CY5682

PRoC™ BLE Touch Mouse Reference Design Kit Guide

Doc. No. 001-94177 Rev. *D

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone (USA): +1.800.858.1810
Phone (Intnl): +1.408.943.2600
www.cypress.com

Copyrights

Source Code

Disclaimer

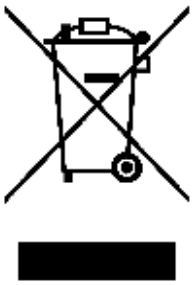
Trademarks

Flash Code Protection

Contents

Safety Information

The CY5682 PRoC™ BLE Touch Mouse Reference Design Kit (RDK) is intended for use as a development, demonstration, and evaluation platform for hardware or software in a laboratory environment. The kit is not intended for general consumer use. It generates, uses, and can radiate radio frequency energy. It has not been tested for compliance with the limits applicable under any standard. Operation of the equipment may cause interference with radio communications, in which case users, at their own expense, will be required to take whatever measures may be required to correct this interference. Cypress recommends that the kit be used only in a shielded room.

	The CY5682 PRoC™ BLE Touch Mouse RDK boards contain electrostatic discharge (ESD)-sensitive devices. Electrostatic charges readily accumulate on the human body and any equipment, which can cause a discharge without detection. Permanent damage may occur to devices subjected to high-energy discharges. Proper ESD precautions are recommended to avoid performance degradation or loss of functionality. Store unused CY5682 PRoC™ BLE Touch Mouse RDK boards in the protective shipping package.
	End-of-Life/Product Recycling The end-of-life cycle for this kit is five years from the date of manufacture mentioned on the back of the box. Contact the nearest recycler to discard the kit.

General Safety Instructions

ESD Protection

ESD can damage boards and associated components. Cypress recommends that the user perform procedures only at an ESD workstation. If an ESD workstation is not available, use appropriate ESD protection by wearing an antistatic wrist strap attached to the chassis ground (any unpainted metal surface) on the board when handling parts.

Handling Boards

The boards provided with CY5682 are sensitive to ESD. This also applies to the boards that are provided with a plastic casing when they are removed from the casing. Hold the boards only by the edges. After removing a board from the box/casing, place it on a grounded, static-free surface. Use a conductive foam pad, if available. Do not slide the board over any surface.

Battery Care and Use

- Use the correct size and type of battery specified in this guide.
- Keep battery contact surfaces and battery compartment contacts clean by rubbing them with a clean pencil eraser or a rough cloth each time you replace batteries.
- Remove the battery from a device when it is not expected to be in use for several months.
- Make sure that you insert the battery into your device properly, with the + (plus) and – (minus) terminals aligned correctly.
- Do not place the battery next to metallic objects such as keys and coins.
- Never throw the battery into fire.
- Do not open up the battery.
- Do not short the battery.
- Do not subject the battery to high temperatures or high humidity.
- Store the battery in a dry place.
- Do not recharge a battery unless it is marked "rechargeable."

Battery Disposal

Batteries can be safely disposed off with normal household waste. Never dispose off batteries in fire because they can explode.

It is important not to dispose off large amounts of batteries in a group. Used batteries are often not completely "dead." Grouping used batteries together can bring these "live" batteries into contact with one another, creating safety risks.

Regulatory Compliance Information

This kit contains devices that transmit and receive radio signals in accordance with the spectrum regulations for the 2.4-GHz unlicensed frequency range. This kit has passed precompliance tests for the following regulatory standards:

- CE
- FCC Part 15
- Canadian RSS-210
- EU EN 300 328

Contact Cypress technical support at bleapps@cypress.com for further details.

.

1. Introduction

Thank you for your interest in the CY5682 PRoC™ BLE Touch Mouse Reference Design Kit (RDK). This kit provides an implementation of a Bluetooth LE (low energy), or Bluetooth Smart, touch mouse using Cypress's PRoC BLE single-chip solution. PRoC BLE is a true single-chip solution that integrates Cypress's industry-leading capacitive touch sensing, an energy-efficient ARM® Cortex®-M0 CPU core, Bluetooth 4.1–compliant Bluetooth Smart connectivity, and a balun to minimize external components for a wide variety of applications.

The kit includes a CySmart™ USB dongle, which can be used to connect the touch mouse to devices such as PCs, laptops, tablets, and smartphones. Both the touch mouse and the dongle can be programmed, enabling users to customize and evaluate firmware per their requirements. Schematics, layout files, ready-to-use firmware, and source code are provided with the kit to enable PC peripheral designers to build their own touch mouse designs.

1.1 Kit Contents

The CY5682 PRoC BLE Touch Mouse RDK includes the following items, as shown in Figure 1-1:

1. PRoC BLE touch mouse
2. CySmart USB dongle
3. MiniProg3 programmer/debugger
4. 10-pin ribbon cable
5. Two AAA batteries
6. USB 2.0 standard A to mini-B cable
7. Tweezers

The kit also includes a quick start guide to enable users to connect the PRoC BLE touch mouse to a PC using the CySmart USB dongle.

Figure 1-1. Kit Contents

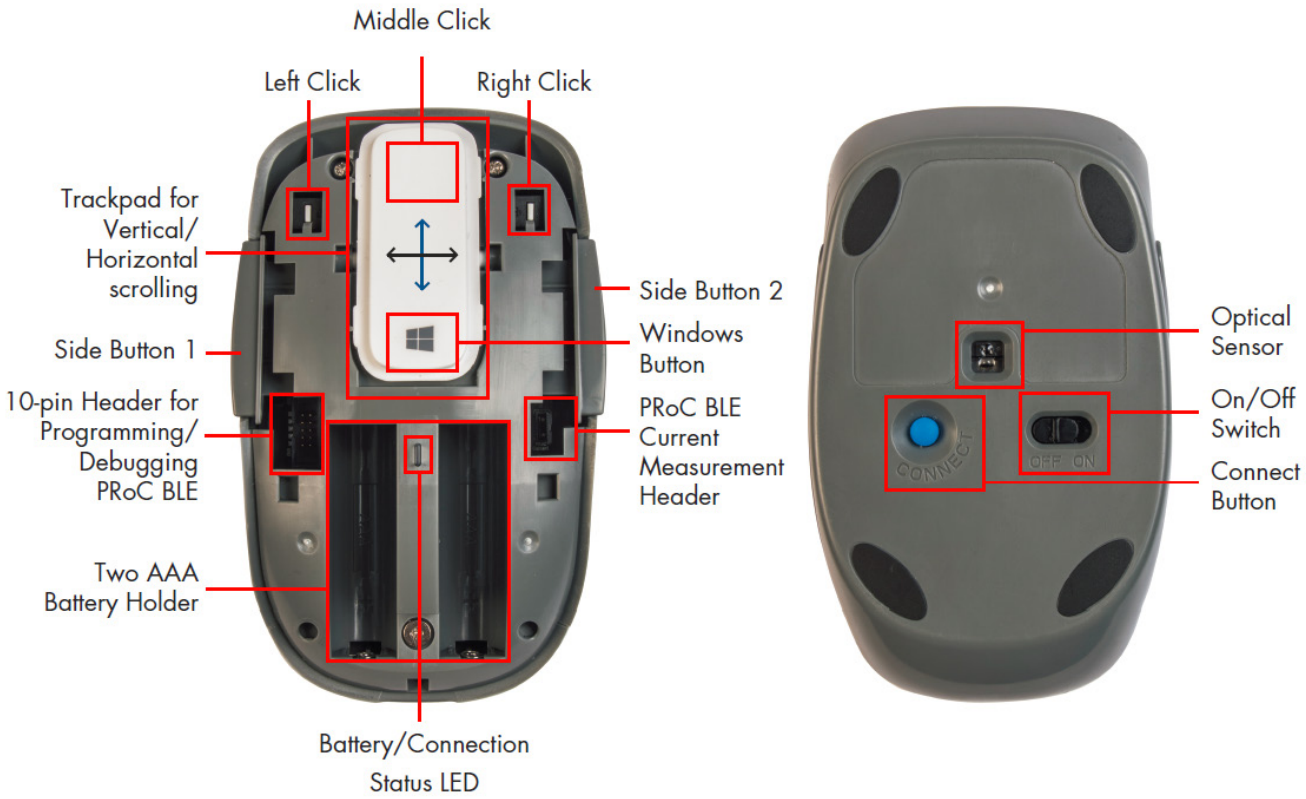
If any part of the kit is missing, contact your nearest Cypress sales office for help: www.cypress.com/go/support.

1.2 PRoC BLE Touch Mouse Details

This section details the blocks and explains the features supported by the PRoC BLE touch mouse.

Note: Figure 1-2 shows the PRoC BLE touch mouse without the top cover, so you can see all its features.

Figure 1-2. PRoC BLE Touch Mouse Features



The quick start guide provided with the kit details the procedure to connect the PRoC BLE touch mouse to a PC using the CySmart USB dongle. You can also connect the PRoC BLE touch mouse directly to any Bluetooth Smart Ready device. Details for this procedure are available in the Connecting PRoC BLE Touch Mouse with Bluetooth Smart Ready Device section.

Once connected to a PC, the PRoC BLE touch mouse provides standard mouse functionality as well as advanced Windows 8.0/8.1–specific functionality, as follows.

Standard mouse features:

- Cursor tracking on PC
- Mouse left-click
- Mouse right-click
- Mouse middle-click
- Moving the finger vertically on the trackpad maps to vertical scrolling
- Moving the finger vertically on the trackpad followed by a liftoff maps to vertical inertial scrolling
- Moving the finger horizontally on the trackpad maps to horizontal scrolling

Advanced features for Windows 8.0/8.1:

- Pressing and releasing the Windows button takes you to the Windows Start screen (opens Start menu on Windows 7 and earlier).
- Pressing the Side button 1 maps to toggling across open apps
- Pressing the Side button 2 maps to the Windows Charms bar

Note: When using the CySmart USB dongle, these features are dependent upon the operating system supporting the generic desktop usage page of the human interface device (HID) class over USB. When the PRoC BLE touch mouse is directly connected to a Bluetooth Smart Ready PC, these features depend on support available for the HID Over GATT Profile (HOGP) over Bluetooth Smart.

The touch mouse also provides LED-based notifications as follows.

- The orange LED shows a slow-breathing effect when the touch mouse is advertising over Bluetooth Smart. It stays on for five seconds and then switches off after a successful connection.
- The orange LED blinks three times on user activity when the touch mouse is doing directed advertisement to re-establish the connection because the existing connection has been lost. The connection can fail due to the mouse being out of range or the host being turned OFF.
- The red LED shows a breathing effect on a low battery level (that is, battery voltage below 1.1 V).

Moreover, the touch mouse offers these features:

- Ability to debug and program the onboard PRoC BLE device using the 10-pin header
- Ability to measure current consumption for PRoC BLE using the current measurement header

1.3 PSoC Creator

PSoC Creator™ is a state-of-the-art, easy-to-use integrated design environment (IDE). It introduces revolutionary hardware and software co-design, powered by a library of verified and characterized PSoC Components™.

With PSoC Creator, you can:

- Drag and drop PSoC Components to build a schematic of your custom design
- Automatically place and route Components and configure GPIOs
- Develop and debug firmware using the included Component APIs

PSoC Creator also enables you to tap into an entire tool ecosystem, including integrated compiler chains and production programmers for PSoC® devices. For more information, visit www.cypress.com/creator.

1.4 Getting Started

This kit guide will help you get fully acquainted with the CY5682 PRoC BLE Touch Mouse RDK.

- The Kit Contents section details the contents of the kit.
- The quick start guide shipped with the kit provides instructions on how to start using the PRoC BLE touch mouse with the CySmart USB dongle.
- The PRoC BLE Touch Mouse Details section covers the features of the touch mouse.
- The Installation chapter describes the installation of the kit software. This includes the PSoC Creator IDE for development, programming, and debugging; PSoC Programmer for programming hex files; and the CySmart tool for Bluetooth LE host emulation.
- The Kit Operation chapter describes common procedures such as programming and debugging the PRoC BLE touch mouse and the CySmart USB dongle. It also introduces the CySmart tool for Bluetooth LE host emulation.
- The Hardware chapter describes the hardware contents of the kit.
- The Firmware chapter describes the firmware on the PRoC BLE touch mouse and the CySmart USB dongle.
- The Advanced Topics chapter explains topics such as connecting the touch mouse to Bluetooth Smart Ready devices, assembling and disassembling the touch mouse, and measuring current/voltage on the touch mouse.
- A. Appendix: Schematics and FAQ provides schematics and a list of frequently asked questions (FAQs) for troubleshooting.

1.5 Additional Learning Resources

Visit www.cypress.com/PRoCBLE for additional learning resources in the form of datasheets, technical reference manuals, and application notes.

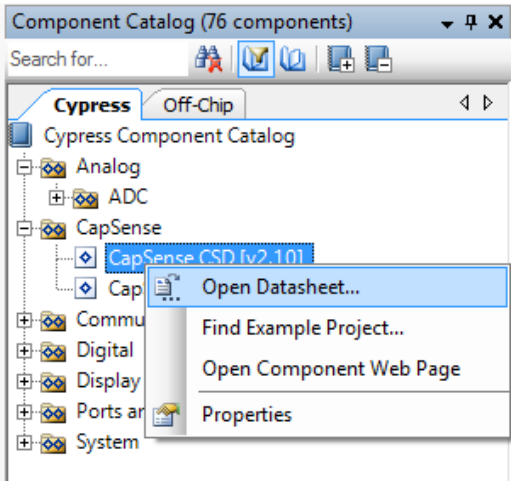
1.5.1 Beginner Resources

PSoC Creator Training: www.cypress.com/go/creatorstart/creatortraining

1.5.2 Component Datasheets

To view the datasheet of a Component from inside PSoC Creator, right-click on the Component and select **Open Datasheet** (see Figure 1-3).

Figure 1-3. Opening Component Datasheet



1.5.3 Learning From Peers: Cypress Developer Community Forums

Visit www.cypress.com/forums to interact with Cypress applications support teams.

1.5.4 Other Kits

Other kits available for PRoC BLE include the following:

- CY8CKIT-042-BLE Bluetooth Low Energy Pioneer Kit (www.cypress.com/CY8CKIT-042-BLE)
- CY5672 PRoC BLE Remote Control RDK (www.cypress.com/CY5672)
- CY5671 PRoC BLE Module (www.cypress.com/CY5671)

1.6 Technical Support

For assistance, visit Cypress Support or contact customer support at +1 (800) 541-4736 Ext. 2 (in the USA) or +1 (408) 943-2600 Ext. 2 (International).

1.7 Document Conventions

Table 1-1. Document Conventions for Guides

Convention	Usage
`Courier New`	Displays file locations, user-entered text, and source code: `C:\ ...cd\icc\`
Italics	Displays file names and reference documentation: Read about the *sourcefile.hex* file in the *PSoC Designer User Guide*.
[**Bracketed, Bold**]	Displays keyboard commands in procedures: [**Enter**] or [**Ctrl**] [**C**]
File > Open	Represents menu paths: File > Open > New Project
Bold	Displays commands, menu paths, and icon names in procedures: Click the **File** icon and then click **Open**.
Times New Roman	Displays an equation: 2 + 2 = 4
Text in gray boxes	Describes Cautions or unique functionality of the product.

1.8 Acronyms

Table 1-2. Acronyms Used in this Document

Acronym	Definition
ADC	Analog-to-Digital Converter
API	Application Programming Interface
BD address	Bluetooth Device address
BLE	Bluetooth Low Energy
CDC	Communications Device Class
ESD	Electrostatic Discharge
GAP	Generic Access Profile
GATT	Generic Attribute
GUI	Graphical User Interface
GPIO	General Purpose Input/Output
HID	Human Interface Device
HOGP	HID (Human Interface Device) over GATT (Generic Attribute) Profile
I^2C	Inter-Integrated Circuit
IAS	Immediate Alert Service
IDAC	Interconnecting Digital-Analog Converter
IDE	Integrated Development Environment
LED	Light-Emitting Diode
MTU	Maximum Transmission Unit
PHY	Physical Layer
PrISM	Precise Illumination Signal Modulation
PRoC	Programmable Radio-on-Chip
PSoC	Programmable Systems-on-Chip
PWM	Pulse-Width Modulation
QFN	Quad Flat No-lead (package)
SAR	Successive Approximation Register
SWD	Serial Wire Debug
UART	Universal Asynchronous Receiver Transmitter
UUID	Universally Unique Identifier

2. Installation

This chapter describes the steps to install the software tools and packages on a PC for the CY5682 PRoC BLE Touch Mouse RDK. The kit installer will install the following software tools and associated documentation on your PC:

- PSoC Creator: IDE used to view, develop, and build firmware for PRoC BLE–based devices
- PSoC Programmer: Software used to download a hex file to the PRoC BLE device
- CySmart: A Bluetooth LE host emulation tool for Windows PCs. The tool works with the CySmart USB dongle and provides an easy-to-use GUI that enables customers to evaluate and debug Bluetooth LE peripheral applications. Refer to the section Using CySmart USB Dongle with CySmart Tool for more details on how to use the CySmart tool.
- Documentation: Includes kit documentation files, hardware schematics, layout, and BOM

2.1 Before You Begin

All Cypress software installations require administrator privileges. Administrator privileges are not required to execute the software after installation. Before you install the kit software, close any other Cypress software that is currently running.

2.2 Install Software

Follow these steps to install the CY5682 PRoC BLE Touch Mouse RDK software.

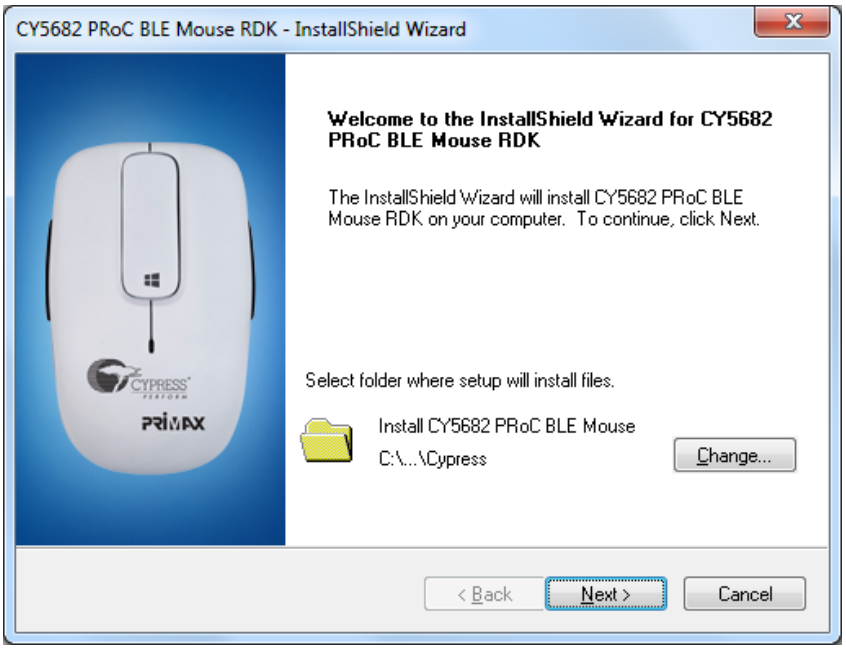
1. Download the CY5682 PRoC BLE Touch Mouse RDK software from www.cypress.com/CY5682. The kit software is available in three different formats for download:
   a. **CY5682 Kit Setup:** This installation package contains all files related to the kit as well as PSoC Creator, PSoC Programmer, and CySmart software. However, it does not include the Windows Installer or Microsoft .NET framework packages. If these packages are not on your computer, the installer directs you to download and install them from the internet.
   b. **CY5682 Kit Only:** This executable file installs only the kit contents, which include the kit source code and firmware, hardware files, and user documents. This package can be used if all the software prerequisites (PSoC Creator, PSoC Programmer, and CySmart) are already installed on your PC.
   c. **CY5682 DVD ISO:** This file is a complete package, stored in a DVD-ROM image format that can be used to create a DVD or extract using an ISO extraction program such as WinZip or WinRAR. The file can also be mounted as a virtual DVD using virtual drive programs such as Virtual CloneDrive and MagicISO. This file includes all the required software, utilities, drivers, hardware files, and user documents.
2. If you have downloaded the ISO file, mount it as a virtual drive. Extract the ISO contents if you do not have a virtual drive to mount. Double-click *cyautorun.exe* in the root directory of the extracted content or mounted ISO if “Autorun from CD/DVD” is not enabled on the PC. The installation window will appear automatically.

   If you are using the CY5682 Kit Setup or CY5682 Kit Only file, then double-click on the file and go to step 4.
3. Click **Install CY5682 PRoC BLE Mouse RDK**, as shown in Figure 2-1, to start the kit installation.

Figure 2-1. Kit Installer Screen

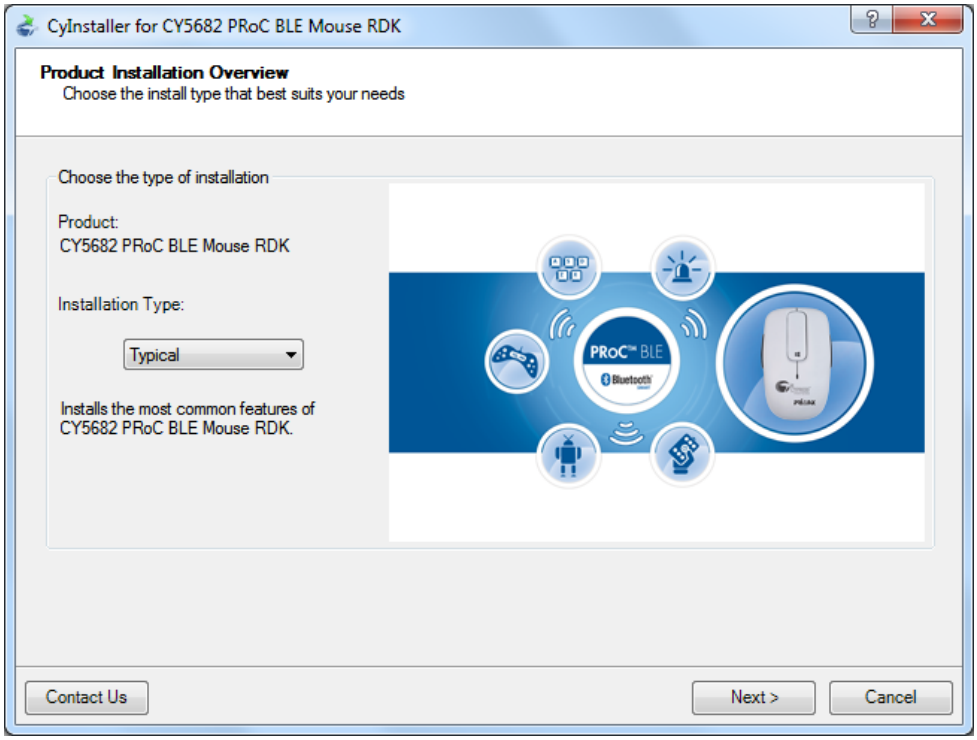
4. If desired, you may select the folder in which you want to install the kit-related files by clicking **Change** as shown in Figure 2-2. Choose the directory and click **Next.**

Figure 2-2. PRoC BLE RDK – InstallShield Wizard



5. Select the “Typical,” “Custom,” or “Complete” installation type in the **Product Installation Overview** window, as shown in Figure 2-3. Click **Next** after you select the installation type.

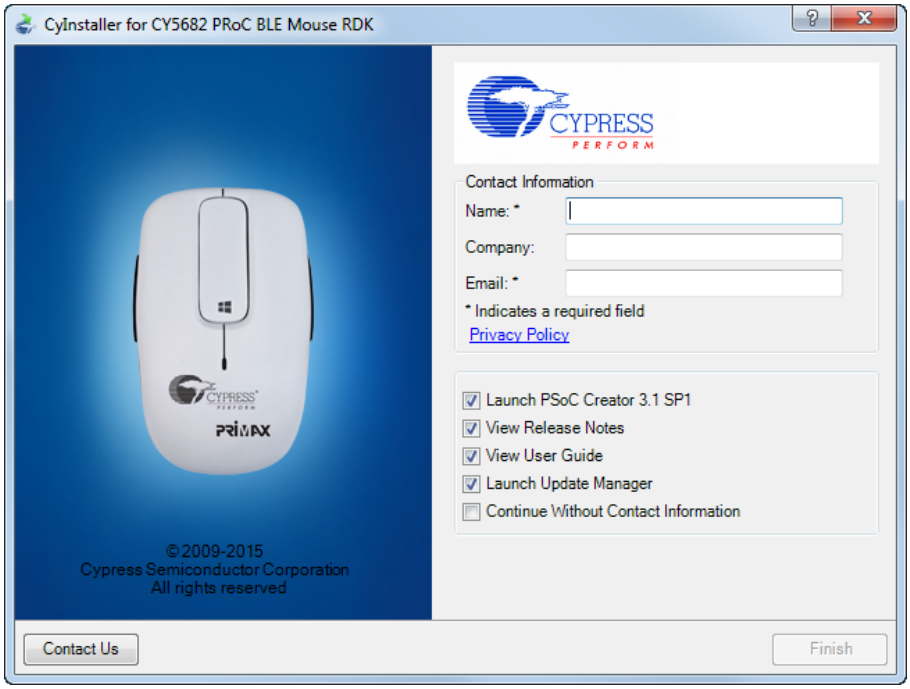
Figure 2-3. Product Installation Overview



6. Read the license agreement and select **I accept the terms in the license agreement** to continue with the installation. Click **Next**. When you click **Next**, the installer automatically installs the required software, if it is not present on your computer. Following is the required software:
   - PSoC Creator 3.1 or later: Download from www.cypress.com/psoccreator.
   - PSoC Programmer 3.21 or later: Download from www.cypress.com/programmer.
   - CySmart 1.0 or later: Download from www.cypress.com/CySmart.

   Note: If you are using the CY5682 Kit Only package, you must download and install this software manually.
7. The installation begins by presenting the list of packages (selected in the previous step) on the installation page. A green check mark appears next to each package after it is installed successfully.
8. Enter your contact information or select the option **Continue Without Contact Information,** as shown in Figure 2-4. Click **Finish** to complete the kit installation.

Figure 2-4. Installer Asking for Contact Information



Note: The PSoC Creator version number displayed in Figure 2-4 may vary depending upon the latest PSoC Creator packaged in the installer.

9. After the installation is complete, the kit contents are available at this location:

   <Install_Directory>\CY5682 PRoC BLE Mouse RDK

   Default location:

   Windows 7 (64-bit):

   C:\Program Files (x86)\Cypress\CY5682 PRoC BLE Mouse RDK

   Windows 7 (32-bit):

   C:\Program Files\Cypress\CY5682 PRoC BLE Mouse RDK

   Note: For Windows 7/8/8.1 users, the installed files and the folder are read-only.

2.3 Uninstall Software

The software can be uninstalled using one of the following methods:

- Go to **Start** > **All Programs** > **Cypress** > **Cypress Update Manager** > **Cypress Update Manager**; click the **Uninstall** button for the appropriate software package.
- Go to **Start** > **Control Panel** > **Programs and Features**; click the **Uninstall/Change** button for the appropriate software package.

3. Kit Operation

This chapter introduces the features of the CY5682 PRoC BLE Touch Mouse RDK that will be used as part of the kit operation. Topics include:

- Connecting PRoC BLE Touch Mouse with CySmart USB Dongle
- Programming and Debugging
- Using CySmart USB Dongle with CySmart Tool

3.1 Connecting PRoC BLE Touch Mouse with CySmart USB Dongle

In this document, “connecting” refers to establishing a Bluetooth Smart wireless link between the PRoC BLE touch mouse and the CySmart USB dongle. The process of establishing the connection for the first time involves “bonding,” which refers to the storage of encryption information from the mouse in the nonvolatile memory of the CySmart USB dongle. This stored information forms the “whitelist,” which is used by a Bluetooth Smart and Bluetooth Smart Ready devices to scan and search for subsequent connections.

The PRoC BLE touch mouse is factory-bonded with the CySmart USB dongle. Therefore, when the dongle is plugged in, it begins to search for a bonded mouse (that is, mouse unit that is present in the dongle’s whitelist). After both AAA batteries are inserted and the ON/OFF switch is set to the ON position, any user activity on the PRoC BLE touch mouse connects it to the dongle. Refer to step 1 in the Accessing Debug Interfaces of the Touch Mouse section for instructions on removing the top cover to insert the two AAA batteries.

The touch mouse needs to be reconnected to the CySmart USB dongle when the Bluetooth device address in the touch mouse firmware is changed or when the dongle is reprogrammed. The steps to reconnect the PRoC BLE touch mouse with the CySmart USB dongle are as follows.

Note: Making changes to the touch mouse firmware or CySmart USB dongle firmware that impact the connection establishment behavior may invalidate the following steps.

1. Insert the dongle into a USB port on the PC. The red power LED on the dongle glows to show that the dongle is powered on, and the green status LED on the dongle glows to show that enumeration is complete on the dongle side. The blue user LED on the dongle shows a slow-breathing effect to indicate that the dongle is scanning for preconnected devices.

   Notes:

   - The time to complete enumeration on the PC side for the first time can vary based on driver installation time.
   - The blue user LED on the CySmart USB dongle does not show a slow-breathing effect if the dongle is reprogrammed. In this case, press the user button on the CySmart USB dongle to start scanning. The blue LED shows the fast breathing effect.

2. Plug the two AAA batteries provided with the kit into the battery holder on the touch mouse.

   Note: The touch mouse can also work with a single AAA battery; however, using two AAA batteries is recommended to prolong the battery life.

3. Set the ON/OFF switch on the touch mouse to the ON position.

4. Press the user button (SW2) on the dongle to start scanning for Bluetooth Smart devices. The blue user LED on the dongle shows a fast-breathing effect to indicate that the dongle is scanning for new devices. Skip this step if the dongle is already in prebond mode, as indicated by the slow-breathing blue user LED.

5. Press the connect button on the touch mouse. The orange LED on the touch mouse shows a slow-breathing effect to indicate that the touch mouse is now in advertising mode. The touch mouse remains in the advertising mode for a maximum interval of 30 seconds.
6. When a successful connection is established, the orange LED on the touch mouse stays on for five seconds and then switches off, while the blue user LED on the CySmart USB dongle glows continuously. If the connection is unsuccessful, the orange LED on the touch mouse stops blinking after a 30-second timeout. In this event, repeat the sequence starting from step 2 to establish a connection.

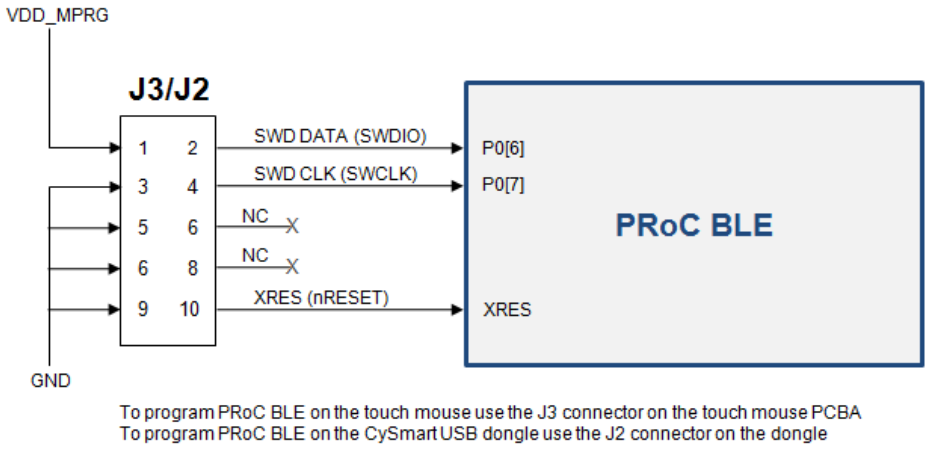
   Note: Do not turn off the mouse while the orange LED is in the ON state for five seconds after successful connection.
7. Place the touch mouse on a flat surface and move it around. If the connection is successful, the mouse cursor on the PC will follow the movement of the touch mouse.

3.2 Programming and Debugging

3.2.1 Programming and Debugging PRoC BLE on Touch Mouse and CySmart USB Dongle

The PRoC BLE touch mouse and the CySmart USB dongle support programming and debugging using the MiniProg3 programmer/debugger provided with the kit. Both expose a 10-pin connector. Figure 3-1 is a block diagram showing the header and associated connections.

Figure 3-1. Programming/Debugging PRoC BLE



Notes:

- Remove the warning sticker before using MiniProg3.
- Before trying to program or debug the device, make sure that PSoC Creator and PSoC Programmer are installed on the PC.

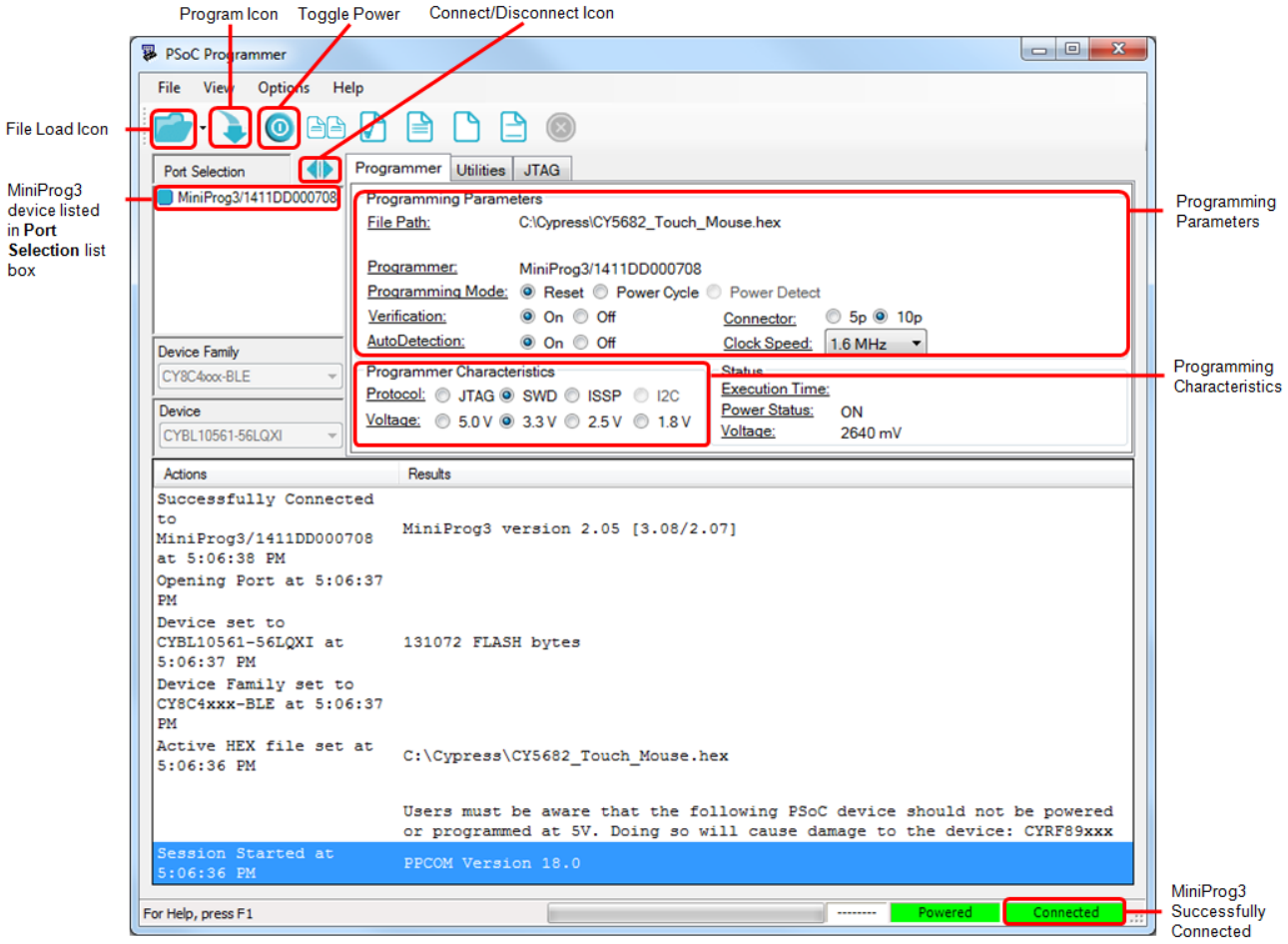
You can program the touch mouse either with PSoC Programmer or directly from PSoC Creator.

To program the touch mouse using PSoC Programmer, follow these steps.

1. Connect MiniProg3 to the PC using the USB A to mini-B cable provided with the kit. When it is properly connected, the four LEDs on the MiniProg3 turn on for a few seconds.
2. Connect one end of the 10-pin ribbon cable, provided with the kit, to the 10-pin header on MiniProg3.
3. Connect the other end of the 10-pin ribbon cable to the 10-pin header (J3 on the touch mouse or J2 on the CySmart USB dongle). Programming headers on both the mouse and the CySmart USB dongle (J3 and J2 respectively) are polarized. Do not apply excessive force; instead, ensure that the connectors are plugged in the correct orientation.
4. Run PSoC Programmer by choosing **Start** > **All Programs** > **Cypress** > **PSoC Programmer**.

   Note: Ensure that only one instance of PSoC Programmer is running on the PC..
5. To establish the connection with the programmer, select either the “MiniProg3” device in the **Port Selection** list box or the **Connect/Disconnect** icon, as shown in Figure 3-2.

Figure 3-2. PSoC Programmer



6. If the connection is successful, the green status LED on MiniProg3 lights up. A blue square with rounded corners appears next to “MiniProg3” in the **Port Selection** list box. Also, the status in the lower right corner of the PSoC Programmer window turns green and shows **Connected**.
7. Ensure that the settings under **Programming Parameters** in the **Programmer** tab are set to the values specified in Table 3-1.

Table 3-1. Programming Parameters

Programming Parameter	Setting
Programming Mode	Touch mouse: Reset (only if batteries are inserted and the ON/OFF switch is in ON position) or Power Cycle Dongle: Reset (when plugged into and powered via USB port) or Power Cycle (when unplugged from USB port)
Verification	On
AutoDetection	On
Connector	10p
Clock Speed	1.6 MHz

Note: Reset mode can be used even when batteries are not inserted if the **Toggle Power** button shown in Figure 3-2 is pressed before step 9.

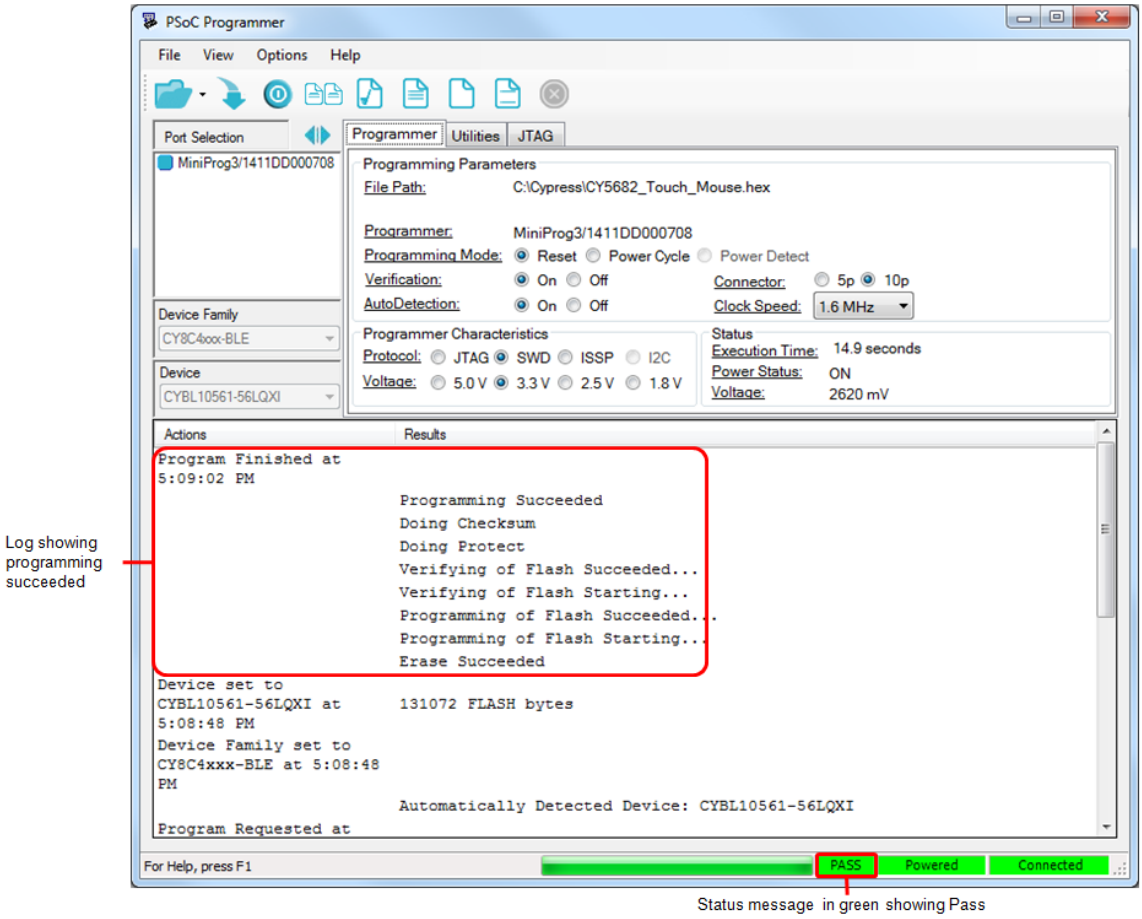
8. Ensure that the parameters under **Programmer Characteristics** are set to the values specified in Table 3-2.

