CapSense® LowPower

1.0

Features

- 6x5 CapSense touchpad interface
- Low system power consumption using Deep-Sleep mode
- Watchdog Timer for periodically waking up from low power modes
- Sends touch coordinates to a host (PC) using UART

General Description

This example project is a PSoC Creator starter design. Any battery driven equipment requires very low system power consumption, while maintaining the required performance. The PSoC 4000 family supports a capacitive touch sensing known as CapSense® and two device low power modes: Sleep and Deep-Sleep. These low power modes enable PSoC 4 to achieve the required performance while operating at very low system power consumption. This example project demonstrates a low power CapSense system using PSoC 4.

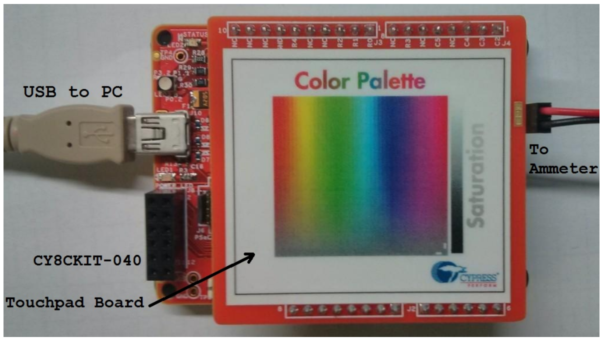
If you are new to CapSense, see PSoC 4 CapSense Design Guide.

Development Kit Configuration

The instructions below describe a stepwise procedure to be followed for testing this starter design with the CY8CKIT-040 Kit. This starter design can be validated on any other PSoC 4000 development platforms. For details, please refer to the Schematic and Pin Mapping section at the end of the document.

1. Use J9 on the CY8CKIT-040 to select 5.0 V.
2. Remove jumper J13 and connect an ammeter for device current measurement.
3. Plug the CapSense touchpad shield board on the baseboard as Figure 1 shows.
4. Connect the DVK to PC using a USB cable
5. Build the CapSense_LowPower project and the program the device with the project.
6. Open the serial port data viewer tool such as HyperTerminal and observe the X-Y coordinates as you touch the capacitive touchpad.

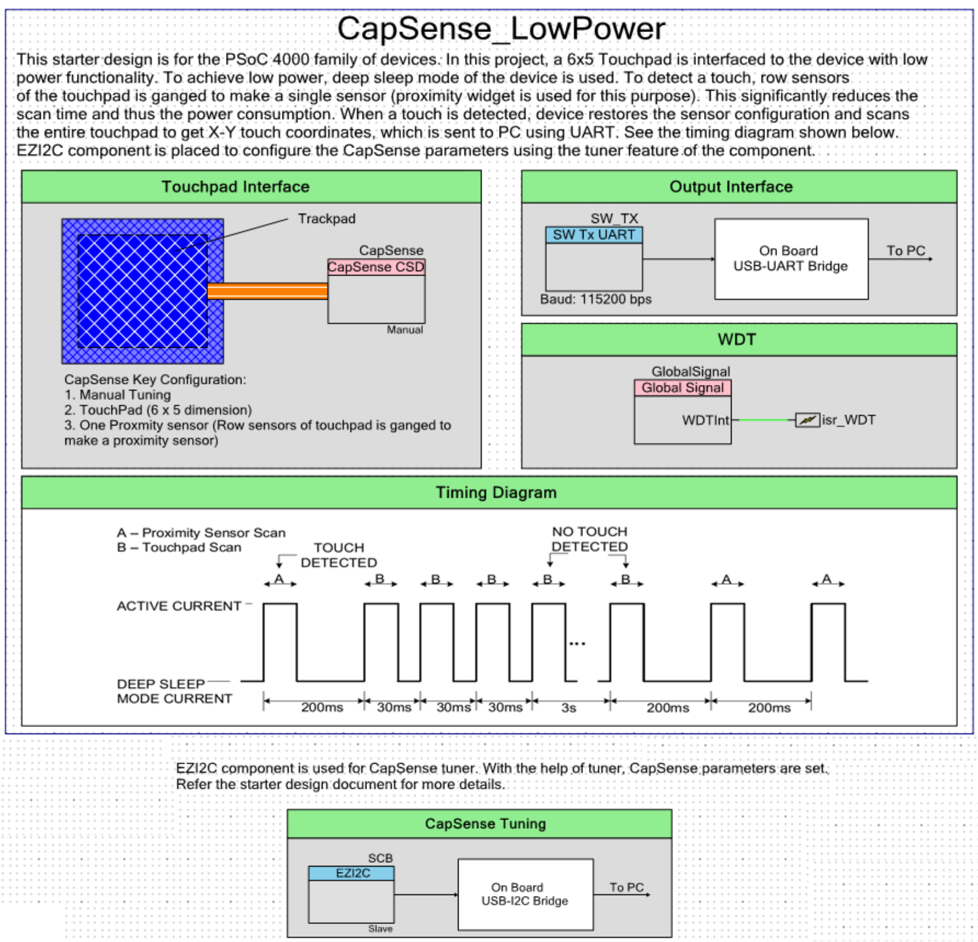
Figure 1. Test Setup



Project Configuration

This example project uses CapSense CSD, SW Tx UART, GlobalSignal and EZI2C Components. Figure 2 shows the Top Design schematic of this project.

