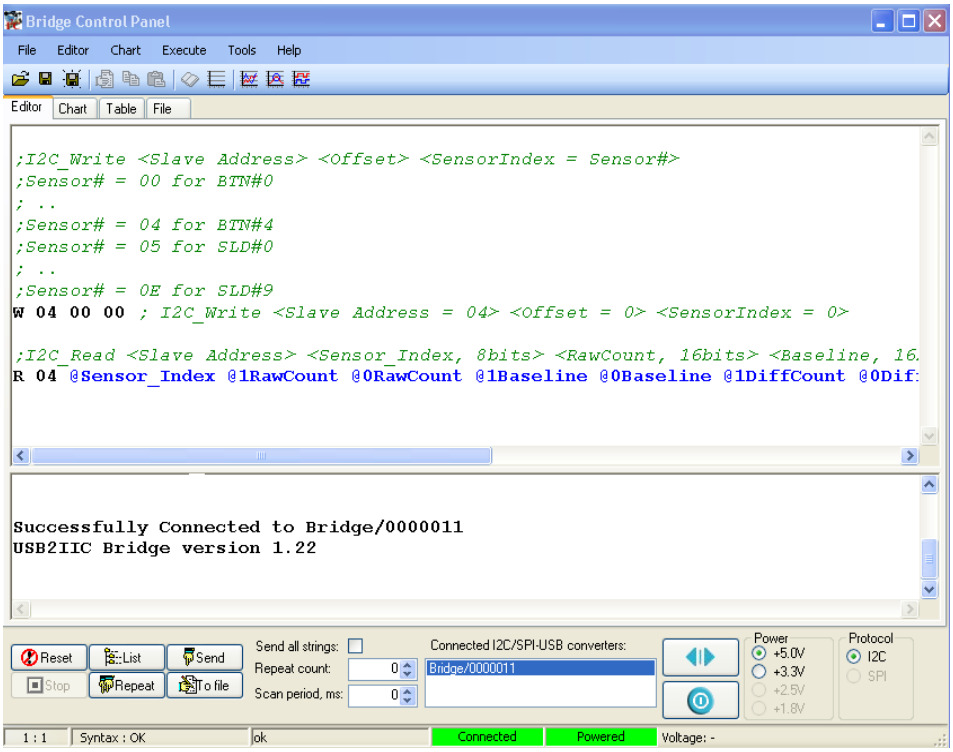
## 8.8 Reading CapSense Data over I²C

Use this procedure to run the code example and to read back the CapSense data on to Bridge Control Panel tool. For additional details on the Bridge Control Panel tool, see AN2397, “CapSense® Data Viewing Tools”.

### 8.8.1 Loading the Bridge Control Panel

1. Use the I2USB bridge/MiniProg3 and a USB A to Mini B cable to connect your computer to the ISSP connector J3 of the CY3280-21x34 Universal CapSense Controller board.
2. On the computer desktop, select **Start** > **All Programs** > **Cypress** > **Bridge Control Panel** *(version)* > **Bridge Control Panel** *(version)*.
   The Bridge Control Panel is a component installed during the PSoC Programmer installation.
3. Select the device from the port selection window.
4. Power the CY3280-21x34 CapSense Controller board at 5 V.
5. From the Bridge Control Panel, select **File** > **Open**. Load the `BCP.iic` file from *BCP Configuration files* folder contained in the code example project folder.

6. Select **Charts** > **Variable Settings**. Load the `BCP.ini` file from *BCP Configuration files* folder contained in the code example project folder. Click **OK** to return to the main window.




## 8.8.2 Reading Raw Count, Baseline, and Difference Count of BTN0

1. Send I$^2$C write instruction W 04 00 00 once.
2. Press the **Repeat** button to send the following I$^2$C read instruction continuously:

```
R 04 @Sensor_Index @1RawCount @0RawCount @1Baseline @0Baseline @1DiffCount
@0DiffCount @ButtonStatus @SliderPosition
```

3. Go to the **Chart** tab to view raw count, baseline, and difference count of BTN0.

## 8.8.3 Reading Raw Count, Baseline, Difference Count, and Slider Position of SLD0:

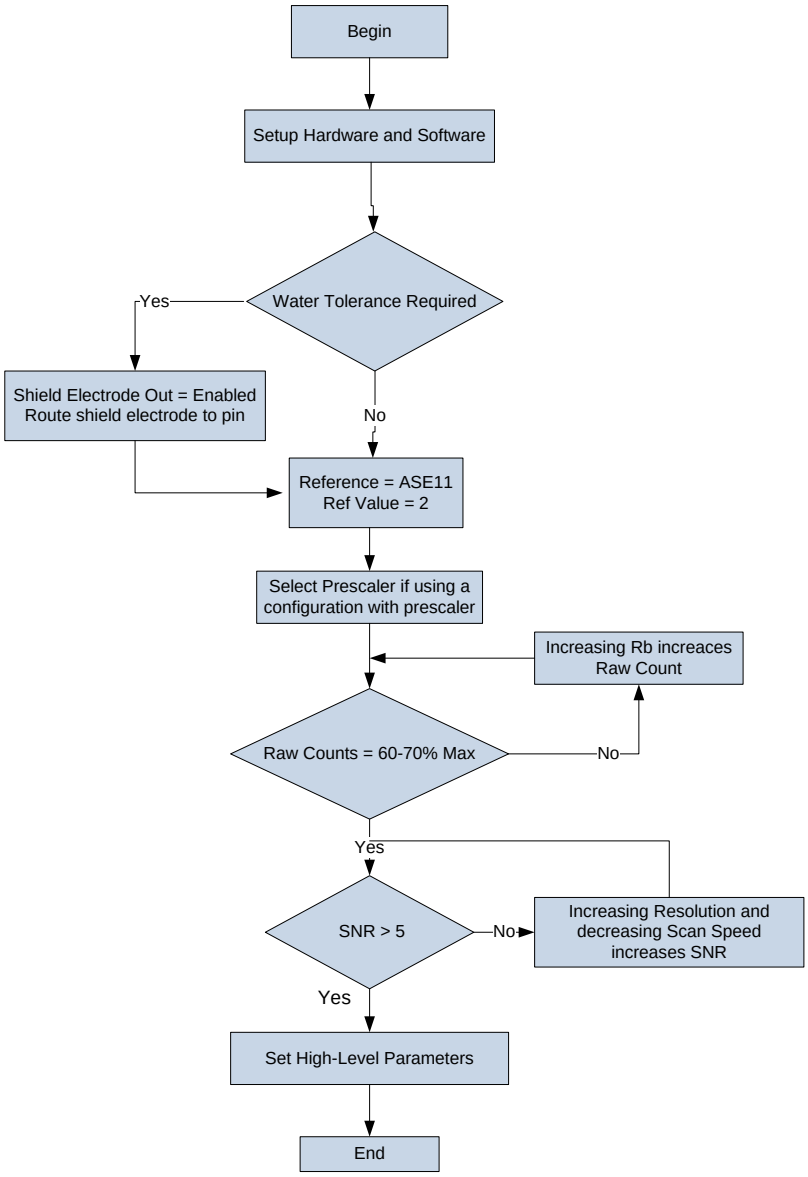
1. Send I$^2$C write instruction W 04 00 05 once.
2. Press the **Repeat** button to send the following I$^2$C read instruction continuously:

```
R 04 @Sensor_Index @1RawCount @0RawCount @1Baseline @0Baseline @1DiffCount
@0DiffCount @ButtonStatus @SliderPosition
```

3. Go to the **Chart** tab to view raw count, baseline, difference count, and slider position of SLD0.

## 8.9 Tuning the CSD User Module

In this section, we tune the user module parameters using the procedure outlined by the following flowchart. For more details, see the CY8C21x34/B CapSense design guide.




### 8.9.1 Configure CSD UM Settings

Use the PSoC Designer tool to input the settings to configure the CSD UM Parameter Settings as described below:

1. This code example uses Rb = 15 kΩ and $C_{MOD}$ = 10 nF. The CY3280-21x34 UCC board comes with a fixed value of Rb. For custom board designs, you can tune the user module by varying Rb.
   Refer to Section 4.2.3 of the CY8C21x34/B CapSense design Guide to know how to select the optimal value of Rb for proper sensitivity of the CapSense sensor.
2. A shield electrode is not required. Therefore, set the **ShieldElectrodeOut** parameter to **None**.

Table 8-2 lists the $C_P$ measured of the sensors using the method described in *Code Example 5: Measuring Absolute Sensor Capacitance with a CY8C21x34/B CapSense Controller*. Since the maximum CP in our example is 16 pF, set the prescaler to 4.