Table 3-2. Programmer Characteristics

Programmer Characteristics	Setting
Protocol	SWD (Serial Wire Debug)
Voltage	3.3 V

9. Click the **File Load** icon. A dialog box appears. Browse to the location of the hex file to be programmed and select the hex file. The selected file appears next to the "File Path" in the **Programmer** tab. The following hex files for the firmware are provided with the kit:
   - PRoC BLE touch mouse:
     <Install_Directory>\CY5682 PRoC BLE Mouse RDK\1.0\Firmware\Hex Files\Touch Mouse\CY5682_Touch_Mouse.hex
   - CySmart USB dongle:
     <Install_Directory>\CY5682 PRoC BLE Mouse RDK\1.0\Firmware\Hex Files\Dongle\BLE_HID_CySmart_Dongle.hex
10. Click the **Program** icon to program the PRoC BLE device on the touch mouse or the CySmart USB dongle with the selected hex file. After successful programming, a "Programming Succeeded" message is displayed in the log area, as shown in Figure 3-3. Also, the status in the lower right corner of the PSoC Programmer window turns green and shows **PASS**.

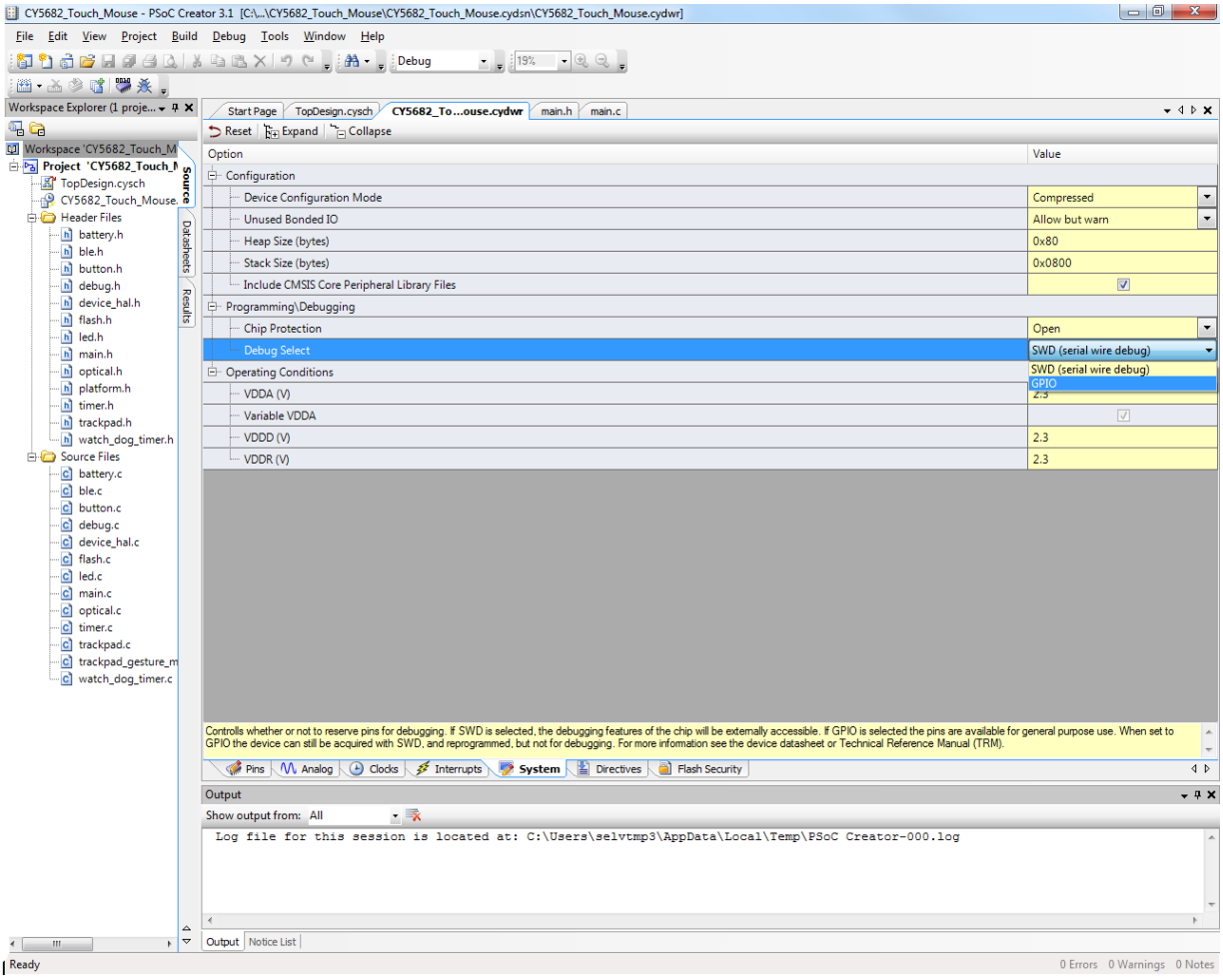
Figure 3-3. Programmer Window after Successful Programming



To program and debug PRoC BLE on the touch mouse or CySmart USB dongle using PSoC Creator, follow these steps.

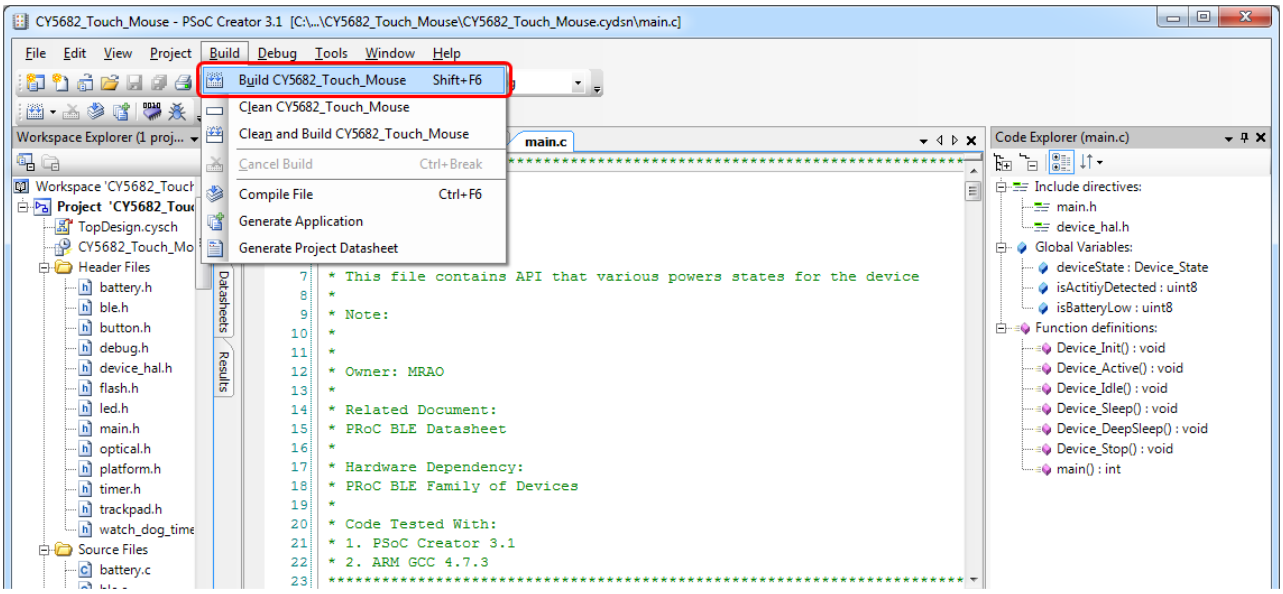
1. Run PSoC Creator by choosing **Start** > **All Programs** > **Cypress** > **PSoC Creator**.
2. Go to the Start Page, expand **Kits** > **CY5682 Touch Mouse RDK**, and then click on the project that you want to open. This allows you to save an editable copy the project to the location of your choice. After you have saved the copy, you can open it using **File** > **Open** > **Project/Workspace**.
3. Note that the debug option is disabled by default in the touch mouse firmware to minimize power consumption. You can enable the debug option using the following procedure:
   - Open *CY5682_Touch_Mouse.cydwr* from Workspace Explorer.
   - Click the **System** tab.
   - Choose the SWD option under **Debug Select,** as shown in Figure 3-4.

Figure 3-4. Enabling Debug Option Using PSoC Creator



4. Build the project by choosing **Build** > **Build <*project name*>**, as shown in Figure 3-5.

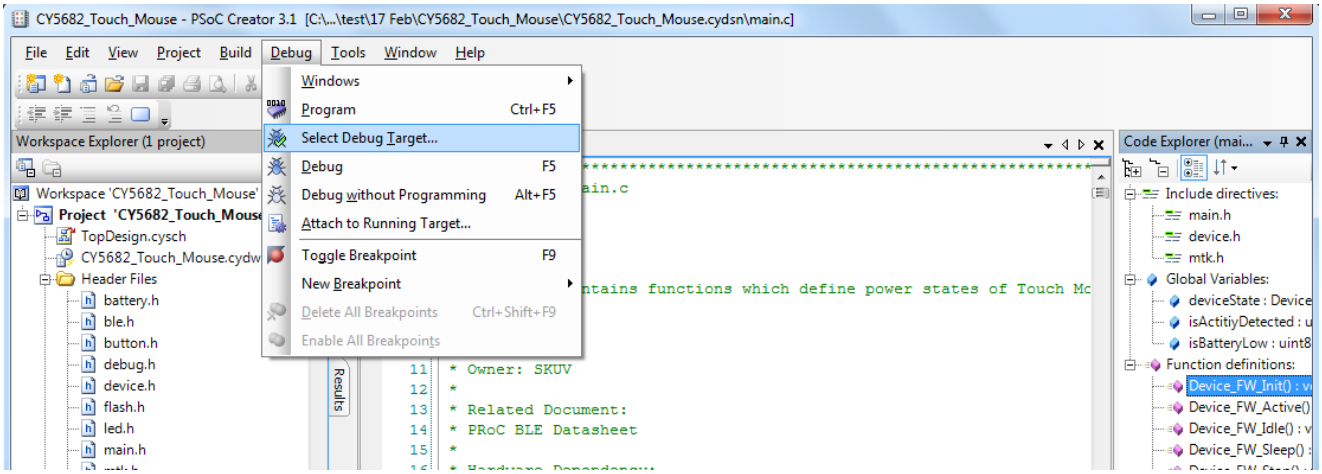
Figure 3-5. Building a Project in PSoC Creator



The **Output** window in PSoC Creator and the status bar display any errors encountered during the build process. Ensure that the project builds without any errors.

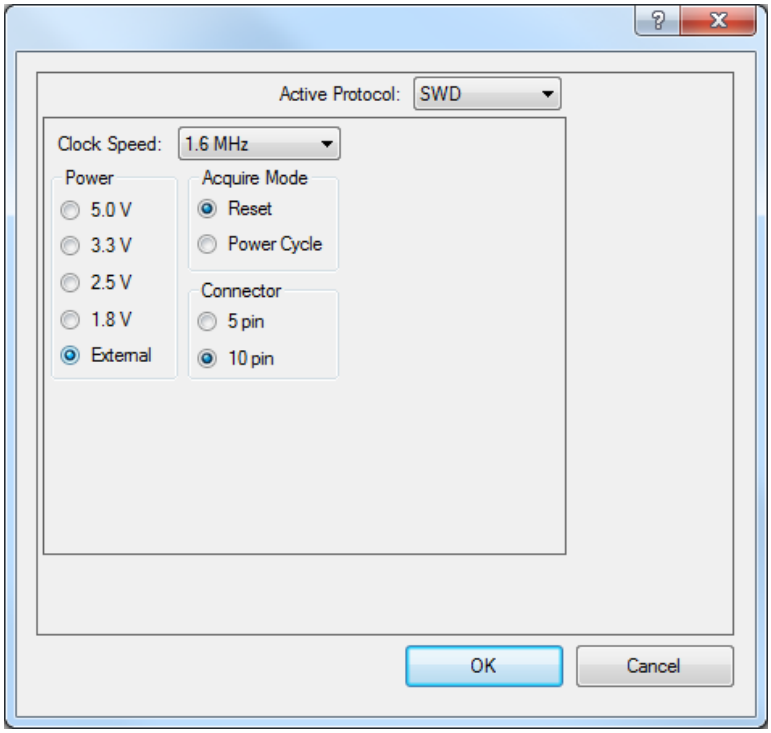
5. Connect the MiniProg3 to the PC using the USB A to mini-B cable provided with the kit. When it is properly connected, the four LEDs on the MiniProg3 turn on for a few seconds.
6. Connect one end of the 10-pin ribbon cable, provided with the kit, to the 10-pin header on the MiniProg3.
7. Connect the other end of the 10-pin ribbon cable to the 10-pin connector on the touch mouse (J3) or on the CySmart USB dongle (J2).
8. Start programming by choosing **Debug** > **Select Debug Target,** as shown in Figure 3-6. This will bring up the **Select Debug Target** window.

Figure 3-6. Programming via PSoC Creator



9. In the **Select Debug Target** window, click the **Port Setting** button and set the configuration options as shown in Figure 3-7.

Figure 3-7. Port Settings

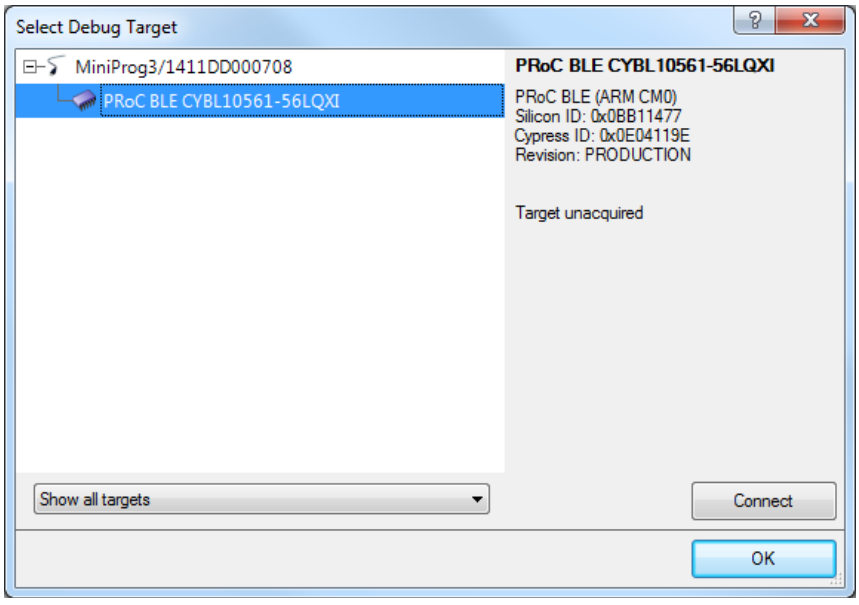


Notes:

- **Acquire Mode** can be set to “Reset” only if batteries are inserted and the ON/OFF switch is in the ON position. Otherwise, set it to “Power Cycle” and choose the Power setting as 1.8 V, 2.5 V, or 3.3 V.
- The “External” selection for the **Power** setting works only if batteries are inserted and the ON/OFF switch is in the ON postion, regardless of the **Acquire Mode** selection.

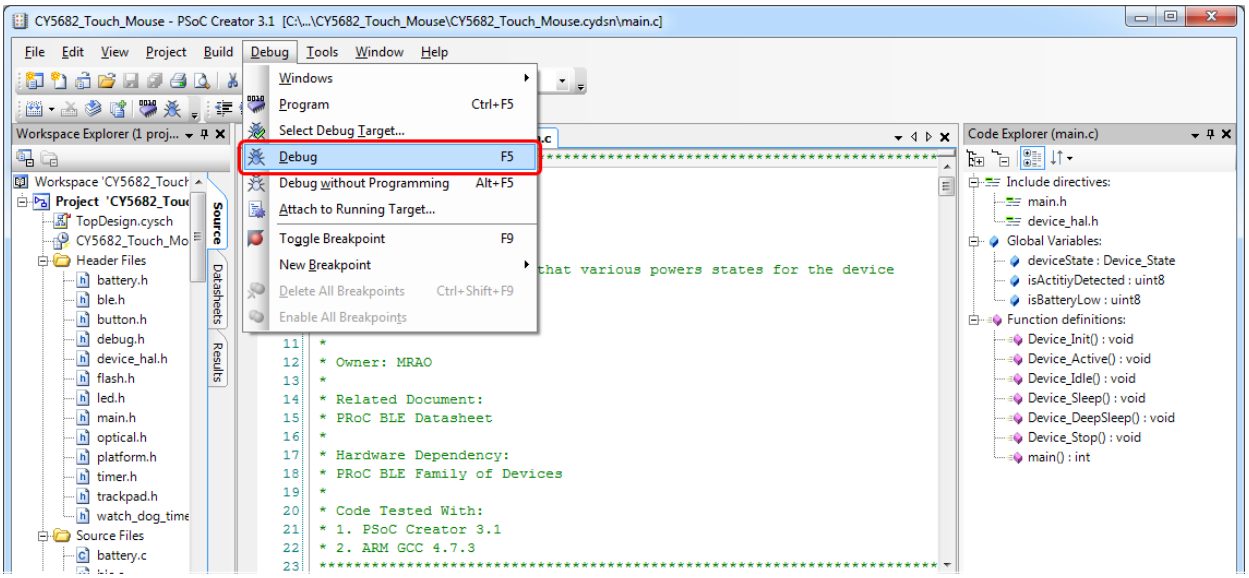
10. In the **Select Debug Target** window, select the PRoC BLE device and click **Connect**, as shown in Figure 3-8 and click **OK**.

Figure 3-8. Select Debug Target Window



11. Select **Debug > Program**.
12. Check the **Output** window and the status bar to check if programming is completed successfully.
13. If you need to debug, choose **Debug** > **Debug** or press [**F5**] in PSoC Creator, as shown in Figure 3-9. In debug mode, you can set breakpoints and step through the code as required.

Figure 3-9. Debugging via PSoC Creator



Detailed PSoC Creator tutorials are available at www.cypress.com/training.

3.2.2 Programming PSoC 5LP on CySmart USB Dongle

The PSoC 5LP controller on the CySmart USB dongle acts as an UART-to-USB bridge, which transfers the data received from PRoC BLE over UART to the USB interfaces listed in Table 3-3. The PSoC 5LP controller provides USB bootloader support to enable users to change the firmware.

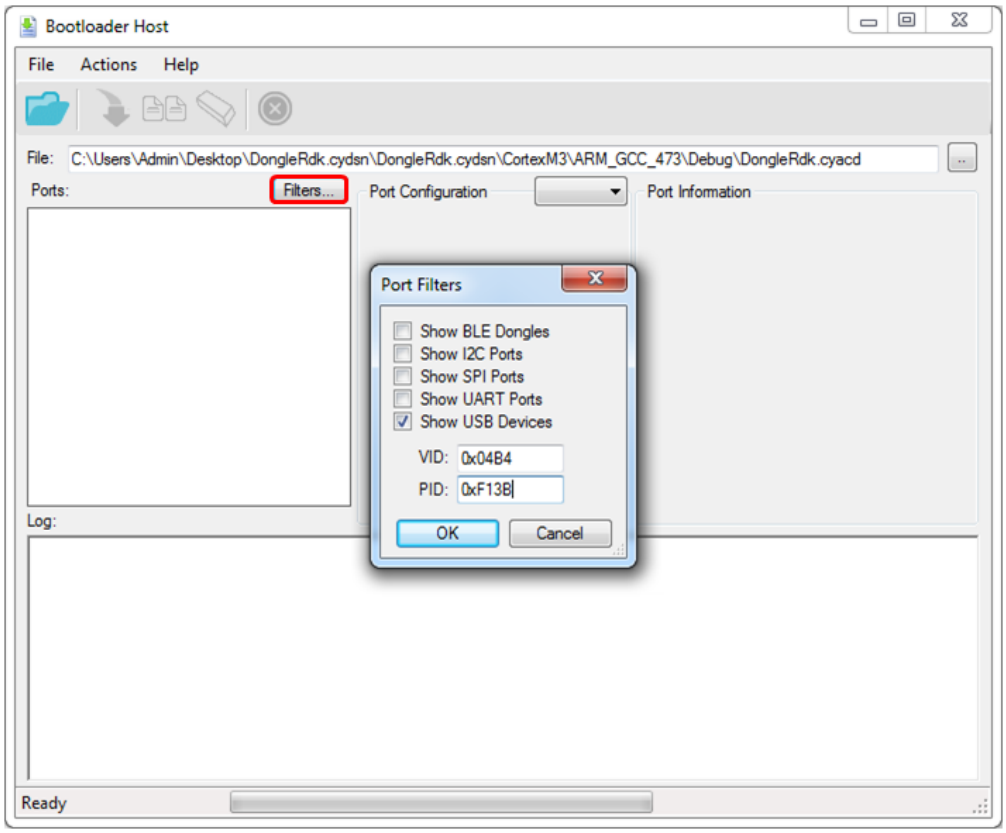
Table 3-3. Exposed CySmart USB Dongle Interfaces

Interface	Description
USB Composite Device	Generic USB device that supports multiple interfaces
USBUART (COM Port)	USB-UART bridge, to work with CySmart tool
Mouse	Standard mouse device for generic mouse functionality
Keyboard	Keyboard device for Windows based shortcuts

Follow these steps to download the bootloadable firmware onto the PSoC 5LP device.

1. Keep the reset switch (SW1) pressed and insert the CySmart USB dongle into a USB port on the PC. If the switch is pressed for more than 100 ms, the PSoC 5LP on the dongle enters into bootloader mode. This is indicated by a blinking green LED on the dongle.
2. Open the Bootloader Host tool from PSoC Creator by choosing **Tools** > **Bootloader Host**.
3. In the Bootloader Host tool, click **Filters** and add a filter to identify the USB device. Set **VID** as “0x04B4” and **PID** as “0xF13B,” and then click **OK,** as shown in Figure 3-10.

Figure 3-10. Filters Tab in Bootloader Host Tool



4. In the Bootloader Host tool, click the **Open File** button (as shown in Figure 3-11) to browse to the location of the bootloadable file *(*.cyacd)*, as shown in Figure 3-12. The bootloadable file for the firmware provided with the kit is *<Install_Directory>\CY5682 PRoC BLE Mouse RDK\1.0\Firmware\Hex Files\Dongle\ CY5682_Dongle_Bridge.cyacd.*

   Note: The **.cyacd* file is generated on building the project for PSoC 5LP, provided as part of the kit installer. This project is *<Install_Directory>\CY5682 PRoC BLE Mouse RDK\1.0\Firmware\Dongle\CY5682_Dongle_Bridge\ CY5682_Dongle_Bridge.cywrk.*

Figure 3-11. Open/Program Bootloadable File from Bootloader Host Tool

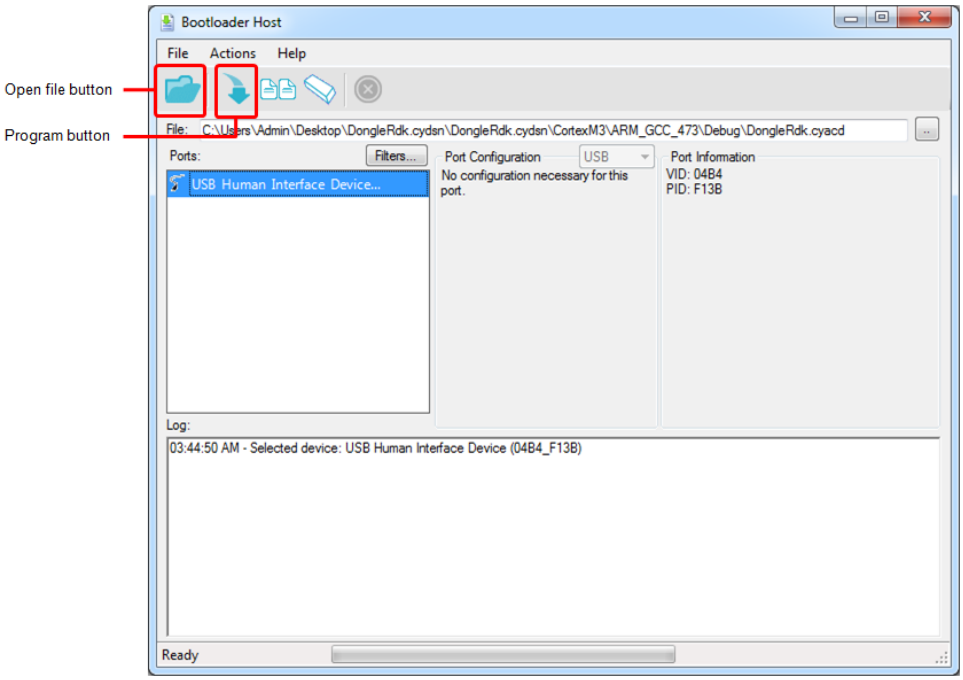
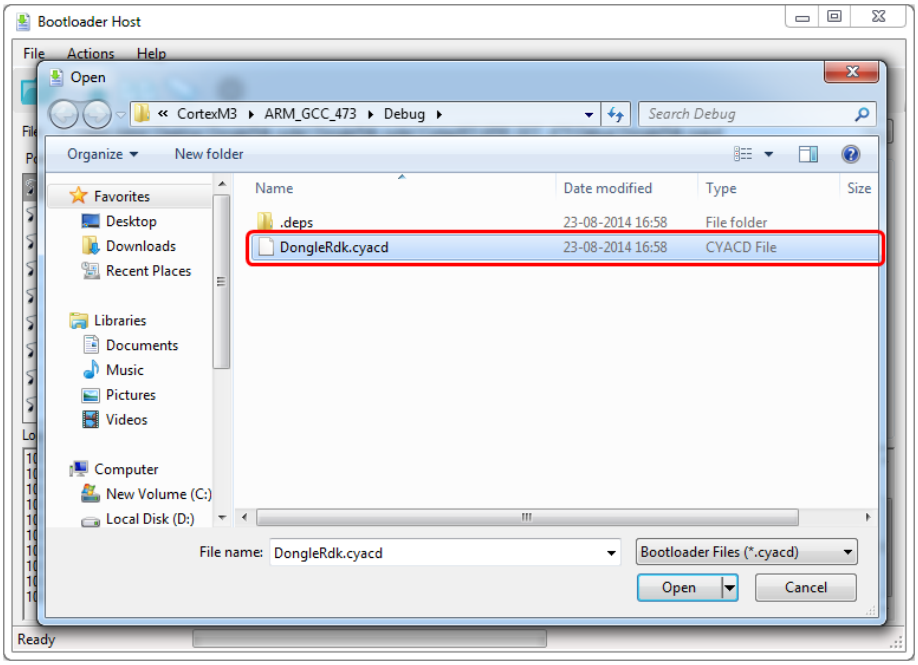


Figure 3-12. Select Bootloadable File from Bootloader Host



5. Click the **Program** button in the Bootloader Host tool to program the device, as shown in Figure 3-11.
6. If the bootload is successful, the log of the tool displays “Successful”; otherwise, it displays “Failed” with a statement describing the failure.

For additional information on bootloaders, refer to Cypress application note AN73503 – *USB HID Bootloader for PSoC 3 and PSoC 5LP*.

3.3 Using CySmart USB Dongle with CySmart Tool

When plugged into a USB port, the CySmart USB dongle enumerates, exposing the USB interfaces listed in Table 3-3. Figure 3-13 and Figure 3-14 show the drivers for the CySmart USB dongle installation on a Windows 7 64- or 32-bit system.

Figure 3-13. Driver Installation in Progress for CySmart USB Dongle on Windows 7

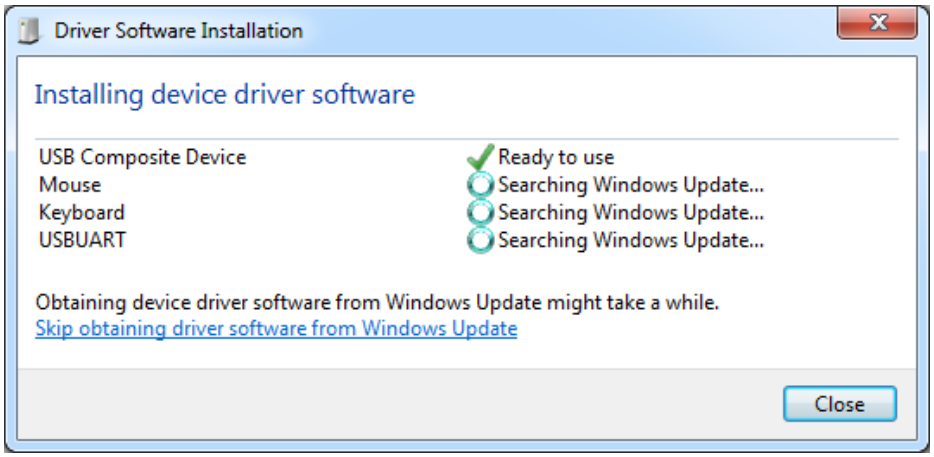
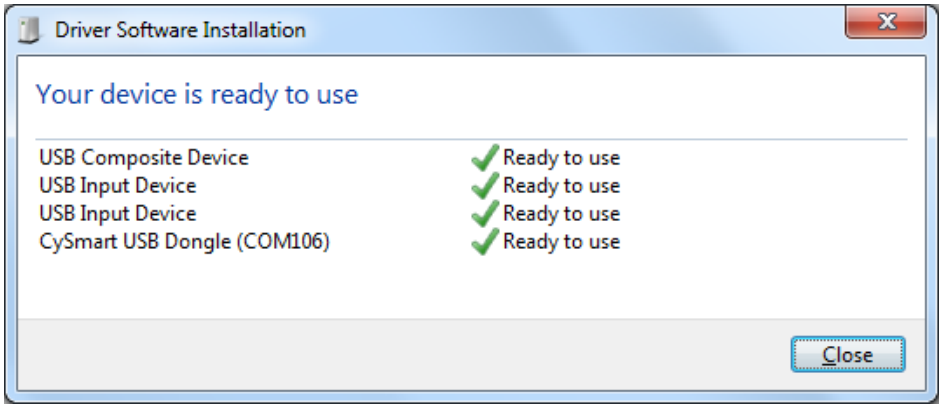


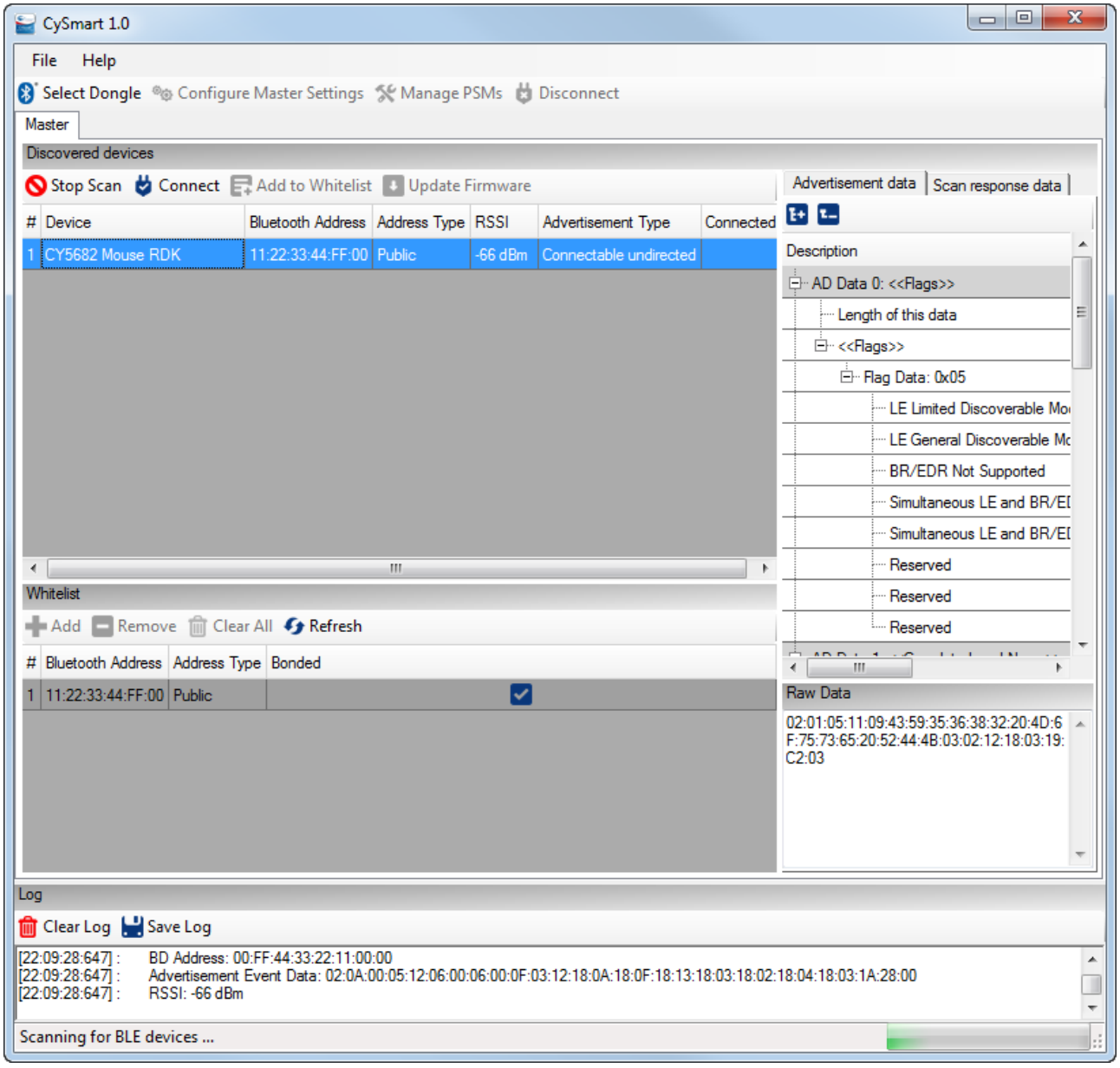
Figure 3-14. Driver Installation Complete for CySmart USB Dongle on Windows 7



Note: The appearance of the windows shown in Figure 3-13 and Figure 3-14 may differ across operating systems and based on settings.

CySmart is a Bluetooth LE host emulation tool for Windows PCs. It provides an easy-to-use GUI that enables customers to test and debug their Bluetooth LE peripheral applications. The tool supports device discovery, connection establishment, pairing and bonding, and it provides the ability to access the Generic Attribute Profile (GATT) server database for the PRoC BLE touch mouse. Figure 3-15 shows a screen shot of the CySmart tool.

Figure 3-15. CySmart Tool



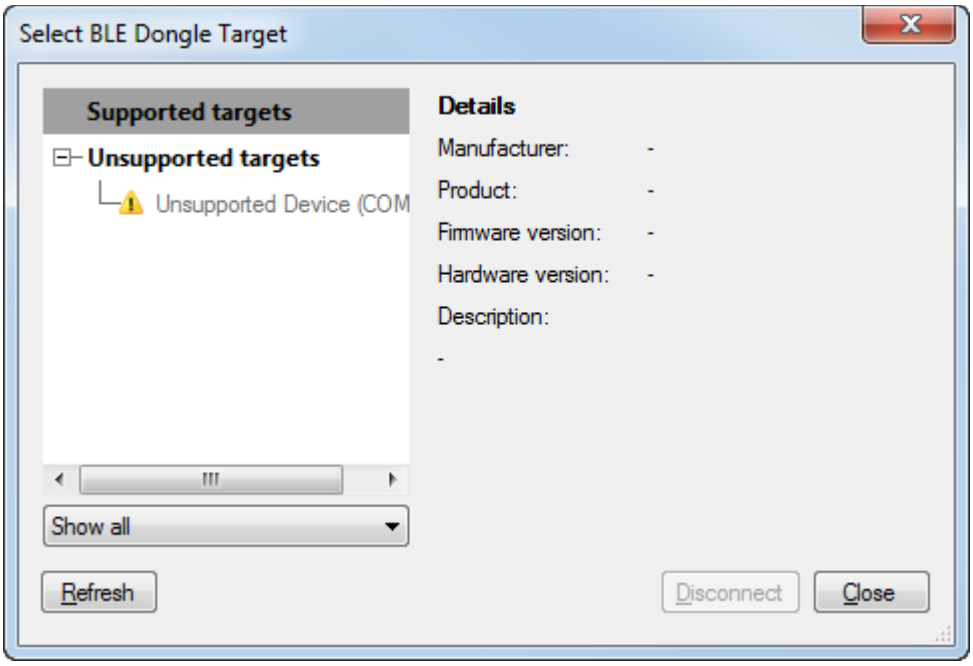
The CySmart tool requires the CySmart USB dongle to be connected to the PC, as shown in Figure 3-16.

Figure 3-16. CySmart USB Dongle Connected to PC

3.3.1 Connecting CY5682 Mouse RDK to CySmart

1. Plug the CySmart USB dongle into the PC.
2. Open CySmart and click on **Select Dongle** button. A window appears as shown in Figure 3-17.

Figure 3-17. BLE Dongle Target Window



3. Click on the **Refresh** button and verify that the Cypress BLE HID dongle is listed as shown in Figure 3-18. Click on "Cypress BLE HID Dongle (COM100)." CySmart tool appears as shown in Figure 3-19.