Figure 2. Top Design



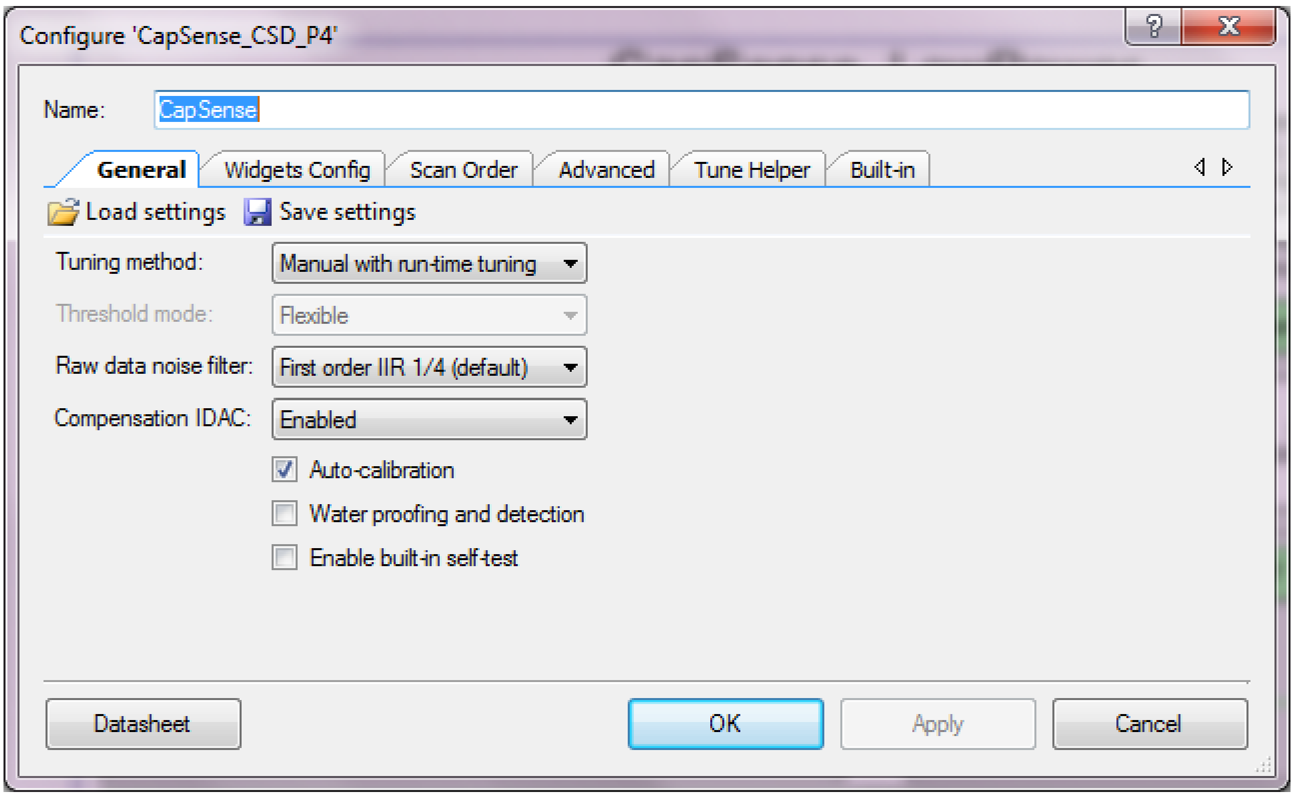
Component Configuration

CapSense

CapSense component is configured with a 6x5 touchpad and a proximity sensor widget. The project provided is already tuned for the capacitive sensors on the CY8CKIT-040 kit using the tuner feature of the component. If you are using a different PSoC 4000 board or the touchpad board, you need to re-tune the sensors. Refer to section- Tuning for more details.