Table 8-2. $C_P$ of All Sensors

| Sensor | Pin | $C_P$ |
|---|---|---|
| BTN0 | P1[6] | 16 pF |
| BTN1 | P1[3] | 14 pF |
| BTN2 | P3[3] | 13 pF |
| BTN3 | P2[1] | 12 pF |
| BTN4 | P2[3] | 11 pF |
| SLD0 | P1[4] | 14 pF |
| SLD1 | P0[6] | 15 pF |
| SLD2 | P0[4] | 14 pF |
| SLD3 | P0[2] | 14 pF |
| SLD4 | P2[6] | 13 pF |
| SLD5 | P2[4] | 12 pF |
| SLD6 | P2[2] | 12 pF |
| SLD7 | P2[0] | 12 pF |
| SLD8 | P3[2] | 11 pF |
| SLD9 | P3[0] | 11 pF |

3. Set the **Precharge Source** to **PRS**.
   The PRS source provides spread-spectrum operation and ensures good immunity from external noise sources.
4. Set the **Prescaler** according to the highest $C_P$ amongst all the sensors. Since the maximum $C_P$ in our example is 16 pF, set the prescaler to 4.
   For more information, refer to the table “Prescaler Setting Based on Precharge Source, IMO, and CP” found in the CSD UM datasheet.
5. If a project has multiple sensors with different $C_P$, set the **Resolution** according to the sensor with highest $C_P$. In our example, the highest $C_P$ is 16 pF and Cf is greater than 0.2 pF. Referring to the figure, “Finger Capacitance (CF) Based on Overlay Thickness and Circular Sensor Diameter” and the table “Resolution Setting Based on Finger Capacitance and CP” both found in the CSD UM datasheet, set the resolution to 13.
6. Select the **Comparator** reference as ASE11.
   ASE11 is recommended for most applications.
7. Set the **Scan** speed to **Normal**.

Table 8-3 lists the noise (variation in raw counts without finger press) and signal (finger response) values for different sensors as observed on the **Bridge Control Panel**.

Table 8-3. Signal of All Sensors

| Sensor | Average Raw Counts without Finger | Signal | Noise | SNR(Approximated to the Nearest Integer) |
|---|---|---|---|---|
| BTN0 | 4451 | 179 | 20 | 9 |
| BTN1 | 3921 | 182 | 20 | 9 |
| BTN2 | 3419 | 184 | 21 | 9 |
| BTN3 | 3186 | 179 | 20 | 9 |
| BTN4 | 2652 | 191 | 21 | 9 |
| SLD0 | 3790 | 100 | 20 | 5 |
| SLD1 | 4432 | 101 | 20 | 5 |
| SLD2 | 4026 | 101 | 19 | 5 |
| SLD3 | 4011 | 103 | 19 | 5 |
| SLD4 | 3739 | 107 | 20 | 5 |
| SLD5 | 3456 | 104 | 20 | 5 |
| SLD6 | 3310 | 110 | 21 | 5 |
| SLD7 | 3158 | 99 | 19 | 5 |
| SLD8 | 2900 | 101 | 19 | 5 |
| SLD9 | 2688 | 115 | 20 | 5 |

8. Since SNR for all sensors is greater than 5:1 and the scan time is low enough that the finger press can glow the LEDs instantaneously, you can set the threshold parameters as described in Table 8-4.
   We use the lowest signal value amongst all sensors as a basis to configure the thresholds. Thus, in our example the signal value is 99.

   Table 8-4. High-Level Parameters of the CSD UM

| Threshold Parameters | Recommended Values | Value | UM Parameter Setting |
|---|---|---|---|
| Finger Threshold | 75% of Signal | 75.3 | 74 |
| Noise Threshold | 40% of Signal | 39.6 | 40 |
| Baseline Update Threshold | 2 Noise Threshold | 80 | 80 |
| Negative Noise Threshold | Noise Threshold | 40 | 40 |
| Hysteresis | 15% of Signal | 14.9 | 15 |

9. Since scan time requirements are met and no false positives reported, set the **Low Baseline Reset** parameter to 10. Set the **Debounce** to 3.

# Code Example 9 Tuning a CSA_EMC with a CY8C20x34 CapSense Controller

## 9.1 Project Name

CE_9_TuningCSAEMC_withCY8C20x34

## 9.2 Overview

This code example demonstrates how to tune a CSA_EMC User Module (UM) for CY8C20x34 devices. To tune a CapSense UM, you must be able to view the CapSense data in a graphical format. This code example uses the EzI2Cs UM to transmit the required CapSense parameters over an $I^2C$ bus and an $I^2C$ to USB bridge. It uses the Bridge Control Panel software on a PC to view the data.

## 9.3 Hardware Setup

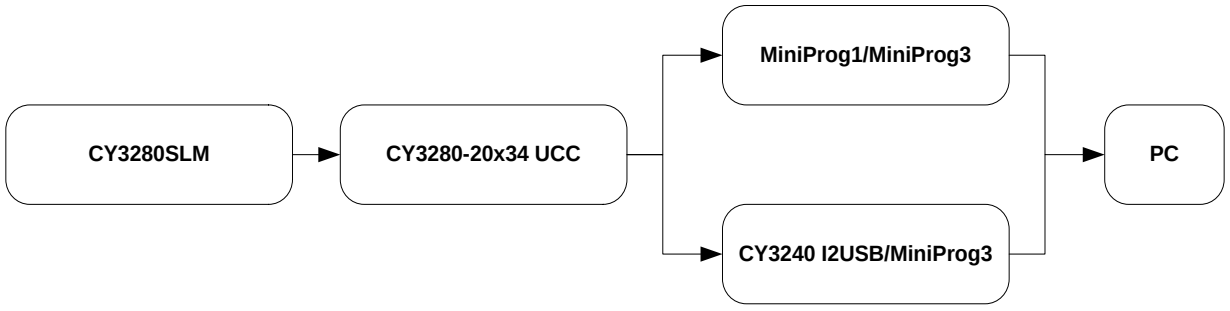
### 9.3.1 Related Hardware

- CY3280-20x34 Universal CapSense Controller board
- CY3280-SLM Universal CapSense Linear Slider Module
- CY3240-I2USB Bridge or CY8CKIT-002 Miniprog3
- CY3217-MiniProg1 Programmer Kit or CY8CKIT-002 Miniprog3
- USB A to Mini-B cable
- PC with a USB port

### 9.3.2 Assembly

Figure 9-1 represents the hardware setup. The CY3280-20x34 UCC kit connects to the CY3280-SLM module through a 22x2_RA_Receptacle. Connect either MiniProg1/Miniprog3 or I2USB bridge to the ISSP header of the kit. The setup uses the MiniProg1/Miniprog3 for programming and the I2USB bridge/Miniprog3 to send data to the PC. They connect to the PC through a USB cable.

Figure 9-1. Hardware Setup Block Diagram

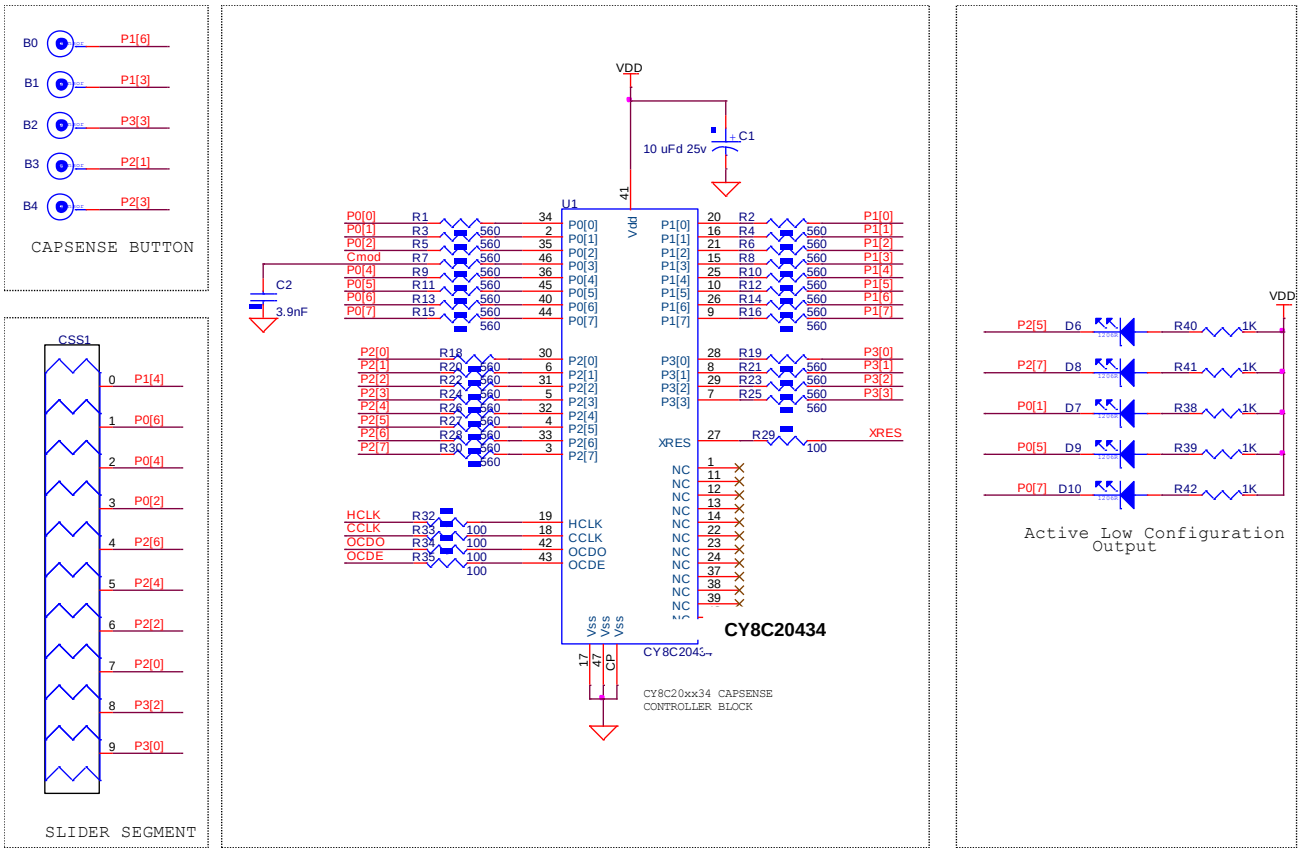


### 9.3.3 Setting Up the Board

Make the following hardware connections:

- Connect header J1 of the CY3280-SLM daughter card to the 22x2_RA_Receptacle (connector P2) on the CY3280-20x34 UCC board.
- Place a jumper on header J1 to short the pins $V_{CC}$ and 5V of the UCC board. This setting allows you to power the CapSense controller from the ISSP connector.
- Place a jumper on header J4 to short the pins XRES and XRES/INT (pins 1 and 2) of the UCC board. This setting routes the XRES pin of the CapSense controller to pin 3 of ISSP connector J3.
- Place a jumper on header J2 to short the pins GND and SHIELD (pins 2 and 3) of the CY3280-SLM board. This setting allows the board hatch pattern on the CY3280-SLM board to connect to ground.
- Connect the MiniProg1/MiniProg3 to the ISSP header on J3 of the UCC board. This connection is required only when the UCC is to be programmed with the code generated hex file. This connection has to be replaced with the I2USB bridge/MiniProg3 when the CapSense data is to be read in the Bridge Control Panel software
- Use the USB A and Mini B cable to connect the other end of the MiniProg1/Miniprog3 (or the I2USB Bridge).

## 9.4 Schematic




The modulator capacitor ($C_{MOD}$) is a 2.2 nF capacitor connected to P0[3]. A 560 Ω resistor is connected in series with each CapSense button to reduce RF interference. LEDs are connected in active low configuration with 1 kΩ series resistors. In total there are five LEDs, five buttons, and one slider with 10 segments available.

Table 9-1. Pin Assignments for LEDs, Buttons, and Slider Segments

| LED | Button | Slider Segment |
|---|---|---|
| LED0 - P2[5] | BTN0 - P1[6] | SLD0 - P1[4], SLD1 - P0[6] |
| LED1 - P2[7] | BTN1 - P1[3] | SLD2 - P0[4], SLD3 - P0[2] |
| LED2 - P0[1] | BTN2 - P3[3] | SLD4 - P2[6], SLD5 - P2[4] |
| LED3 - P0[5] | BTN3 - P2[1] | SLD6 - P2[2], SLD7 - P2[0] |
| LED4 - P1[2] | BTN4 - P2[3] | SLD8 - P3[2], SLD9 - P3[0] |

# 9.5 Software Setup

## 9.5.1 Tools Required

- PSoC® Designer (version 5.2 or higher)
- PSoC Programmer (version 3.13 or higher)
- Bridge Control Panel

## 9.5.2 User Module List

The following table lists the user modules used in this code example and the hardware resources occupied by each user module.

| User Module | Hardware Resources |
|---|---|
| CSA_EMC | CapSense Block |
| EzI2Cs | $I^2C$ Block |

## 9.5.3 User Module Parameter Parameters, Global Resources

The `ReadMe.txt` file in the project describes the parameter settings for the user modules used in this code example. It also includes a list of the Global Resources.

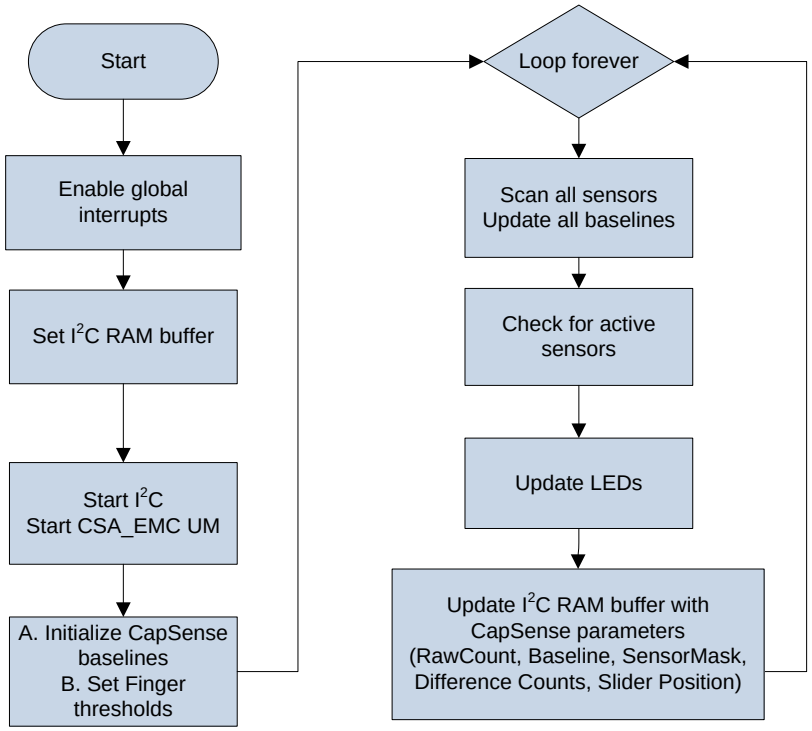
**Note** We tuned the code example for the 1.5 mm acrylic overlay that comes with the CY3280-SLM kit. To check the example with a thicker overlay, tune the project as described in this code example.

# 9.6 Operation

On reset, firmware performs the following operations:

- Define a structure MyI2C_Regs store the button number, raw count, difference count, baseline, the centroid position, and the status of the CapSense button corresponding to the given button number.
- Enables global interrupt, and then starts the CSA_EMC UM.
- Starts the EzI2Cs UM, and then sets the structure MyI2C_Regs as the $I^2C$ RAM buffer.
- Performs the following operations in an infinite loop:
  - Scans all sensors continuously and updates the structure MyI2C_Regs with the raw count, difference count, baseline, slider centroid position, and the status of the requested CapSense button. The $I^2C$ master can request CapSense data of a particular button by writing the button number into first byte of the $I^2C$ buffer of EzI2Cs slave.
  - When the firmware detects a button press, it switches the corresponding LED to ON. It switches the LED to OFF when the button is released.
  - When slider is touched, the firmware turns the LED to ON indicating the touch position.

Figure 9-2. Functional Flow for Code Example 9




## 9.7 Running the Code Example

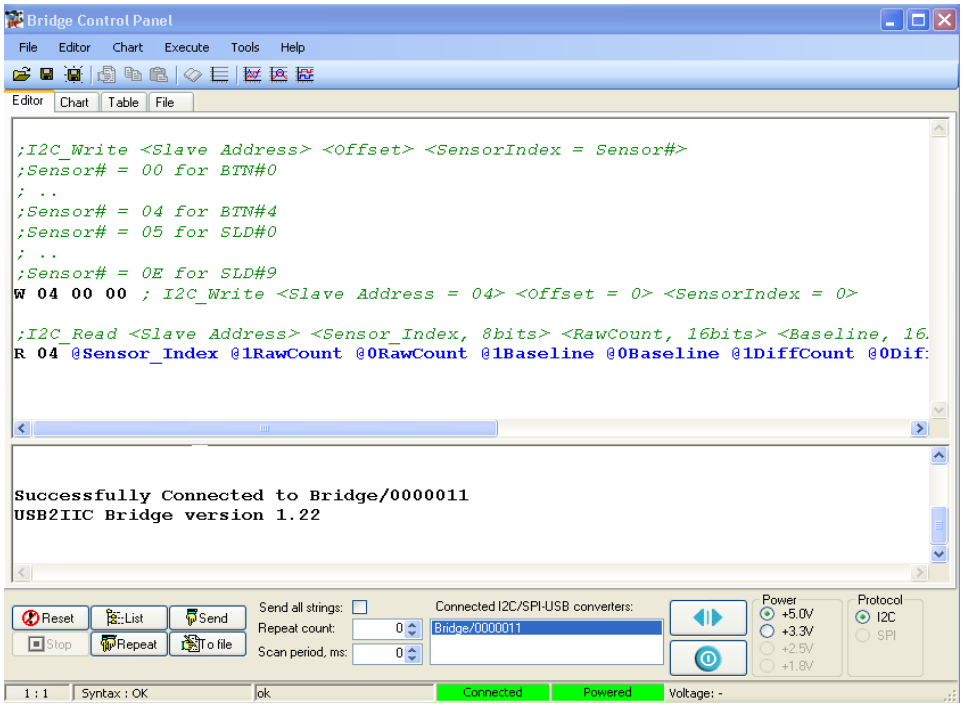
Program the board with the project, and then use this procedure to run the code example. For details on how to program the UCC board, refer to Chapter 5 of CY3280-20x34 kit guide.

1. Use MiniProg1/Miniprog3 or any of the sources mentioned in the CY3280-20x34 UCC kit guide to power the board at 5 V.
2. Touch the linear slider on the CY3280-SLM module board.
   The corresponding LEDs on the CY3280-SLM board light up.
3. Touch a button.
   The corresponding LED on the CY3280-SLM module board lights up. Multiple buttons can be touched at the same time. The linear slider and buttons can also be used simultaneously.

## 9.8 Reading CapSense Data over I²C

Use this procedure to run the code example and to read back the CapSense data on to Bridge Control Panel tool. For additional details on the Bridge Control Panel tool, see AN2397, “CapSense® Data Viewing Tools”.