Figure 3-18. BLE Dongle Target Window

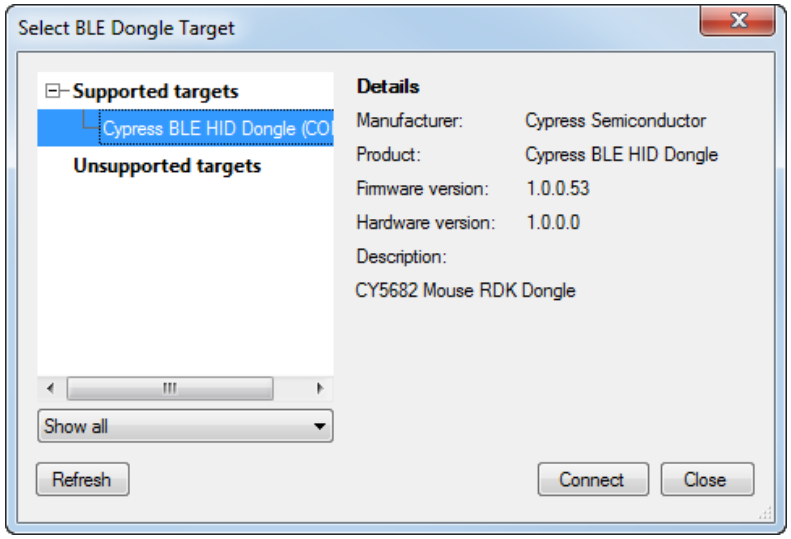
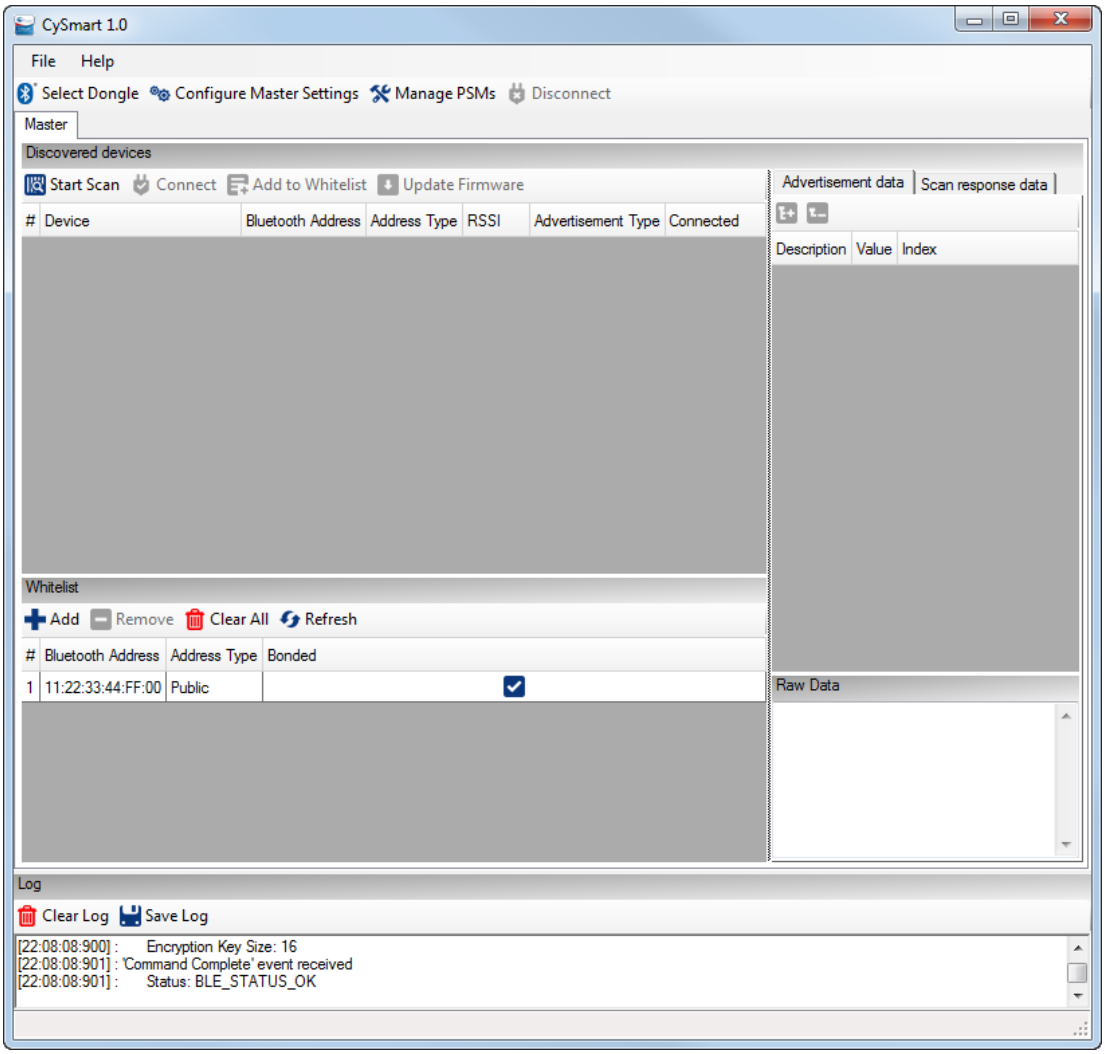


Figure 3-19. CySmart Window



4. Insert two AAA batteries that are provided with the kit into the battery holder on the touch mouse.

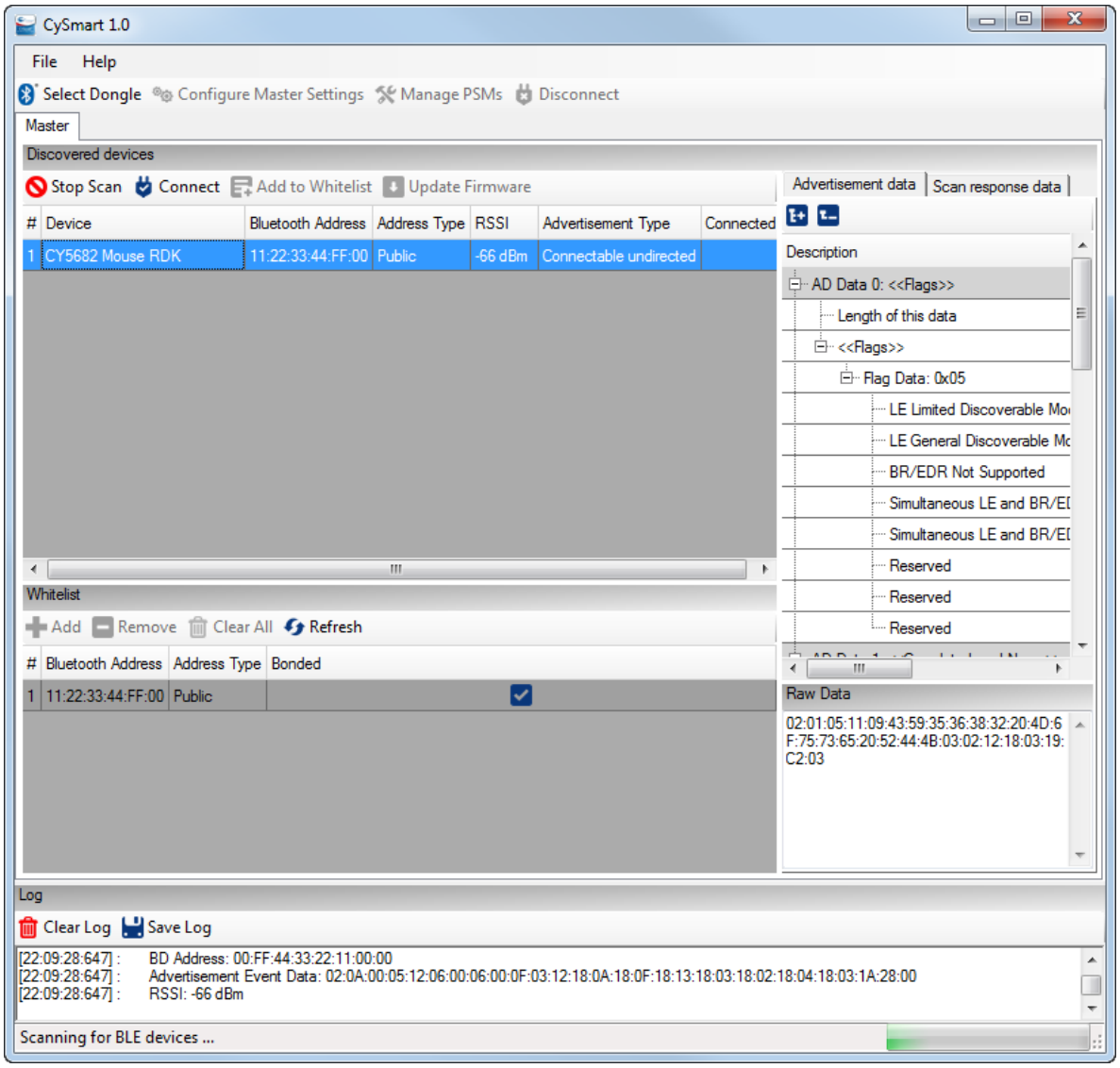
   Note: The touch mouse can also work with a single AAA battery; however, using two AAA batteries is recommended to prolong battery life.

5. Set the ON/OFF switch on the touch mouse to the ON position.

6. Press the connect button on the touch mouse. The orange LED shows the breathing effect to indicate that the touch mouse is now in the advertising mode.

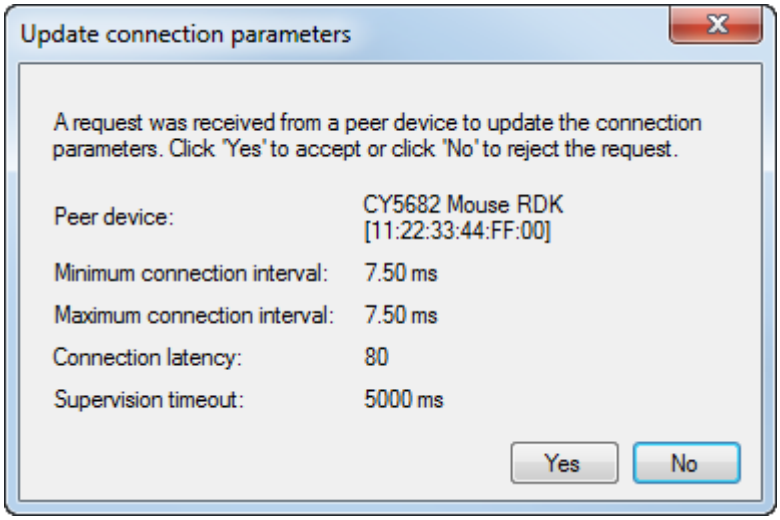
7. Click on **Start Scan** button on the CySmart tool to start scanning for BLE devices that are advertising. Verify that "CY5682 Mouse RDK" is listed as shown in Figure 3-20.

Figure 3-20. CySmart Window with the List of Bluetooth Devices



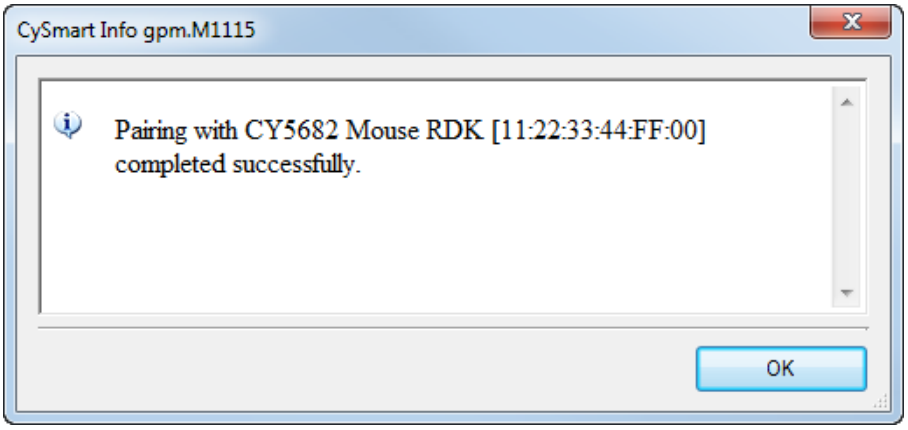
8. Double-click on CY5682 Mouse RDK. Verify that the "Update connection parameters" window appears as shown in Figure 3-21. Click **Yes**.

Figure 3-21. Update Connection Parameters Window



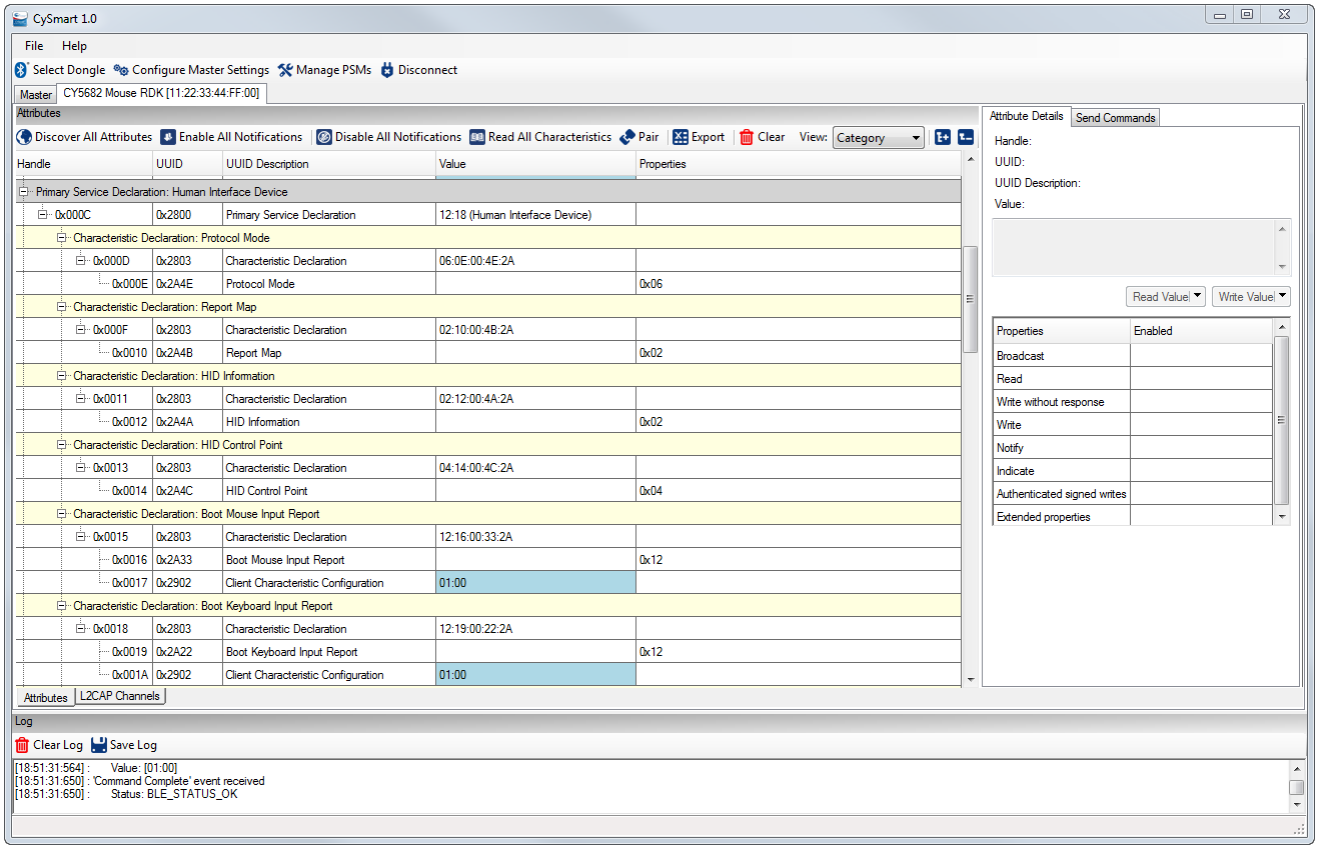
9. Set the **Security level:** “Unauthenticated with Encryption” in **Configure Master Settings** > **Security Parameters** as described in the CySmart User Guide. Press the **Pair** button and verify that the Pairing Successful window appears as shown in Figure 3-22.

Figure 3-22. Pairing Successful Window



10. Click on the **Discover All Attributes** button and verify that all attributes are discovered.
11. Click on the **Enable All Notifications** button and verify that notifications are enabled as shown in Figure 3-23.

Figure 3-23. CySmart Tool with Notifications Enabled



12. Move the touch mouse and verify that notifications are received on the CySmart tool.

For detailed instructions on using the CySmart tool, download the guide for the CySmart tool from www.cypress.com/go/CySmart.

4. Hardware

This chapter describes the CY5682 RDK hardware.

Note: “Module” in this chapter refers to a self-contained assembly of electronic components and circuitry such as the trackpad module.

4.1 Board Details

The PRoC BLE touch mouse has two boards connected via a 15-pin flexible flat cable (FFC):

- Main board
- Trackpad module

The main board consists of the following key components:

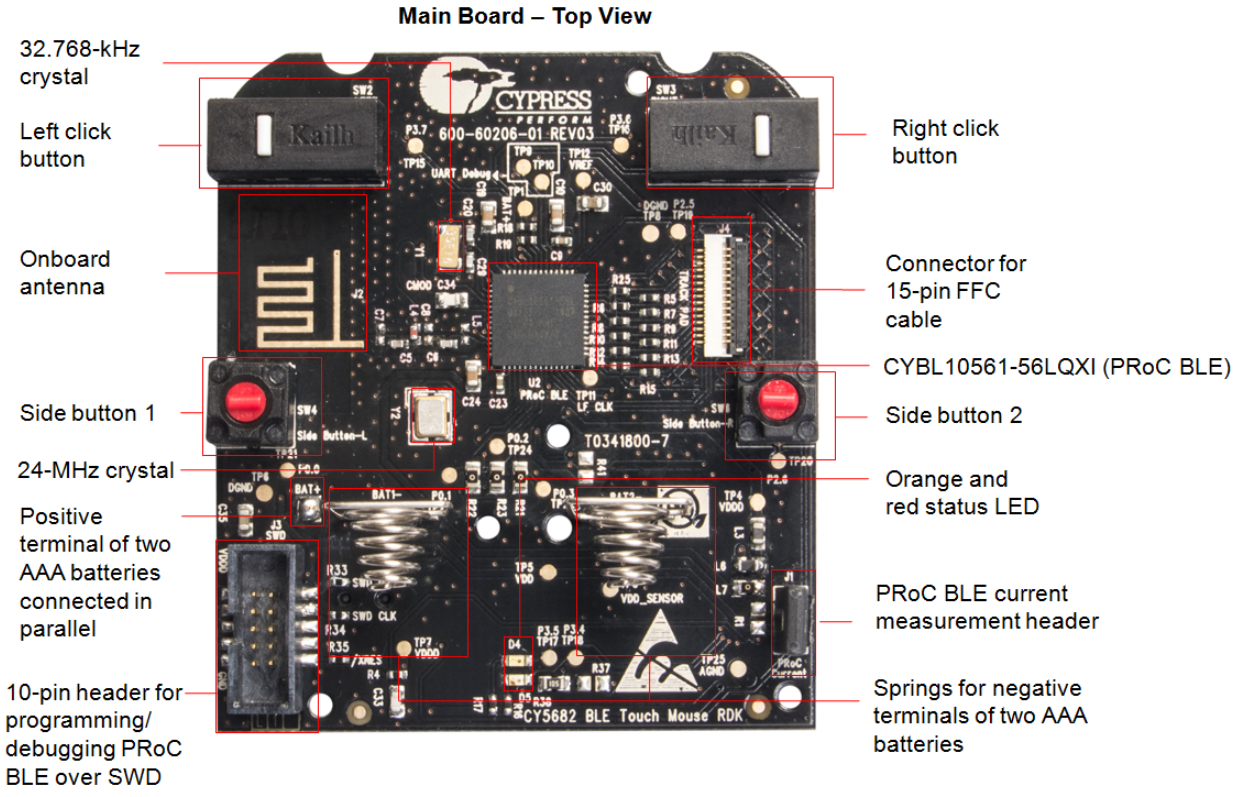
- CYBL10561-56LQXI (PRoC BLE)
- Boost regulator with two AAA batteries in parallel through a switch
- Optical sensor with IR LED for sensing mouse movements
- Overvoltage protection circuit and crystals for PRoC BLE
- 10-pin header for programming and debugging PRoC BLE using SWD
- Connector for 15-pin FFC cable (to connect to the trackpad module)
- Onboard antenna, including an antenna-matching network
- Status LED (orange and red) for indications
- ON/OFF single pole, double throw (SPDT) switch
- Five buttons:
    - Left button
    - Right button
    - Connect button
    - Side button 1
    - Side button 2

The trackpad module board consists of the following key components:

- Trackpad sensors
- Buttons
    - Windows button
    - Middle button
- Connector for 15-pin FFC cable (to connect to the main board)

Top and bottom views of the main board and the trackpad module, with several key components marked, are shown in Figure 4-1 and Figure 4-2 respectively.

Figure 4-1. CY5682 PRoC BLE Touch Mouse Main Board and Trackpad Module Top View (with FFC Cable)



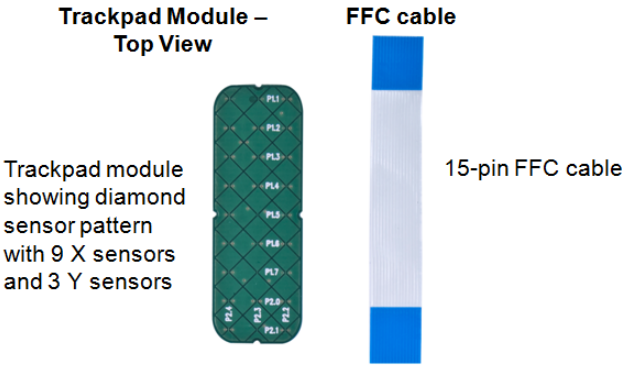
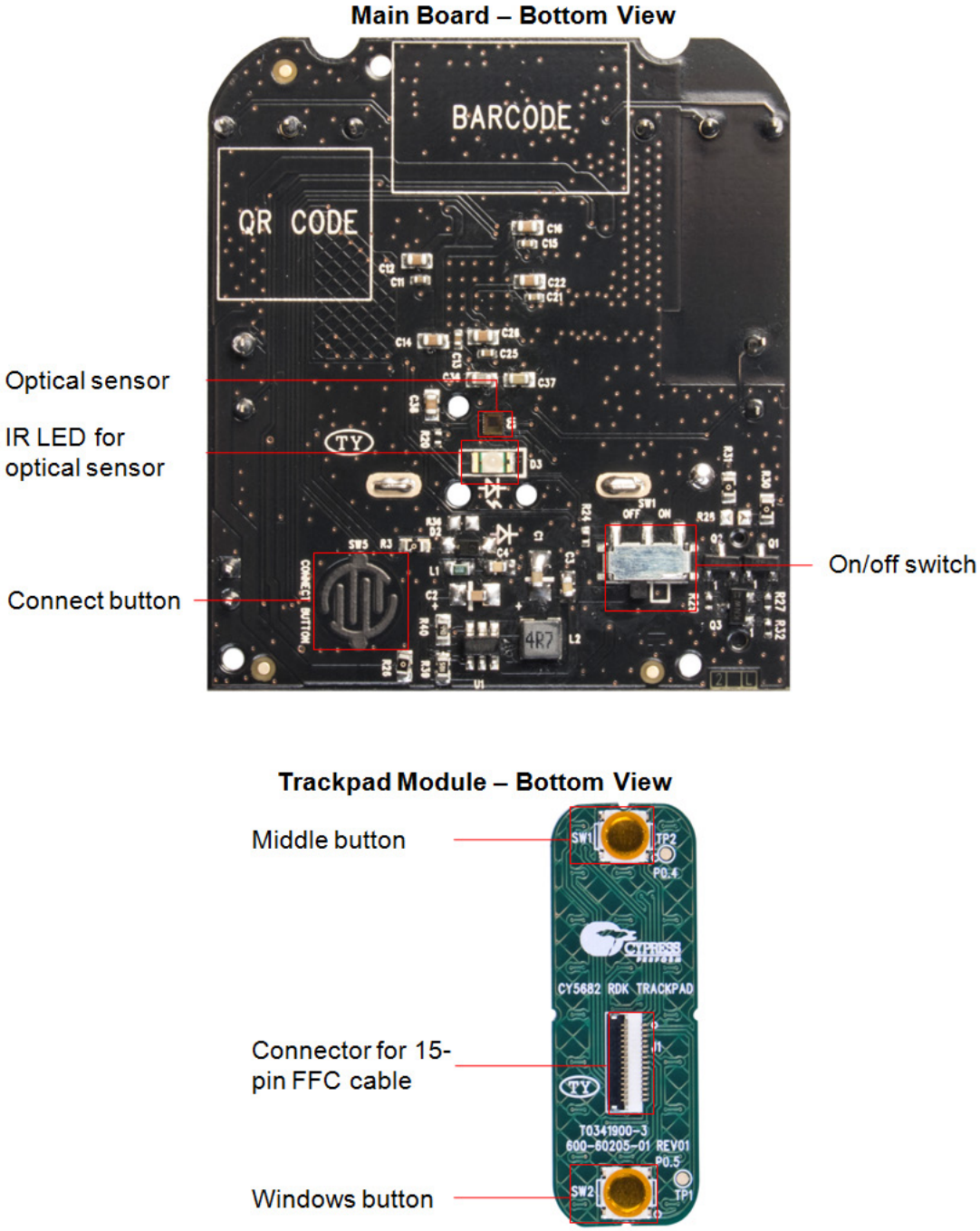


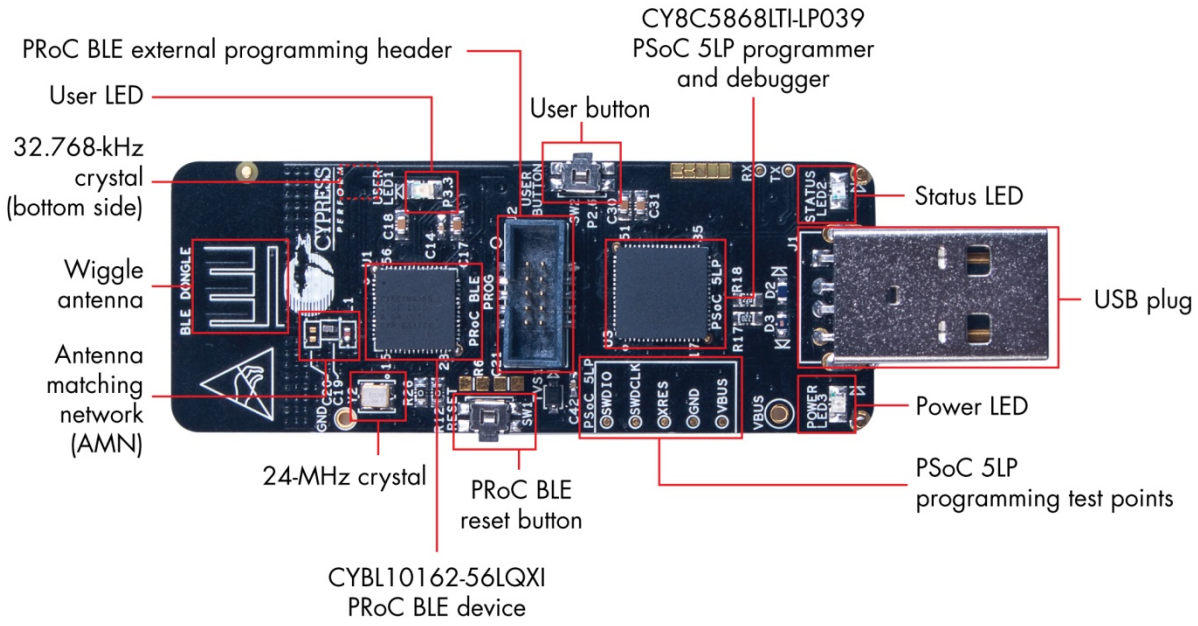
Figure 4-2. CY5682 PRoC BLE Touch Mouse Main Board and Trackpad Module Bottom View



The CySmart USB dongle board consists of the following key components, as shown in Figure 4-3:

- CYBL10162-56LQXI PRoC BLE controller
- Reset and user buttons
- PSoC 5LP controller
- 10-pin header for programming and debugging PRoC BLE using SWD
- Onboard antenna for BLE, including an antenna-matching network
- USB 2.0 type-A plug (USB plug)
- Power, status, and user LEDs
- 24-MHz and 32.768-kHz crystals

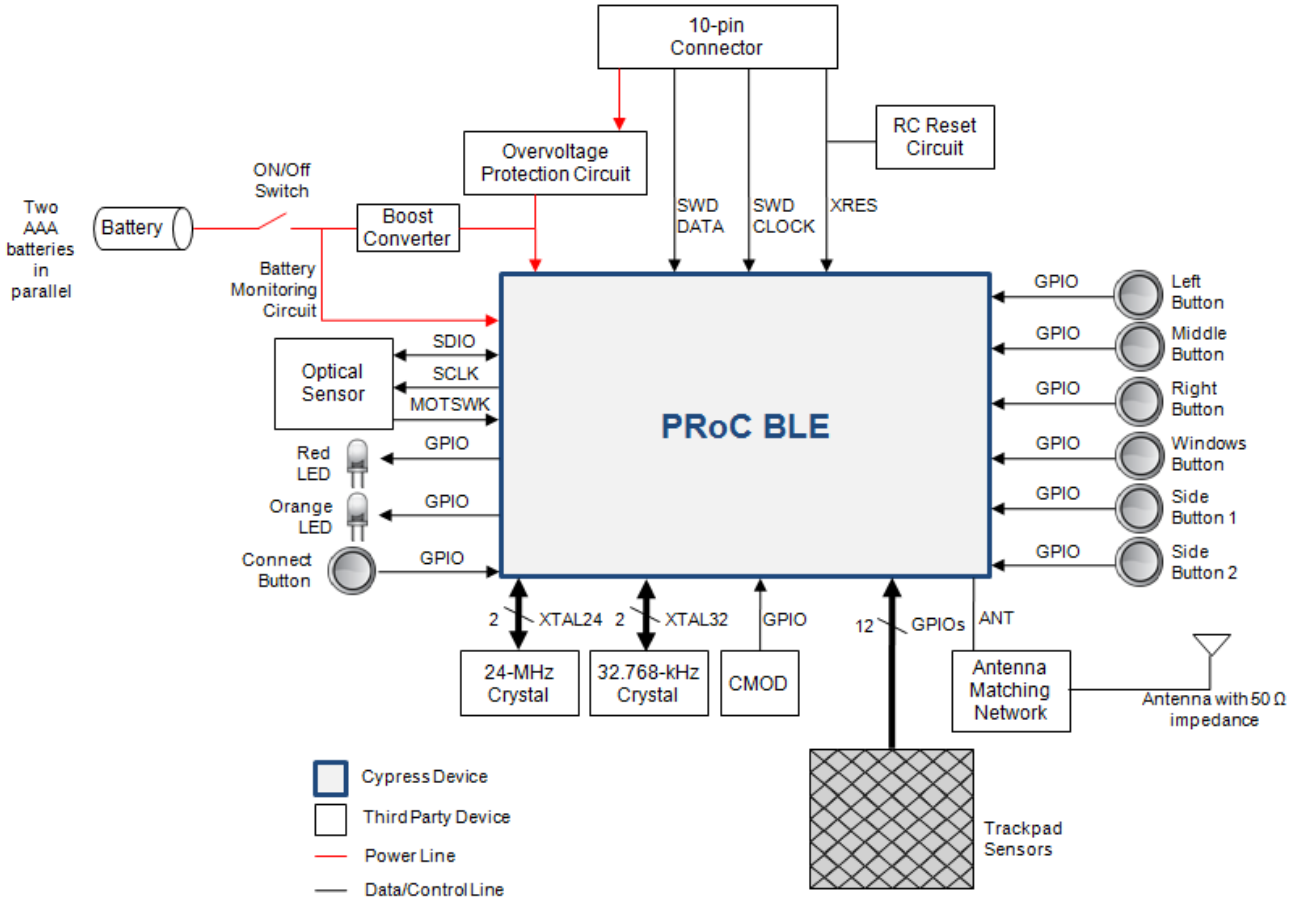
Figure 4-3. CySmart USB Dongle



4.2 Theory of Operation

The CY5682 RDK contains a PRoC BLE–based touch mouse device and a PRoC BLE–based CySmart USB dongle. Figure 4-4 shows the block diagram for the PRoC BLE touch mouse (covers the PRoC BLE touch mouse main board and trackpad module).

Figure 4-4. PRoC BLE Touch Mouse Block Diagram

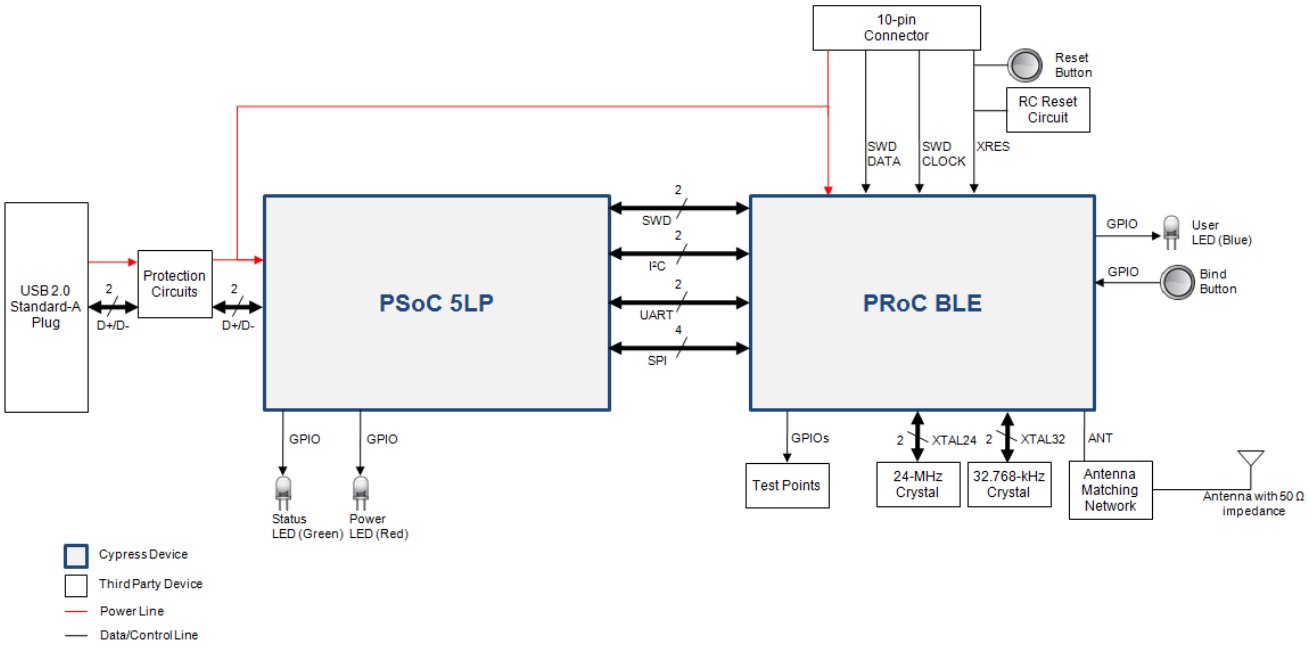


The touch mouse is based on the CYBL10561-56LQXI PRoC BLE controller. It includes the following features:

- Capacitive touch sensing area or trackpad for vertical scroll (Z-wheel) and horizontal scroll (H-wheel), and an optical sensor for cursor movement
- Mechanical buttons for left-click, right-click, and middle-click; the Windows button; and two side buttons (side button 1 and side button 2)
  - Side button 1: Triggers [**Win**] [**Ctrl**] [**Backspace**] (to toggle across open apps in Windows 8.0/8.1)
  - Side button 2: Triggers [**Win**] [**C**] (to open the Charms bar in Windows 8.0/8.1)
- ON/OFF switch
- Boost converter onboard to boost the supply from two AAA batteries in parallel to 2.3 V
- Headers for programming/debugging and for measuring current across PRoC BLE

The CySmart USB dongle is a Bluetooth Smart-to-USB bridge for USB-enabled devices (PCs, smart TVs, tablets, and so on). The dongle includes a PRoC BLE device, which is programmed to receive data from the touch mouse over a Bluetooth Smart link. It also contains a PSoC 5LP device, which functions as a serial-to-USB bridge by transferring the data received from PRoC BLE over UART to the PC over USB, as shown in Figure 4-5.

Figure 4-5. CySmart Dongle Block Diagram



The CySmart USB dongle is based on the CYBL10162-56LQXI PRoC BLE controller. It is intended to make it as easy as possible for Cypress customers/distributors to get started with Cypress's PRoC BLE–based reference designs (CY5682 PRoC BLE Touch Mouse RDK and CY5672 PRoC BLE Remote Control RDK). The dongle can be connected to any device such as a PC or a smart TV that supports HID over USB. Note that the PRoC BLE touch mouse can also be directly connected to any Bluetooth Smart Ready device without the use of the dongle; see the Connecting PRoC BLE Touch Mouse with Bluetooth Smart Ready Device section for details.

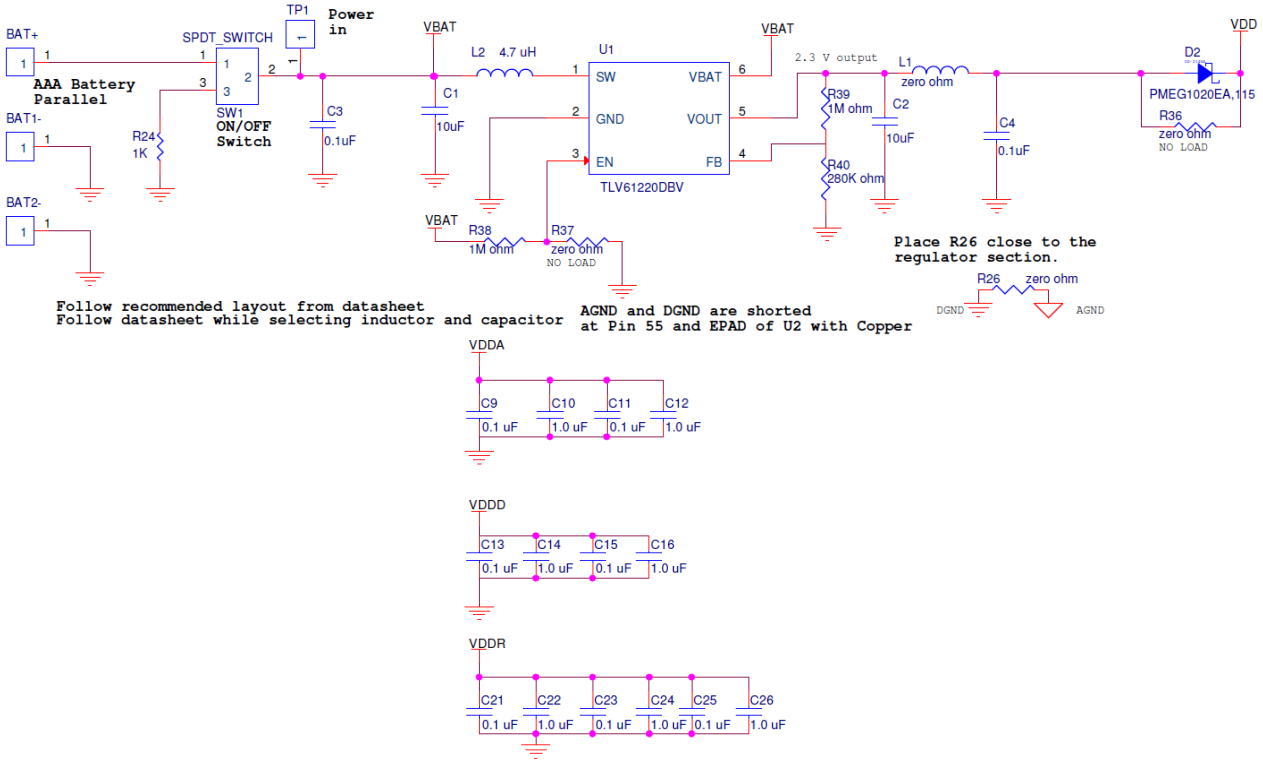
The CySmart dongle has a USB A-type plug to connect the PSoC 5LP device to the USB port of the host PC. The PSoC 5LP device then communicates with the PRoC BLE device over UART, multiplexed I^2C, or SPI. The firmware provided with the kit installer uses the UART interface. The dongle has an onboard printed wiggle antenna. It also has a user LED, a user button, and a reset button for the PRoC BLE device. The CySmart USB dongle is powered directly through the USB port (VBUS) at 5.0 V. PSoC 5LP enables bootloading over USB to change the firmware for PSoC 5LP. See the Programming PSoC 5LP on CySmart USB Dongle section for details on bootloading PSoC 5LP.

4.3 Functional Description – PRoC BLE Touch Mouse

4.3.1 Power Supply

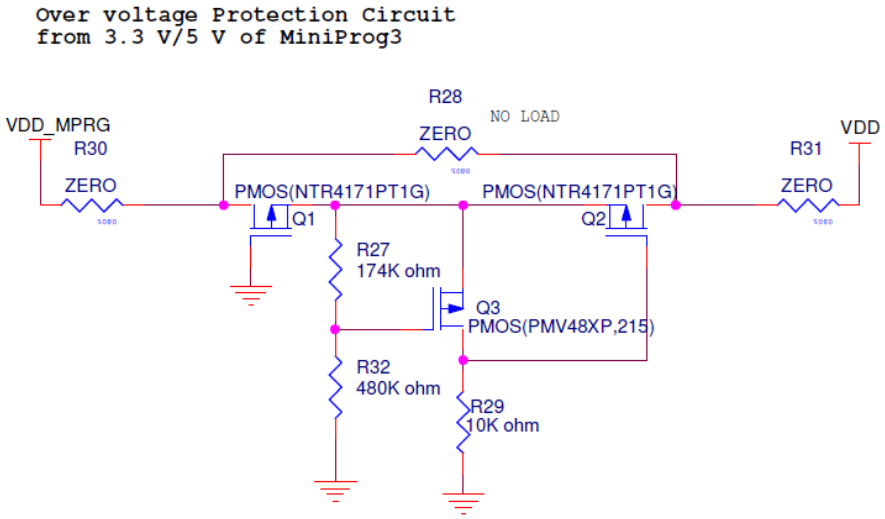
The touch mouse is powered by two AAA batteries connected in parallel to provide an output voltage of 1.5 V. Figure 4-6 shows the boost regulator circuit that boosts 1.5 V to 2.3 V. The touch mouse includes notification LEDs, which require an operating voltage of 2.3 V. The touch mouse can be turned ON/OFF using the SPDT switch SW1.

Figure 4-6. Power Supply



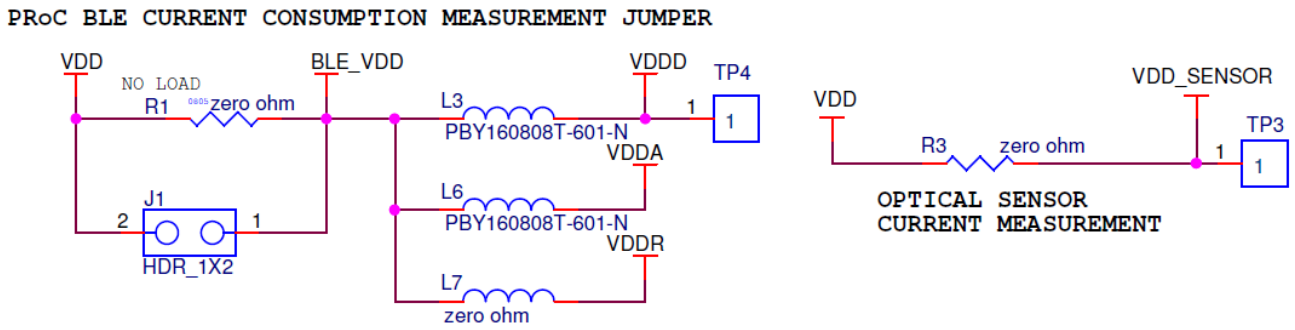
The overvoltage protection circuit shown in Figure 4-7 ensures that using MiniProg3 at 5 V for programming does not damage the components such as the optical sensor in the system.

Figure 4-7. Overvoltage Protection Circuit



The current consumption for the PRoC BLE device and optical sensor can be measured by using the circuits shown in Figure 4-8. See the Current and Voltage Measurement section for more information.