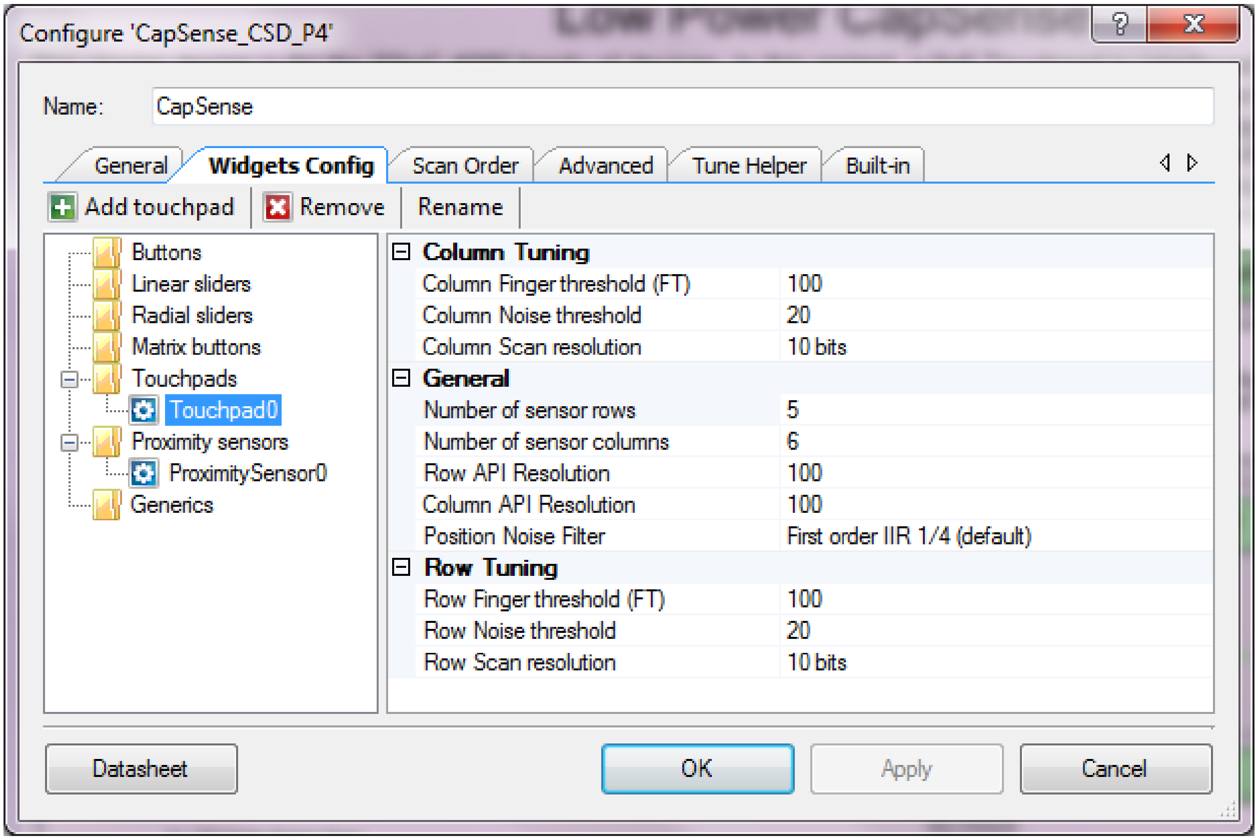
Figure 3 shows the CapSense component's widget configuration window. Manual tuning is used to achieve minimum scan resolution. This is important because lower scan resolution results in lower scan time, which in turn reduces the average power consumption by allowing the device to spend more time in Deep-Sleep mode.

Figure 3. General Settings



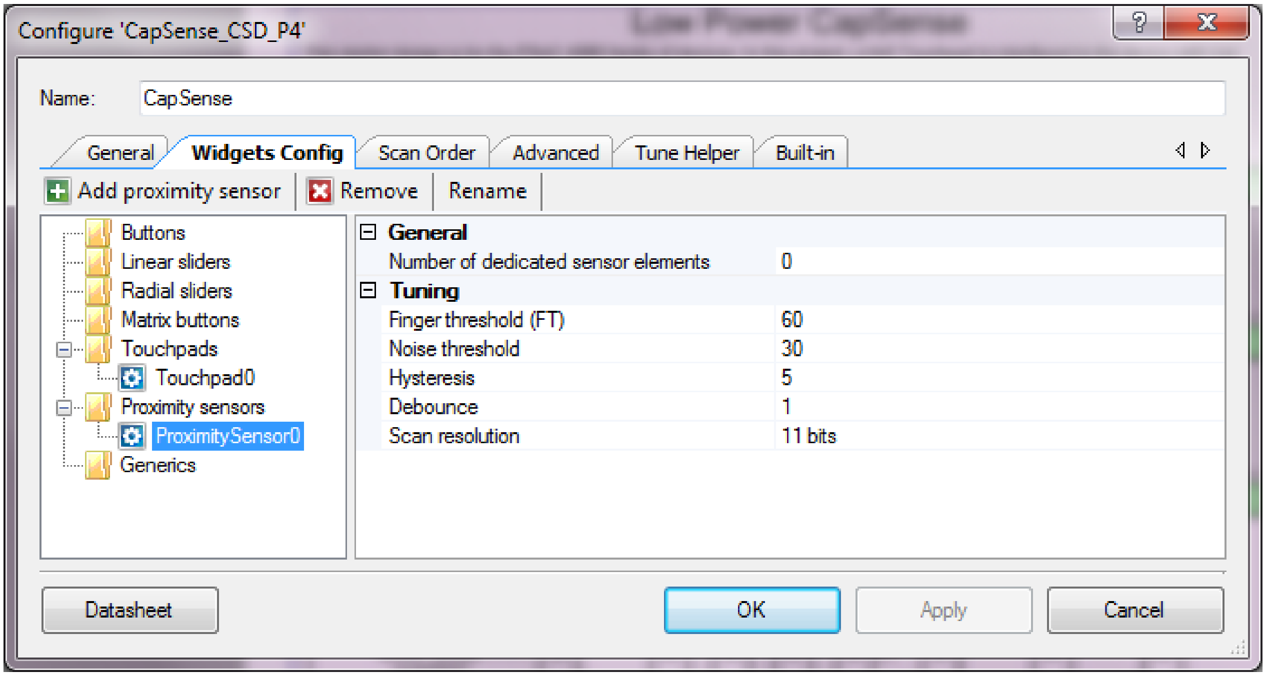
- Manual tuning to have full control over scan time
- Default filter (First order IIR 1/4) for raw counts
- Compensation IDAC enabled to reduce the effect of base sensor capacitance
- Auto-calibration is enabled for IDAC