1. Use the I2USB bridge/MiniProg3 and a USB A to Mini B cable to connect your computer to the ISSP connector J3 of the CY3280-20x34 Universal CapSense Controller board.
2. On the computer desktop, select **Start** > **All Programs** > **Cypress** > **Bridge Control Panel** *(version)* > **Bridge Control Panel** *(version)*.
   The Bridge Control Panel is a component installed during the PSoC Programmer installation.
3. Select the device from the port selection window.
4. Power the CY3280-20x34 CapSense Controller board at 5 V.
5. From the Bridge Control Panel, select **File** > **Open**. Load the *BCP.iic* file from the *BCP Configuration* files folder contained in the code example project folder.
6. Select **Charts** > **Variable Settings**. Load the `BCP.ini` file from *BCP Configuration files* folder contained in the code example project folder. Click **OK** to return to the main window.




### 9.8.1 Reading Raw Count, Baseline, and Difference Count of BTN0

1. Send I$^2$C write instruction W 04 00 00 once.
2. Press the **Repeat** button to send the following I$^2$C read instruction continuously:

```
R 04 @Sensor_Index @1RawCount @0RawCount @1Baseline @0Baseline @1DiffCount
@0DiffCount @ButtonStatus @SliderPosition
```

3. Go to the **Chart** tab to view raw count, baseline, and difference count of BTN0.

### 9.8.2 Reading Raw Count, Baseline, Difference Count, and Slider Position of SLD0

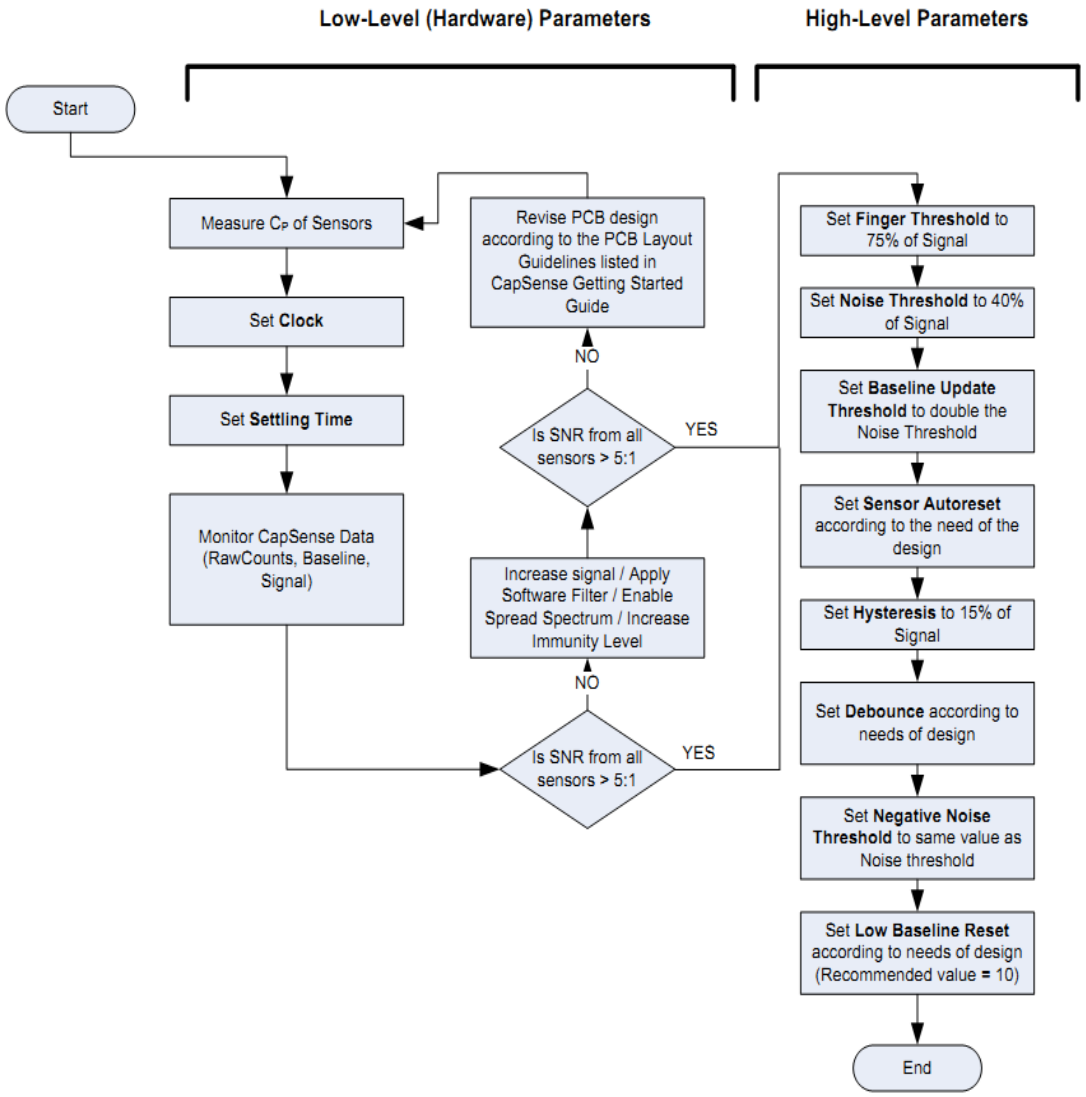
1. Send I$^2$C write instruction W 04 00 05 once.
2. Press the **Repeat** button to send the following I$^2$C read instruction continuously:

```
R 04 @Sensor_Index @1RawCount @0RawCount @1Baseline @0Baseline @1DiffCount
@0DiffCount @ButtonStatus @SliderPosition
```

3. Go to the **Chart** tab to view raw count, baseline, difference count, and slider position of SLD0.

## 9.9 Tuning the CSA_EMC User Module

In this section, we tune the user module parameters using the procedure outlined in the following flowchart. For more details, see the CY8C20x34 CapSense design guide.




### 9.9.1 Configure CSD UM Settings

Use the PSoC Designer tool to input the settings to configure the CSD UM Parameter Settings as described below:

1. Set CPU_CLK equal to SysClk/2 in the Global Resources window of the PSoC Designer tool. Configure the remaining parameters in the Global Resources window based on your design needs.
2. Table 9-2 lists the $C_P$ measured of all sensors using the method in *Code Example 6: Measuring Absolute Sensor Capacitance with a CY8C20x34 CapSense Controller*. Since the highest $C_P$ in our example is 15 pF and IMO is 12 MHz, set the Clock to IMO/4.

   For more information on the IMO, see Table 15 in the CY8C20x34 CapSense Design Guide.

Table 9-2. $C_P$ of All the Sensors

| Sensor | Pin | $C_P$ |
|---|---|---|
| BTN0 | P1[6] | 15 pF |
| BTN1 | P1[3] | 13 pF |
| BTN2 | P3[3] | 12 pF |
| BTN3 | P2[1] | 11 pF |
| BTN4 | P2[3] | 9 pF |
| SLD0 | P1[4] | 14 pF |
| SLD1 | P0[6] | 14 pF |
| SLD2 | P0[4] | 13 pF |
| SLD3 | P0[2] | 12 pF |
| SLD4 | P2[6] | 12 pF |
| SLD5 | P2[4] | 11 pF |
| SLD6 | P2[2] | 10 pF |
| SLD7 | P2[0] | 10 pF |
| SLD8 | P3[2] | 10 pF |
| SLD9 | P3[0] | 9 pF |

3. Calculate the Settling Time parameter with the equation:

$$Settling\ Time = \frac{5 \times C_{INT}}{\left(Clock \times C_P \times 25\left(\frac{1}{F_{CPU}}\right)\right)}$$

$C_{INT}$ = 3.9 nF

Clock = IMO/4 = 3 MHz

$C_P$ = 15 pF

$F_{CPU}$ = SysClk/2 = 12 MHz/2 = 6 MHz (SysClk is set to 12 MHz in this example)

Substituting these values in the above formula for settling time, we get Settling Time = 80

With the new setting in place, use the Bridge Control Panel to measure the SNR. Table 9-3 lists the values observed for this code example.

Table 9-3. Observed Signal of All the Sensors

| Sensor No. | Avg. Raw Counts without Finger | Noise | Signal | SNR (Approximated to the Nearest Integer) |
|---|---|---|---|---|
| BTN0 | 2000 | 14 | 130 | 9 |
| BTN1 | 2140 | 15 | 140 | 9 |
| BTN2 | 2210 | 18 | 140 | 8 |
| BTN3 | 2230 | 16 | 140 | 9 |
| BTN4 | 1930 | 20 | 150 | 7 |
| SLD0 | 1985 | 11 | 100 | 9 |
| SLD1 | 2310 | 14 | 95 | 7 |
| SLD2 | 2075 | 14 | 105 | 7 |
| SLD3 | 2160 | 13 | 95 | 7 |

| Sensor No. | Avg. Raw Counts without Finger | Noise | Signal | SNR (Approximated to the Nearest Integer) |
|---|---|---|---|---|
| SLD4 | 2500 | 14 | 100 | 7 |
| SLD5 | 2370 | 15 | 105 | 7 |
| SLD6 | 2200 | 16 | 105 | 7 |
| SLD7 | 2180 | 14 | 110 | 8 |
| SLD8 | 1960 | 14 | 110 | 8 |
| SLD9 | 2050 | 15 | 110 | 7 |

4. Since the lowest SNR = 7 (rounded to the nearest integer) is greater than 5:1, set the High Level Parameters as Described in Table 9-4.