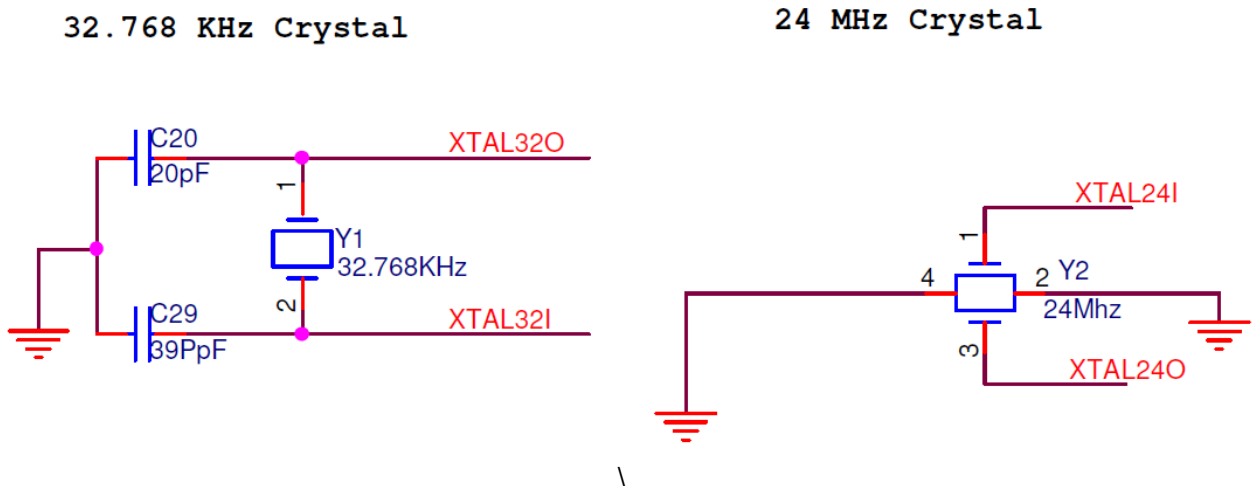
Figure 4-8. Current Measurement Circuit



4.3.2 Clock and Reset

The PRoC BLE device requires two crystal oscillators, one running at 24 MHz and the other at 32.768 kHz, as shown in Figure 4-9. Both oscillators are in the Pierce configuration. The 24-MHz oscillator is used by the Bluetooth Smart radio in the Active state, whereas the 32.768-kHz oscillator is used by Bluetooth Smart radio in low power modes.

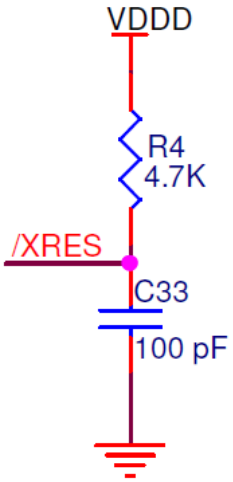
Figure 4-9. Clock Design



\

The RC reset circuit shown in Figure 4-10 provides the power-on reset pulse for the PRoC BLE device.

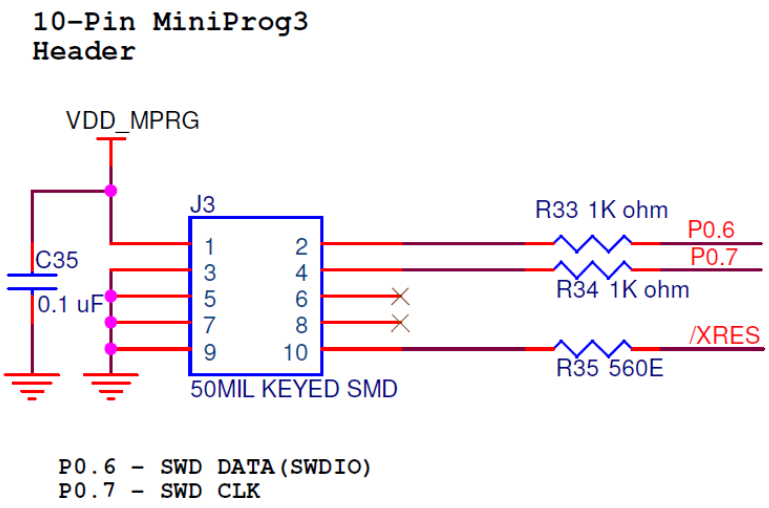
Figure 4-10. Reset Circuit



4.3.3 Program and Debug Circuit

PRoC BLE exposes the SWD interface on the 10-pin header J3, as shown in Figure 4-11. The Programming and Debugging PRoC BLE on Touch Mouse and CySmart USB Dongle section describes how to use the J3 header to debug or program PRoC BLE on the touch mouse.

Figure 4-11. SWD Debug Header

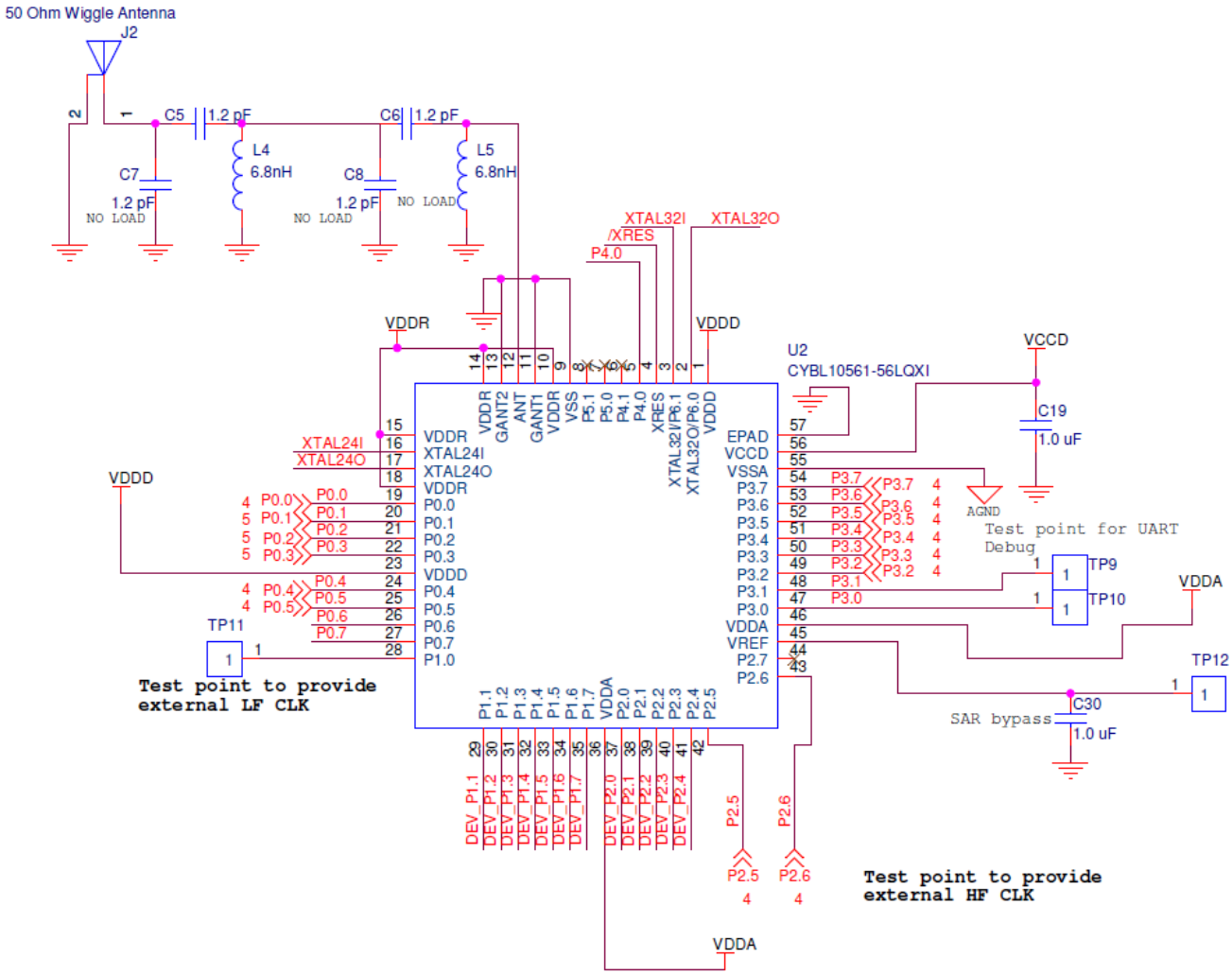


4.3.4 PRoC BLE Device

The CYBL10561-56LQXI PRoC BLE is a 32-bit, 48-MHz ARM Cortex-M0 solution with CapSense®, 12-bit SAR ADC, 4 time/counter/pulse-width modulators (TCPWMs), 36 GPIOs, 2 serial communication blocks (SCBs), an LCD direct drive, inter-IC sound (I^2S), and an integrated Bluetooth Smart radio with a balun. PRoC BLE includes a royalty-free BLE stack compatible with Bluetooth 4.1 and provides a complete, programmable, and flexible solution for HID. In addition, PRoC BLE provides a simple, low-cost way to add BLE connectivity to any existing system.

GPIOs and serial interfaces are used to create the essential components of a working mouse such as an optical sensor, a trackpad, buttons, and LEDs with PRoC BLE. Data collected from peripherals of the mouse is transmitted over the air using the Bluetooth Smart link. The PRoC BLE device is connected to a wiggle antenna through an LC filter circuit for optimum RF performance. Figure 4-12 shows the hardware connections for the PRoC BLE device.

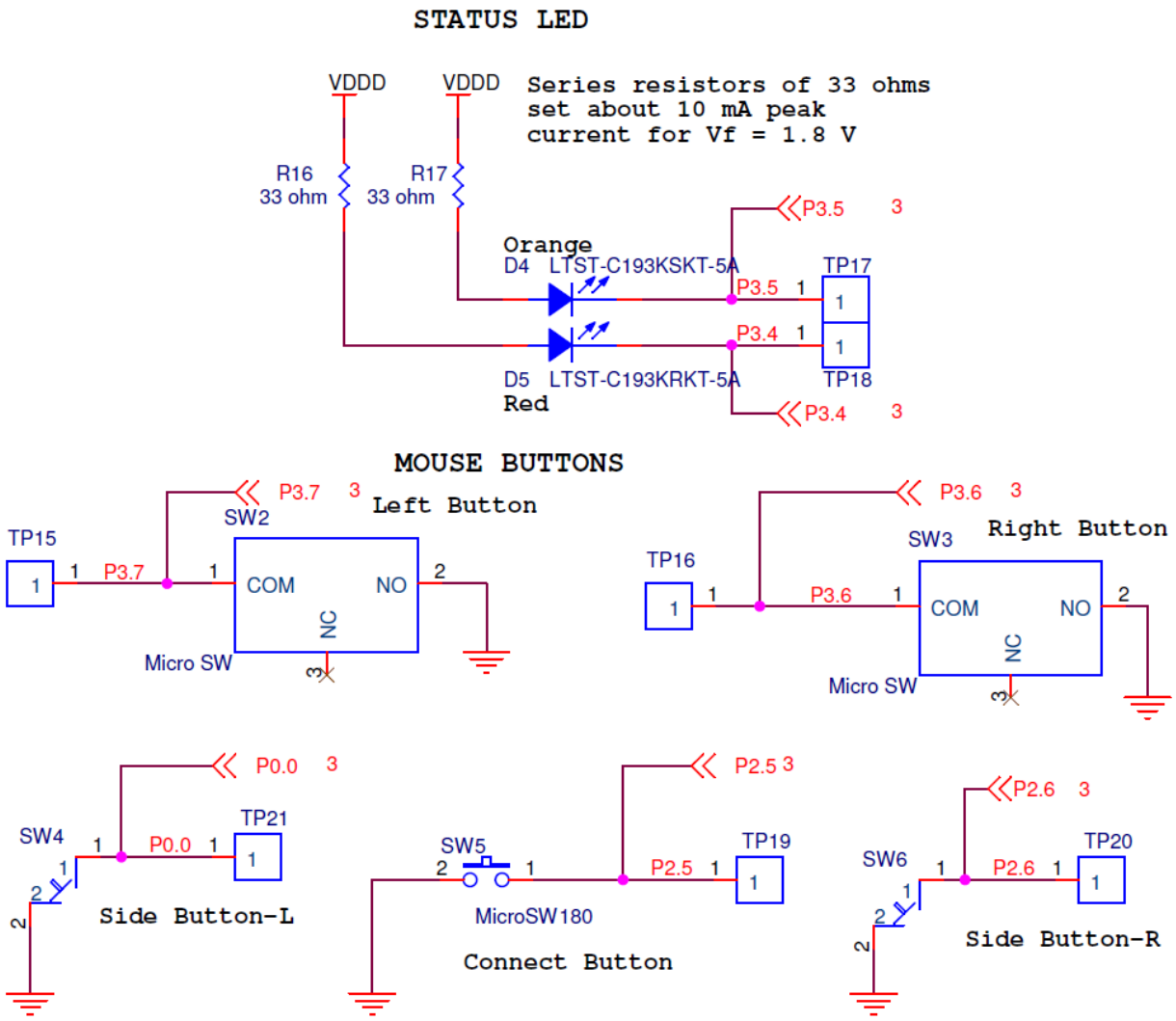
Figure 4-12. PRoC BLE Connections



4.3.5 LEDs and Mouse Buttons

The mouse has two LEDs, as shown in Figure 4-13. These LEDs can indicate different notifications depending on the firmware state. The LED Subsystem section describes the indications displayed by the firmware provided with the kit. The mouse includes seven buttons, as shown in Figure 4-13, of which five are on the main board. The other two buttons are on the trackpad module, and the hardware connections for them are shown in the Trackpad Interface section. The Button Subsystem section describes the functions for each button according to the firmware provided with the kit.

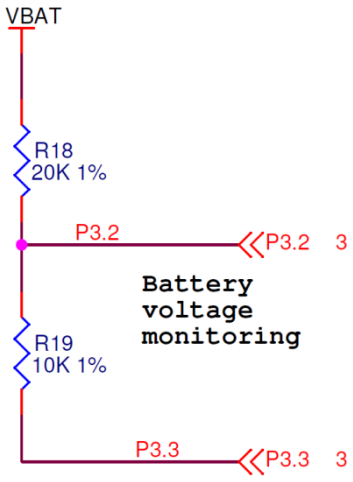
Figure 4-13. Status LEDs and Mouse Buttons



4.3.6 Battery Monitoring

The battery monitoring circuit is a voltage divider circuit, as shown in Figure 4-14. The input to this circuit is VBAT (battery voltage), and the output is connected to pin 3.2 of PRoC BLE. The GND end of the divider is connected to pin 3.3 of PRoC BLE. With this feature, the firmware can control when the divider is active or inactive. This helps to measure the battery status by connecting the voltage divider to GND and to prevent leakage current during all other times by disconnecting it from GND.

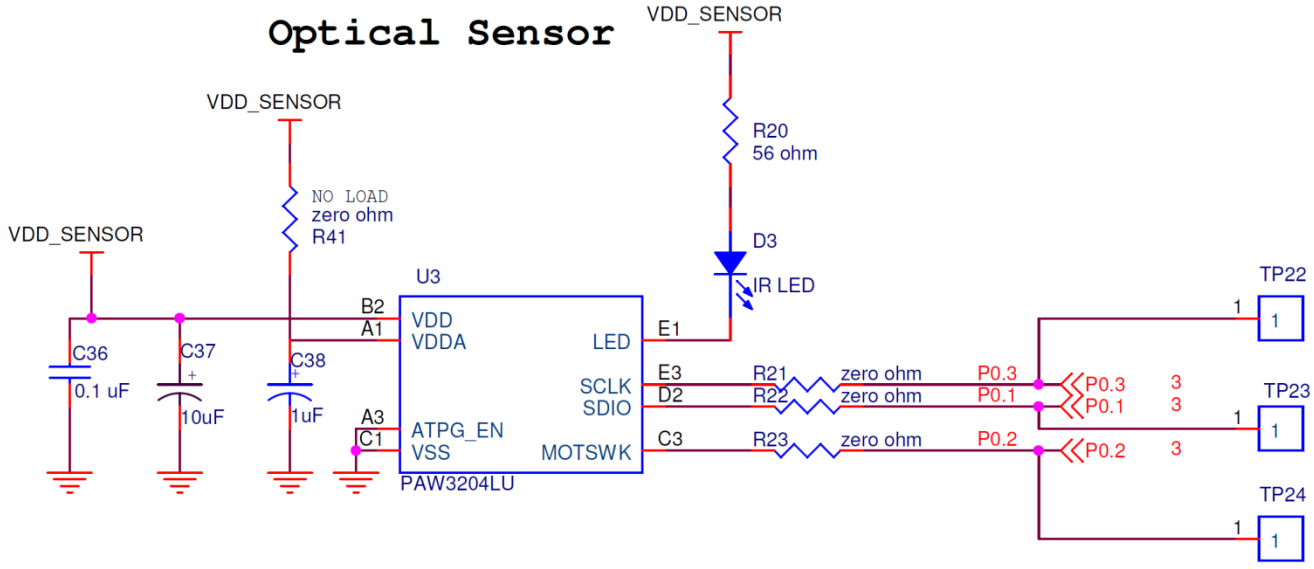
Figure 4-14. Battery Monitoring



4.3.7 Optical Sensor

The optical sensor (PAW3204LU) captures the mouse movement information using an IR LED and uses image processing to determine the distance and direction of mouse movement (X and Y coordinates relative to the previous location). The optical sensor then provides an interrupt using the MOTSWK pin, which is connected to pin P0.2 of the PRoC BLE device. PRoC BLE, on receiving the interrupt, reads the X and Y coordinates over the two-wire serial interface shown in Figure 4-15. Finally, PRoC BLE sends the coordinates over the Bluetooth Smart link.

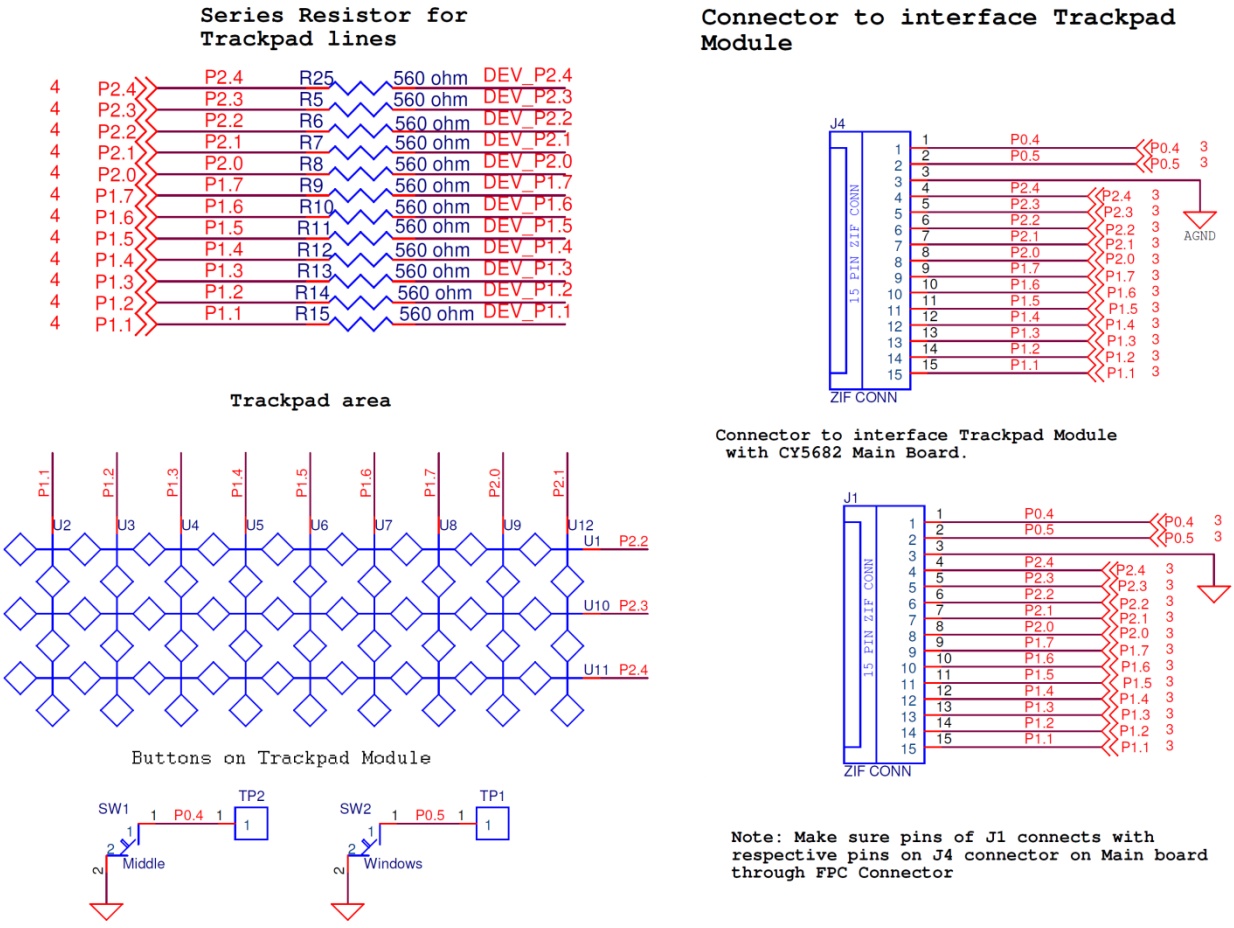
Figure 4-15. Optical Sensor



4.3.8 Trackpad Interface

This section describes the implementation of the trackpad module and its interface with the PRoC BLE device, as shown in Figure 4-16. The capacitive sensor array from the trackpad module is connected to the PRoC BLE GPIOs through series resistors, and they are internally configured as CapSense input lines. The CapSense Gesture Component in PRoC BLE scans and processes these lines to determine the coordinates for finger movement on the trackpad. Note that the trackpad module also has the middle-click and Windows buttons. Both the trackpad module and the main board have a 15-pin connector. An FFC cable plugs into these connectors to connect the boards.

Figure 4-16. Trackpad Module and Associated Connectors



4.3.9 Test Points

Table 4-1 lists the test points available on the CY5682 hardware and the associated signal names.

Table 4-1. Test Points

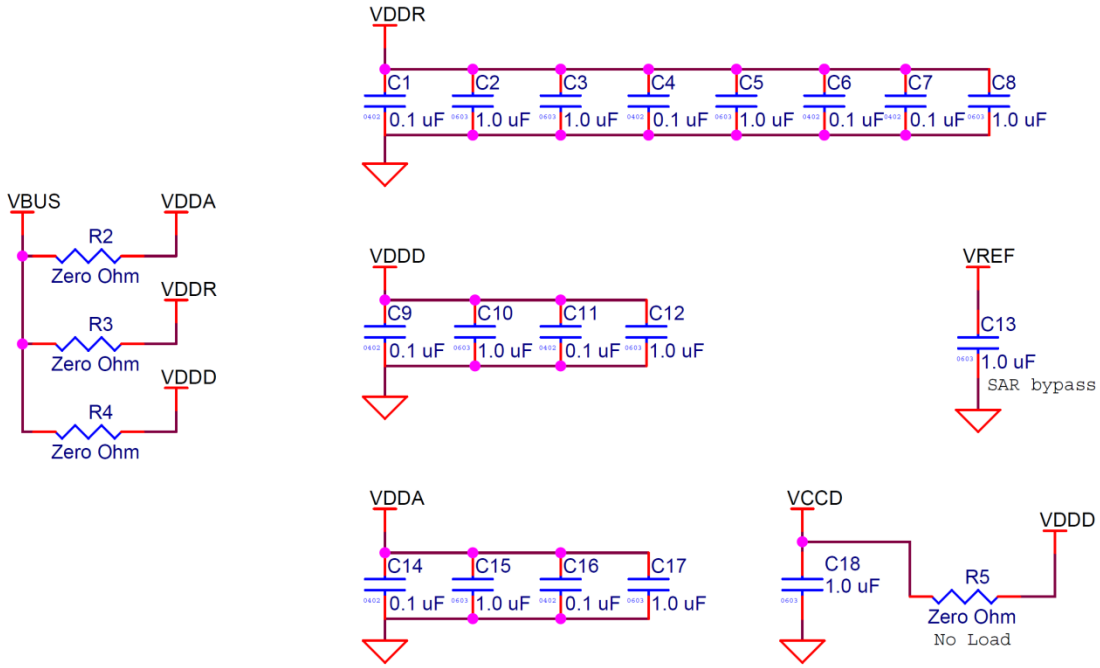
Test Point	Signal Name
TP1	AAA battery output through the SPDT switch
TP3	VDD_SENSOR
TP4	VDDD for PROC BLE current measurement
TP5	System power VDD
TP6	System power GND
TP7	PRoC BLE device power VDDD
TP8	PRoC BLE device power GND
TP9	UART Rx debug
TP10	UART Tx debug
TP11	External LF CLK
TP12	External HF CLK
TP15	SW2 or Left button
TP16	SW3 or Right button
TP17	Orange LED
TP18	Red LED
TP19	SW5 Connect button
TP20	SW6 Side button 2
TP21	SW4 Side button 1
TP22	Optical Sensor SCLK
TP23	Optical Sensor SDIO
TP24	Optical Sensor MOTSWK
TP25	AGND for CMOD capacitor

4.4 Functional Description – CySmart USB Dongle

4.4.1 Power Supply

All devices on the CySmart USB dongle are powered using the 5-V power supply from USB (VBUS). The decoupling capacitors and the current measurement circuitry are shown in Figure 4-17. The decoupling capacitors decouple the noise present in the power supply.

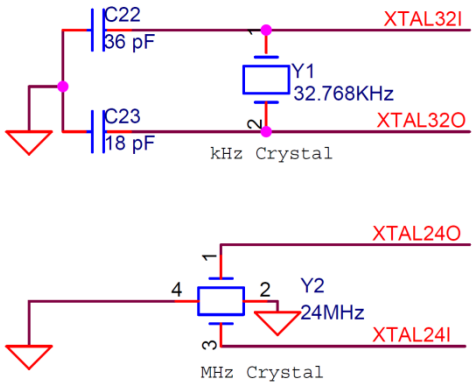
Figure 4-17. Decoupling Capacitors and Current Measurement Circuit



4.4.2 Clock and Reset

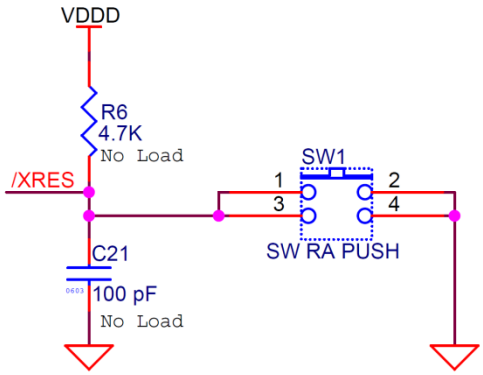
The PRoC BLE device requires two crystal oscillators, one running at 24 MHz and the other at 32.768 kHz, as shown in Figure 4-18. Both oscillators are in the Pierce configuration. The 24-MHz oscillator is used by the Bluetooth Smart radio in the Active state, whereas the 32.768-kHz crystal is used to maintain Bluetooth Smart link timing in various sleep modes.

Figure 4-18. Crystal Circuit



The RC reset circuit for PRoC BLE, including the reset button SW1, is shown in Figure 4-19. The PRoC BLE device is reset when SW1 is pressed.

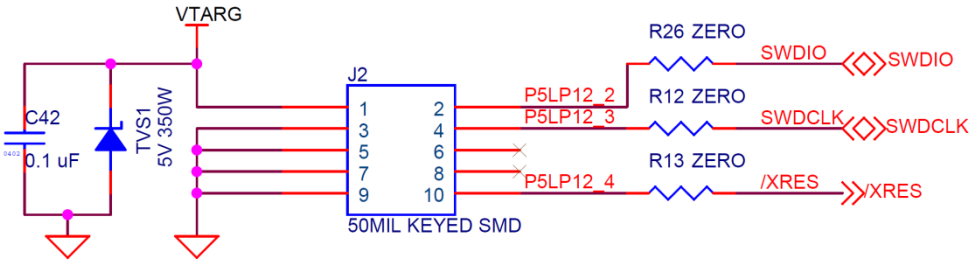
Figure 4-19. Reset Circuit



4.4.3 Program and Debug Circuit

PRoC BLE exposes the SWD interface on the 10-pin header J2, as shown in Figure 4-20. The Programming and Debugging PRoC BLE on Touch Mouse and CySmart USB Dongle section describes how to use the J2 header to debug or program PRoC BLE on the CySmart USB dongle.

Figure 4-20. SWD Debug Header

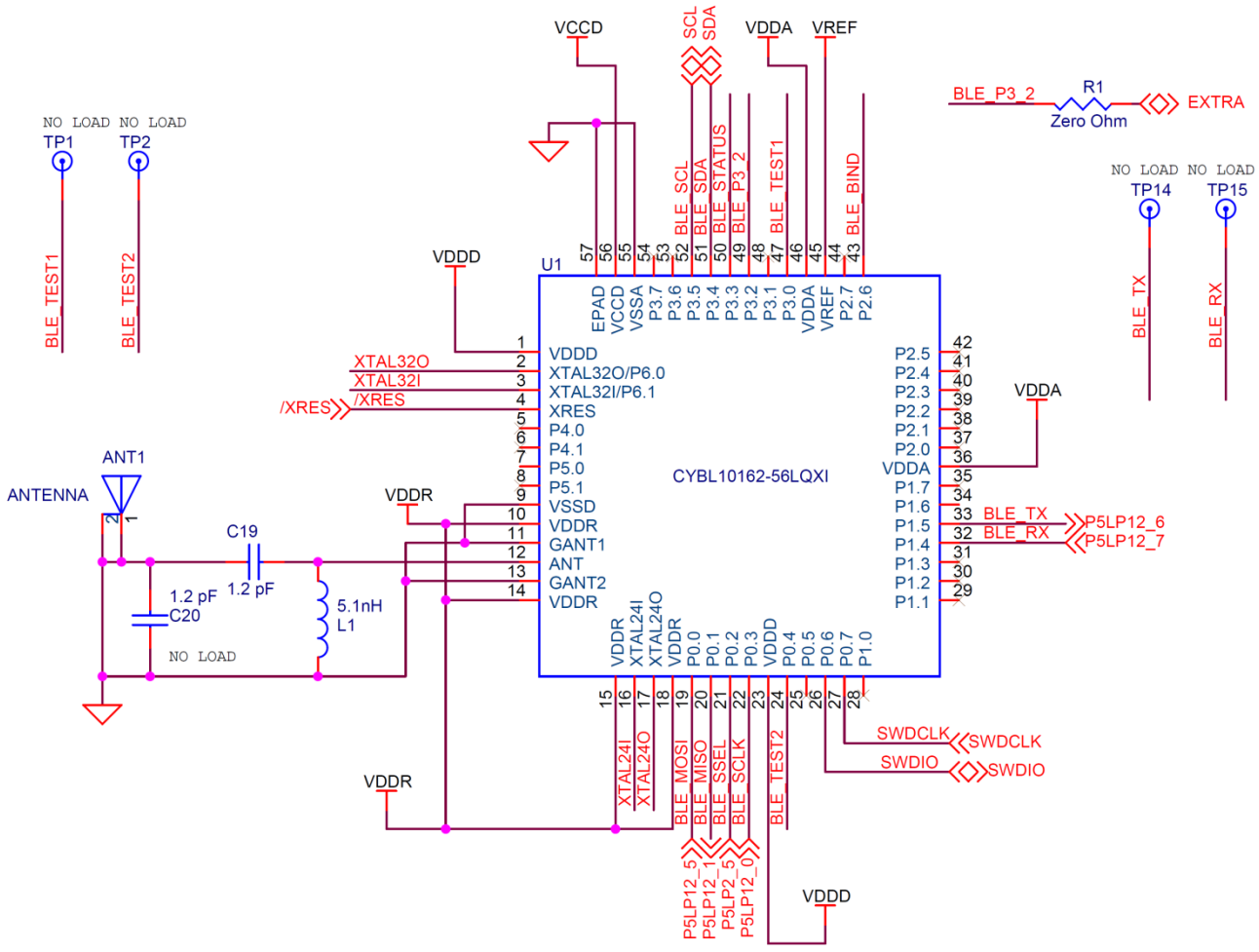


4.4.4 PRoC BLE Device

The CYBL10162-56LQXI PRoC BLE is a 32-bit, 48-MHz ARM Cortex-M0 solution with CapSense, 12-bit SAR ADC, 4 TCPWMs, 36 GPIOs, 2 SCBs, an LCD direct drive, I^2S, and an integrated Bluetooth Smart radio with a balun. PRoC BLE includes a royalty-free BLE stack compatible with Bluetooth 4.1 and provides a complete, programmable, and flexible solution for HID. In addition, PRoC BLE provides a simple, low-cost way to add BLE connectivity to any existing system.

The PRoC BLE device on the CySmart USB dongle receives the data transmitted from the mouse/remote control over the Bluetooth Smart link and sends it to the PC over USB through the PSoC 5LP device. The PRoC BLE device is connected to a wiggle antenna through an LC filter circuit for an optimum performance. Figure 4-21 shows the hardware connections for the PRoC BLE device.

Figure 4-21. PRoC BLE Connections



4.4.5 PSoC 5LP Device

The PSoC 5LP device acts as a USB-to-UART bridge between PRoC BLE and the host device such as a PC. In addition to the UART connection, there are SPI and I²C connections between the PSoC 5LP and the PRoC BLE devices. Figure 4-22 shows the hardware connections for the PSoC 5LP device The I²C bus contains firmware-controlled pull-ups using FET, which can be enabled or disabled using the PSoC 5LP pins, as shown in Figure 4-23. Note that in the firmware provided as part of the kit, the UART connection between PSoC 5LP and PRoC BLE is used.

Figure 4-22. PSoC 5LP Connections

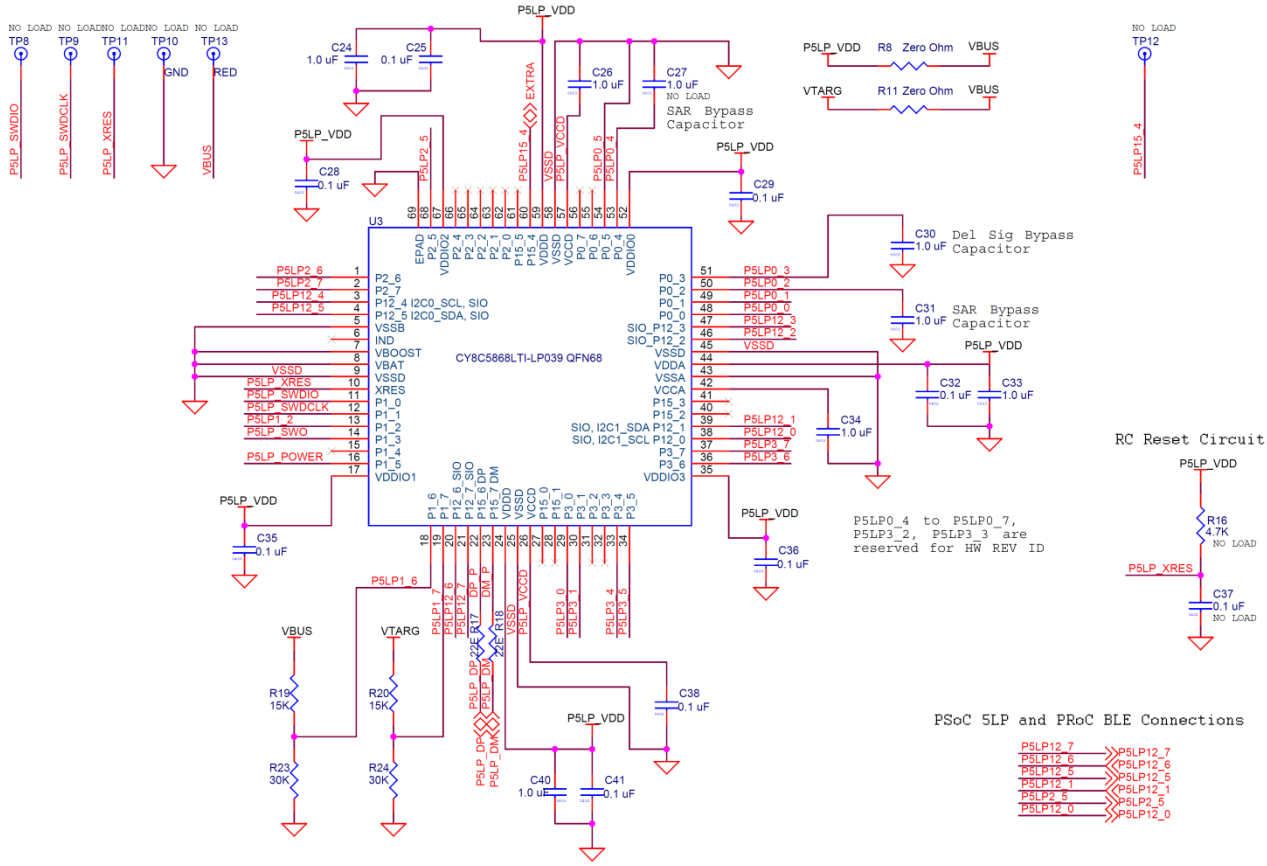
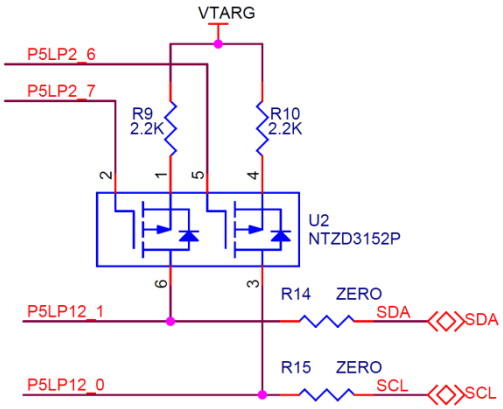


Figure 4-23. I²C Connection

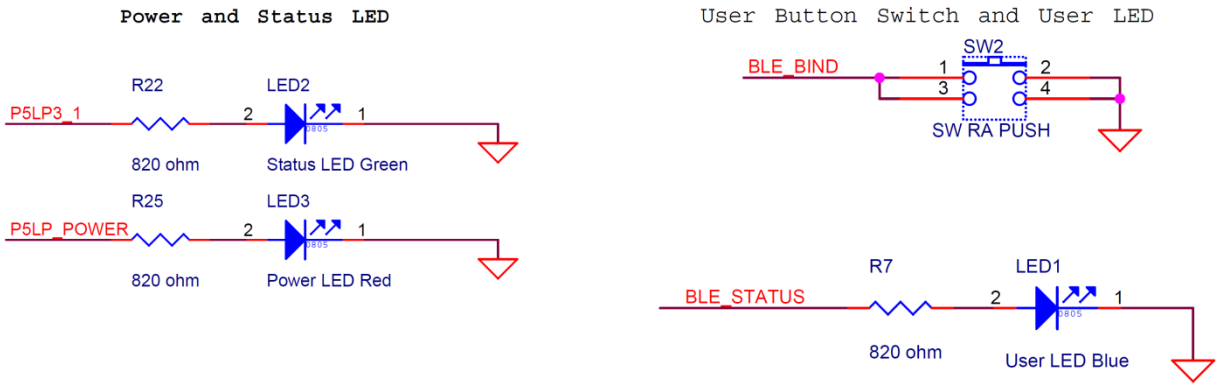


4.4.6 LEDs and Buttons

There are three LEDs for status indications and a button for providing inputs to the PRoC BLE device, as shown in Figure 4-24. In the example firmware provided with the kit, the button (SW2 or user button) is used as a bind initiation input button. The red LED indicates power status. This LED is always on when the dongle is plugged into the USB port. The green LED switches on and stays on once USB enumeration is complete on the dongle side. The blue LED indicates Bluetooth Smart states. The behavior of the green and blue LEDs is controlled by the firmware on the PSoC 5LP and PRoC BLE devices respectively.

Note: Completion of enumeration on the dongle side is not the same as on the host (such as a PC) side. The green LED stays on when enumeration is complete on the dongle side. The host may take additional time to load drivers and make the device available for use with other applications.

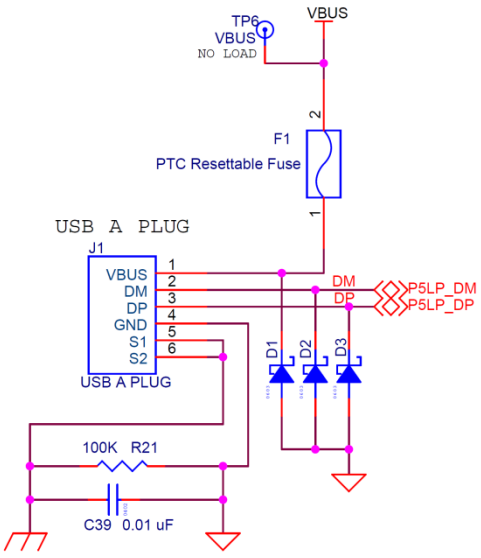
Figure 4-24. LEDs and Buttons



4.4.7 USB Connection

The CySmart dongle uses the USB port to communicate with the PC and for its power supply, as shown in Figure 4-25. The USB differential lines are connected to the PSoC 5LP device, which acts as a bridge between the PC and the PRoC BLE device.

Figure 4-25. USB Circuitry



5. Firmware

This chapter explains the firmware provided with the CY5682 PRoC BLE Touch Mouse RDK. Topics covered include the following:

- Firmware for PRoC BLE on Touch Mouse
- Firmware for PRoC BLE and PSoC 5LP on CySmart USB Dongle

Note: "Subsystem" in this chapter refers to self-contained firmware that enables the complete functionality for a hardware feature such as the UART subsystem. Multiple subsystems are integrated to form the complete firmware for an application such as a touch mouse.