Figure 4. Widget Configuration



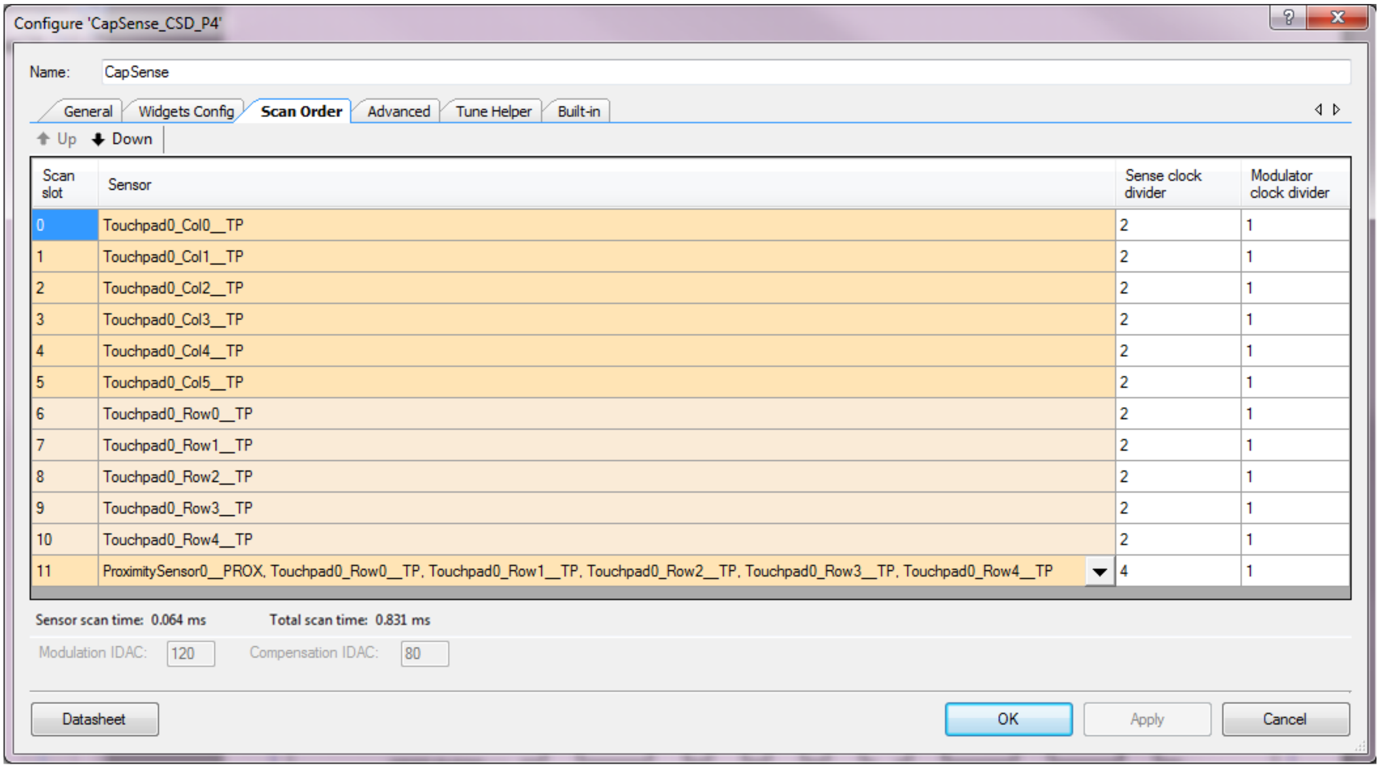
- Touchpad widget is placed with 6 x 5 sensors
- Column and row scan resolution is set to 10 bits
- API Resolution along the row and the column is set to 100 counts
- Finger Threshold and Noise Threshold are set using the Tuner