   Table 9-4. High-Level Parameters of the UM

| Threshold Parameters | Recommended Values | Value | UM Parameter setting |
|---|---|---|---|
| Finger threshold | 75% of signal | 71.25 | 71 |
| Noise threshold | 40% of signal | 38 | 38 |
| Baseline Update threshold | 2* noise threshold | 76 | 76 |
| Negative noise threshold | noise threshold | 38 | 38 |
| Hysteresis | 15% of signal | 14.25 | 14 |

5. Leave the other parameters to their default values.

# Code Example 10 Power Consumption Optimization for One CapSense Button Using SmartSense on CY8C20xx6A

## 10.1 Project Name

CE_10_PowerOptimization_withCY8C20xx6A

## 10.2 Overview

The objective of this code example is to achieve a power consumption of less than 50 µA for one button using SmartSense. This code example incorporates a SmartSense and a 16-bit timer UMs. The device operates in Sleep mode for a certain time period to reduce the average current consumption. We use the 16-bit timer as a scan timer to wake up the device after a configured sleep time.

## 10.3 Hardware Setup

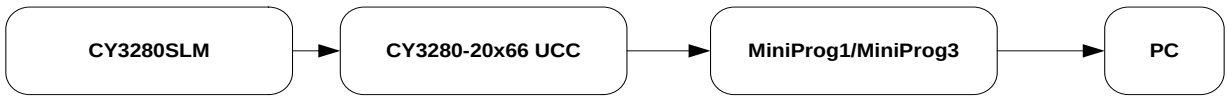
### 10.3.1 Related Hardware

- CY3280-20x66 Universal CapSense Controller board
- CY3280-BSM Universal CapSense Simple button Module
- CY3217-MiniProg1 Programmer Kit or CY8CKIT-002 Miniprog3
- USB A to Mini B cable
- Ammeter
- Power supply

### 10.3.2 Assembly

Figure 10-1 represents the hardware setup. The CY3280-20x66 UCC kit connects to the CY3280-BSM module through a 22x2_RA_Receptacle. The setup uses a MiniProg1/Miniprog3 connected to the ISSP header of the kit for programming.

Figure 10-1. Hardware Setup Block Diagram

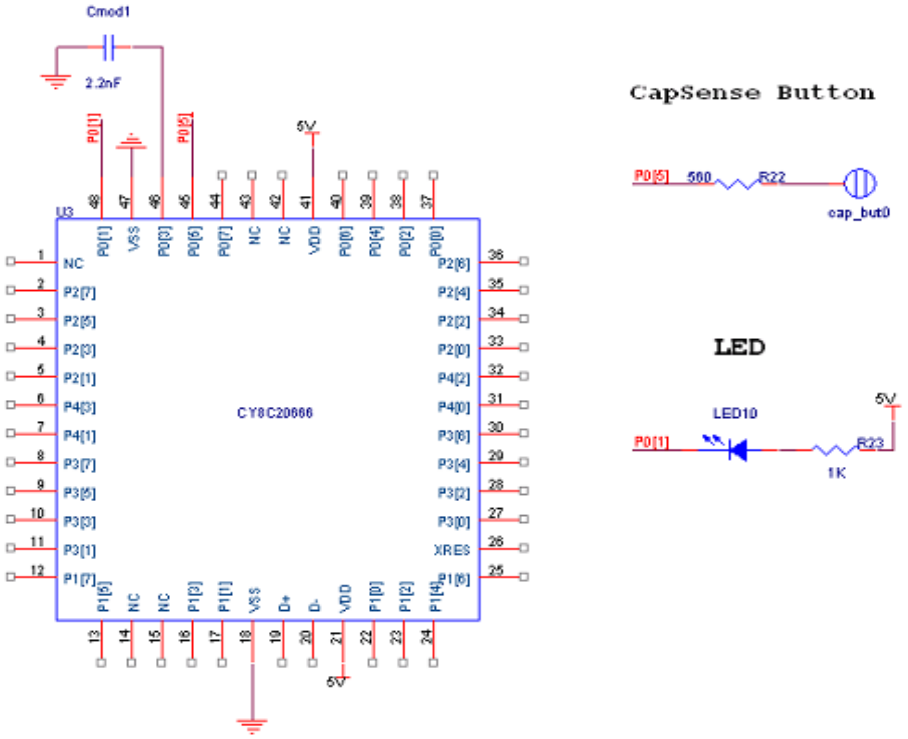


## 10.4 Setting Up the Board

Make the following hardware connections:

- Connect header J1 of the CY3280-BSM daughter card to the 22x2_RA_Receptacle (connector P2) on the CY3280-20x66 UCC board.
- Place a jumper on header J7 to short the pins $V_{CC}$ and $V_{CC\_PROG}$ of the UCC Board. This setting allows you to power the CapSense controller from the ISSP connector.
- Place a jumper on header J4 to short the pins XRES and XRES/INT (pins 1 and 2) of the UCC Board. This setting routes the XRES pin of the CapSense controller to pin 3 of ISSP connector J3.
- Place a jumper on header J2 to short the pins GND and SHIELD (pins 2 and 3) of the CY3280-BSM board. This setting allows the board hatch pattern on the CY3280-BSM board to connect to ground.
- Connect the MiniProg1/MiniProg3 to the CY3280-20x66 UCC on the J3 header.

## 10.5 Schematic




**Note** Remove power LEDs D1 and D2 from the CY3280-20x66 Universal CapSense Controller board while measuring power consumption of the device. Power consumption of the device with SmartSense UM depends on the SNR of system. When the board layout design follows the guidelines in the Getting Started with CapSense guide, then power consumption is minimized.

Table 10-1. Pin Assignments Used for Code Example 10

| Component | Pin |
|---|---|
| Button 0 | P0[5] |
| LED | P0[1] |
| $C_{MOD}$ | P0[3] |

## 10.6 Software Setup

### 10.6.1 Tools Required

- PSoC® Designer (version 5.2 or higher)
- PSoC Programmer (version 3.13 or higher)

### 10.6.2 User Module List

The following table lists the user modules used in this code example and the hardware resources occupied by each user module.

| User Module | Hardware Resources | User Module Name |
|---|---|---|
| SmartSense | CapSense Block, Timer1 (default) | SmartSense |
| Timer16 | Timer 0 | Scan_Timer |

### 10.6.3 User Module Parameter Parameters, Global Resources

The `ReadMe.txt` file in the project describes the parameter settings for the user modules used in this code example. It also includes a list of the Global Resources.

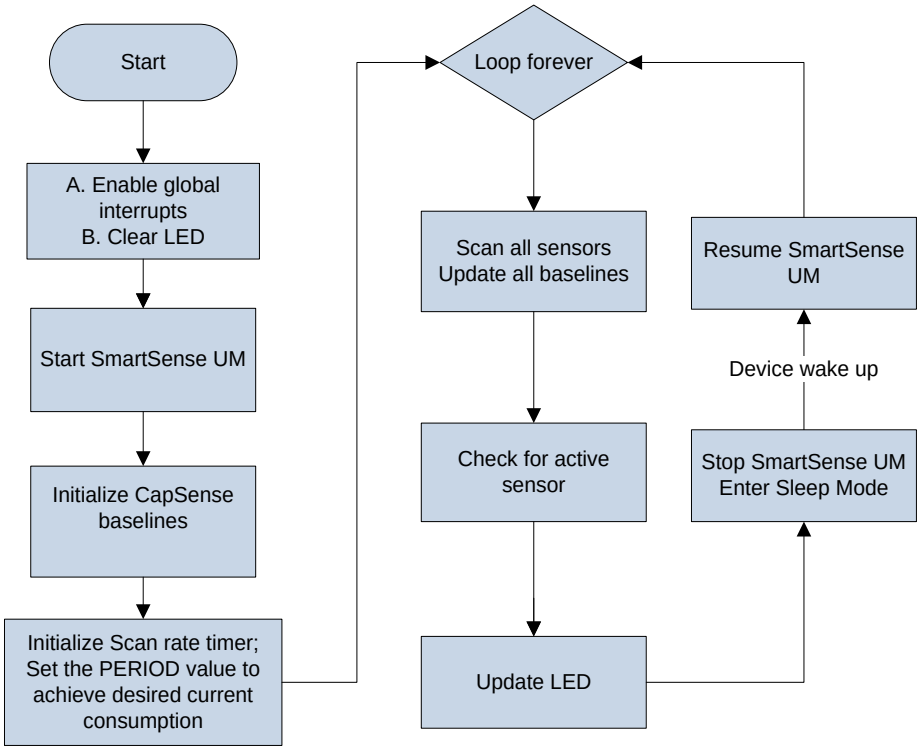
## 10.7 Operation

On reset, the program loads all hardware settings from the device configuration into the device and executes *main.c*. Then the firmware performs the following operations:

- Initializes the SmartSense UM, the ScanRate_Timer UM, and the LED.
- Enables global interrupt, and starts the SmartSense and ScanRate_Timer user modules.
- When the firmware detects a button press, it switches the LED to ON. It switches the LED to OFF when the button is released. This is done to verify the functionality of the device.
- The device enters Sleep mode, while the firmware continuously scans the sensor. The device wakes up from sleep on the ScanRate_Timer interrupt.