5.1 Firmware for PRoC BLE on Touch Mouse

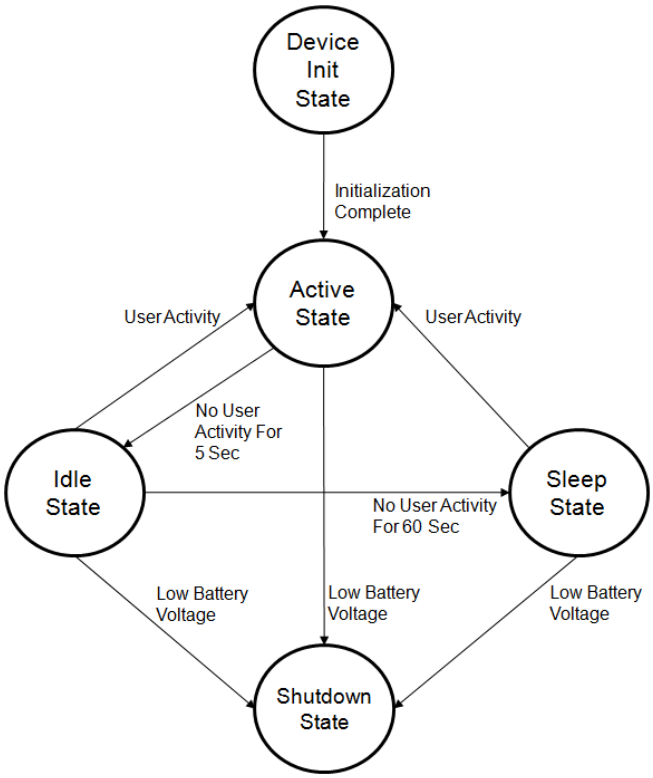
The PRoC BLE device is at the heart of the touch mouse. The firmware (see Figure 5-1) on the PRoC BLE controls the following:

- System power states
- BLE-based wireless communication
- Trackpad scanning and gesture detection
- Optical sensor
- Button scanning
- LED glowing
- Battery monitoring

5.1.1 Firmware Architecture

The touch mouse application firmware implements a state machine that has five states, as shown in Figure 5-1.

Figure 5-1. State Machine for Touch Mouse



- **Device Init state:** The mouse firmware enters this state after the device is powered on. All subsystems present (optical sensor, ISR for buttons, and trackpad) in the mouse firmware are initialized in this state. The firmware transitions to Active state after initialization process is complete.
- **Active state:** The mouse firmware enters this state:
  - After device initialization from the Device Init state
  - On user activity (buttons, optical sensor, and trackpad) from the Idle state
  - On user activity (buttons and optical sensor) from the Sleep state

  The mouse application demands a report rate of 125 Hz in the Active state; to satisfy this requirement, every input subsystem (such as the optical sensor, trackpad, and buttons) in the mouse is polled at a 7.5-ms interval. The firmware detects user activity and collects the data from the corresponding input subsystems. The device, in the connected state, sends this data over the Bluetooth Smart link per the HOGP specification.

  The firmware transitions to the Idle state if there is no user activity for five seconds.

  When the device is not connected to a peer device (such as a CySmart USB dongle), it starts advertising on user activity to re-establish a connection for 1.28 seconds. After that time, the firmware transitions to the Idle state and then to the Sleep state if there is a Bluetooth Smart link disconnection or if there is no peer device (such as a CySmart USB dongle).

  The battery voltage is measured every three seconds. When the firmware detects a low-voltage condition (at approximately 1.1 V), it initiates a red LED blinking indication on user activity to notify the user about the low battery condition. The firmware transitions to the Shutdown state when it detects a battery voltage below the critical voltage condition (at approximately 0.9 V).
- **Idle state**: The mouse firmware enters this state from the Active state if there is no user activity for five seconds. In this state, the firmware polls for user activity from the input subsystems at a poll interval of 125 ms to minimize power consumption. The firmware also maintains the Bluetooth Smart link with a 7.5-ms connection interval and a slave latency of 80. It returns to the Active state if any user activity (buttons, optical sensor, or trackpad) is detected. In this mode, the PRoC BLE device is put into the Deep Sleep state at every possible opportunity.

The battery voltage is measured every three seconds. The firmware transitions to the Sleep state if there is no user activity for 60 seconds or to the Shutdown state if the battery voltage is below the critical voltage of 0.9 V.

- **Sleep state**: The mouse firmware enters this state from the Idle state if there is no user activity for 60 seconds. In this state, the firmware stops polling for user activity (trackpad, buttons, optical sensor) and activates GPIO interrupts. In addition, the optical sensor transitions to a low-power state. The firmware maintains the Bluetooth Smart link with a 7.5-ms connection interval and a slave latency of 80.

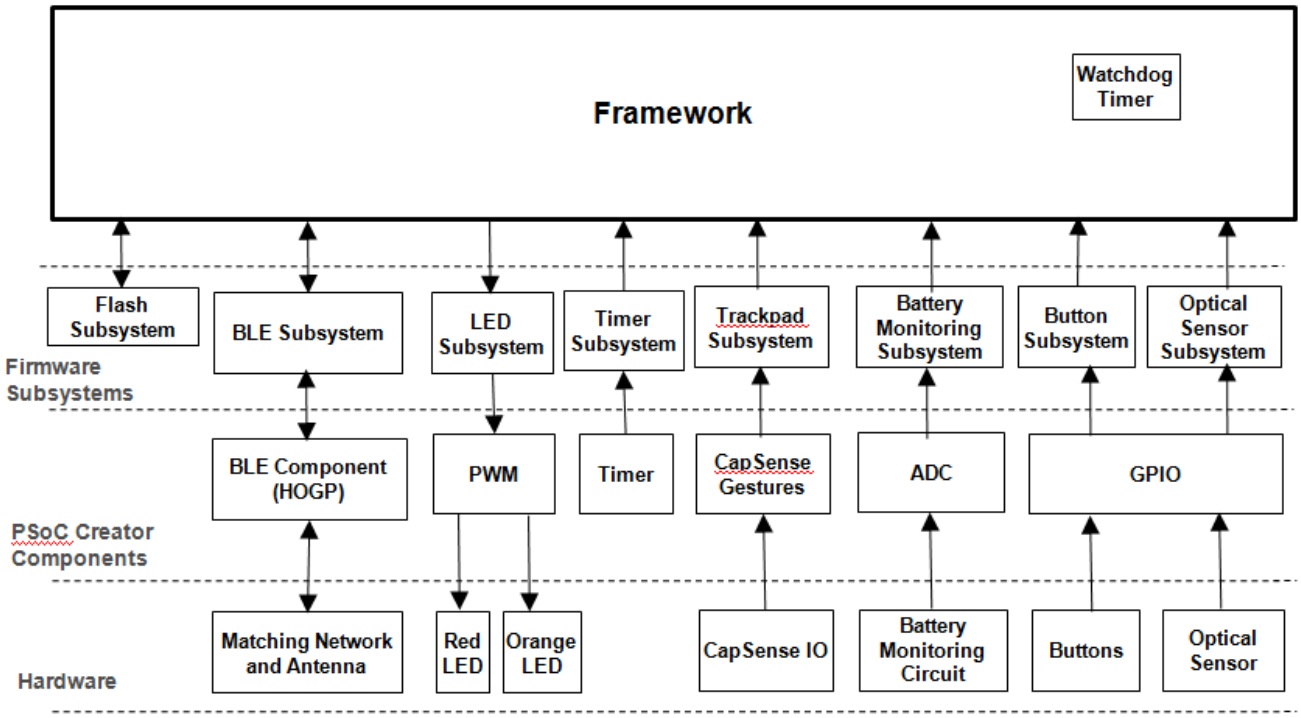
  While in this state, an optical sensor or button activity will generate a GPIO interrupt, which wakes up the MCU and transitions the firmware state to Active. The battery voltage is measured every three seconds. The firmware transitions to the Shutdown state if the battery voltage is below the critical voltage of 0.9 V.

  Touch gestures are not functional in this state. The system power consumption is considerably lower in this state compared to that in the Active and Idle states.

- **Shutdown state:** The firmware enters the Shutdown state when the battery voltage is below the critical voltage of 0.9 V. In this state, the firmware shuts down all input and output subsystems, including the BLE subsystem, and enters the MCU stop mode (PRoC BLE stop mode). This effectively renders the touch mouse nonfunctional. The user is expected to replace the batteries in this state to make the touch mouse operational again.

The firmware implements these states in an application framework with the help of the components noted in Figure 5-2.

Figure 5-2. Firmware Architecture



The firmware architecture of the touch mouse comprises an application framework along with the input and output subsystems. Input subsystems detect and record user activity. Output subsystems transmit data wirelessly or provide indications of status to the user. Input and output subsystems use the Components provided with PSoC Creator as well as application logic to interface with the hardware blocks noted in Chapter 4. Hardware. The application framework initializes the subsystems, implements the system power states, and invokes the input/output subsystems as required.

The touch mouse firmware implements a state machine, shown in Figure 5-1, with the help of the following input and output subsystems:

- Timer subsystem
- Watchdog timer

- BLE subsystem
- Trackpad subsystem
- Optical sensor subsystem
- Button subsystem
- Battery monitoring subsystem
- LED subsystem
- Flash subsystem
- Debug subsystem

These subsystems use PSoC Components. Figure 5-3 gives a schematic view of PRoC BLE.

Figure 5-3. PSoC Creator Schematic View of PRoC BLE Touch Mouse

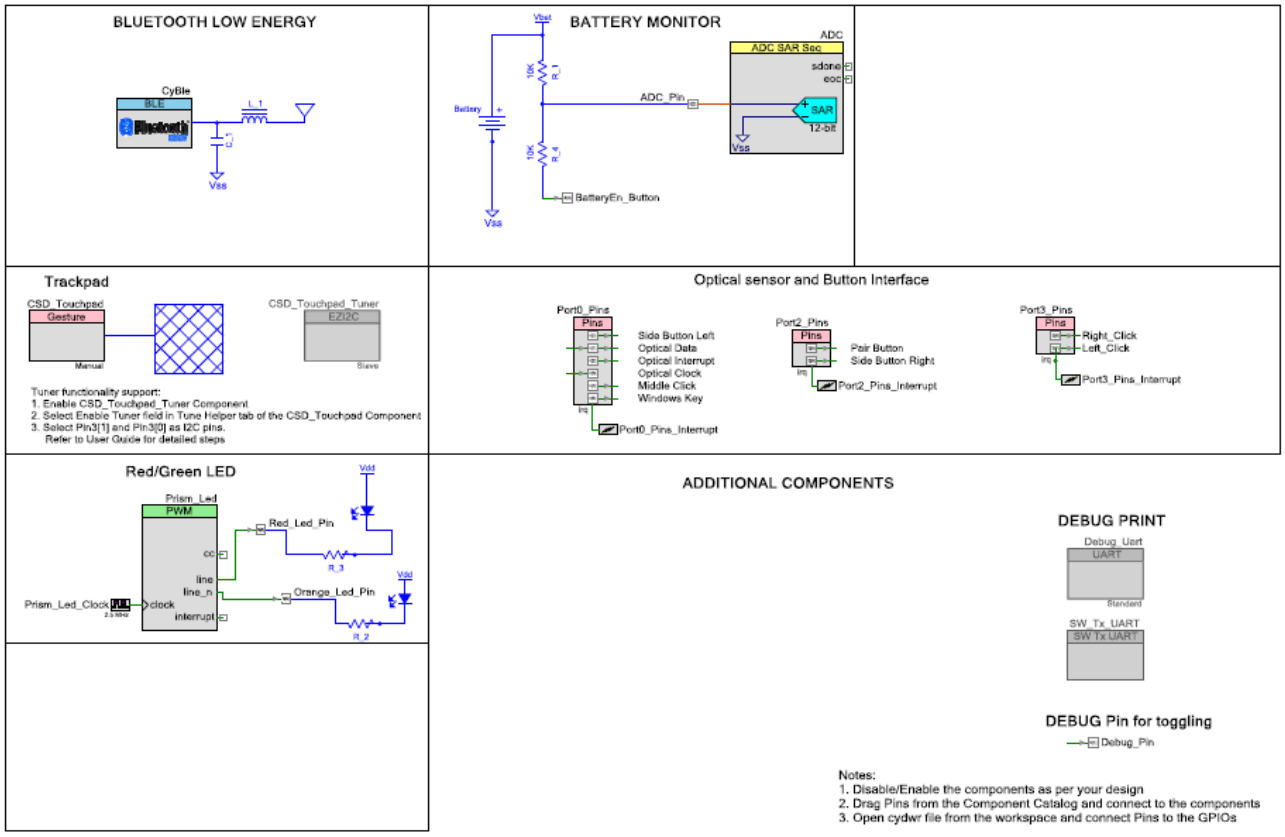


Table 5-1 lists the file structure used by the mouse project and lists key functions implemented in each file:

Table 5-1. File Structure in Touch Mouse Project

File Name	Details
main.c	This file is the top-level application, which initializes the system and handles the switching between power states based on user activity. This file has following functions: • ***main()*** – The main function for the application. • ***Device_FW_Init()*** – Initializes all system blocks • ***Device_FW_Active()*** – Watchdog timer counter1 is configured to trigger interrupt every 1.25 ms. All system blocks are polled at every 7.5 ms and notifications are sent to connected host when data is available. • ***Device_FW_Idle()*** – Watchdog timer counter1 is configured to trigger interrupt every 125 ms. All system blocks are polled every 125 ms. • ***Device_FW_Sleep()*** – Watchdog timer counter1 is disabled and system blocks are put into a low-power mode. • ***Device_FW_Stop()*** – All blocks are disabled and system is put into the hibernate mode
device.c	This file handles the initialization of individual system blocks, time keeping, polling for events, low-power mode implementation, and sending notifications to the connected host when data is available. • ***Device_Init()*** – Initializes all system blocks. • ***Devic_Timer_Callback()*** – This function is a callback from watchdog timer counter1. The compare value for the Prism_Led (Pseudo random PWM) is incremented. • ***Device_Active ()*** – Polls the data (based on activity) from all system blocks every 7.5 ms and sends a notification to the connected host. • ***Device_Low_Power_State()*** – Polls the data from all system blocks every 125 ms in idle mode. In Sleep mode, all system blocks are disabled and GPIO interrupt mechanism is enabled. • ***Device_ShutDown ()*** – Disables all system blocks.
timer.c	This file handles initialization and generation of timer tick every 1.25 ms • ***Timer_CallBack()*** – Increments the tick timer by a value equal to tickincrement. Tickincrement depends on the state of system. • **Timer_Init()** – Initializes the timer block • ***Timer_Get_Time_Stamp()*** – Returns the current timestamp • **Timer_Time_Elapsed()** – Returns true if the time exceeded an interval specified as one of parameters. • ***Timer_Set_Period()*** – Sets the time period between timer interrupts
watchdog_timer.c	This file handles watch dog timer reset functionality. Watch dog timer system reset is configured to occur after 6 seconds. • ***Watch_Dog_Timer_Init()*** – Initializes the watchdog timer counter0 and configures it to trigger a reset interrupt • ***Watch_Dog_Timer_Clear()*** – Clears the watchdog timer counter0 • ***Watch_Dog_Timer_Disable()*** – Disables the watchdog timer counter0 • ***Watch_Dog_Timer_Enable()*** – Enables the watchdog timer counter0

File Name	Details
ble.c	This file handles BLE initialization, configuration, advertisement, notifications and response to BLE events • ***Ble_Update_Serial_Number_String()*** –Updates the serial number string in the device information service with the silicon ID • ***Ble_AppCallBack()*** – Callback to get GAP, GATT, and L2CAP events from the BLE Component • ***Ble_BasCallBack()*** – Callback registered with the Battery service • ***Ble_ScpsCallBack()*** – Callback registered with the Scan Parameter service • ***Ble_HidsCallBack()*** – Callback registered with the HID service • ***Ble_IasCallBack()*** – Callback registered with the Immediate Alert service • ***Ble_LinkLossCallBack()*** – Callback registered with the Link Loss service • ***Ble_TxPowerCallBack()*** – Callback registered with the Tx Power service • ***Ble_WriteBondedList()*** – Writes the bond information into flash for future retrieval • ***Ble_Init()*** – Initializes the BLE Component • ***Ble_Configure()*** – Configures the BLE Component after the BLE stack on event • ***Ble_Set_Address()*** – Sets the public BLE address for the device • ***Ble_Is_Init_Completed()*** – Returns the status of BLE initialization • ***Ble_Send_Battery_Data()*** – Sends the battery data to a connected client • ***Ble_Send_Data()*** – Sends the mouse or keyboard report to a connected client • ***Ble_StartAdvertisement()*** – Starts a directed or undirected advertisement • ***Ble_StopAdvertisement()*** – Stops a directed or undirected advertisement • ***Ble_Get_State()*** – Returns the current state of BLE • ***Ble_Set_State()*** – Sets the state of BLE based on the system state • ***Ble_Stop()*** – Stops the BLE block • ***Ble_Enter_LowPowerMode()*** – Puts BLE into Deep Sleep mode
trackpad.c	This file handles trackpad initialization, configuration, polling, centroid calculation, and gesture detection • ***Trackpad_Init()*** – Initializes the trackpad module • ***Trackpad_Start_Poll()*** – Starts the trackpad sensor scanning • ***Trackpad_IsComplete()*** – Returns the status of trackpad sensor scanning • ***Trackpad_Poll()*** – Detects the number of fingers and decodes the gestures on trackpad • ***Trackpad_Get_Report()*** – Generates the report for gestures on trackpad • ***Trackpad_Set_State()*** – Sets the state of the trackpad module based on the system state • ***Trackpad_IsActive()*** – Returns the user activity on the trackpad; returns true if the finger is present on the trackpad • ***Trackpad_Stop()*** – Stops trackpad scanning
trackpad_gesture_mapping.c	This file maps the detected with appropriate HID commands which are recognized by PC • ***Trackpad_Gesture_Map()*** –Maps trackpad gestures to the appropriate HID report format

File Name	Details
optical.c	This file handles optical sensor initialization, configurations, read/write, and reset functionality.. • ***Optical_Reset()*** – Resets the optical sensor block • ***Optical_Write()*** – Updates the registers of the optical sensor • ***Optical_Read()*** – Reads the data from optical sensor registers • ***Optical_Set_Data()*** – Writes the data into specific registers of the optical sensor • ***Optical_Get_Data()*** – Reads the data from specific registers of the optical sensor • ***Optical_Init()*** – Initializes the optical sensor • ***Optical_Get_Report()*** – Returns the delta x and delta y movement of the optical sensor • ***Optical_Set_DPI()*** – Configures the DPI of the optical sensor • ***Optical_Set_State()*** – Configures the optical sensor GPIO setting based on the system state • ***Optical_Poll()*** – Polls for optical sensor activity
button.c	This file handles GPIO initialization for buttons and decodes the state transition of buttons. • ***Button_Debounce()*** – Handles button debouncing and provides the correct button status. • ***Button_Interrupt_CallBack()*** – Button interrupts are disabled and an event is generated to switch the system state. Button interrupts are enabled in Idle and Sleep modes. • ***Button_Init()*** – Registers the call-back function for button interrupts. • ***Button_Poll()*** – Polls for the button status. • ***Button_Set_State()*** – Modifies the configuration of buttons based on the system state (enabling and disabling interrupts) • ***Button_IsPairButtonPressed()*** – Returns the status of the 'pair' button • ***Button_Get_Report()*** – Returns the button report only when there is a transition in button state • ***Button_Is_Active()*** – Returns true if any button is in the pressed state • ***Button_Stop()*** – Stops the button module. All button interrupts are disabled.
battery.c	This file handles the initialization of the ADC and the measurement of voltage using the ADC. • ***Battery_Init()*** – Starts the ADC Component • ***Battery_Get_Value()*** – Measures and returns the battery voltage • ***Battery_Stop()*** – Stops the ADC Component

File Name	Details
led.c	This file handles initialization of LED and the creation of various effects in glowing LEDs. • ***Led_Update()*** – Callback function. The compare value for the Prism_Led (Pseudo random PWM) is incremented in this function. • ***Led_Init()*** – Initializes the LED block • ***Led_Blinking()*** – Creates a specific LED effect based on the configuration of the LED breathing effect • ***Led_On()*** – Keeps the LED ON with a specific intensity • ***Led_LowBattery()*** – Triggers a low-battery LED indication • ***Led_LinkLoss( )*** – Triggers the link loss LED indication • ***Led_IsComplete()*** Returns the status of the LED breathing effect (complete/not complete) • ***Led_Stop()*** – Stops the LED module and resets all variables
flash.c	This file handles storage/retrieval of information to/from flash. • ***Flash_Save ()*** – Stores the information into flash • ***Flash_Load ()*** – Restores the information from flash and copies it into the RAM buffer • ***Flash_Calc_Checksum()*** – Calculates checksum for the specified buffer
debug.c	This file handles the print debug messages (logs) over UART. • ***Debug_Putc()*** **–** Prints a character on the UART • ***Debug_Byte_To_ASCII()*** **–** Maps a byte to its corresponding ASCII value • ***Debug_Print_Start()*** **–** Initializes the debug functionality • ***Debug_Print()*** **–** Prints a message (log) on the UART

File Name	Details
platform.h	This file contains macros that control various options in the firmware related to debug, touchpad tuning, usage of MCU power states, manufacturing test kit (MTK), and BLE power states.. • ***ENABLE_DEBUG_PIN*** **–** Define this macro to enable toggling of debug pin macros. • ***DEBUG_PRINT*** **–** Define this macro to enable debug print feature over UART. SCB UART should be enabled along with this macro. PRoC BLE Pin 3[1] should be configured as UART Tx. • ***SOFTWARE_UART*** **–** Define this macro to enable the debug print feature over UART. Software Transmit UART Component should be enabled along with this macro. PRoC BLE Pin 3[1] should be configured as UART Tx. • ***ENABLE_BLE_LOW_POWER_MODE*** **–** Define this macro to enable BLE low-power mode (Deep Sleep mode) • ***ENABLE_BONDING*** **–** Define this macro to enable the storage of bonding information into flash. The touch mouse retrieves this information while doing directed advertisement to connect to a previously connected host. • ***TOUCHPAD_TUNER*** **–** Define this macro to enable the touchpad tuning functionality • ***MCU_SLEEP_DISABLED*** **–** Define this macro to disable the MCU from entering sleep state. • ***MCU_DEEP_SLEEP_DISABLED*** **–** Define this macro to disable the MCU entering Deep Sleep state. • ***ENABLE_MTK_MODE*** **–** Define this macro to enable the manufacturing mode.

5.1.2 Timer

The touch mouse derives two functionalities from the watchdog timer:

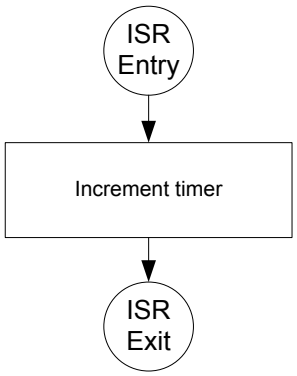
1. Tick timer functionality (1.25-ms tick timer)
2. Reset functionality (six seconds watchdog reset). The reset functionality is explained in the next section.

The PRoC BLE watchdog timer has two 16-bit counters (counter 0 and counter 1) and one 32-bit counter (counter 2).

5.1.2.1 Tick Timer Functionality

The touch mouse firmware utilizes watchdog counter 1 to generate an interrupt on terminal count: in Active state, a timer interrupt is generated every 1.25 ms; in Idle state, a timer interrupt is generated every 125 ms. Figure 5-4 shows the flowchart for the timer subsystem.

Figure 5-4. Timer Subsystem Flowchart



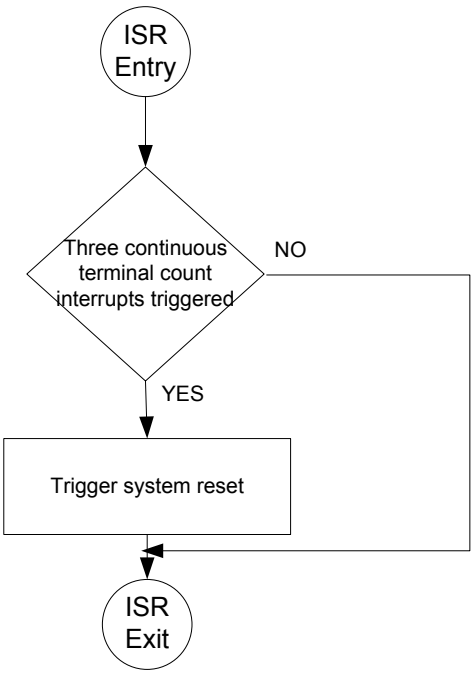
5.1.2.2 Watchdog TimerReset Functionality

The touch mouse firmware uses watchdog counter 0 to ensure that the system recovers from any undesired firmware behavior. Counter 0 is configured to generate a system reset interrupt after three continuous unhandled interrupts. A counter 0 interrupt is generated on reaching the specified terminal count. The terminal count value of touch mouse firmware is set to 0xFFFF, which converts to approximately a two-second interval. Based on this terminal count setting, a system interrupt is generated approximately after six seconds, which is after three continuous interrupts are unhandled.

A system reset interrupt will cause the MCU to reset, and firmware execution will start from the beginning. To prevent a system reset, counter 0 must be cleared before the 6-second period.

Watch_Dog_Timer_Clear() is used to clear the watchdog timer counter 0. This function is called at least once in every loop to clear the watchdog timer counter 0. The loop time in Active mode is 7.5 ms, in Idle mode is 125 ms, and in low-power state us 600 ms. Figure 5-5 shows the flowchart for the watchdog timer subsystem.

Figure 5-5. Watchdog Timer Subsystem Flow Chart



5.1.3 Bluetooth LE Subsystem

The BLE subsystem transmits data provided by the application framework over a Bluetooth Smart link. It implements the HOGP profile for the touch mouse. Figure 5-6 shows the BLE subsystem flow chart. The HOGP provides the following services:

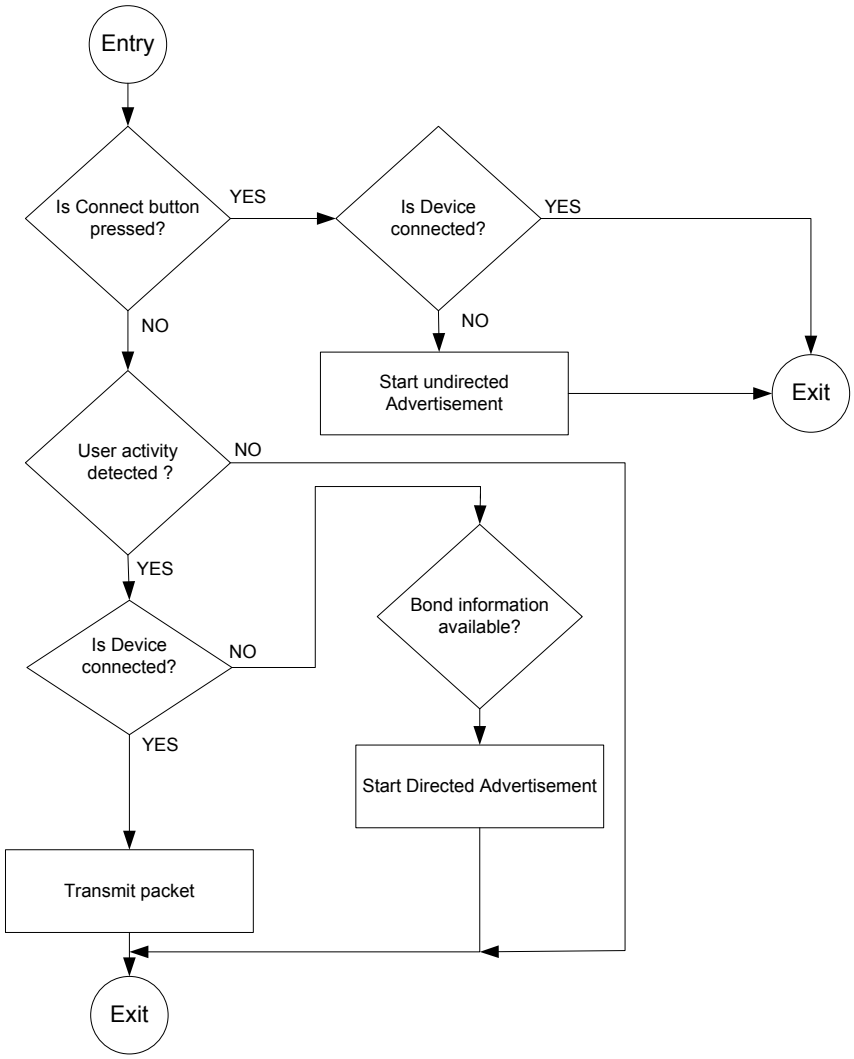
- Single HID service for transmitting the mouse and keyboard reports
- Battery service for transmitting the battery status to the peer device
- Device information service that exposes manufacturer and/or vendor information about a device
- Scan parameter service that enables a GATT client to store the BLE scan parameters being used with a GATT server device. This allows the GATT server to use the information and adjust behavior to optimize power consumption and/or reconnection latency.

In the Active state, the BLE Component connection interval is set to 7.5 ms, and the slave latency is set to 80. The subsystem enters the Idle state after five seconds of inactivity. The subsystem transitions from the Idle state and then to the Sleep state on a disconnect event or if no peer device is present.

In the Idle state, the BLE Component connection interval is set to 7.5 ms, and the slave latency is set to 80. The subsystem enters the Sleep state after 60 seconds of inactivity.

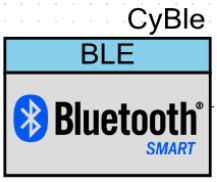
In the Sleep state, the BLE Component connection interval is set to 7.5 ms, and the slave latency is set to 80.

Figure 5-6. BLE Subsystem Flowchart



The BLE subsystem is built using the BLE Component (Figure 5-7) provided in PSoC Creator.

Figure 5-7. BLE Component of PSoC Creator

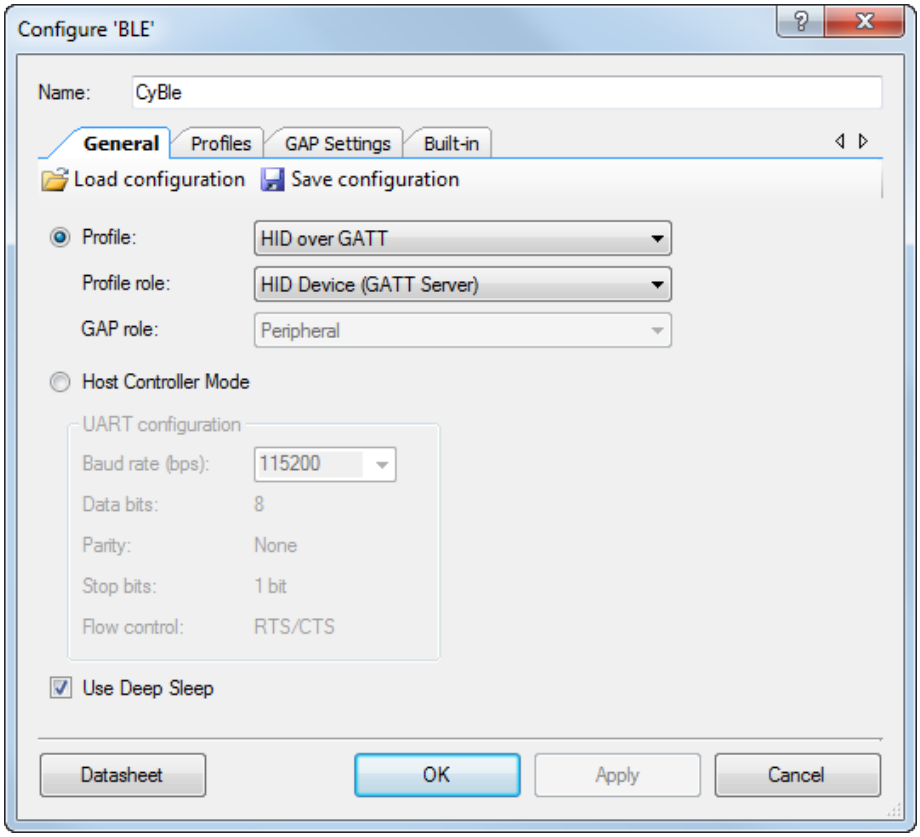


This Component is configured in PSoC Creator using a GUI consisting of three main tabs as well as several sub-tabs. The settings on these tabs are discussed below. Additional details can be found in the Component datasheet.

5.1.3.1 General Settings

The BLE Component can be configured for multiple profiles and roles. The touch mouse operates on the HID over GATT profile. When the profile mode is configured to HID over GATT, the profile role will be automatically selected as HID Device (GATT Server) as shown in Figure 5-8.

Figure 5-8. General Tab of BLE Component



The settings used for a touch mouse application for the BLE subsystem **General** tab are as follows:

- **Profile:** The touch mouse configures this parameter as “HID over GATT,” since it is an HID.
- **Profile role:** The touch mouse configures this parameter as “HID Device (GATT Server),” as it is acting as a GATT server, which sends notifications to the GATT client whenever data is available.
- **GAP role:** Based on the **Profile** and **Profile role**, the **GAP role** is automatically selected as a peripheral for the touch mouse.
- **Use Deep Sleep:** This parameter allows the firmware to use the Deep Sleep state to optimize the power consumption of the touch mouse.

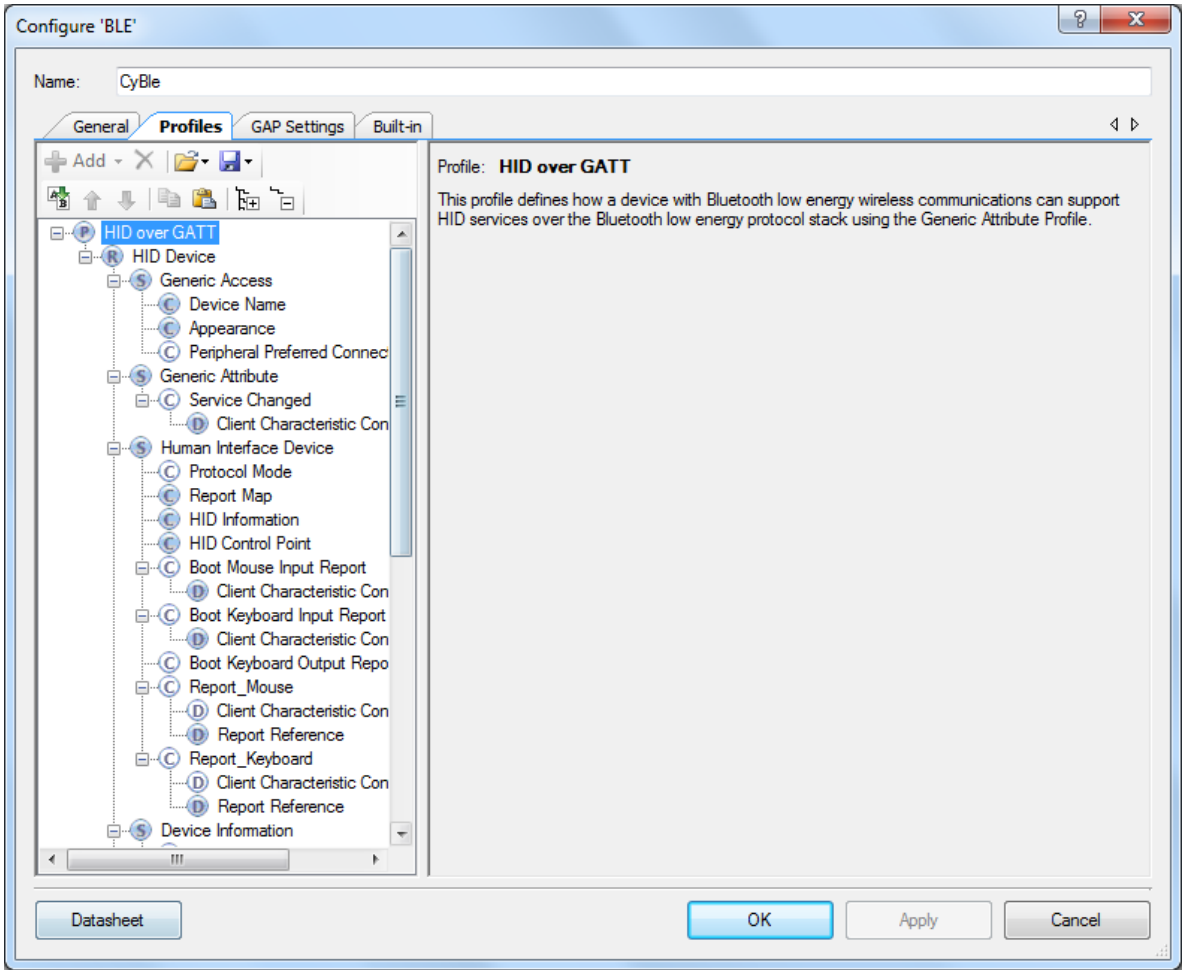
5.1.3.2 Profiles Settings

The **Profiles** tab of the BLE Component is shown in Figure 5-9. It consists of set of services, characteristics, and descriptors.

Generic Access Service	Device Name	Configure the value of this characteristic as “CY5682 Mouse RDK”
	Appearance	Configure the value of this characteristic as “HID: Mouse”
	Peripheral Preferred Connection Parameters	Characteristic values are automatically populated by the Component based on Advertisement settings configured in the GAP Settings tab of the BLE Component.
Generic Attribute Service	The default BLE configuration of this service is used for touch mouse	
Human Interface Device Service	Protocol Mode	The default BLE configuration of this characteristic is used for the touch mouse.
	Report Map	The Report Map characteristic defines the HID descriptor for the touch mouse. The report map for the touch mouse consists of mouse report structure and keyboard report structure.
	HID Information	The default BLE configuration of this characteristic is used for the touch mouse

	HID Control Point	The default BLE configuration of this characteristic is used for the touch mouse	
	Boot Mouse Input Report	The default BLE configuration of this characteristic is used for the touch mouse	
	Boot Keyboard Input Report	The default BLE configuration of this characteristic is used for the touch mouse	
	Boot Keyboard Output Report	The default BLE configuration of this characteristic is used for the touch mouse	
	Report Mouse	Modify read permissions to “Encryption Required” and configure the Report ID of the Report Reference descriptor to “1”. The rest of the parameters are left as default values set by the BLE component.	
	Report Keyboard	Modify read permissions to “Encryption Required” and configure the Report ID of the Report Reference descriptor to “2”. The rest of the parameters are left as default values set by the BLE component.	
	Device Information	Manufacturer String Name	Configure the value as “Cypress Semiconductor”
		Firmware Revision String	Configure the value as “1.0.0.0”
		PnP ID	Configure Vendor ID as “0x4B4”, Product ID as “0x5683” and Product Version as “0x0001”
		Hardware Revision String	Configure the value as “1.0.0.3”
		Serial Number String	This value is over written by firmware with BD Address generated using silicon ID
		Software Revision String	Configure the value as “PSoC Creator 3.1”
		Model String Number	Configure the value as “CY5682”
Battery Service	Configure the value as “100”, the rest of the values are default values of the BLE component.		
Scan Parameter Service	The default BLE configuration of this service is used for the touch mouse.		
Link Loss Service	The default BLE configuration of this service is used for the touch mouse.		
Immediate Alert Service	The default BLE configuration of this service is used for the touch mouse.		
TX Power Service	Configure the Tx Power Level as “0”. The remaining default BLE configuration of this service is left at default values.		

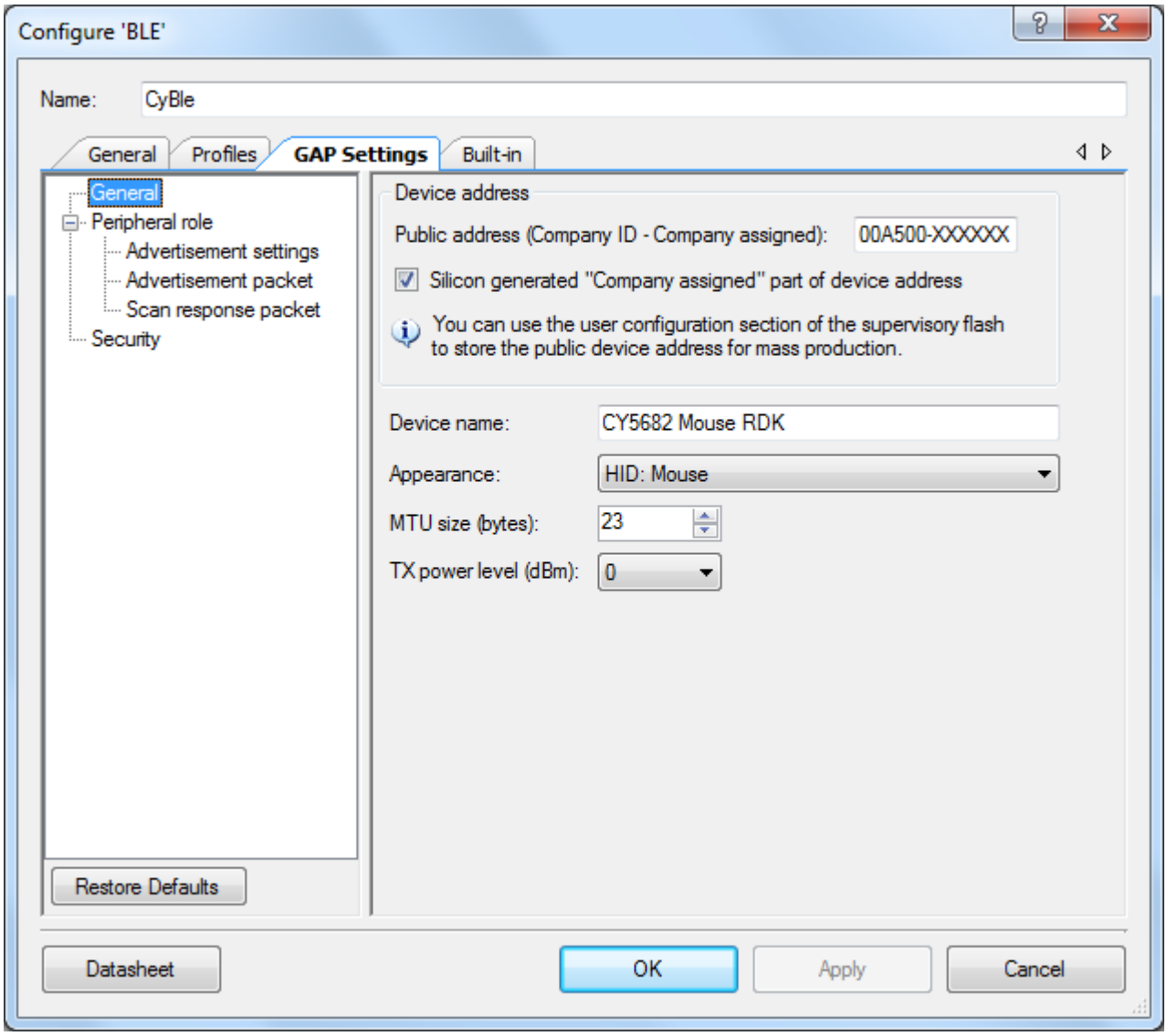
Figure 5-9. Profiles Tab of BLE Component



5.1.3.3 GAP Settings

Figure 5-11 shows the **General** settings for the touch mouse firmware.