Figure 5. Widget Configuration - Proximity Sensor



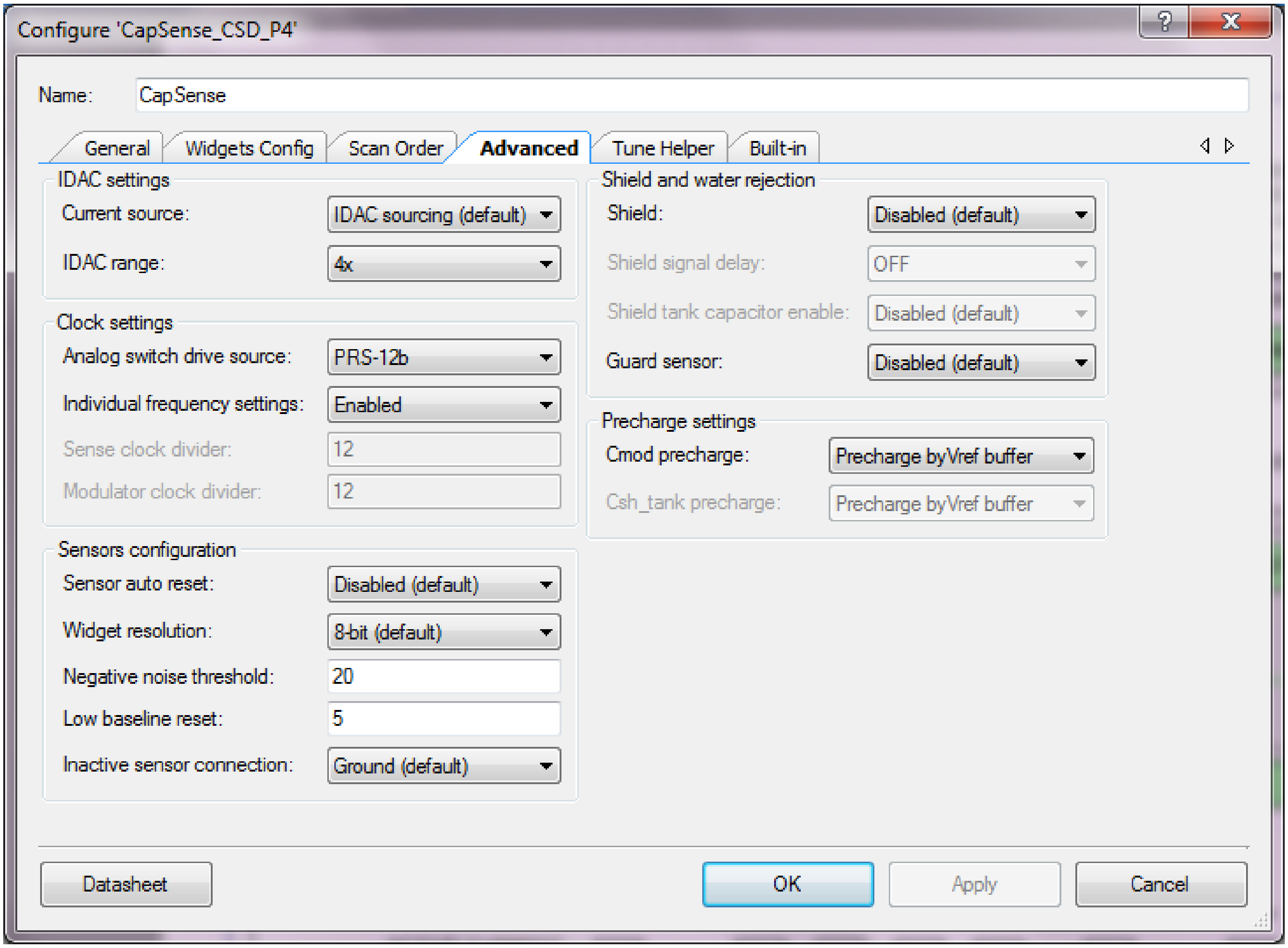
- Proximity sensor is selected, but with zero dedicated elements. Proximity sensor is formed by ganging the row sensor elements. See Figure 6 for details.
- Scan resolution of 11 bits. It is a tradeoff between sensitivity and scan time.
- Finger Threshold and Noise Threshold are configured using the tuner.

Figure 6. Scan Order and IDAC



- Sense clock divider and modulator clock divider are set as Figure 6 shows.
- Proximity sensor (Scan slot 11) is formed by combining the row sensors of the touchpad widget.

Figure 7. Advanced Settings

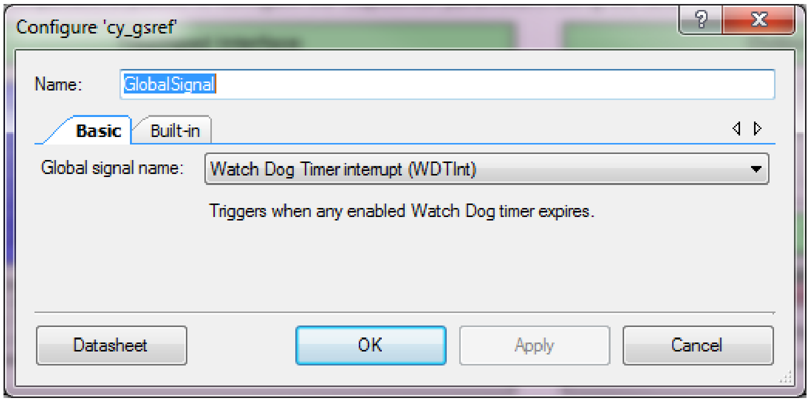


- IDAC is set to source mode with 4x range.
- Analog switch drive source is set to PRS-12b.
- Set the rest of the parameters as shown in Figure 7.