  To change the scan rate, change the macro SCANRATE_TIMER_VALUE. This controls the period of the ScanRate_Timer. Response improves as scan rate reduces.

Figure 10-2. Functional Flow of Code Example 10




## 10.8 Running the Code Example

Program the board with the project, and then use this procedure to run the code example. For details on how to program the UCC board, refer to Chapter 5 of the CY3280-20x66 kit guide.

1. To change the scan rate, edit the macro SCANRATE_TIMER_VALUE in the code example. This changes the parameter period of the ScanRate_Timer user module. Scan rate is variable by the relation:

   $$\text{Scan rate} = \text{Period} / \text{Clock select frequency}$$

   Where, $\text{Period} = \text{SCANRATE\_TIMER\_VALUE}$

2. Use an ammeter to measure the current for different scan rates at 250, 500, 750, and 1000 ms. The following graph and table show the power consumption at different scan rates for different operating voltages.

Figure 10-3. Average Current at Varying Scan Rates and Voltage

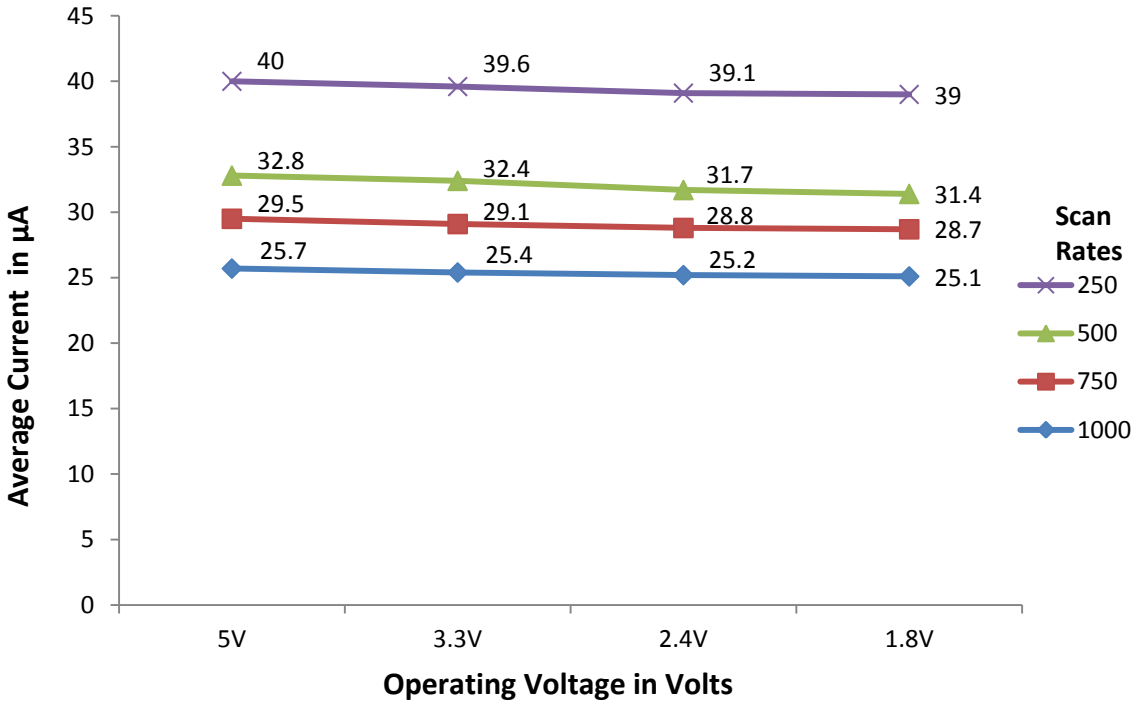



Table 10-2. Average Current at Varying Scan Rates and Voltage

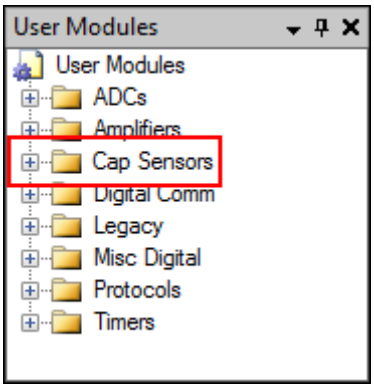
| Scan rate (ms) | Average Current (µA) | | | |
|---|---|---|---|---|
| | 5 V | 3.3 V | 2.4 V | 1.8 V |
| 1000 | 25.7 | 25.4 | 25.2 | 25.1 |
| 750 | 29.5 | 29.1 | 28.8 | 28.7 |
| 500 | 32.8 | 32.4 | 31.7 | 31.4 |
| 250 | 40 | 39.6 | 39.1 | 39 |

# Appendix

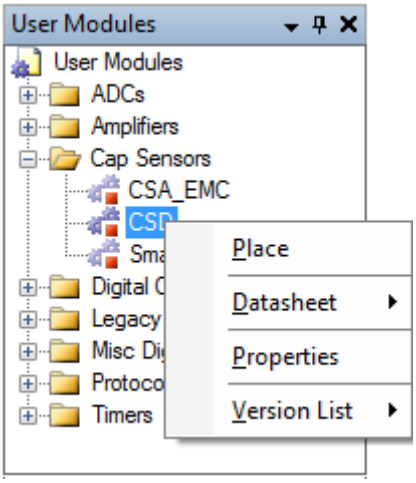
## Configuring the CSD User Module

To modify the projects available with this design guide, you must be able to configure the CapSense User Modules (UM) available with the PSoC® Designer software. Configuring UMs helps you to port any project from one device to other. All CapSense UMs use a similar process for configuration. In the following example, we use a software wizard to to configure the CSD:

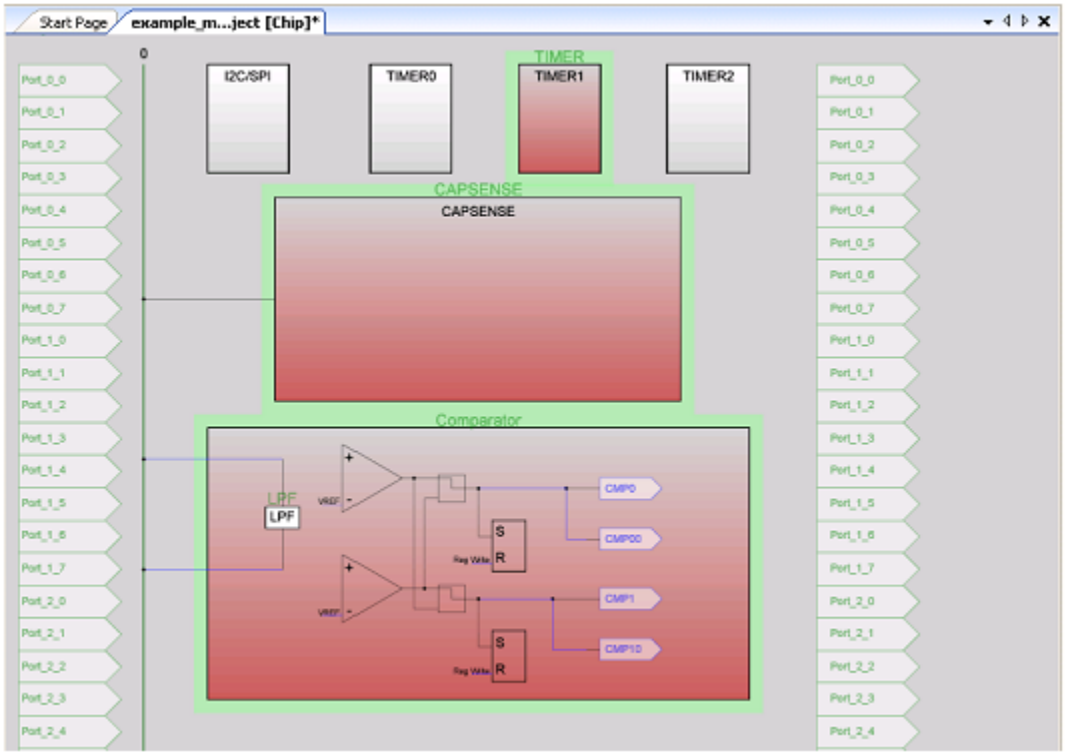
1. In the **User Modules** window of the PSoC Designer, expand the **Cap Sensors** folder.



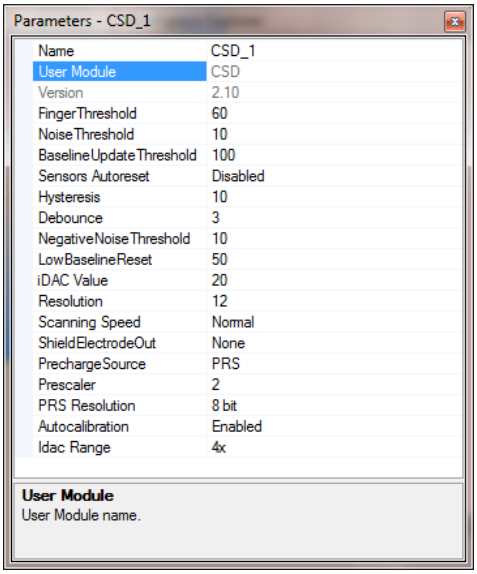

2. In the **Cap Sensors** folder, right-click **CSD** and then select **Place**.



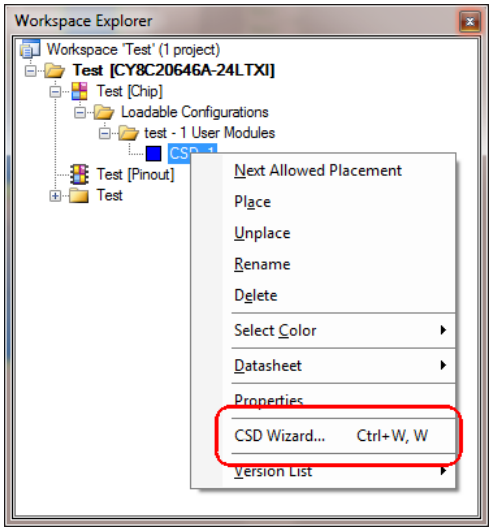
The Designer tool places the user module in the available block in the chip editor.