Figure 5-10. GAP Settings Tab of BLE Component Showing General Configuration

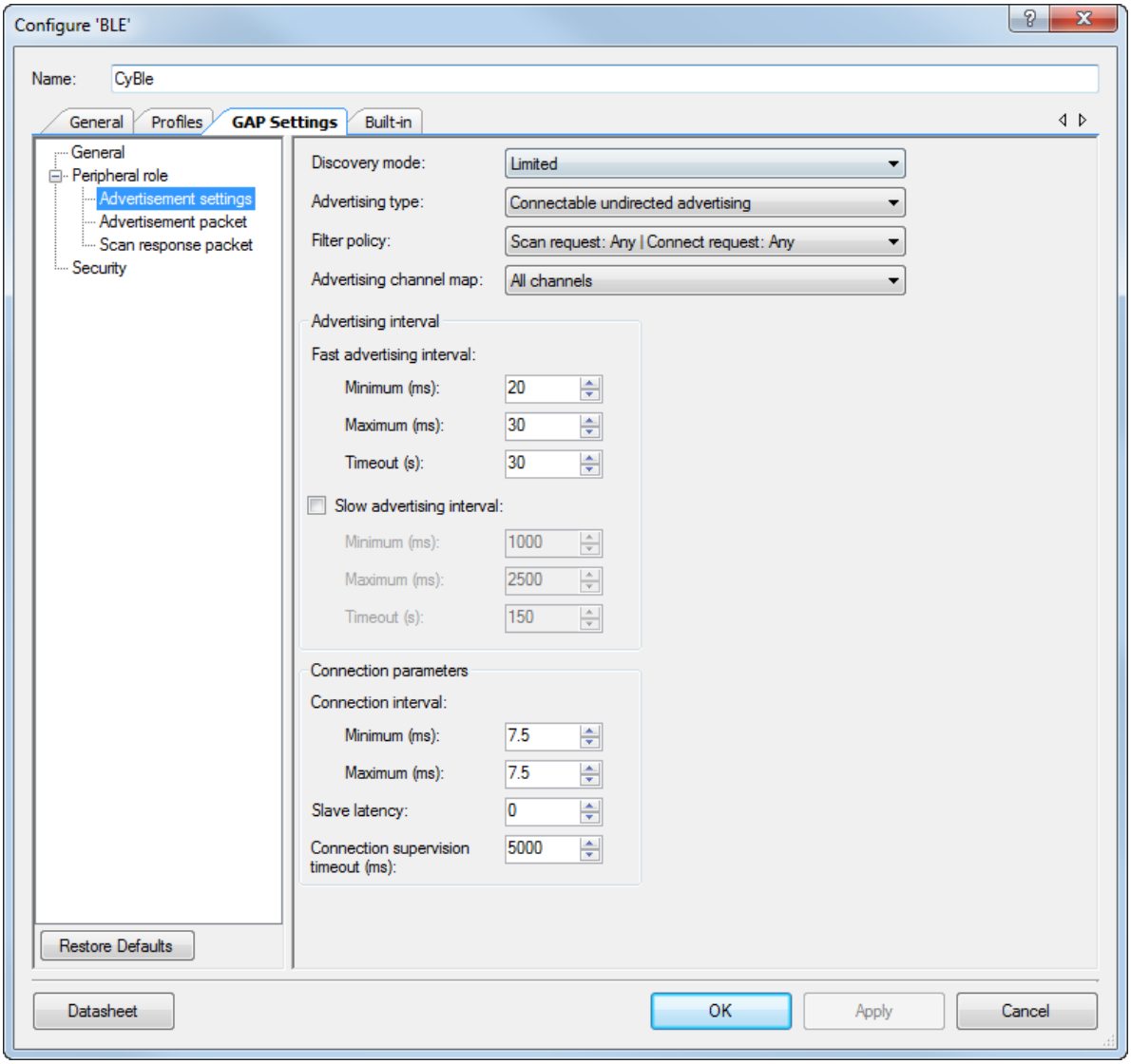


The following settings are used for the touch mouse application for the BLE subsystem (**GAP Settings** > **General)**:

- **Public address:** This parameter defines the BD Address (Bluetooth Device Address) of the touch mouse. It consists of six bytes. First three bytes define the company ID and the next three bytes are a unique value. The company address is set to “00A500,” which is Cypress ID. The next three bytes are generated using the silicon ID of PRoC BLE as “Silicon generated “**Company assigned” part of device address**” check box is selected.
- **Device name:** This parameter defines the name of the touch mouse. It is configured as “CY5682 Mouse RDK” for the touch mouse project.
- **Appearance :** Choose “HID: Mouse” for the touch mouse project.
- **MTU size (bytes) :** This parameter defines the packet size. It is configured as 23 bytes for the touch mouse project.
- **Tx power level (dBm):** This parameter defines the transmit power of the PRoC BLE radio. It is configured as “0” for the touch mouse project.

Figure 5-11 shows the **Advertisement settings** chosen for the touch mouse firmware.

Figure 5-11. GAP Settings Tab of BLE Component Showing Advertisement Settings



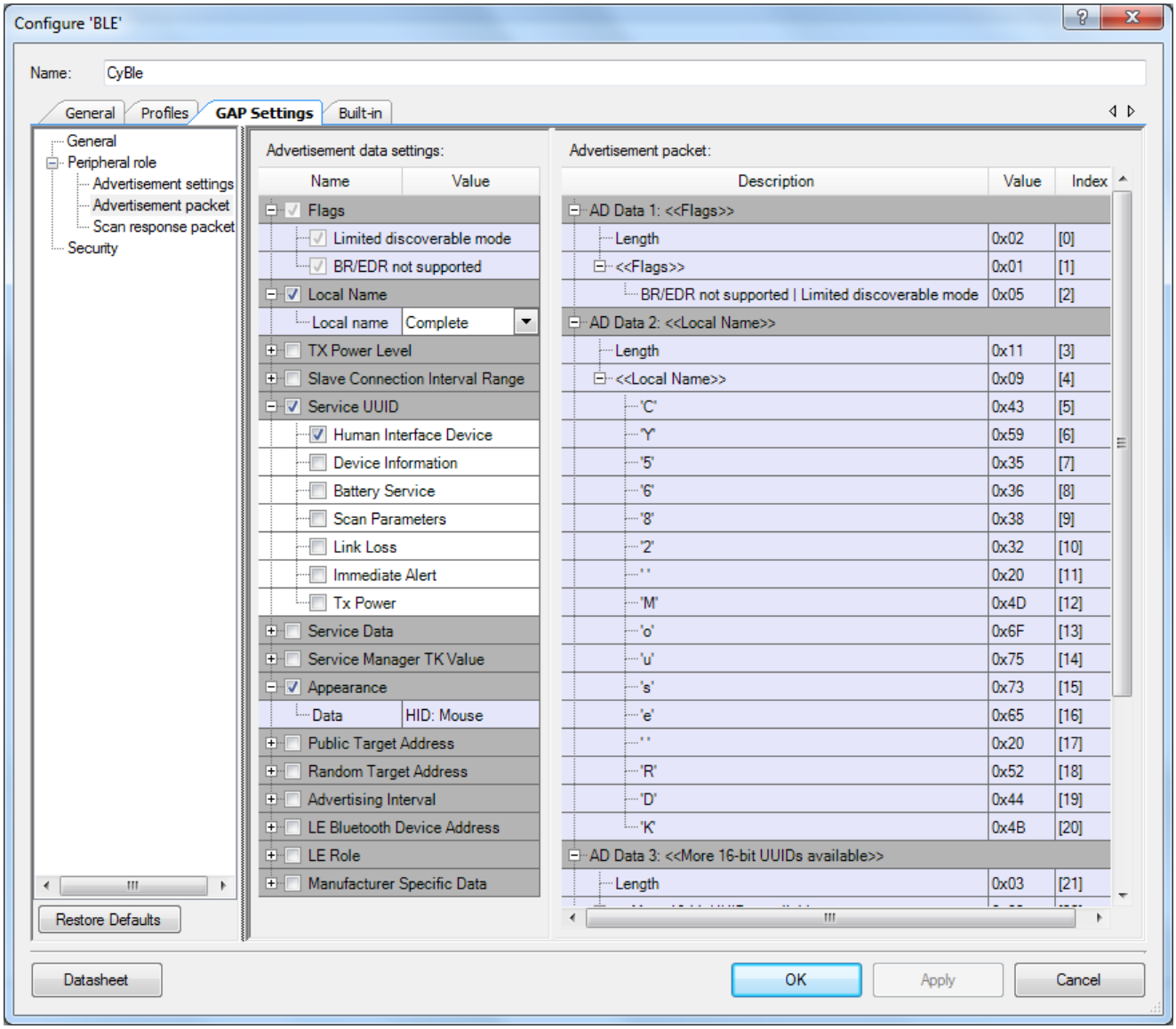
The following settings are used for the touch mouse application for the BLE subsystem (**GAP Settings** > **Advertisement settings**):

- **Discovery mode:** This mode is used by devices that need to be discoverable only for a limited period of time or for a specified event. The timeout duration is defined by the Fast advertising interval timeout parameter. The touch mouse configures this parameter as “Limited” so that it advertises for a limited period of time when the connect button is pressed.
- **Advertising type:** This parameter defines the advertising type to be used by the link layer of PRoC BLE. The touch mouse configures this parameter as “Connectable undirected advertising,” which allows any other device to connect to it.
- **Filter policy:** This parameter defines how the scan and connection requests are filtered. The touch mouse configures this parameter as “Scan request: Any | Connect request: Any”, which allows it to process connection requests and scan requests from all devices.
- **Advertising channel map:** This parameter is used to enable a specific advertisement channel. The touch mouse configures this parameter as “All channels” so that it allows advertisement on all three possible BLE advertisement channels.

- **Advertising interval:** This parameter defines the interval between two advertising events.
- **Fast advertising interval:** This advertisement interval results in a faster connection.
  - **Minimum (ms):** This parameter is the minimum interval for advertising the data and establishing a connection. It can be configured in steps of 0.625 ms. The valid range is from 20 ms to 10240 ms. The minimum value is configured as 20 ms for the touch mouse.
  - **Maximum (ms):** This parameter is the maximum interval for advertising the data and establishing a connection. It is configured to increment in multiples of 0.625 ms. The valid range is from 20 ms to 10240 ms. The maximum value is configured as 30 ms for the touch mouse.
  - **Timeout (s):** This parameter is the timeout value of advertising with fast advertising interval parameters. A timeout interval of 30 seconds is configured for the touch mouse.
- **Connection parameters:** These para**meters defin**e the connection event timing for a central device communicating with the touch mouse.
- **Connection interval:** This parameter separates consecutive connection events.
  - **Minimum (ms):** This parameter is the minimum permissible connection time value to be used during a connection event. It can be configured in steps of 1.25 ms. The range is from 7.5 ms to 4000 ms. The minimum connection interval is configured as 7.5 ms for the touch mouse to achieve a report rate above 125 Hz.
  - **Maximum (ms):** This parameter is the maximum permissible connection time value to be used during a connection event. It can be configured in steps of 1.25 ms. The range is from 7.5 ms to 4000 ms. The maximum connection interval is configured as 7.5 ms for the touch mouse to achieve a report rate above 125 Hz.
- **Slave latency:** This parameter defines the latency of the slave in responding to a connection event in consecutive connection events. It is expressed in terms of multiples of connection intervals, where only one connection event is allowed per interval. The range is from 0 to 499 events. A slave latency of 0 is used in the touch mouse firmware during connection establishment and it is updated to 80 after the connection is successfully established.
- **Connection supervision timeout (ms):** This parameter defines the link supervision timeout interval. It defines the timeout duration for which a link needs to be sustained in the case of no response from a peer device over the BLE link. The time interval is configured in multiples of 10 ms. The range is from 100 ms to 32000 ms. A connection supervision timeout of 5000 ms is configured for the touch mouse.

Figure 5-11 shows the **Advertisement packet** for the touch mouse firmware.

Figure 5-12. GAP Settings Tab of BLE Component Showing Advertisement Packet



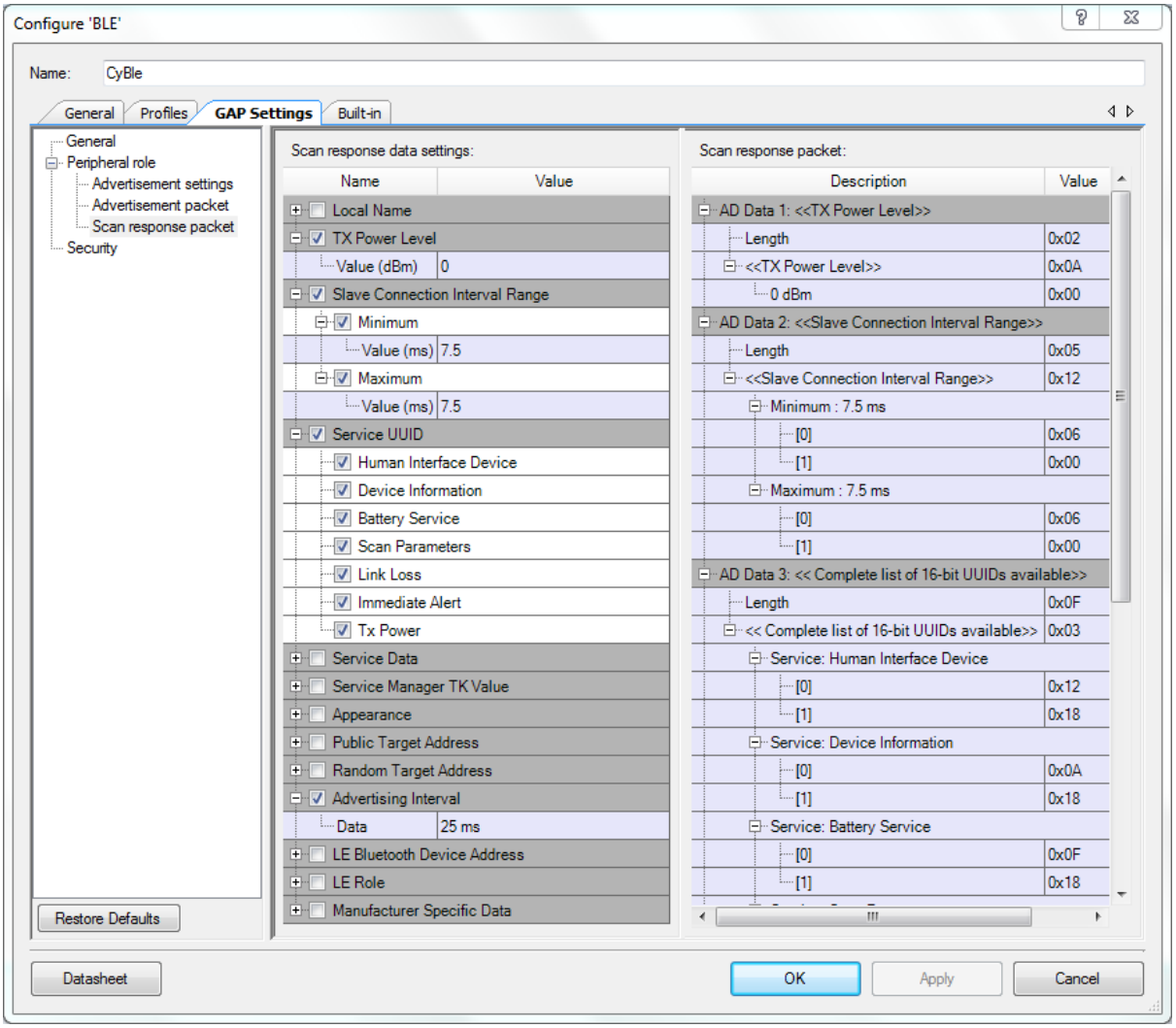
The following settings are used for the touch mouse application for the BLE subsystem (**GAP Settings** > **Advertisement packet**):

- **Local Name**: This parameter is interpreted by the host after receiving an advertisement packet from the touch mouse as the device name. It displays this name in the list of scanned devices. Select the **Local Name** check box and configure it as “complete” for the touch mouse.
- **Service UUID**: This parameter is interpreted by the host after it receives an advertisement packet from the touch mouse as the primary service. The host understands that an HID device is trying to connect, if the “Human Interface Device” check box is selected under **Service UUID** for the touch mouse.
- **Appearance**: This parameter is interpreted by the host after it receives an advertisement packet, as HID: Mouse. Select the Appearance check box for the touch mouse

  Leave the remaining configuration of the **Advertisement packet** tab at default values.

Figure 5-13 shows the **Scan response packet** for the touch mouse.

Figure 5-13. GAP Settings Tab of BLE Component Showing Scan Response Packet



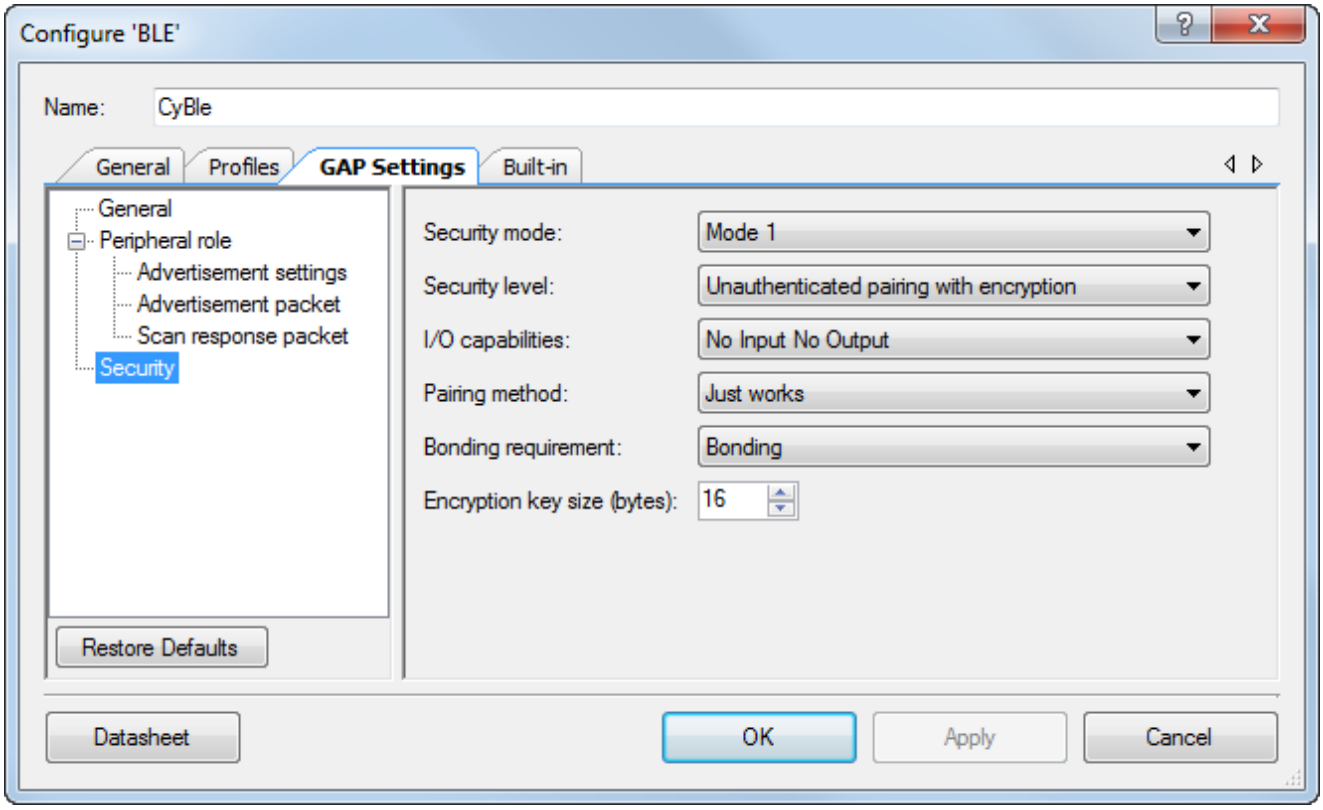
Settings used for the touch mouse application for the BLE subsystem (**GAP Settings** > **Scan Response packet)** are as follows:

- **Slave Connection Interval Range**: This parameter defines the minimum and maximum connection interval for the touch mouse. Select this check box for the touch mouse project. The host interprets these value and sets the appropriate connection interval.
- **Service UUID**: This parameter defines the services supported the touch mouse. Select all services to indicate the services supported by the touch mouse.

Leave the remaining configuration of the Advertisement packet tab at default values.

Figure 5-14 shows the **Security** settings for the mouse firmware.

Figure 5-14. GAP Settings Tab of BLE Component Showing Security



The settings used for the touch mouse application for the BLE subsystem **GAP Settings** > **Security** are as follows:

- **Security mode:** This parameter defines the GAP security modes for the Component. The touch mouse configures this mode as “Mode 1,” which is chosen when data encryption is required.
- **Security level:** The touch mouse configures this parameter as “Unauthenticated pairing with encryption.” With this level of security, the touch mouse will send encrypted data after establishing a connection with the client device.
- **I/O capabilities:** This parameter refers to the device's input and output capability that can enable or restrict a particular pairing method or security level. The touch mouse configures this parameter as “No Input No Output,” as it does not have the capability to enter and display authentication key data.
- **Pairing method:** The touch mouse configures this parameter as “Just works.” The device will use the simple pairing procedure without authentication. With this method, the transferred data will be vulnerable to “man in the middle” attacks.
- **Bonding requirement:** The touch mouse configures this parameter as “Bonding.” It will store the link key of a connection after pairing with the client device. The touch mouse uses a previously stored key for the connection when the connection is lost and then re-established.
- **Encryption key size (bytes):** This parameter defines the encryption key size based on the profile requirement. The valid values are 7 to 16 bytes. The touch mouse configures this parameter as 16 bytes.

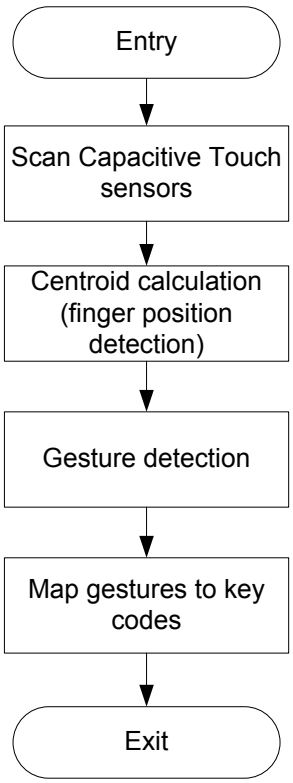
The datasheet identifies the APIs supported by the BLE Component as well as additional details on configuration. To go to the datasheet, click the **Datasheet** button at the bottom left corner of the BLE Component GUI, as shown in Figure 5-14.

5.1.4 Trackpad Subsystem

The trackpad for the touch mouse is composed of nine rows and three columns of sensors. It is capable of decoding one-finger vertical and horizontal scroll gestures. The firmware detects these gestures and reports them to the application framework.

The PSoC Creator CapSense Gesture Component supports capacitive touch sensor (trackpad) scanning, centroid calculation, and gesture detection. It provides an API for each of these functionalities. The trackpad subsystem calls the relevant APIs to detect user activity. The trackpad subsystem is executed every 7.5 ms in the Active state and every 125 ms in the Idle state. The flow chart in Figure 5-15 shows how the trackpad subsystem operates.

Figure 5-15. Trackpad Subsystem Flow Chart



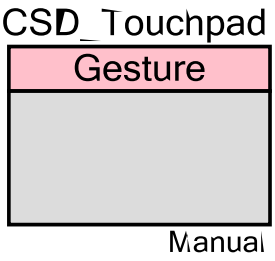
- **Scan:** Detecting the trackpad activity involves scanning the capacitive touch sensors. The presence of a finger on the trackpad will result in non-zero values for signals on those particular sensors.
- **Centroid calculation:** Signal values from the scanning stage are used as inputs for the centroid calculation algorithm. This algorithm estimates the total number of fingers and their positions on the trackpad.
- **Gesture detection:** The number of fingers and their positions calculated in the centroid calculation stage are used as inputs to the gesture detection algorithm. Table 5-2. shows the gestures detected by default by the kit firmware.

Table 5-2. Supported Gestures

Gesture	Action Performed
Vertical scroll	Single-finger movement in a up/down motion
Horizontal scroll	Single-finger movement in a left/right motion
Vertical inertial scroll	Single-finger swipe in a up/down direction (single-finger movement followed by a lift-off)

Figure 5-16 shows the CapSense Gesture Component. The following sections explain its configuration for the touch mouse.

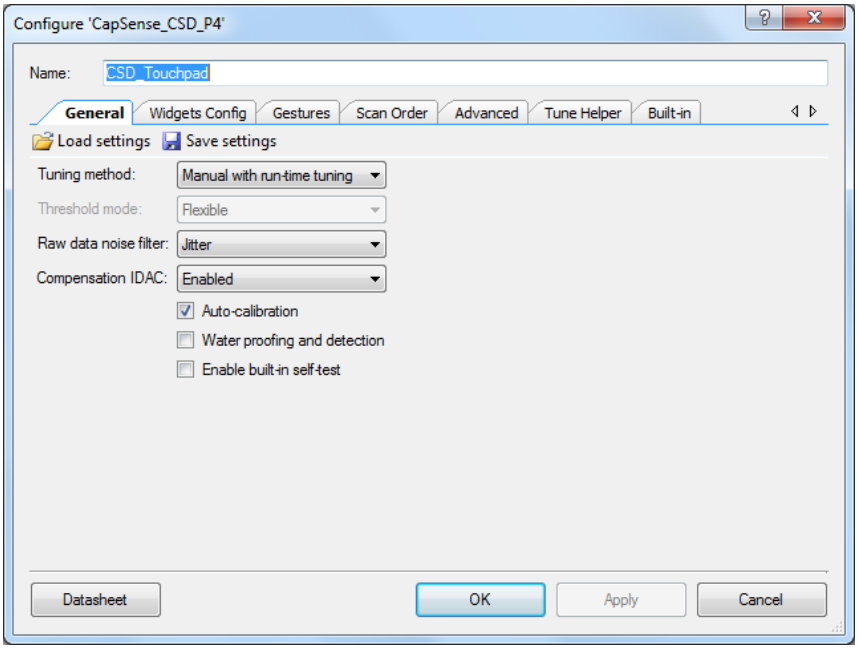
Figure 5-16. CapSense Gesture Component of PSoC Creator



5.1.4.1 General Tab

The parameters under the 'General' Tab are configured as shown in Figure 5-17.

Figure 5-17. General Tab of CapSense Gesture Component



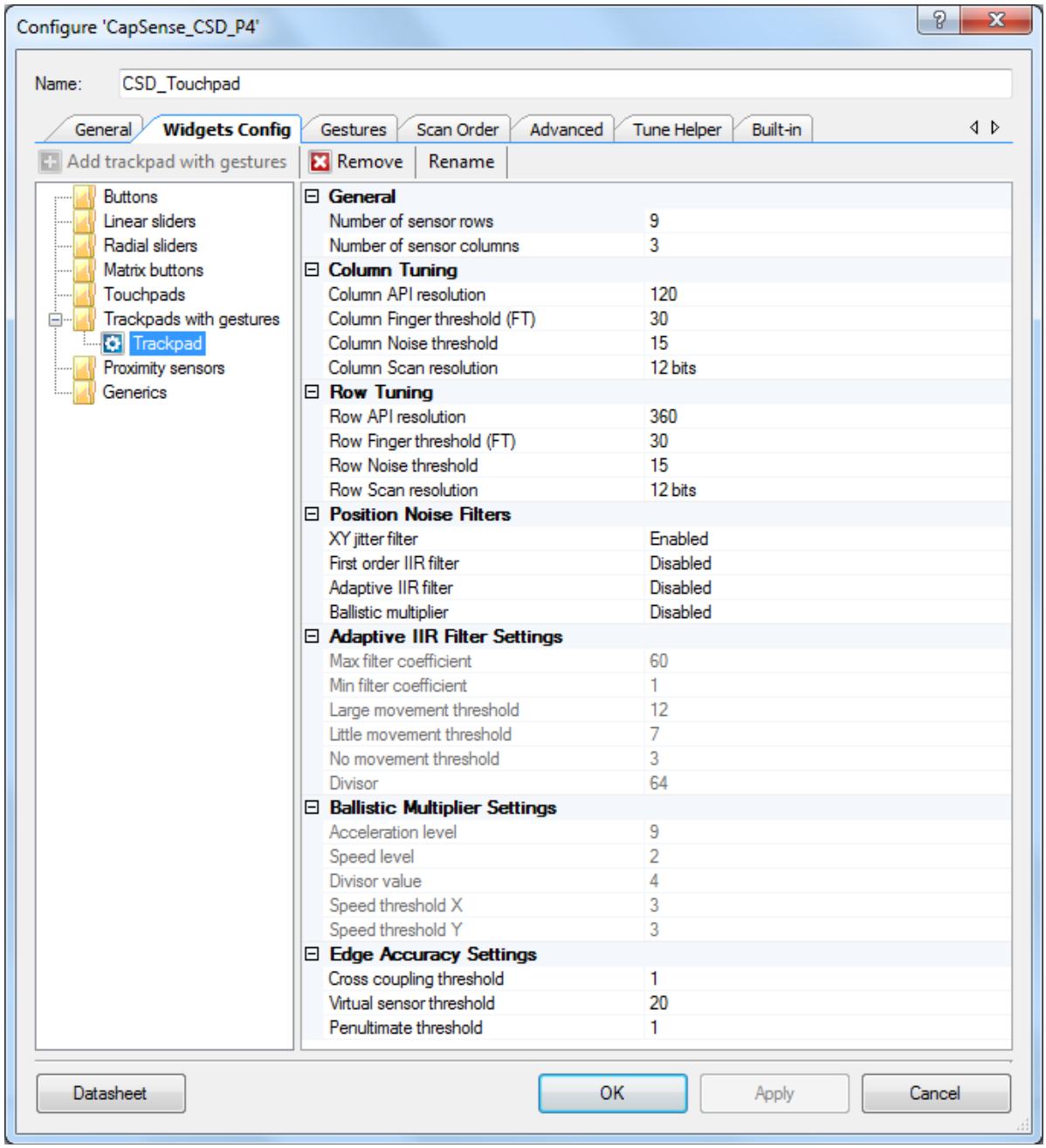
The settings used for a touch mouse application for the BLE subsystem **General** tab are as follows:

- **Tuning method:** The touch mouse configures this parameter as "Manual with run-time tuning." This option allows run-time tuning of the CSD_Touchpad Component.
- **Raw data noise filter:** This parameter selects the raw data filter. Only one filter can be selected, which is applied to all sensors. The filter is used to reduce the effect of noise during sensor scans. The touch mouse configures this parameter as "Jitter"
- **Compensation IDAC:** This mode increases sensitivity and SNR. The **Compensation IDAC** is connected to the AMUX bus full-time during CapSense operation and is intended to compensate for the sensor's parasitic capacitance. The touch mouse configures this parameter as "Enabled."
- **Auto-calibration:** The touch mouse sets this parameter, which allows IDAC auto-calibration.

5.1.4.2 Widgets Config Tab

The **Trackpad with gestures** widget is added as shown in Figure 5-18.

Figure 5-18. Widgets Config Tab of CapSense Gesture Component

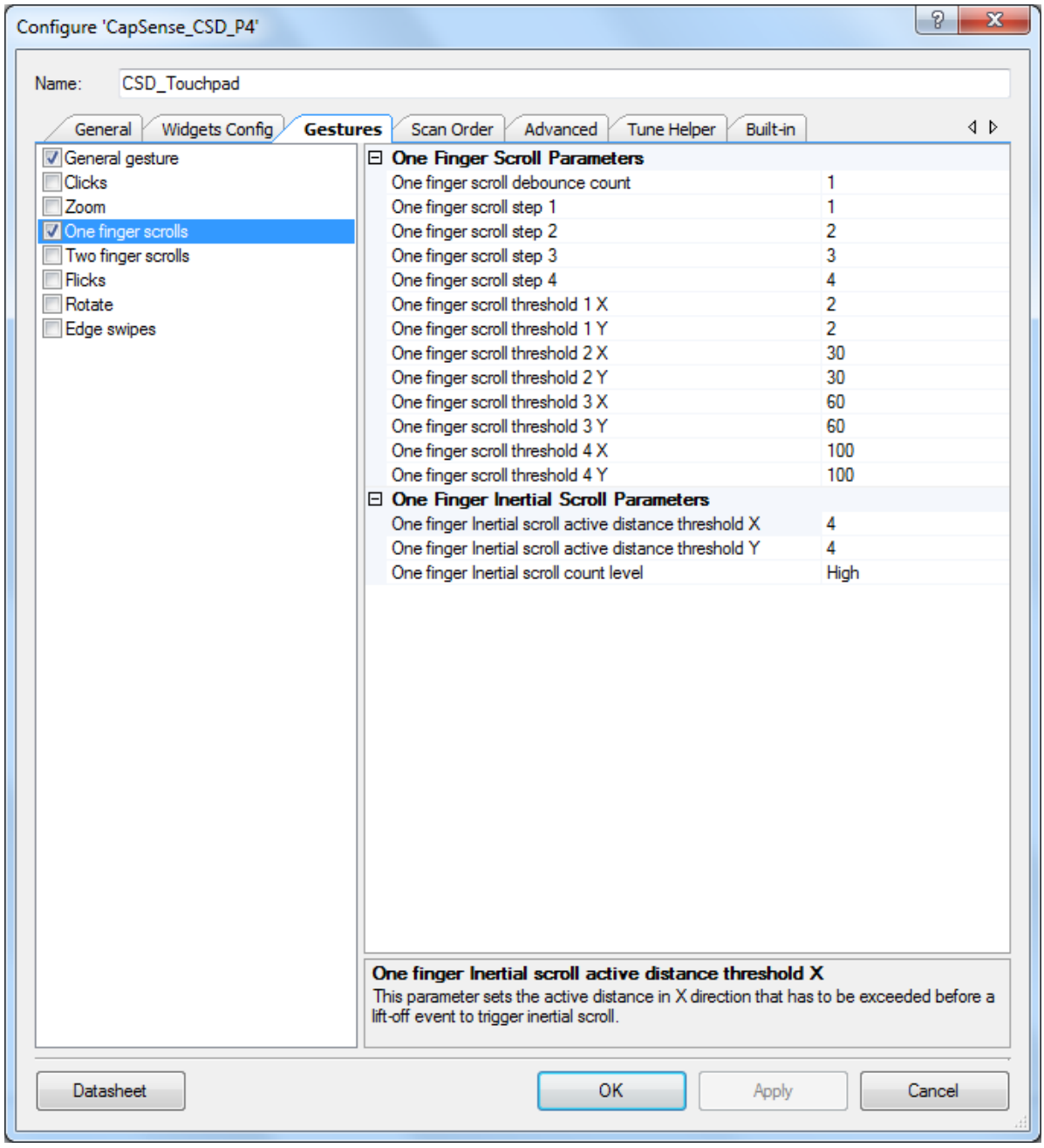


- **Number of sensor rows** and **Number of sensor columns**: These parameters depend on the number of sensors along the X- and Y-axis. Values for these parameters are set based on the trackpad design. The touch mouse configures them as nine rows and three columns.
- **Column Tuning** and **Row Tuning**: Tuning parameters for the touch mouse are selected to achieve optimum touch performance.

5.1.4.3 Gestures Tab

The **One finger scrolls** and **General gesture** are selected. The configuration for these gesture parameters shown in Figure 5-19 is the default.

Figure 5-19. Gestures Tab of CapSense Gesture Component

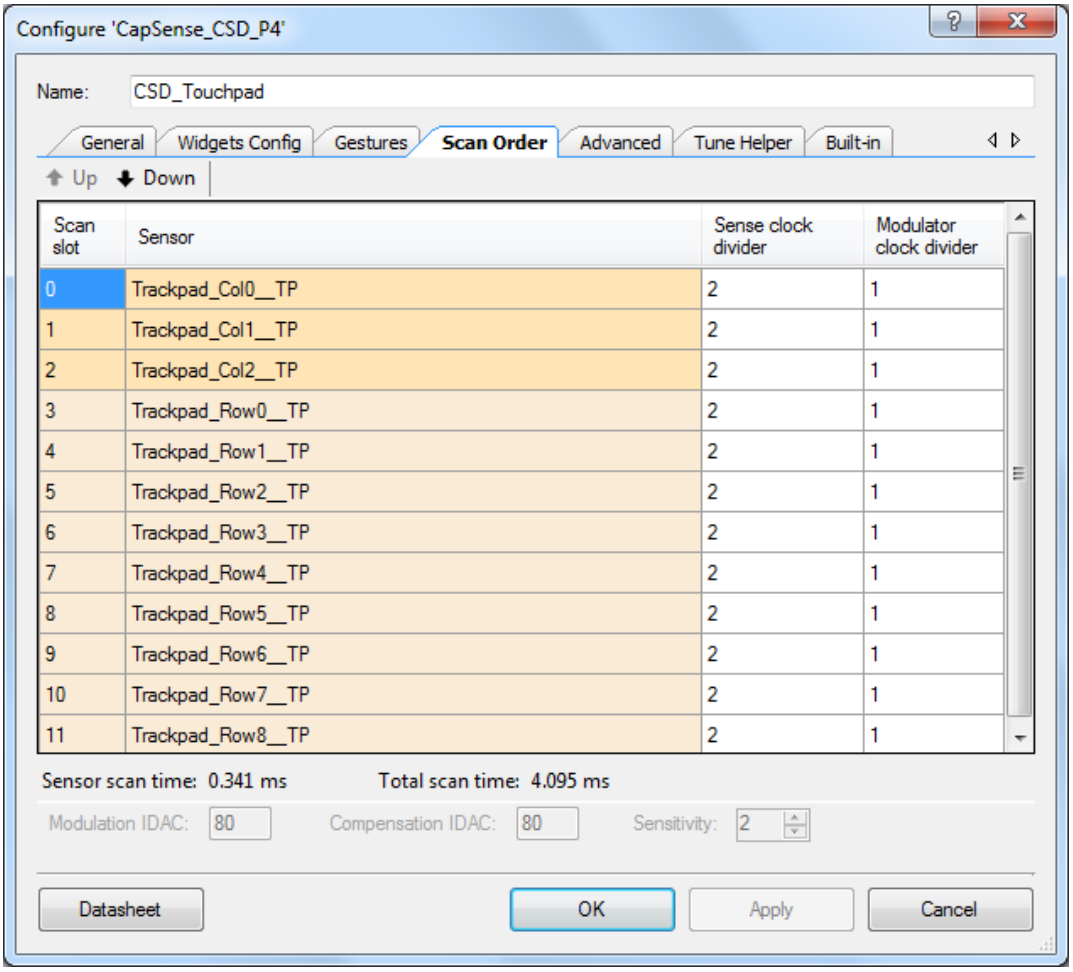


5.1.4.4 Scan Order Tab

The settings in the scan order tab are shown in Figure 5-21. The sense clock divider is modified to '2'. All other parameters are left at their default settings.

The datasheet provides a detailed table to select the sense clock divider under "Sense clock divider" section. Click the **Datasheet** button at the bottom-left corner of the CapSense Gesture Component GUI, as shown in Figure 5-21.