GlobalSignal Component

The component is configured to generate Watch Dog Timer Interrupt signal as Figure 8 shows.

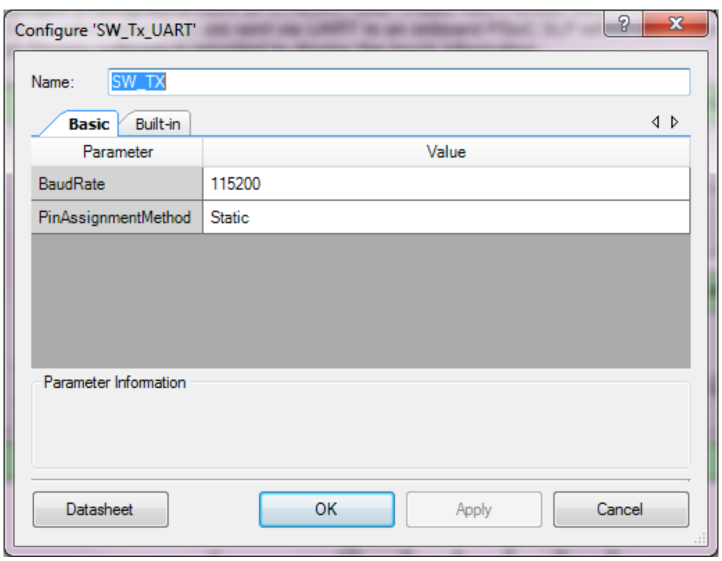
Figure 8. WDT Interrupt



SW Tx UART Component

This component is placed to send the touch position information to the external host via UART. Note that this component is only a transmitter.

Figure 9. SW TX Component Configuration



Project Description

An infinite loop in the main code of the design alternates the PSoC 4 device between Deep-Sleep and Active modes. Watchdog timer is used to generate the interrupts with the wake up period configured to either 30 ms or 200 ms depending on whether the touchpad is active or not. If the touchpad is active, the wakeup period is 30 ms, else it is 200 ms. The proximity sensor widget, which is formed by ganging the touchpad axis sensors, is used to detect the touch. It significantly reduces the scan time as compared to scanning the entire touchpad, thereby reducing the active power consumption.

When a finger is present on the touchpad, proximity sensor detects it. The touchpad sensor configuration is then restored and the device wake up period is configured to 30 ms. During this 30 ms period, device scans the entire touchpad, sends the previous touch position information using UART and enters Deep-Sleep power mode. CPU sends the touch position data during the touchpad scan; thereby utilizing active time of the device. After the scan is complete, the new touch position is calculated and the device is put to Deep-Sleep mode.

If the touch is not detected for 3 seconds, the wake up period is set to 200ms. Device then scans only the proximity sensor upon the periodic wakeup events.

The timing diagram of this process is shown in Figure 10 and the firmware flowchart is shown in Figure 11.