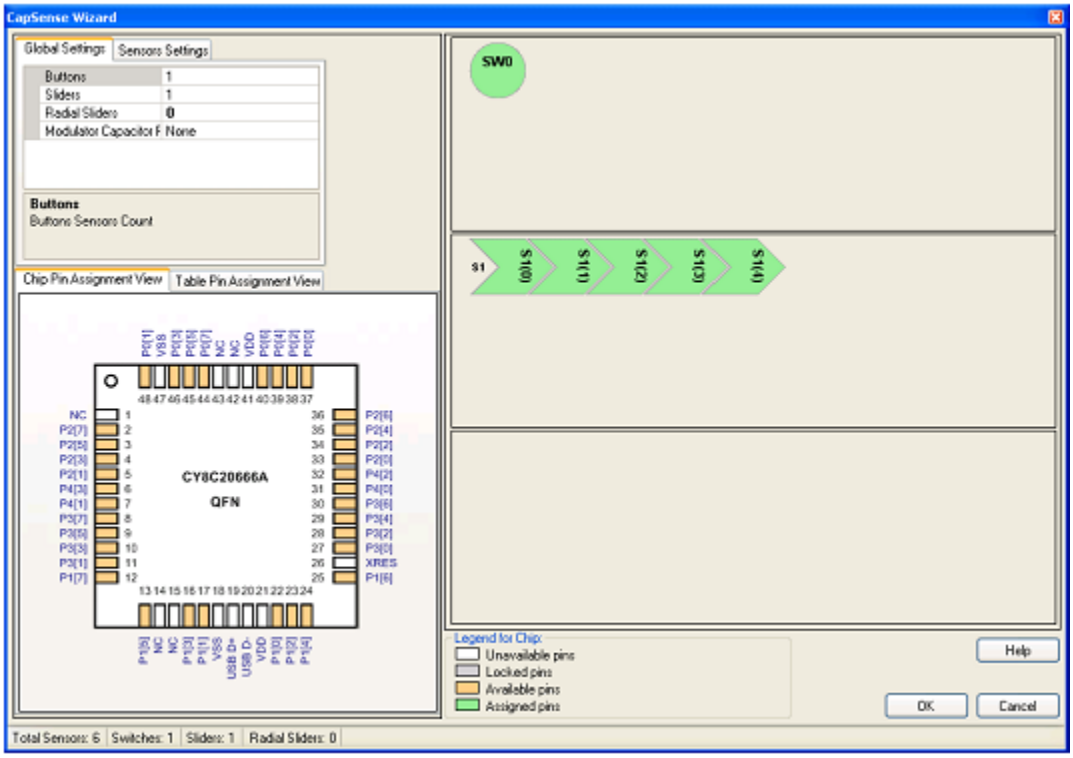

3. Configure the CSD_1 properties, as shown in the following screenshot:



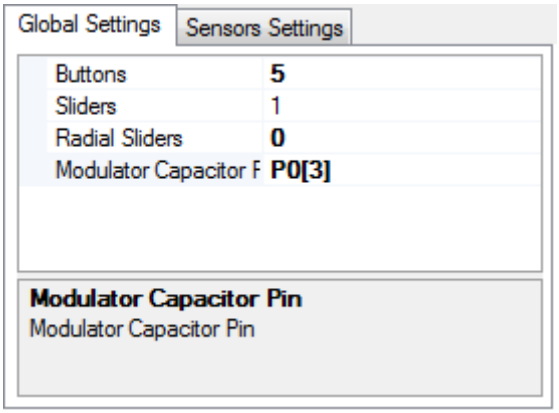
4. Right-click the CSD user module and then select **CSD Wizard**.
   We use the wizard to assign pins to the sensors.



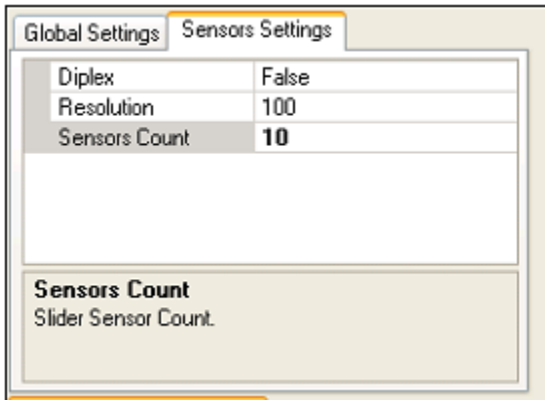

The CapSense Wizard appears.



5. Use the **CSD Wizard** window to configure the **Global Settings**.
   The settings in the screenshot are a project with five buttons and one slider. Use the **Global Settings** to configure the number of sensors required for your project.




6. Click the slider in the **CSD Wizard** to view sensor settings. Configure the **Sensor Settings**, as shown in the following figure.




7. To assign the sensor on a particular pin, click and drag from the sensor block to the required pin in the Pin Assignment window. Drag and drop SW0 to pin P1 [6]. Sensor pin assignment can be done in Table Pin Assignment View (Figure 10-5) or Chip Pin Assignment View (Figure 10-4).

Figure 10-4. Assign Sensors to Pins - Chip Pin Assignment View

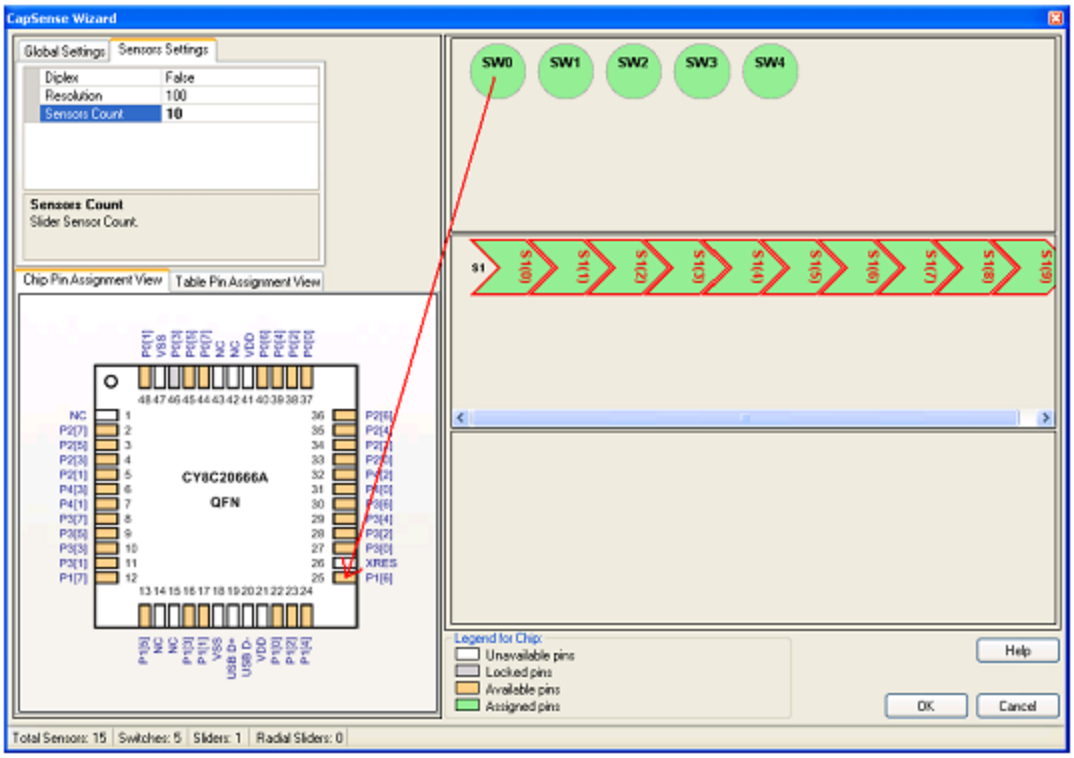
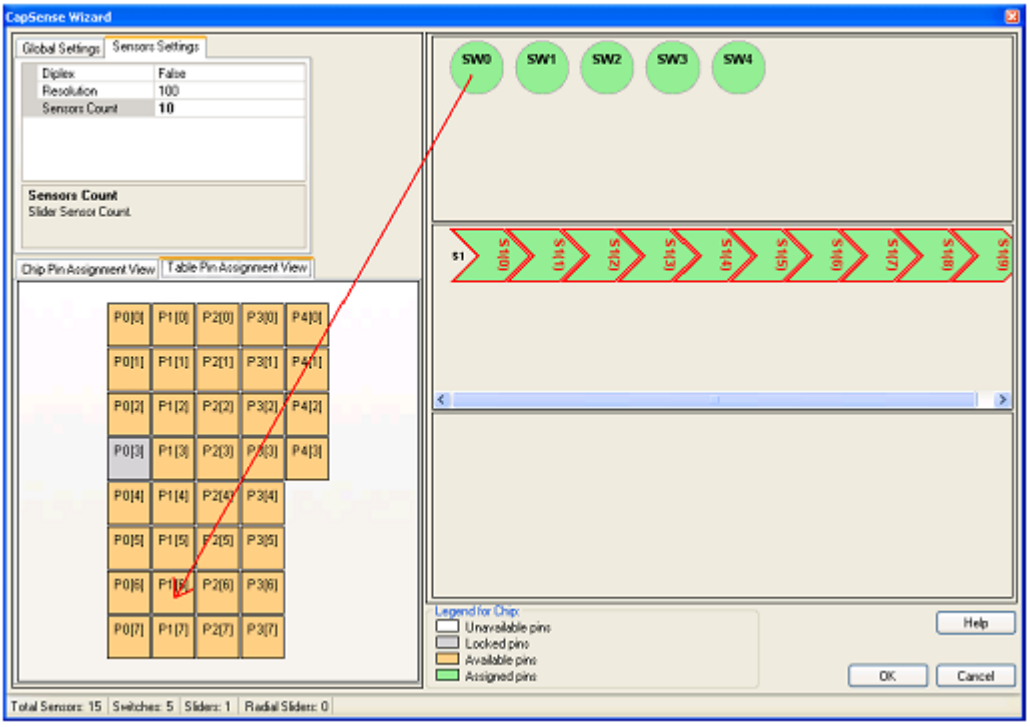