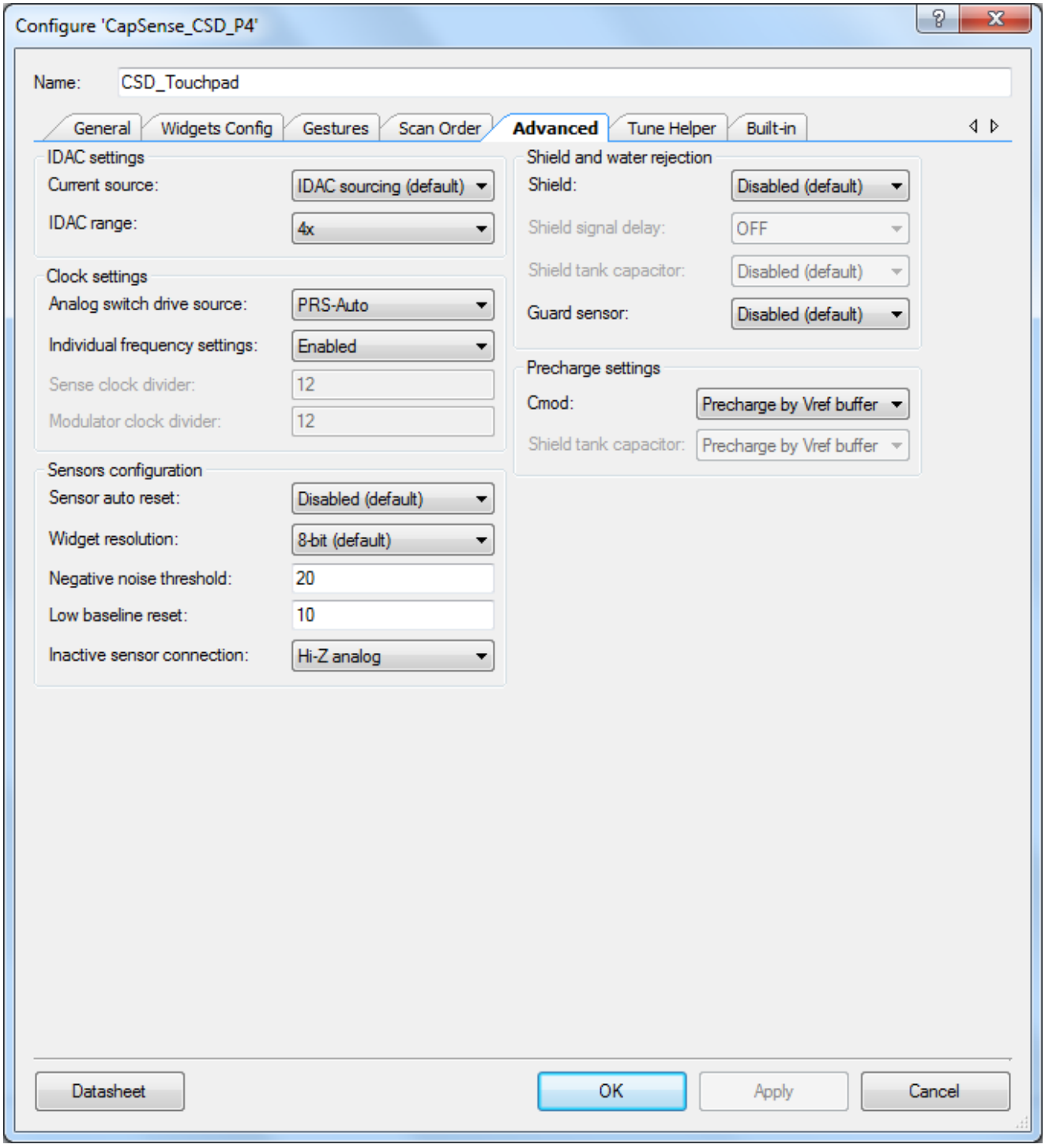
Figure 5-20. Scan Order Tab of CapSense Gesture Component



5.1.4.5 Advanced Tab

The settings in the advanced tab are shown in Figure 5-21. Analog Switch Drive Source is configured as PRS-Auto, Inactive sensor connection setting is configured to Hi-Z Analog, Negative noise threshold is configured to 20 and Low baseline reset is configured to 10. Other parameters are left at their default values generated by the Component.

Figure 5-21. Advanced Tab of CapSense Gesture Component



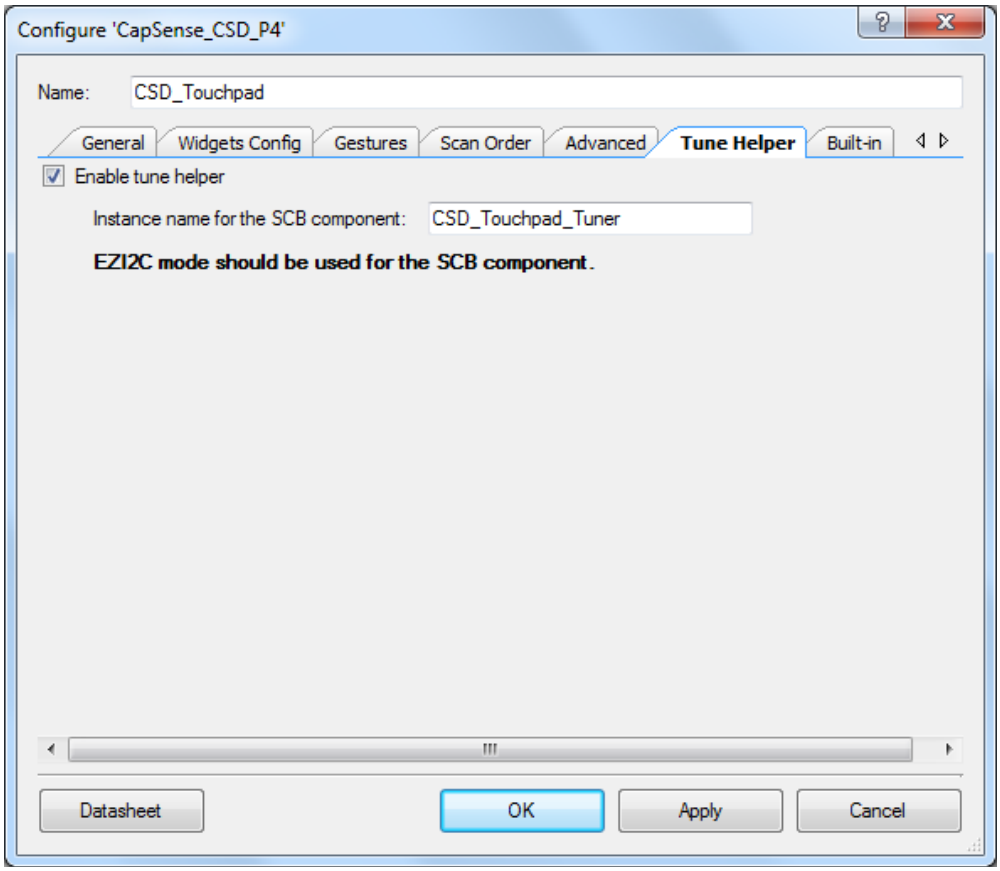
The datasheet lists the APIs supported by the CapSense Gesture Component. It also provides documentation for the configurable parameters mentioned previously in this section. To go to the datasheet, click the **Datasheet** button at the bottom left corner of the CapSense Gesture Component GUI, as shown in Figure 5-21.

5.1.4.6 Trackpad Tuning

The trackpad of the touch mouse RDK is already tuned; therefore, it is not necessary to repeat the tuning exercise. Tuning of trackpad is required during the development phase. Every new design needs tuning to ensure optimal performance. Do the following to connect to the tuner:

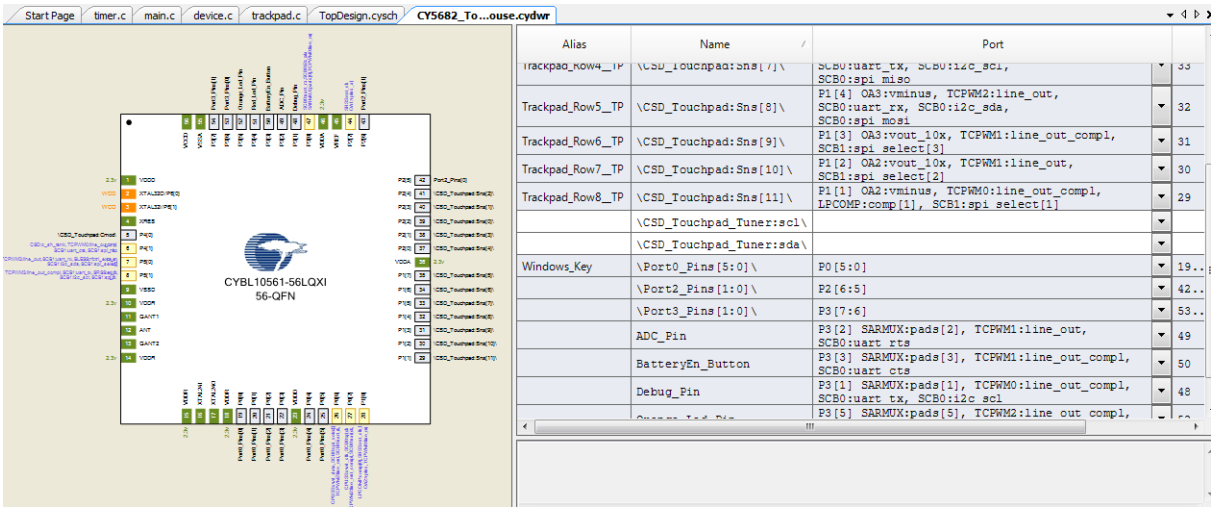
1. Enable the “TOUCHPAD_TUNER” macro in the *platform.h* file.
2. Right-click on the “CSD_Touchpad_Tuner” Component and select “Enable” to enable the Component
3. Select Tune Helper tab of CSD_Touchpad Component and select “Enable tune helper” check box as shown in Figure 5-22.

Figure 5-22. Tune Helper Tab of CapSense Gesture Component



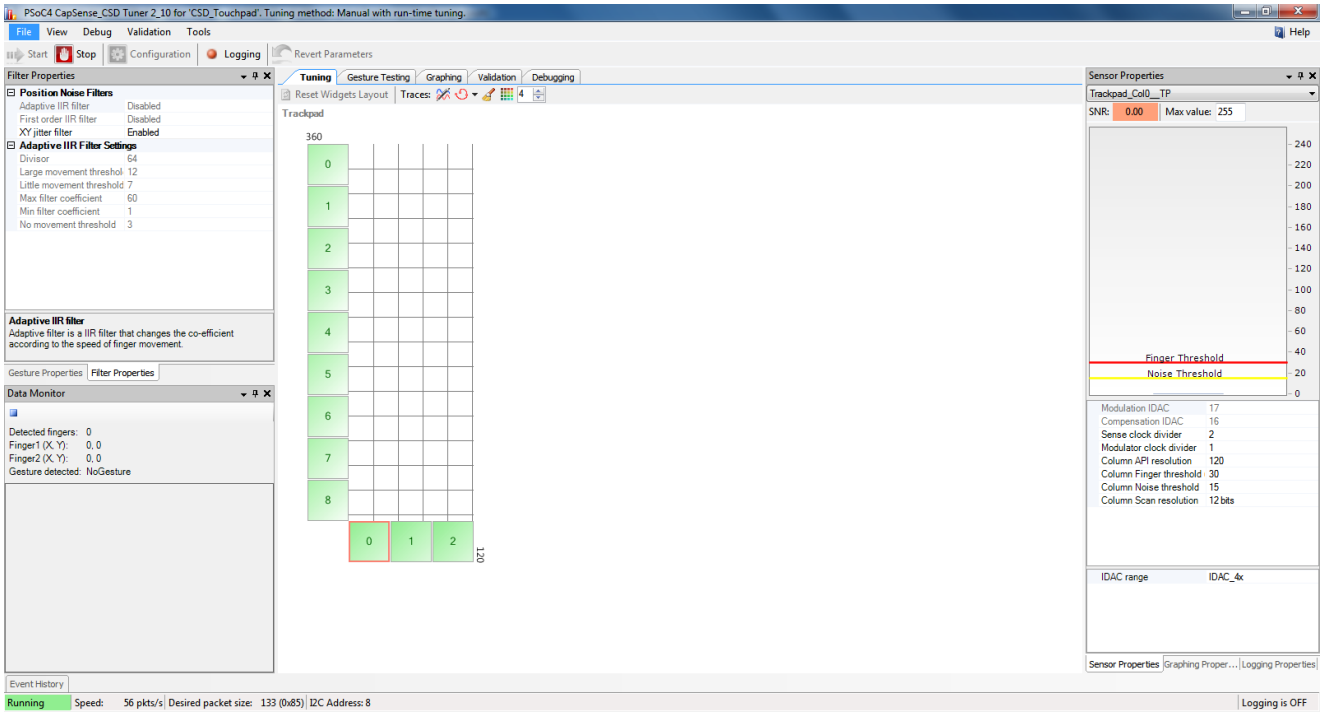
4. CSD_Touchpad_Tuner:scl and CSD_Touchpad_Tuner:sda pins appear in CY5682_Touch_Mouse.cydwr as shown in Figure 5-23.

Figure 5-23. CY5682_Touch_Mouse.cydwr View



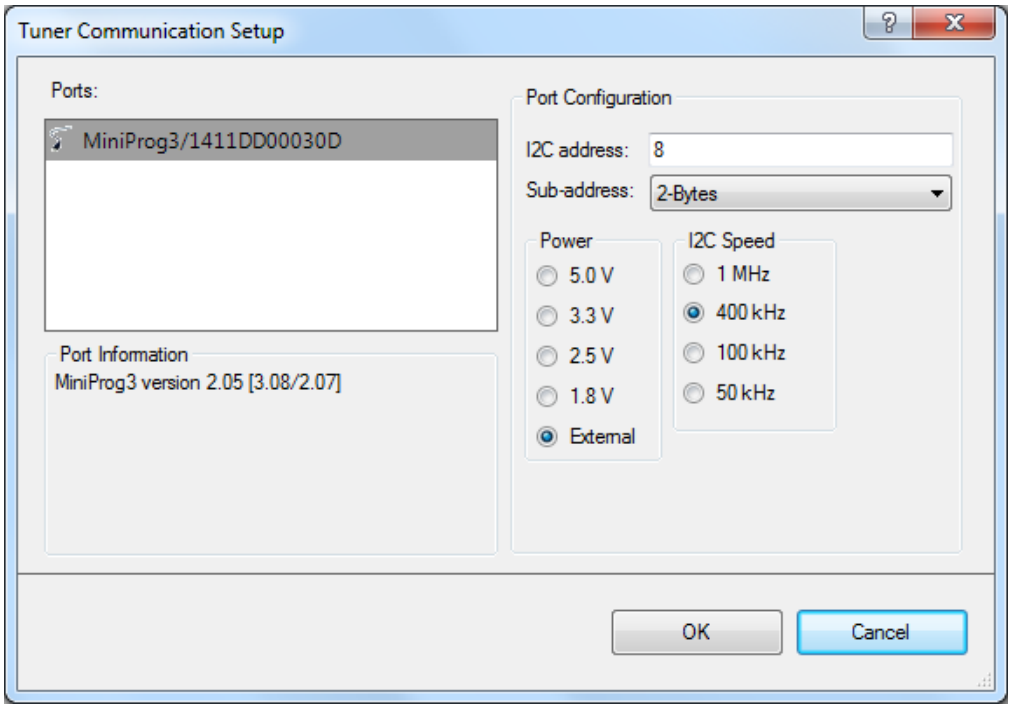
5. The debug pin is in the enabled state and is connected to Pin3[1]. Disable the debug pin Component or assign the debug pin to Pin 1[0] because Pin3[1] is needed for tuning the trackpad. Assign Pin3[1] and Pin3[0] to CSD_Touchpad_Tuner:scl and CSD_Touchpad_Tuner:sda respectively.
6. Rebuild the mouse by choosing **Build** > **Clean and Build CY5682 Touch Mouse**. Program the hex into the touch mouse.
7. Solder four wires from the touch mouse to a 5-pin male connector in the following order: VDD (TP5), GND (TP6), no connect, Pin3[1], and Pin3[0]
8. Connect the 5-pin male header to 5-pin female header of MiniProg3.
9. Connect MiniProg3 to a PC using a USB cable.
10. Right-click on the CSD_Touchpad Component and choose **Launch Tuner**. The Tuner tool appears as shown in Figure 5-24.

Figure 5-24. Tuner Tool



11. Click on **Configuration** button to invoke Tuner Communication Setup window. Configure the settings as shown in Figure 5-25.

Figure 5-25. CY5682_Touch_Mouse.cydwr View

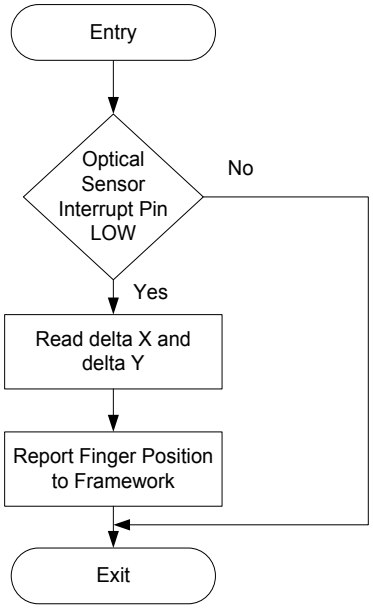


12. Insert two AAA batteries in the touch mouse and set the ON/OFF switch to the ON position.
13. Click on **Start** button and verify that the tuner starts running. For more information about how to use Tuner GUI, see the PSoC® 4 CapSense® Tuning Guide

5.1.5 Optical Sensor Subsystem

The optical sensor subsystem reads the delta X (change in X coordinates from the previous read operation) and the delta Y (change in Y coordinates from the previous read operation) values from the optical sensor. It implements a two-wire custom serial protocol to communicate with the optical sensor. The optical sensor has an active LOW interrupt line to indicate the presence of valid XY data. The subsystem reads the status on this interrupt line before accessing the X/Y registers over the serial interface. This subsystem is polled every 7.5 ms in the Active state. In the Idle state, the firmware is configured such that new data is read only when an interrupt is received rather than through polling. The application framework sends the values read from the optical sensor over the Bluetooth Smart link per the HOGP specification. Figure 5-26 shows the flow chart for the optical sensor subsystem.

Figure 5-26. Optical Sensor Subsystem Flow Chart



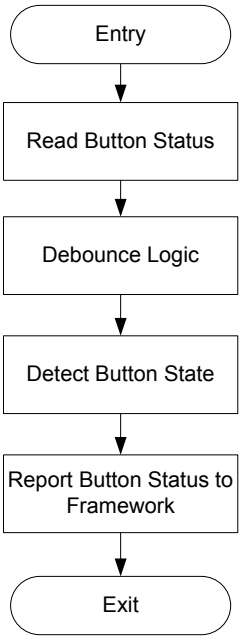
5.1.6 Button Subsystem

The touch mouse consists of seven buttons: left-click, right-click, middle-click, Windows button, side button 1, side button 2, and the connect button. The button subsystem polls the button status every 7.5 ms in the Active state. In the Idle state, each button is configured to generate an interrupt. Table 5-3 lists the buttons and the default functionalities provided by the kit firmware. Figure 5-27 shows the flow chart for the button subsystem.

Table 5-3. Buttons and Functionalities

Button	Functionality
Left	Left mouse click
Right	Right mouse click
Middle	Middle mouse click
Windows	Invoke [**Windows**] key
Side Left (Side Button 1)	Invoke [**Windows**] [**Ctrl**] [**Backspace**] key combination (toggle across open apps in Windows 8 and later)
Side Right (Side Button 2)	Invoke [**Windows**] [**C**] key combination (open Charms bar in Windows 8 and later)
Connect	Initiate advertisement for connecting with a host

Figure 5-27. Button Subsystem Flow Chart



Buttons are connected to the PRoC BLE GPIOs, and they are internally pulled HIGH (1). The default state of all buttons is read as HIGH (1). When a button is activated, the GPIO state changes to LOW (0). The firmware reads the port status and applies debounce logic to derive the button state. The button state is then reported to the application framework, which, in turn, sends it over the Bluetooth Smart link per the HOGP specification.

5.1.7 Battery Monitoring Subsystem

The battery monitoring subsystem measures the voltage at the battery terminal. This subsystem is polled every three seconds. The PSoC Creator's **ADC SAR Seq** Component is used to measure the battery voltage. Figure 5-28 shows the flow chart for this subsystem.

Figure 5-28. Battery Monitoring Subsystem Flow Chart

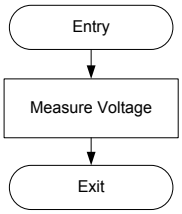
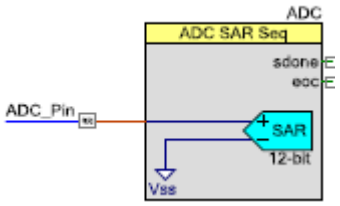


Figure 5-29 shows the schematic of the Sequencing SAR ADC Component. The following sections discuss the configuration settings.

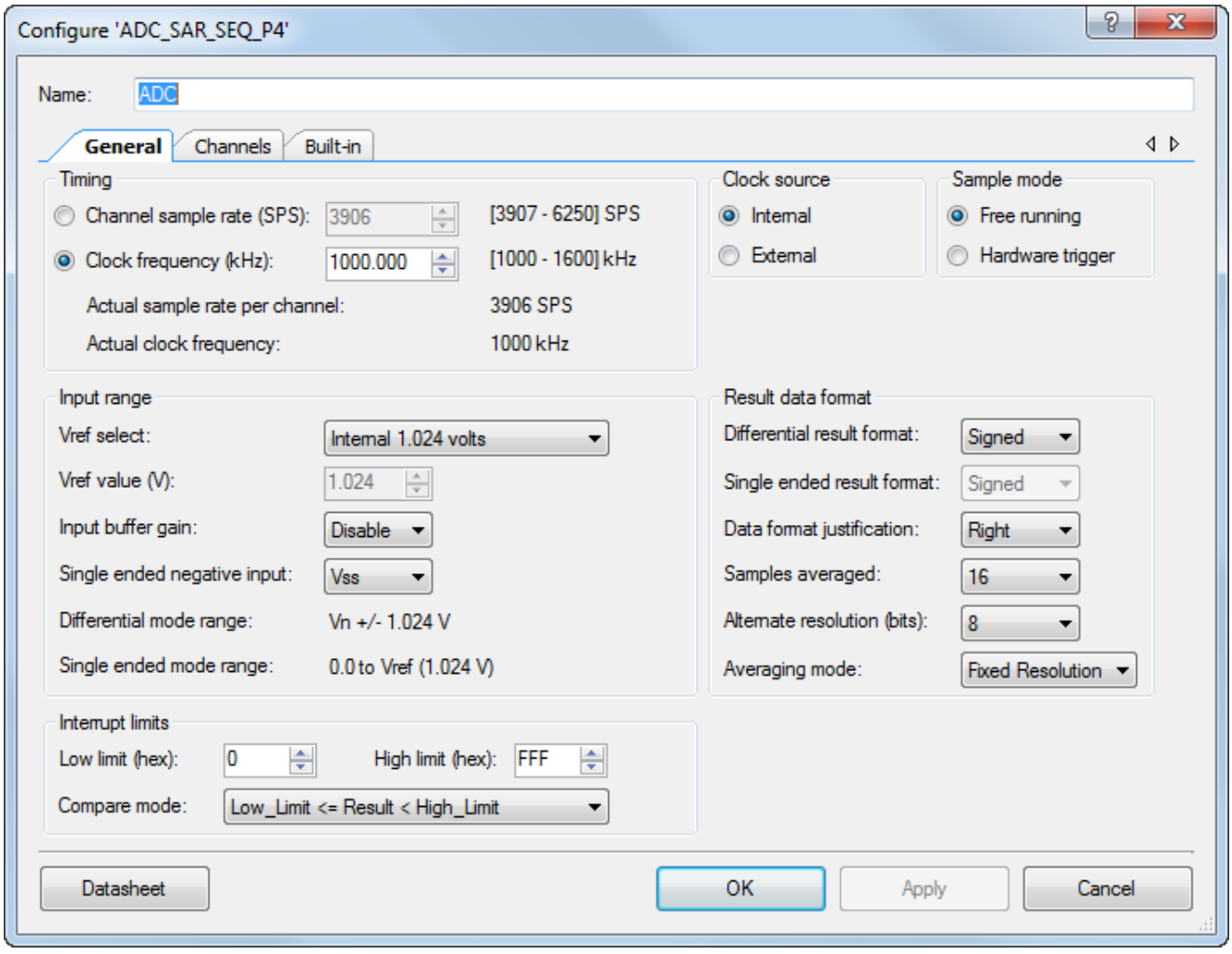
Figure 5-29. Sequencing SAR ADC Component of PSoC Creator



5.1.7.1 General Tab

The **Clock frequency** and other parameters shown in Figure 5-30 are selected to achieve a sampling rate of 3,906 samples per second.

Figure 5-30. General Tab of Sequencing SAR ADC Component

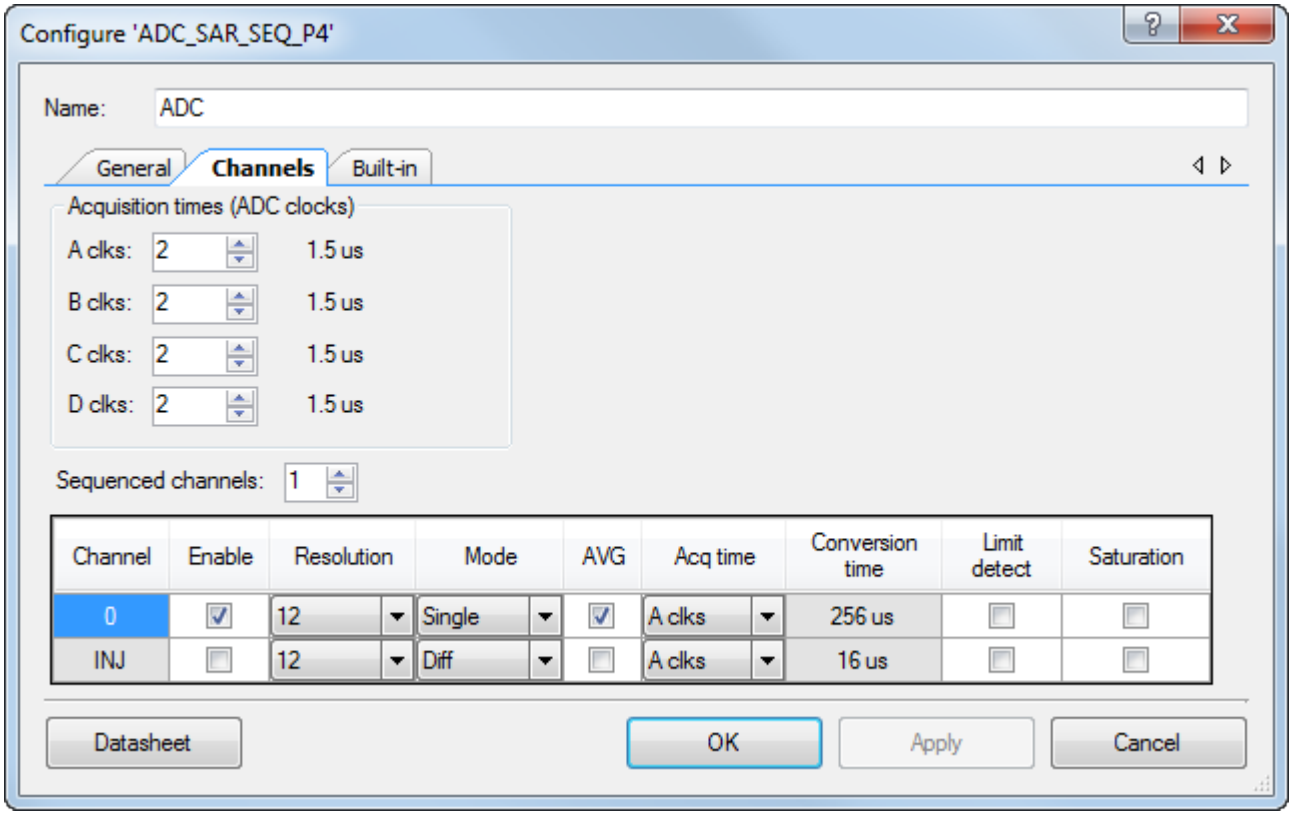


- **Clock frequency (kHz):** The touch mouse configures this parameter as “1000.000” to minimize power consumption.
- **Vref select:** The touch mouse configures this parameter as “Internal 1.024 volts,” as it has a potential divider circuit for battery voltage.
- **Samples averaged (bits):** The touch mouse configures this parameter as “16”. This value is sufficient to eliminate noise and save time and power for the touch mouse application. A smaller value for this parameter would result in an increase in noise. A higher value for this parameter would result in an increase in execution time and therefore, power.

5.1.7.2 Channels Tab

The ADC operates with a single-ended 12-bit resolution input. The clock is set to **A clks** to get a conversion time of 256 µs, as shown in Figure 5-31. Note that averaging is enabled for this channel (AVG is checked).

Figure 5-31. Channels Tab of Sequencing SAR ADC Component

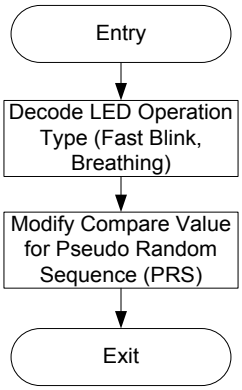


The datasheet provides a list of APIs supported by the Sequencing SAR ADC Component as well as additional details on configuration. To go to the datasheet, click the **Datasheet** button at the bottom left corner of the Sequencing SAR ADC Component GUI, as shown in Figure 5-31.

5.1.8 LED Subsystem

The LED subsystem shows device notifications to the user through red and orange LEDs and supports blinking (at different rates) and the breathing effects. This subsystem is executed every 1.25 ms. Figure 5-32 shows the flow chart for the LED subsystem.

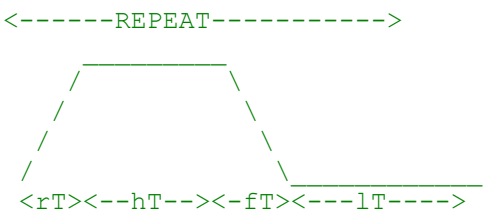
Figure 5-32. LED Subsystem Flow Chart



Touch Mouse LED Effects:

- The orange LED slow breathing effect is used to notify general or undirected advertisement. (This behavior is observed when the Connect button on the touch mouse is activated)
- The orange LED fast breathing effect is used to notify directed advertisement to the host (This behavior is observed if the touch mouse was bonded to at least one device earlier and it is either out of range or received a disconnect)
- The red LED slow breathing effect is used to notify a low-battery condition (This behavior is observed when the battery voltage is between 0.9 V and 1.1 V)

The slow or fast breathing effect can be realized by configuring the parameters as shown below. The firmware (*led.c* file) takes these parameters as inputs and generates specific patterns through the Timer, Counter, PWM (TCPWM) Component configured as a PWM to achieve the desired breathing effect.



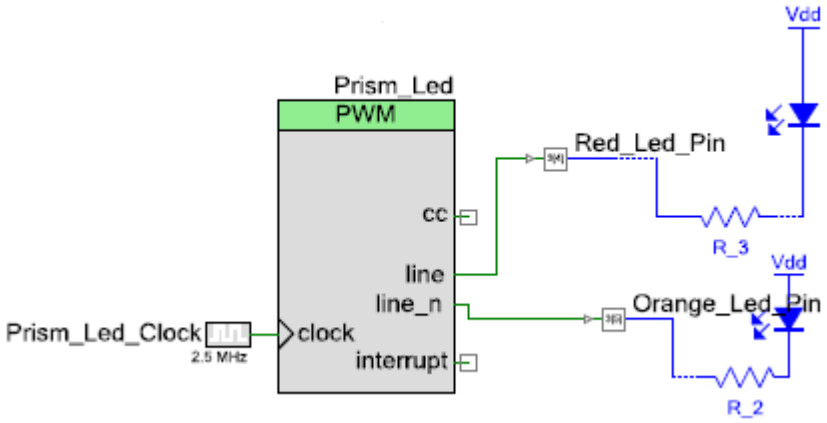
Parameters:

- riseTime(rT) - Time in millisecond for rise time
- highTime(hT) - Time in millisecond for high standby time
- fallTime(fT) - Time in millisecond for fall time
- lowTime(lT) - Time in millisecond for low standby time
- REPEAT - Repeating the above pattern

The blinking or breathing effect on the LEDs is generated via the PWM Component (Figure 5-33) of PSoC Creator. The following sections cover the configuration of the PWM Component.

As explained in the earlier sections, the touch mouse has two LEDs, but only of them is active at any given time. Therefore, two LEDs connected to one PWM Component will serve the required purpose. By default, both LED GPIOs are put in high-impedance mode to disable the corresponding LED. The drive mode of the required LED is modified to strong based on the touch mouse events which trigger LED notifications.

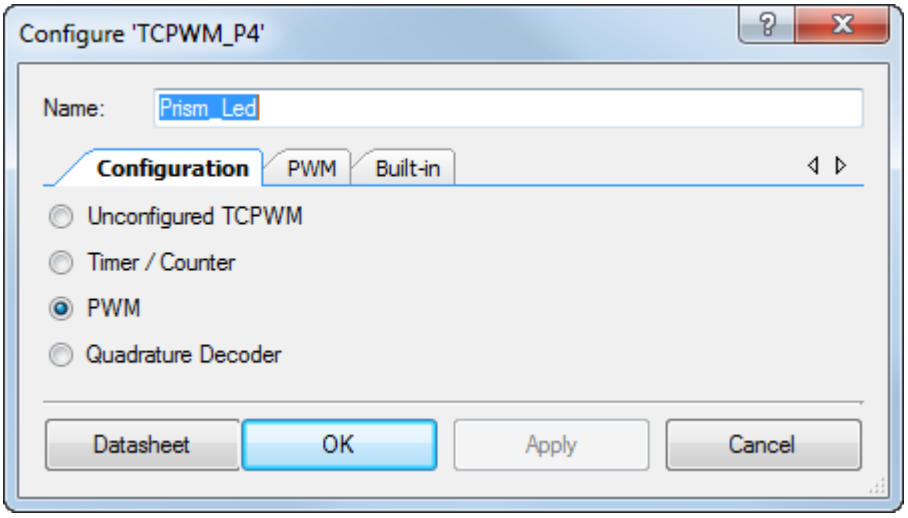
Figure 5-33. PWM Component of PSoC Creator



5.1.8.1 Configuration Tab

The **PWM** option is selected, as shown in Figure 5-34.

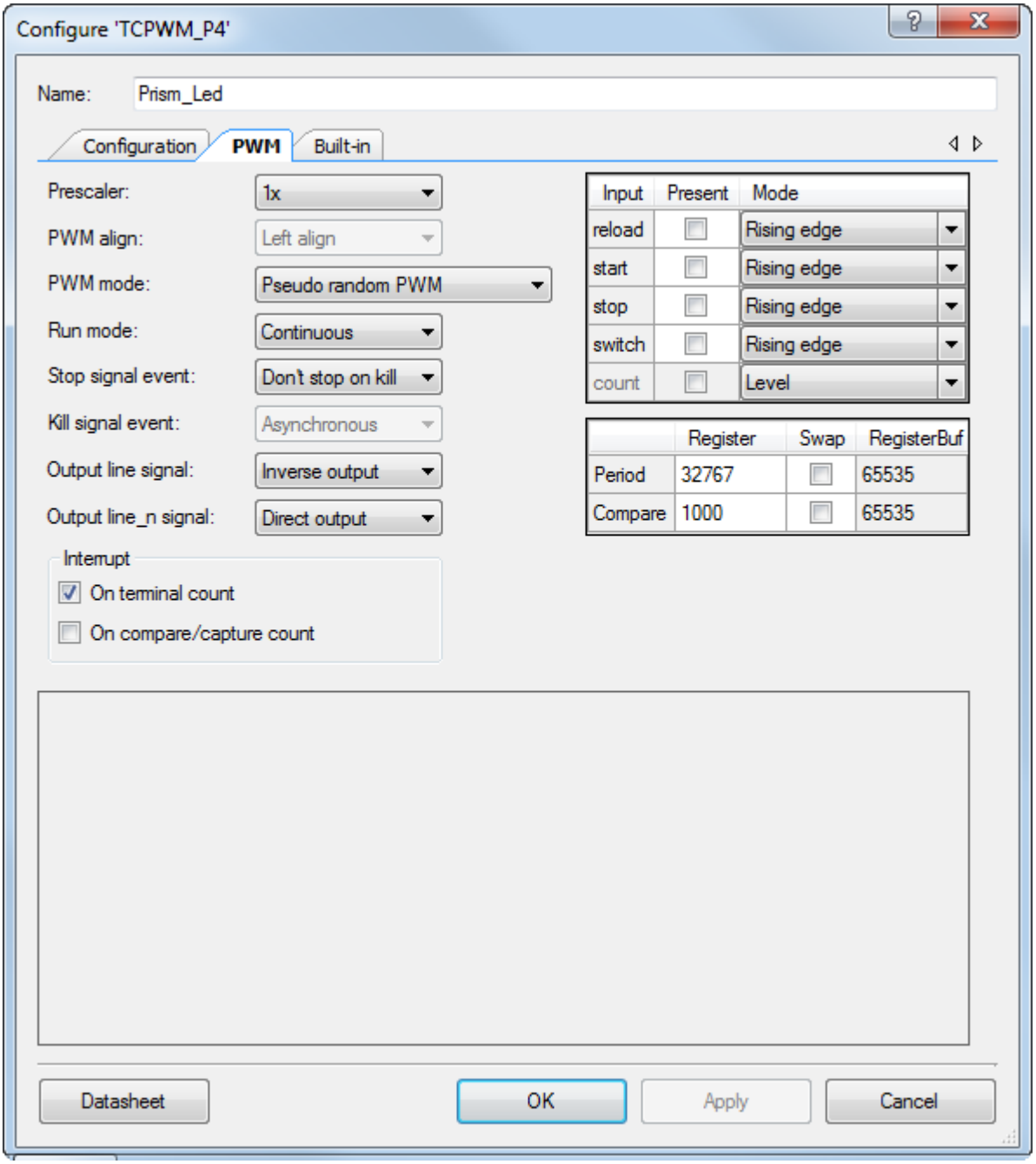
Figure 5-34. Configuration Tab of TCPWM Component



5.1.8.2 PWM Tab

The **PWM mode** is set as “Pseudo random PWM” to reduce the noise radiation from the LED. The **Period** and **Compare** values are set to “32767” and “1000” respectively, as shown in Figure 5-35, which gives a default duty cycle of approximately 30%. The period value along with the 2.5-MHz input clock results in an output waveform with a frequency of approximately 76 Hz.

Figure 5-35. PWM Tab of TCPWM Component



The datasheet provides a list of APIs supported by the TCPWM Component as well as additional details on configuration. To go to the datasheet, click the **Datasheet** button at the bottom left corner of the TCPWM Component GUI shown in Figure 5-35.

5.1.9 Flash Subsystem

The flash subsystem supports storing and retrieving of the peer (client) device BD (Bluetooth Device) address. The touch mouse receives the BD address of the peer device on a connection event. This address is stored in flash memory. The peer device BD address is needed for directed advertisement; therefore, it is retrieved from flash every time the touch mouse system power is reset.

The firmware provides an option to disable writes to flash by disabling macro “ENABLE_BONDING” in the *platform.h* file.

5.1.10 Debug Subsystem

The debug subsystem is used print debug information on the UART terminal. This subsystem is disabled by default. The firmware provides an option to enable debug by enabling the “DEBUG_PRINT” or “SOFTWARE_UART” macro in the *platform.h* file. If the “DEBUG_PRINT” macro is enabled, you should select the serial communication block UART Component in schematic view of PSoC Creator; if “SOFTWARE_UART” macro is enabled, select the software Transmit UART Component.

The debug subsystem provides an option to print selected debug logs by setting the appropriate debug level. For example, if the default debug level is set to ‘5’, during Device_FW_Initi() debug prints with setting greater than ‘5’ are printed.

5.1.11 Project Options (platform.h)

You can use the options provided in the firmware to enable or disable debug, touchpad tuning, usage of MCU power states, manufacturing test kit (MTK), and BLE power states.

- ***ENABLE_DEBUG_PIN –*** Define this macro to enable debug pin macros. Pin3[1] is configured as debug pin; this pin is accessible through a test point (TP9). The debug pin is enabled by default.
- ***DEBUG_PRINT –*** Define this macro to enable the debug print feature over UART. SCB UART should be enabled along with this macro. PRoC BLE Pin 3[1] should be configured as UART Tx. Disable the debug pin Component or configure the debug pin to Pin 3[0] while using the debug print feature.
- ***SOFTWARE_UART –*** Define this macro to enable the debug print feature over UART. The Software Transmit UART component should be enabled along with this macro. PRoC BLE Pin 3[1] should be configured as UART Tx. Disable the debug pin Component or configure the debug pin to Pin 3[0] while using the debug print feature.
- ***ENABLE_BLE_LOW_POWER_MODE –*** Define this macro to enable BLE to enter Deep Sleep mode.
- ***ENABLE_BONDING –*** Define this macro to enable the storage of bonding information in to flash. The touch mouse retrieves information from flash while doing directed advertisement to connect to a previously connected host.
- ***TOUCHPAD_TUNER –*** Define this macro to enable the touch pad tuning functionality.
- ***MCU_SLEEP_DISABLED –*** Define this macro to disable the MCU from entering Sleep state.
- ***MCU_DEEP_SLEEP_DISABLED –*** Define this macro to disable the MCU from entering Deep sleep state.
- ***ENABLE_MTK_MODE –*** Define this macro to enable manufacturing mode. The manufacturing mode firmware is used on the production line to screen out defective samples. For details, contact Cypress technical support (bleapps@cypress.com).

5.2 Firmware for PRoC BLE and PSoC 5LP on CySmart USB Dongle

The CySmart USB dongle has two modes of operation:

- HID host: In this mode, the dongle acts as the HID host and transfers the BLE data to USB. This is the normal operating mode.
- CySmart emulator: In this mode, the dongle performs all BLE operations triggered via the CySmart tool. This is used for examining the BLE data directly.