Figure 10. Timing Diagram

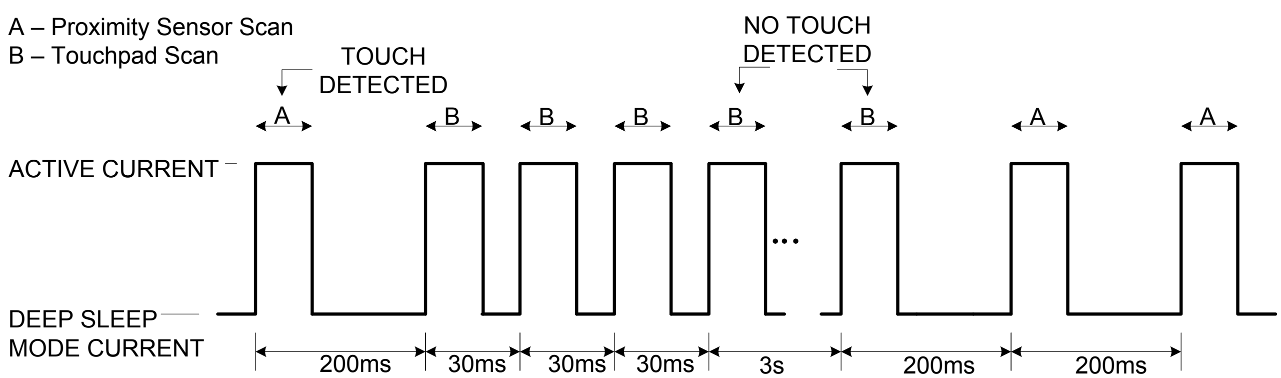
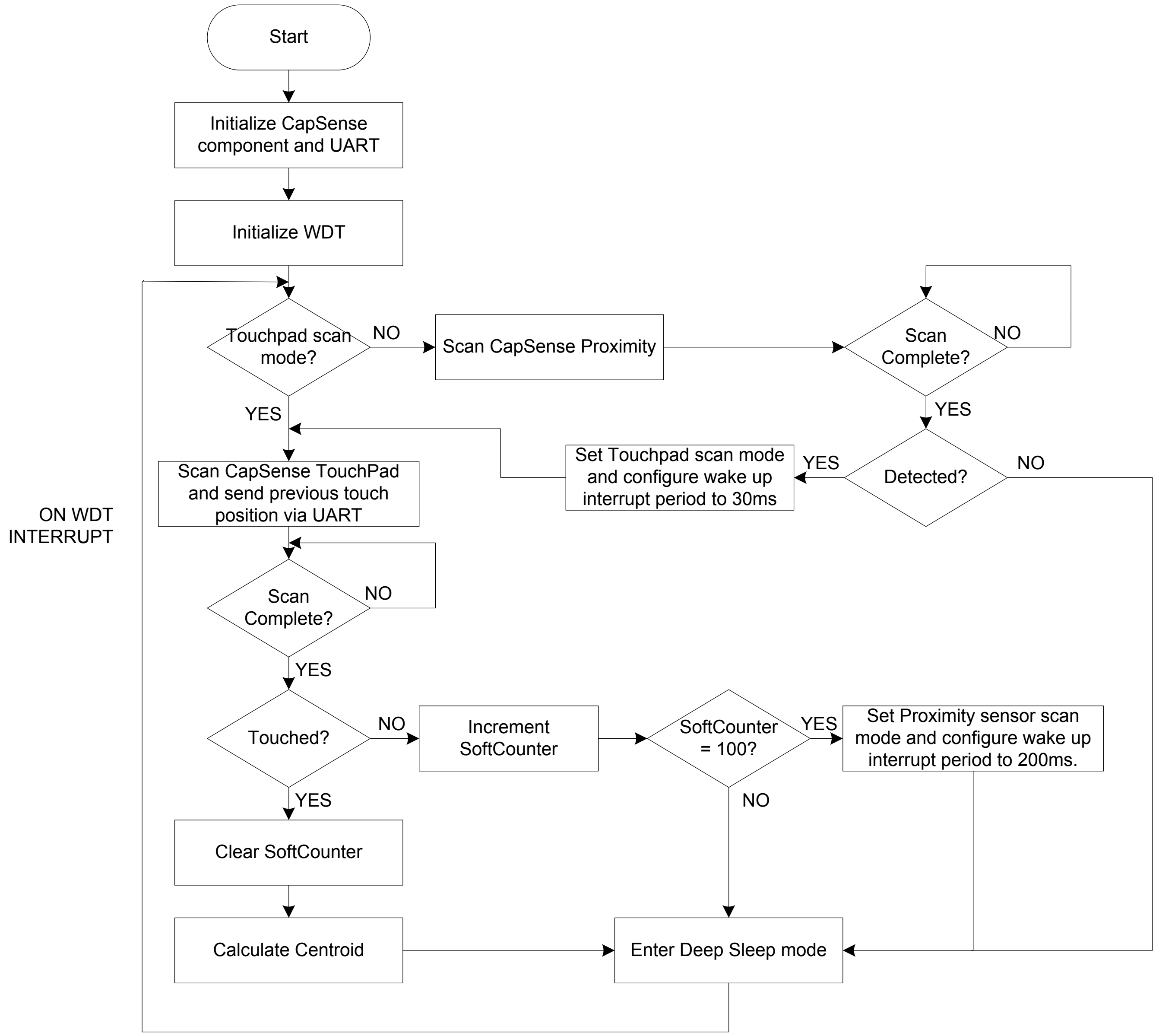


Figure 11. Firmware Flowchart



UART Data Viewing

PSoC 5LP device on the CY8CKIT-040 kit acts as a USB-UART Bridge. PSoC 4 device sends the X-Y touch position information to the PSoC 5LP device using UART. PSoC 5LP translates it to a USB packet and sends the data to the PC with USB enumerated as a virtual COM port. For viewing this data, use the serial port viewer tools such as Hyper-terminal or Tera Term software. Configure the serial port with following settings in the tool:

Baud Rate : 115200bps.

Data Bit : 8 bits

Parity : None

Stop : 1 bit

Flow Control : None

The X and Y counts will be from 0 to 100, which is set in Row API Resolution and Column API Resolution in the CapSense configuration tool as Figure 4 shows. Note that each record consists of 9 ASCII character codes. For example:

20 35 36 20 20 20 32 39 0D <space>56<space><space><space>29<carriage return>

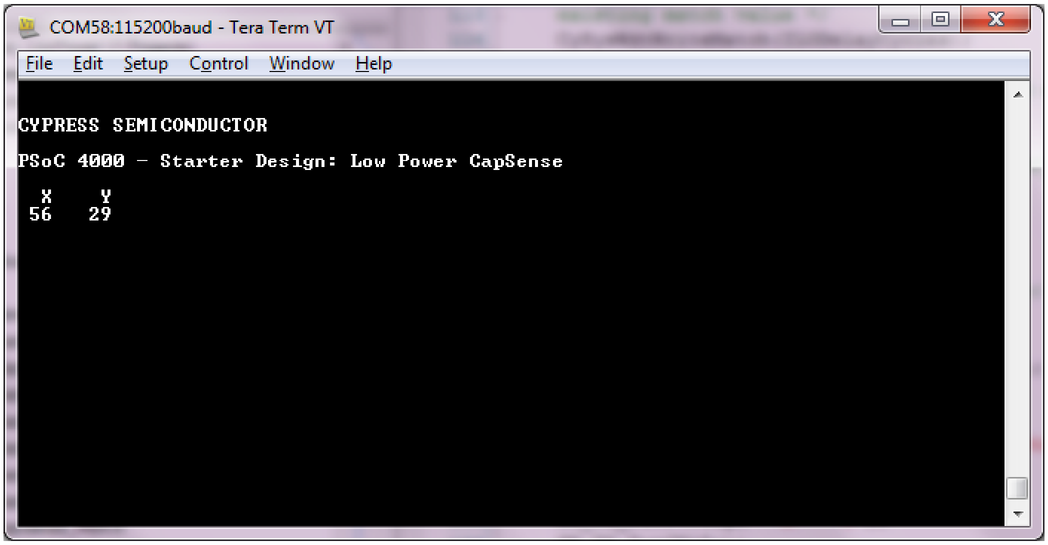
Expected Results

X-Y Coordinates is displayed in the serial port data viewer tool as Figure 12 shows. The current measurement readings are shown in Table 1.

Table 1. Expected Current Measurement Results

State	Current consumption	PC Application
When the touchpad is inactive	~8.3uA	X-Y counts on the UART terminal will be static
When the touchpad is active	~300uA	X-Y counts on the UART terminal should match the finger position on the touchpad

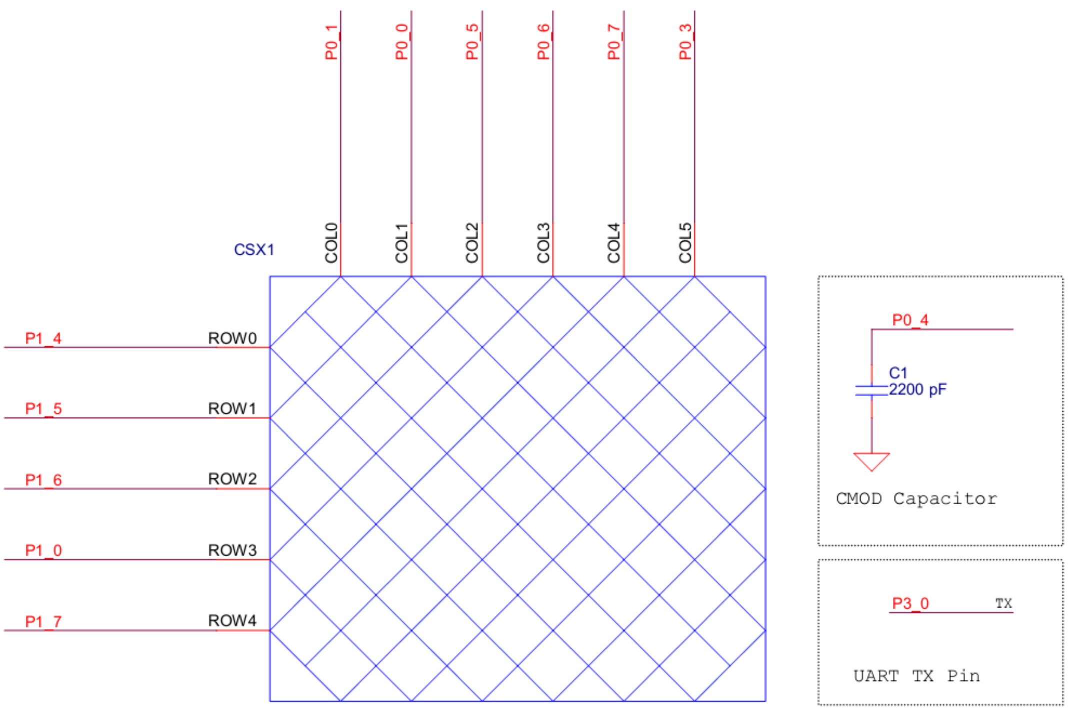
Figure 12. X-Y Touchpad Touch Coordinates on Serial Port Data Viewer



Schematic and Pin Mapping

Circuit connections for touchpad are given in Figure 13.

Figure 13. Touchpad Connections



Tuning

CapSense component can be tuned to any touchpad board / proximity sensor using the Tuner feature. The CapSense Tuner uses I^2C communication to configure the CapSense parameters.

In this project, an EZI2C Component configured as an I^2C slave is used for tuning purpose. The CY8CKIT-040 kit has an on-board PSoC 5LP device, which can act as I2C-USB Bridge. Thus, you can tune the CapSense component without using any other hardware.

In the project, uncomment the macro #define TUNING_ENABLE 1 in main.h file to enable the tuner code. Note that when tuning is enabled, UART will not function.

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

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