Figure 10-5. Assign Sensors to Pins - Table Pin Assignment View




8. Similarly, other sensors can be assigned to the device pins. Click **OK** to finish

# Migrating Code Examples to Other CapSense Devices

To select the right device for your end application, refer to the Getting Started with CapSense Guide. All the code examples provided with this document are for a particular device (target kit), as mentioned in the Table 1. To use the code examples with other CapSense devices, you need to migrate (or port) the project to the desired device.

Before porting, you should be aware of the CapSense User Modules supported by each CapSense device. *Table 5-1* in Getting Started with CapSense Guide shows the mapping of the CapSense product family to their supported CapSense UMs.

The following sections provide step-by-step procedures to port a project from one device to another.

## SmartSense

The SmartSense UM is supported by the CY8C20xx6A and CY8C21x34B device families. For more information, refer to *Code Example 1: Buttons and Sliders with SmartSense with a CY8C20xx6A CapSense Controller*. Currently, we do not provide any kit for the CY8C21x34B part, but you can use SmartSense with a CY8C21x34B device in your environment. Write to capsense@cypress.com to request a kit for CY8C21x34B.

## Communication with CapSense Controller

*Code Example 2: Host Communication Through I2C with a CY8C20xx6A CapSense Controller* and *Code Example 3: Data Transmission Through UART with a CY8C20xx6A CapSense Controller* demonstrate use of $I^2C$ and UART protocols to interface the CapSense controller CY8C20xx6A with the host processor. Porting these projects to other CapSense controllers is simple and does not need much modification.

**Port Code Example 3 to other devices**

1. Create a new project with your desired target device.
   The procedure to create a new project is available in the respective UCC kit guides.
2. Place the UART UM along with a CapSense UM (CSD, CSA, CSA_EMC, or SmartSense).
3. For reliable operation, tune the CapSense UM with the methods described in Code Example 7, Code Example 8, or Code Example 9 as appropriate. This step is not needed for SmartSense.
4. Configure the CapSense and UART UMs as desired.
   You may use the same settings as used in the Code Example 3 for the UART UM.
5. Open the existing *main.c* file in workspace explorer. Replace the existing *main.c* content with the content of the *main.c* file of the Code Example 3 project.
6. Add the files *display.c* and *display.h* from the Code Example 3 folder to your project folder location.
7. Complete *3.7 Running the Code Example* to run your project.
   Refer to *Plotting CapSense Data with MultiChart* to see the plotted data in MultiChart tool.

**Use $I^2C$ Communication with a CY8C24x94 Device**

$I^2C$ protocol is used for data transfer in the tuning code examples found in this document. Refer to the Code Example 7, Code Example 8, and Code Example 9 projects to use $I^2C$ with the specified devices. To use $I^2C$ communication with a CY8C24x94 device, follow the steps below.

1. Create a new project with CY8C24x94 as target device.
   Refer to the CY3280-24x94 UCC kit guide to learn how to create a new project with the kit.
2. Place the $I^2C$ UM along with a CSD UM to your project.
3. For reliable operation, tune the CSD UM with the method described in Code Example 7.
4. Configure the CapSense and $I^2C$ UMs as desired.
   You may use the same settings as used in Code Example 2 for the $I^2C$ UM.
5. Add the *main.c*, *display.c* and *display.h* files from the Code Example 2 project folder to your project.
6. Open the existing *main.c* file in Workspace Explorer. Replace the existing *main.c* content with the content of *main.c* file of the Code Example 2 project.
7. Add the files *display.c* and *display.h* from the Code Example 2 folder to your project folder location.
8. Complete *2.7 Running the Code Example* to run your project.
9. To see the CapSense data over $I^2C$, copy the *BCP Configuration Files* folder from the Code Example 1 project folder to your project location, and then complete *1.8 Reading CapSense Data over I2C*.

### Absolute Sensor Capacitance Measurement

*"CE #4 - Measuring Absolute Sensor Capacitance with a CY8C20xx6A CapSense Controller", "CE #5 - Measuring Absolute Sensor Capacitance with a CY8C21x34/B CapSense Controller", and "CE #6 - Measuring Absolute Sensor Capacitance with a CY8C20x34 CapSense Controller"* measures the absolute capacitance of sensors for CY8C20xx6A, CY8C21x34/B, and CY8C20x34 devices. Due to CSD UM architectural similarity, method to calculate sensor capacitance for CY8C21x34/B has to be used for CY8C24x94 device. To know more about the CSD UM in CY8C24x94, refer to CSD datasheet for CY8C24x94.

**Port Code Example 5 to a CY8C24x94 Device**

1. Create a new project with CY8C24x94 as target device.
   Refer to the CY3280-24x94 UCC kit guide to learn how to create a new project with the kit.
2. Place the CSD and UART UM to your project. Configure the UM as desired.
   You can use the same settings as used in Code Example 5 for your project.
3. Open the existing *main.c* file in workspace explorer. Replace the existing *main.c* content of *main.c* file of the Code Example 5 project.
4. Refer to *CY8C21x34/B_Calculsensitivity.xls* file to know how to calculate the sensitivity. Replace the sensitivity value in the *main.c* file with the calculated value.
5. Complete *6.7 Running the Code Example* to display the absolute capacitance on the HyperTerminal tool.
   Code Examples 7 through 9 describe how to tune the CapSense UMs is specific to the UMs and devices used in the projects and cannot be used with other devices.

## Related Documents

### Design Guides

- Getting Started with CapSense
- CY8C20xx6A CapSense Design Guide
- CY8C20x34 CapSense Design Guide
- CY8C21x34/B CapSense Design Guide

### Datasheets

- CSD Datasheet
- CSA_EMC Datasheet
- SmartSense Datasheet

### Application Notes

- AN2397 – CapSense Data Viewing Tools

### Kit Guides

- CY3280-20x66 Universal CapSense Controller Kit Guide
- CY3280-20x34 Universal CapSense Controller Kit Guide
- CY3280-21x34 Universal CapSense Controller Kit Guide

# Acronyms

| Acronym | Definition |
|---|---|
| AN | Application Note |
| API | Application Programming Interface |
| BSM | Simple Button Module |
| CE | Code Example |
| COM | Communication |
| CSA | CapSense Successive Approximation |
| CSD | CapSense Sigma Delta |
| EMC | Electromagnetic Compatibility |
| PC | Personal Computer |
| SLM | Linear Slider Module |
| UART | Universal Asynchronous Receiver/Transmitter |
| UM | User Module |
| USB | Universal Serial Bus |

# Document Revision History

| Revision | Issue Date | Origin of Change | Description of Change |
|---|---|---|---|
| ** | 07/10/2012 | ZINE | New document |
| *A | 08/17/2012 | ZINE | Added text in the Overview section to provide details about capsense@cypress.com. |
| *B | 05/16/2013 | ZINE | Corrected pin details in Section 4.3. |
| *C | 10/08/2013 | DCHE | Added MiniProg3 in the Hardware Setup section.<br>Modified Code Example 3 to view slider centroid position via UART.<br>Modified Code Example 4 and Code Example 5 to view sensor Cp via UART/I2C. |
| *D | 01/13/2015 | SLAN | Replaced “I2USB bridge”, “I2C-USB Bridge” with “CY3240-I2USB bridge” in all instances across the document. |
| *E | 07/15/2015 | DIMA/SLAN | Updated Measuring Absolute Sensor Capacitance with a CY8C20xx6A CapSense Controller:<br>Updated Operation:<br>Updated description (to clarify how Cp is measured).<br>Updated attached code examples to PSoC Designer 5.4 SP1.<br>Updated to new template.<br>Completing Sunset Review. |
| *F | 05/26/2017 | AESATP12 | Updated logo and copyright. |