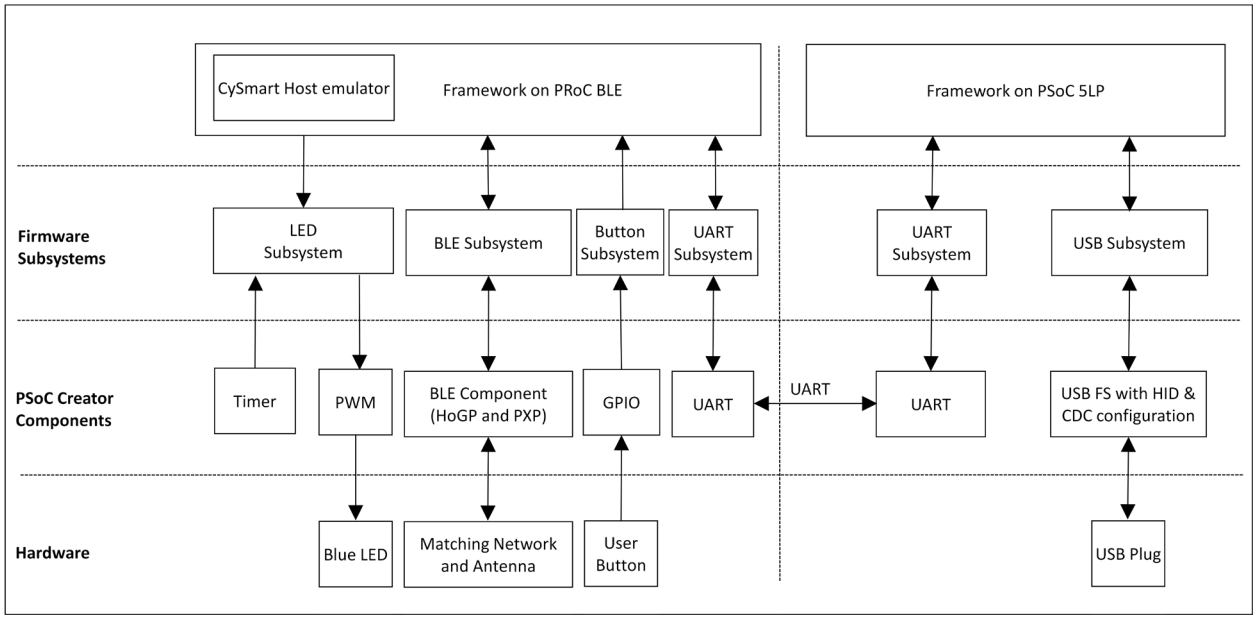
When powered on, the dongle is in the HID host mode and communicates with the HOGP-complaint BLE device. On connecting the dongle to the CySmart tool, the dongle switches to the CySmart emulator mode.

The CySmart USB dongle firmware comprises the firmware for both PRoC BLE and PSoC 5LP. Note that these two devices each have their own separate PSoC Creator projects although they are described together below..

5.2.1 Firmware Architecture

The firmware for the CySmart USB dongle consists of the framework (on PRoC BLE and PSoC 5LP) and various input/output subsystems, as shown in Figure 5-36.

Figure 5-36. PRoC BLE Firmware Architecture for CySmart USB Dongle



In HID mode, USB is configured with two endpoints - one for keyboard data and one for mouse data. The PRoC BLE device will format the HID packet along with a report ID and send it to the PSoC 5LP with a unique code for each of the HID endpoints.

In CySmart mode, only the UART interface will be used; the HID interface will be inactive. The UART data is sent to the PC by the PSoC 5LP using a virtual COM port via the USB interface.

The dongle firmware includes the following subsystems:

- BLE subsystem (PRoC)
- LED subsystem (PRoC and PSoC 5LP)
- Button subsystem (PRoC)
- UART subsystem (PRoC and PSoC 5LP)
- USB subsystem (PSoC 5LP)

These subsystems use PSoC Components. Figure 5-37 shows a schematic view of the PRoC BLE device and Figure 5-37 shows a schematic view of the PSoC 5LP device.

Figure 5-37. Schematic View of BLE HID CySmart Dongle Firmware (PRoC)

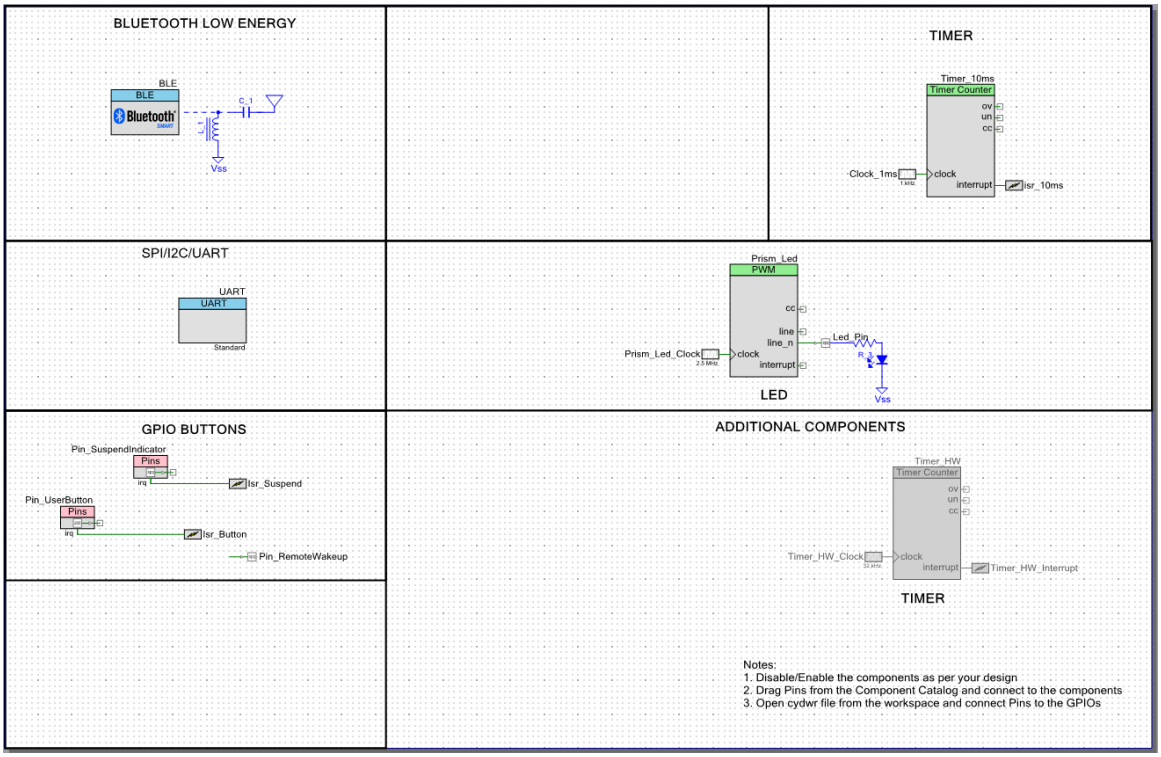
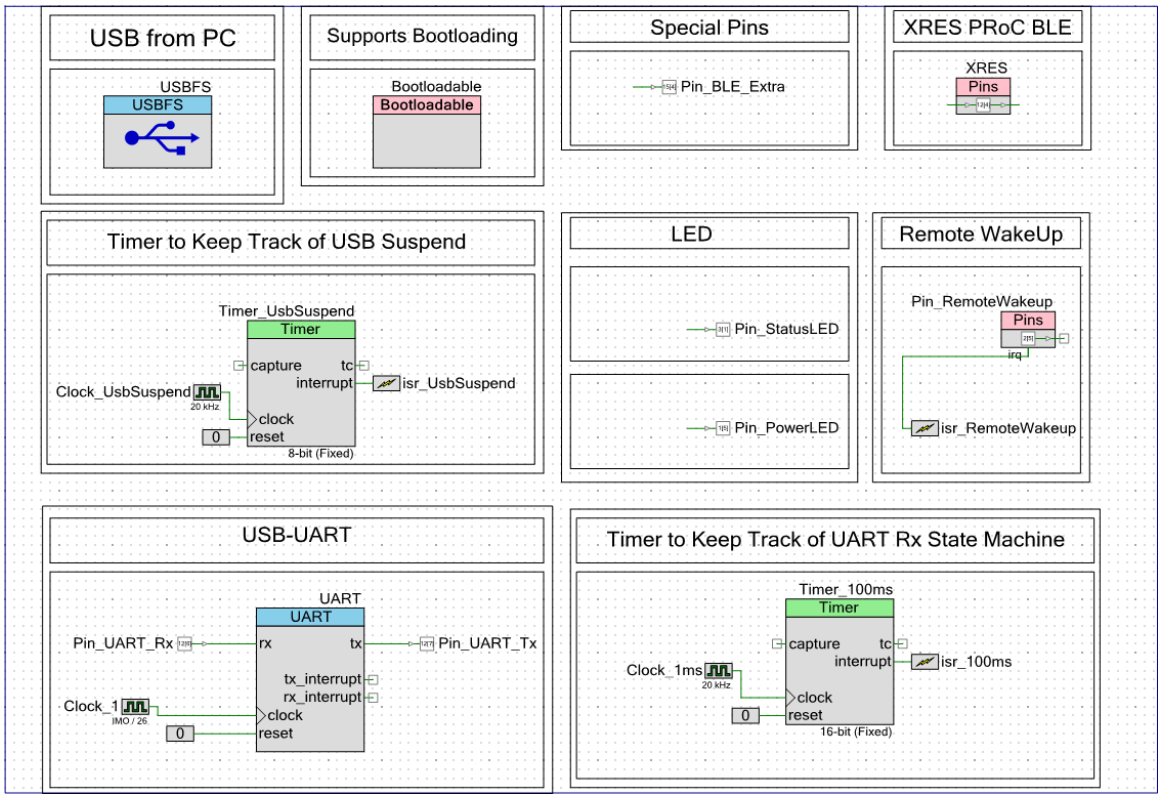
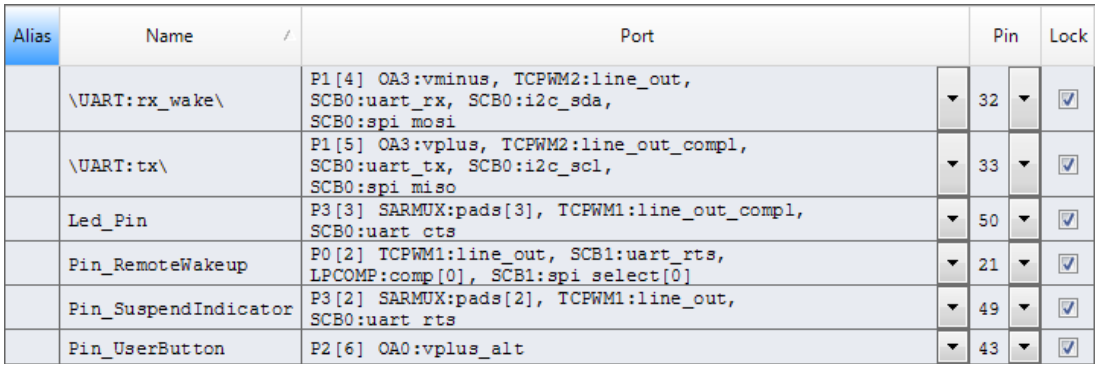


Figure 5-38. Schematic View of CY5682 Dongle Bridge Firmware (PSoC 5LP)



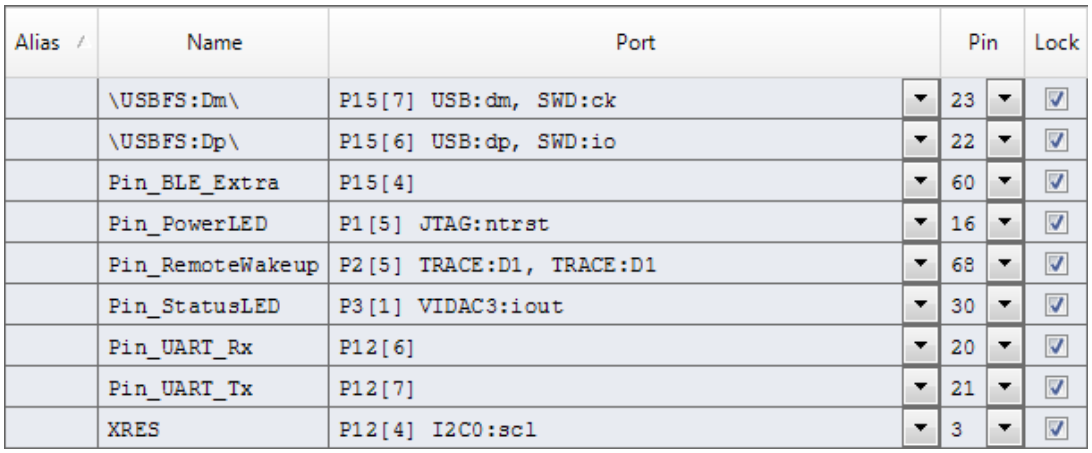
The GPIOs are configured per the dongle hardware connections, as shown in Figure 5-39.

Figure 5-39. GPIO Configuration of BLE HID CySmart Dongle Firmware (PRoC)



Alias	Name	Port	Pin	Lock
	\UART:rx_wake\	P1[4] OA3:vminus, TCPWM2:line_out, SCB0:uart_rx, SCB0:i2c_sda, SCB0:spi_mosi	32	☑
	\UART:tx\	P1[5] OA3:vplus, TCPWM2:line_out_compl, SCB0:uart_tx, SCB0:i2c_scl, SCB0:spi_miso	33	☑
	Led_Pin	P3[3] SARMUX:pads[3], TCPWM1:line_out_compl, SCB0:uart_cts	50	☑
	Pin_RemoteWakeup	P0[2] TCPWM1:line_out, SCB1:uart_rts, LPCOMP:comp[0], SCB1:spi_select[0]	21	☑
	Pin_SuspendIndicator	P3[2] SARMUX:pads[2], TCPWM1:line_out, SCB0:uart_rts	49	☑
	Pin_UserButton	P2[6] OA0:vplus_alt	43	☑

Figure 5-40. GPIO Configuration of CY5682 Dongle Bridge Firmware (PSoC 5LP)



Alias	Name	Port	Pin	Lock
	\USBFS:Dm\	P15[7] USB:dm, SWD:ck	23	☑
	\USBFS:Dp\	P15[6] USB:dp, SWD:io	22	☑
	Pin_BLE_Extra	P15[4]	60	☑
	Pin_PowerLED	P1[5] JTAG:ntrst	16	☑
	Pin_RemoteWakeup	P2[5] TRACE:D1, TRACE:D1	68	☑
	Pin_StatusLED	P3[1] VIDAC3:iout	30	☑
	Pin_UART_Rx	P12[6]	20	☑
	Pin_UART_Tx	P12[7]	21	☑
	XRES	P12[4] I2C0:scl	3	☑

Table 5-4 describes the file structure used by the projects and lists key functions implemented in each file:

Table 5-4. File Structure and Key Functions

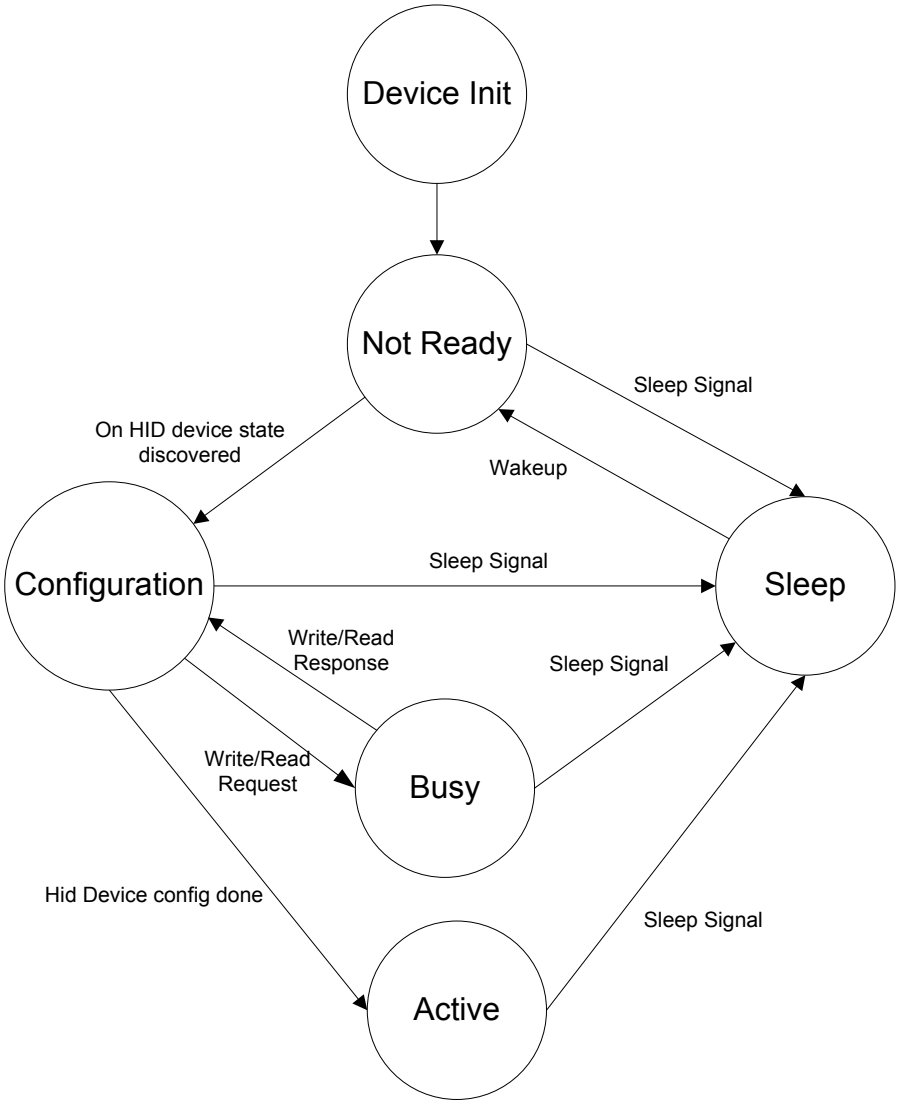
File name	Details
Project: BLE_HID_CySmart_Dongle (PRoC) main.c	This is the top-level application file. It initializes the system and runs the main loop. This file has the following functions: • *main()* – The main function for the application, handles the mode switching between HID mode & CySmart mode • *Device_Init()* – Initializes all the blocks(BLE, UART, LED, Timer and GPIOs) of the system • *Device_Timer_Callback()* – Handles the timer interrupt configured for LED module

File name	Details
Project: BLE_HID_CySmart_Dongle (PRoC) Application.c	This file handles the HID mode functionality of the project. It handles the BLE events and the state machine of HID host. This file has the following functions: • *BLE_Init()* – Handles the BLE-specific initialization and registers the callback functions for each profile. The BLE Component is started with HID mode configuration. • *BLE_DeInit()* – De-registers the callback functions for each profile and stops the HID mode BLE Component. • *Dongle_Ble_State_Handler()* – Handles the state machines for BLE scanning, connection, GATT Discovery, and profile-level configurations. • *FilterScanResponsePackets()* – Parses the advertisement packet from the device and filters the packets for HID service UUID. The Dongle will connect to devices containing HID service or to devices sending directed advertisement to the Dongle. • *SendEmptyReports()* – Sends dummy report on BLE disconnection. If any key down packet is sent by device before disconnection, this dummy report will ensure that the key press state is reverted. • *BleCallBack()* – Handles general events from the BLE stack like disconnect, timeout, scan response, authentication events, etc. • *HidsCallBack()* – Handles the HID-specific events. This function gathers the report IDs of each Report reference descriptor to match with the notification data. On receiving HID notification, this function matches it to the expected report ID and sends the report to the USB Bridge controller. • *BasCallBack ()* – Handles the BAS-specific events for notification enable and notification events. The Battery notification will be sent as HID feature report to USB interface. • *Process_Suspend_Command()* – Handles USB suspend command from USB Bridge controller. The system will go into Sleep mode as long as the suspend command is active. This function also sends the HID Suspend command to the HID device if it is connected. This will inform the device to enter Sleep mode.
Project: BLE_HID_CySmart_Dongle (PRoC) led.c	This module is same as described in mouse project. See the LED Subsystem section.
Project: BLE_HID_CySmart_Dongle (PRoC) timer.c	This module is same as described in mouse project. See the Timer section. Watchdog timer-based tick timer is used by default. The firmware has provision to use the “Timer_HW” Component instead of the watchdog timer by doing the following: • Enable **ENABLE_TIMER_COMPONENT** macro in *Timer.h*. • Disable **ENABLE_WDT_TIMER** macro in *Timer.h*. • Right-click on the **Timer_HW** Component in the schematic view and select “enable”.
Project: CY5682_Dongle_Bridge (PsoC 5LP) main.c	This is the top-level application file. It initializes the system and runs the main loop. This file has the following functions: • *main()* – Handles the USB reset and runs the main loop to monitor UART and USB data transfers. • *Device_Init()* – Initializes all the blocks of the system, waits for the USB enumeration and sets the enumeration LED. This function also resets the PRoC BLE after system initialization to ensure that both the controllers are in sync.

File name	Details
Project: CY5682_Dongle_Bridge (PsoC 5LP) USB_Interface.c	This file handles the bridge functionality between USB and UART interfaces. This file has the following functions: • *UARTTransmit()* – Checks for USB CDC OUT data, if present sends it to UART by configuring the expected baud rate by PRoC-BLE. • *PktRxStateMachine()* – Parses the UART packets and loads them into respective endpoint buffers. This runs a state machine on each RX byte and looks for the packet format. Predefined opcodes are used to identify the endpoint to which the decoded packet is to be sent. If a proper packet is detected, which does not belong to known opcodes, it will be routed to the CDC endpoint, which will be used by the CySmart tool. • *SendToUSB()* – Monitors each endpoint status and loads the pending data when the endpoint is available. For the touch mouse and keyboard endpoints, both endpoints must be free before a new packet can be sent so that the HID combo key order is maintained.

The framework running on PRoC BLE follows the state machine for the HID host mode, as shown in Figure 5-41.

Figure 5-41. State Machine for PRoC BLE Firmware



The following sections describe the states in the state machine.

5.2.1.1 Device Init State

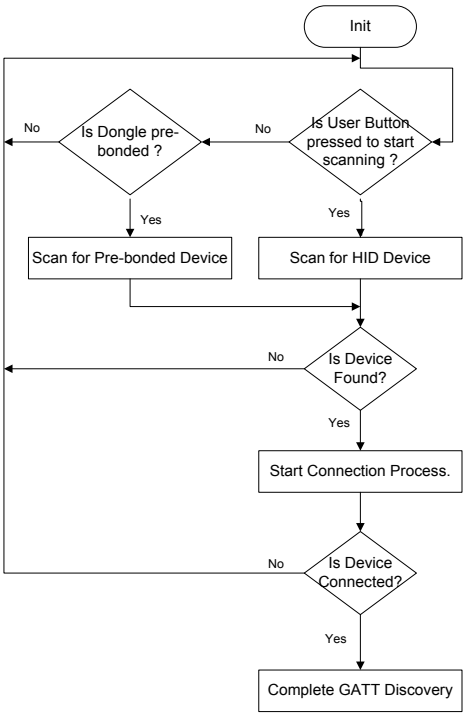
In the Device Init state, the following actions are performed:

- All global variables are initialized.
- The UART Component is initialized and started.
- ISRs for UART, button, and sleep mode are initialized and started.
- The LED module is initialized and started.
- The BLE Component is started and the callback functions for events are registered.

5.2.1.2 Not Ready State

The dongle will enter the Not Ready state (Figure 5-42) on each power cycle. This state represents that the BLE Component is not ready to accept any commands. In this state, the firmware continuously scans for a peripheral device until it gets a scan response from an HID device, and then it issues a connect request. On successfully connecting to the HID device, the dongle starts discovering the characteristics exposed by the server. When the discovery is complete, the device transitions to the Configuration state.

Figure 5-42. Code Flow in BLE Not Ready State



5.2.1.3 Configuration State

In the Configuration state, the dongle starts configuring the connected HID device per the required behavior. It performs the following operations on every new connection:

- Completes the bonding procedure with the security level “Unauthenticated pairing with encryption.” On successful bonding, the keys will be stored into flash.
- Reads the “Peripheral preferred connection parameters” attribute and updates the connection parameters of the BLE link
- Configures the “HID Control Point” characteristic with the value “Exit Suspend,” signaling to the HID device that the HID host device is exiting the power-saving mode
- Configures the HID protocol mode with the value “Report Protocol Mode”

- Configures the Immediate Alert characteristic with “No alert”, this alert level will be changed when the path loss is below the threshold value.
- Reads the Tx Power Level to calculate the path loss as required for the proximity monitor: Path loss = Tx Power – RSSI
- Read the “Report Reference descriptor index” which contains the Report ID mapping for each type of HID report configured by the HID Server, The dongle is configured to support a maximum of six HID Input records. The dongle doesn’t support HID output reports.
- Configures the Link Loss Alert Level characteristic with a high alert level
- Enables notifications for the Scan Refresh and battery-level characteristics
- Enables notifications for all the HID report types: Mouse, Keyboard, multimedia and power reports

On each GATT request, which requires the response, the PRoC BLE application framework enters the busy state until the corresponding pass/fail response is received from a peripheral such as a mouse or remote control. When the configuration is complete, the PRoC BLE application framework enters the Active state.

5.2.1.4 Busy State

This state indicates that the dongle is waiting for a response from a peer device and is not available to issue the next BLE command.

5.2.1.5 Active State

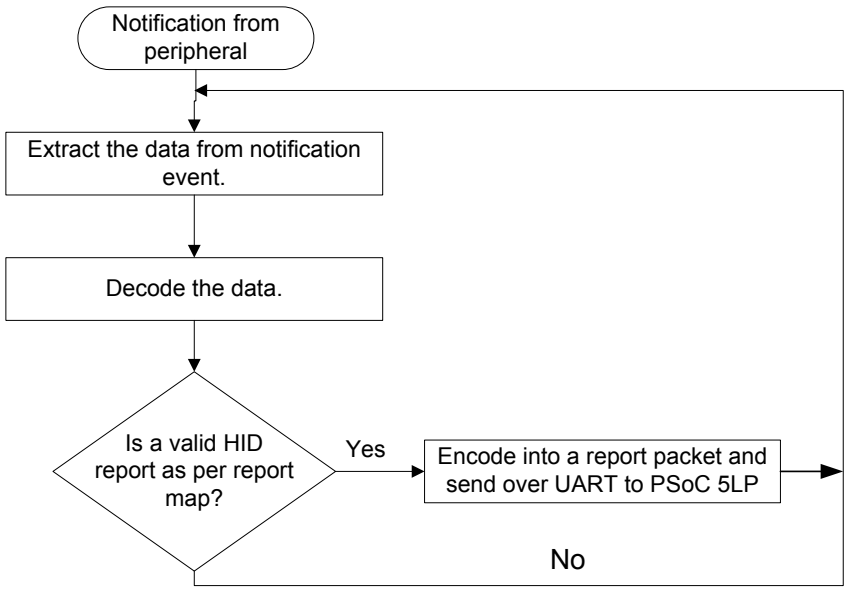
This state indicates that the dongle has completed the configuration of the HID device and is available for BLE commands.

The RSSI level is monitored continuously; when the signal strength is below the threshold, the dongle sets an immediate alert with “High alert,” which will be used by the mouse to signal the user about the signal quality. This alert will revert to “No alert” if the signal strength exceeds the threshold.

Note: By default, the proximity profile implementation is disabled. It can be enabled using the ENABLE_PATH_LOSS_ALERT macro in *Application.c* source file.

When the dongle reaches the Active state, it starts processing HID report notifications, encodes them into the UART packet format, and sends them to the PSoC 5LP. Figure 5-43 shows the flow chart of this operation when the HID report is received from the BLE HID device.

Figure 5-43. Code Flow for HID Notifications



5.2.1.6 Sleep State

A host PC issues the USB suspend command if the PC is in sleep mode. To enable the PRoC BLE device to detect the suspend command, one GPIO of PSoC 5LP is used as a suspend indicator. This GPIO is triggered by PSoC 5LP on receiving the USB suspend command.

PRoC BLE monitors this pin for sleep and wake commands by configuring a GPIO interrupt. The dongle enters the low-power mode on suspend; it exits the low-power mode when the pin state changes. This transition can happen in all other states.

5.2.2 Bluetooth LE Subsystem

The PRoC BLE-based firmware configures the BLE Smart Component available in PSoC Creator with HOGP and the profile role as Report and Boot Host (GATT client) with support for the following services:

- Device Information
- Scan Parameters
- Battery
- Human Interface Device
- Immediate Alert
- Link Loss
- Tx Power

A GATT server (HID device) configured as a peripheral (GAP role) can start an advertisement to connect to the central device (CySmart USB dongle). When the peripheral advertises, the client detects the device and connects to it. After connecting to the device, the client discovers the services provided by the HID device and enables notifications for the characteristics it supports.

After the client configures the peripheral device, it receives data when the peripheral device sends a notification. The received data is then decoded by the application logic and transmitted to the PSoC 5LP firmware via UART.

The BLE subsystem is built on the BLE Component (Figure 5-44) provided in PSoC Creator.

Figure 5-44. BLE Component of PSoC Creator

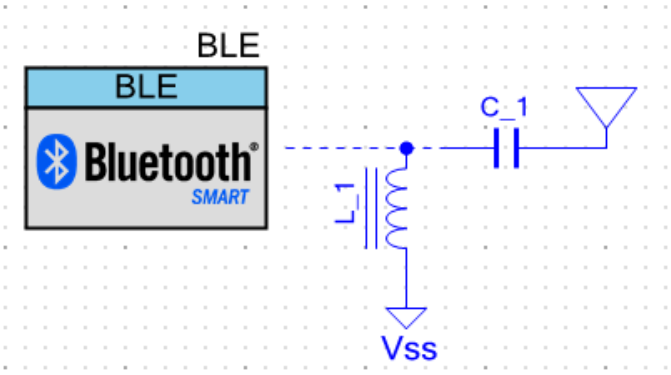


Figure 5-45 and Figure 5-46 show the configuration for the BLE Component.

Figure 5-45. BLE Component General Configuration

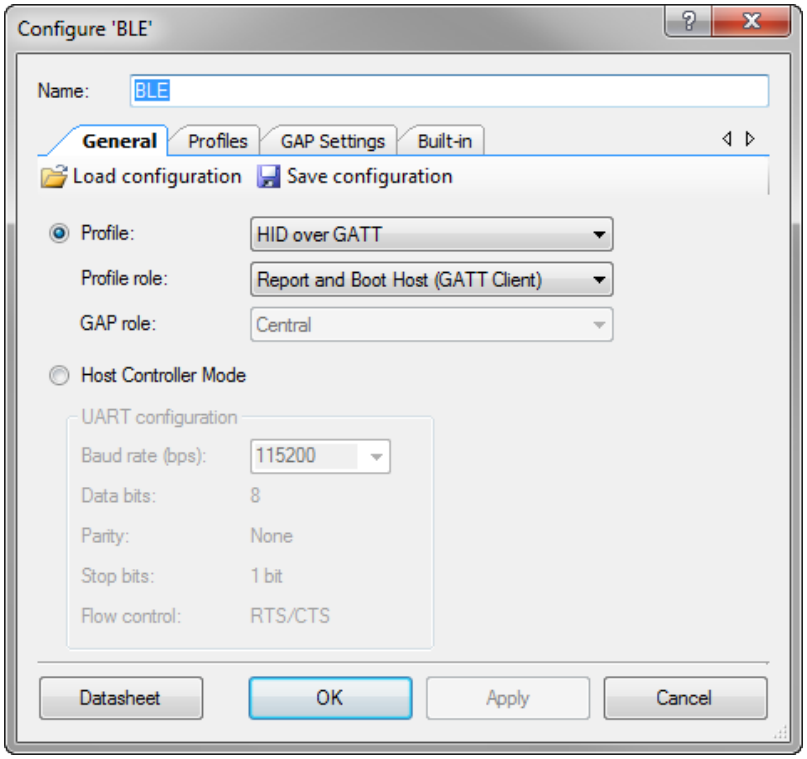
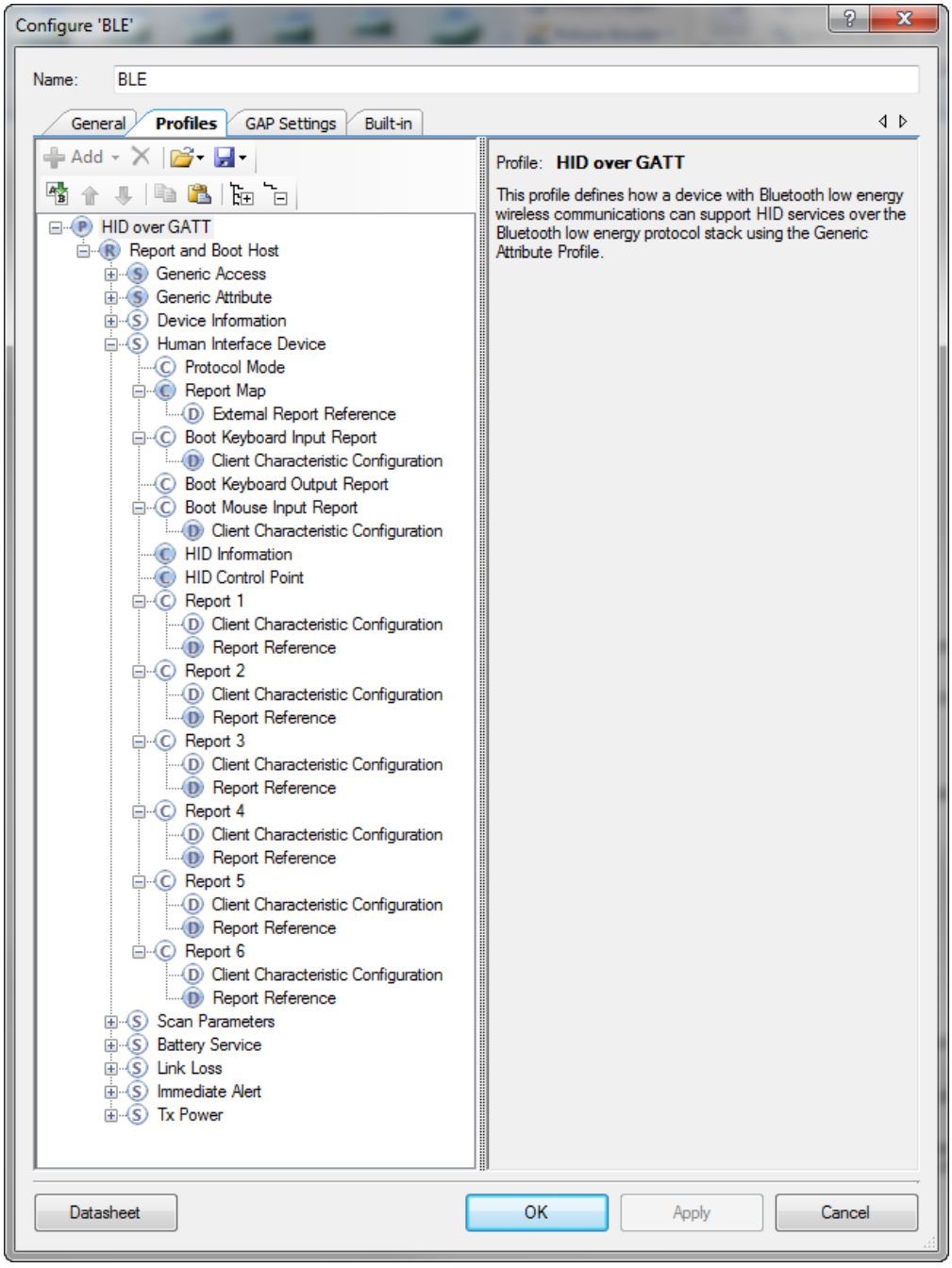


Figure 5-46. Profiles Supported by the Host Device



Open the BLE Component wizard from the project for more details of Component configuration.

All HID report notification except keyboard are appended with the report ID from the corresponding report reference before sending to UART to match the report with USB HID Report map configured in USB subsystem.

5.2.3 LED Subsystem

The dongle has three LEDs. The red and green LEDs are connected to PSoC 5LP, while the blue LED is connected to the PRoC BLE. The blue LED has three modes of operation:

- Blinking effect: The dongle is scanning for new HID devices; this is initiated by pressing the user button (SW2).
- Slow-breathing effect: The dongle is scanning for preconnected devices.

- Stay on: The BLE connection to the HID device is active.

The LED subsystem uses the PWM and Timer Components to create the effects required by the dongle. The design is the same as that of the mouse explained in the LED Subsystem section.

PSoC 5LP has two LEDs:

- The red LED indicates the power status. It is on if the dongle is powered.
- The green LED indicates the USB status:
  - Stays on upon successful enumeration
  - Blinks if the dongle is in the bootloader mode

5.2.4 Button Subsystem

The dongle has one user button, which is configured for an edge-triggered input. This button is used as the pairing button, which triggers the dongle to scan for new BLE HID peripherals.

The dongle, by default, will only scan for pre-bonded devices if they are present in the bonded list. If no devices are present in the list, the dongle will wait for a button press to start scanning.

When this button is pressed, the dongle disconnects the current connection if one exists. It starts scanning for new devices by ignoring the prebonded device list. So, whenever you want to connect a new HID device to the dongle, press the pairing button.

5.2.5 UART Subsystem

The UART subsystem is the core subsystem in dongle operation. The UART interface connects the **PRoC BLE** and **PSoC 5LP** controllers to each other.

The PRoC BLE UART subsystem frames the packet for each type of data using the format as described below. The PSoC 5LP UART subsystem decodes the packet and routes it to the corresponding USB endpoints.

5.2.5.1 PRoC BLE UART

The UART subsystem is built on the UART Component provided in PSoC Creator. The UART Component is configured as shown in Figure 5-49.

Figure 5-47. UART Component of PSoC Creator

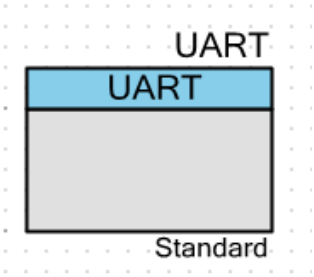


Figure 5-48. UART Component UART Basic Tab

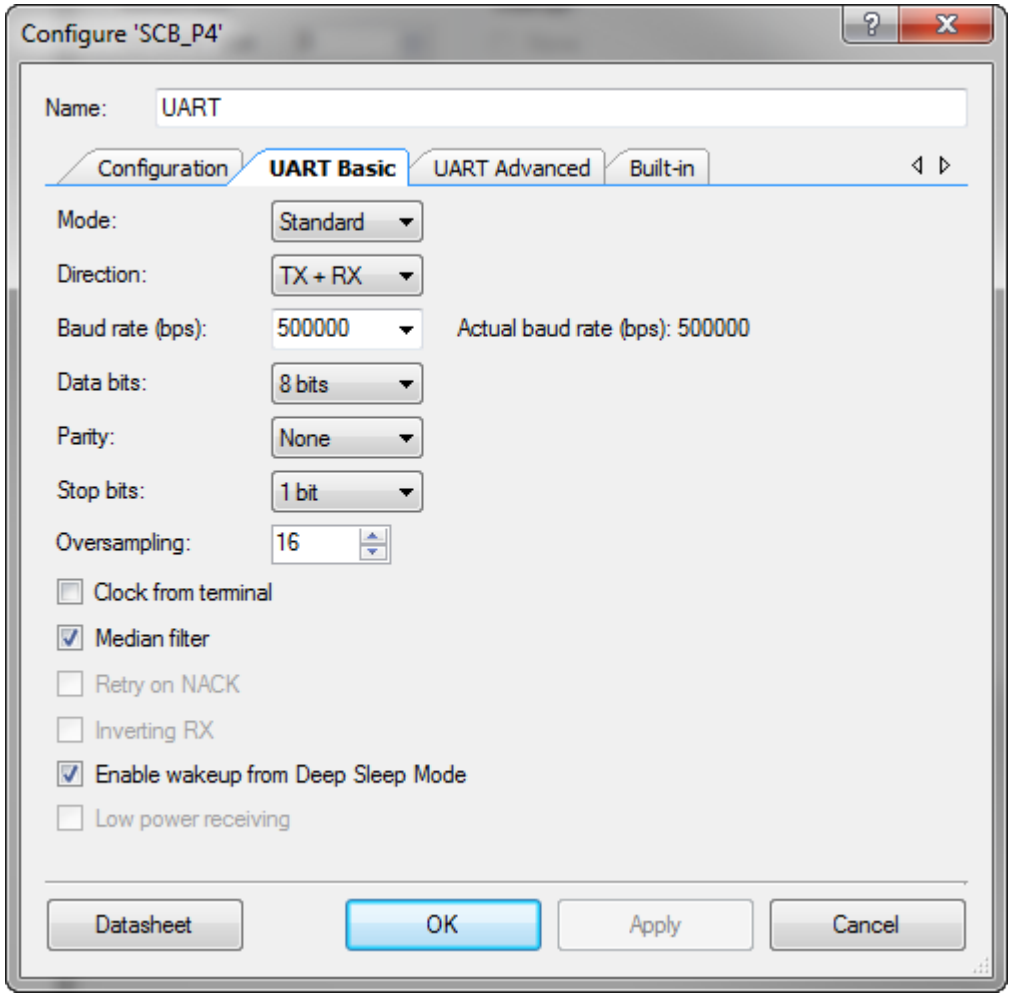
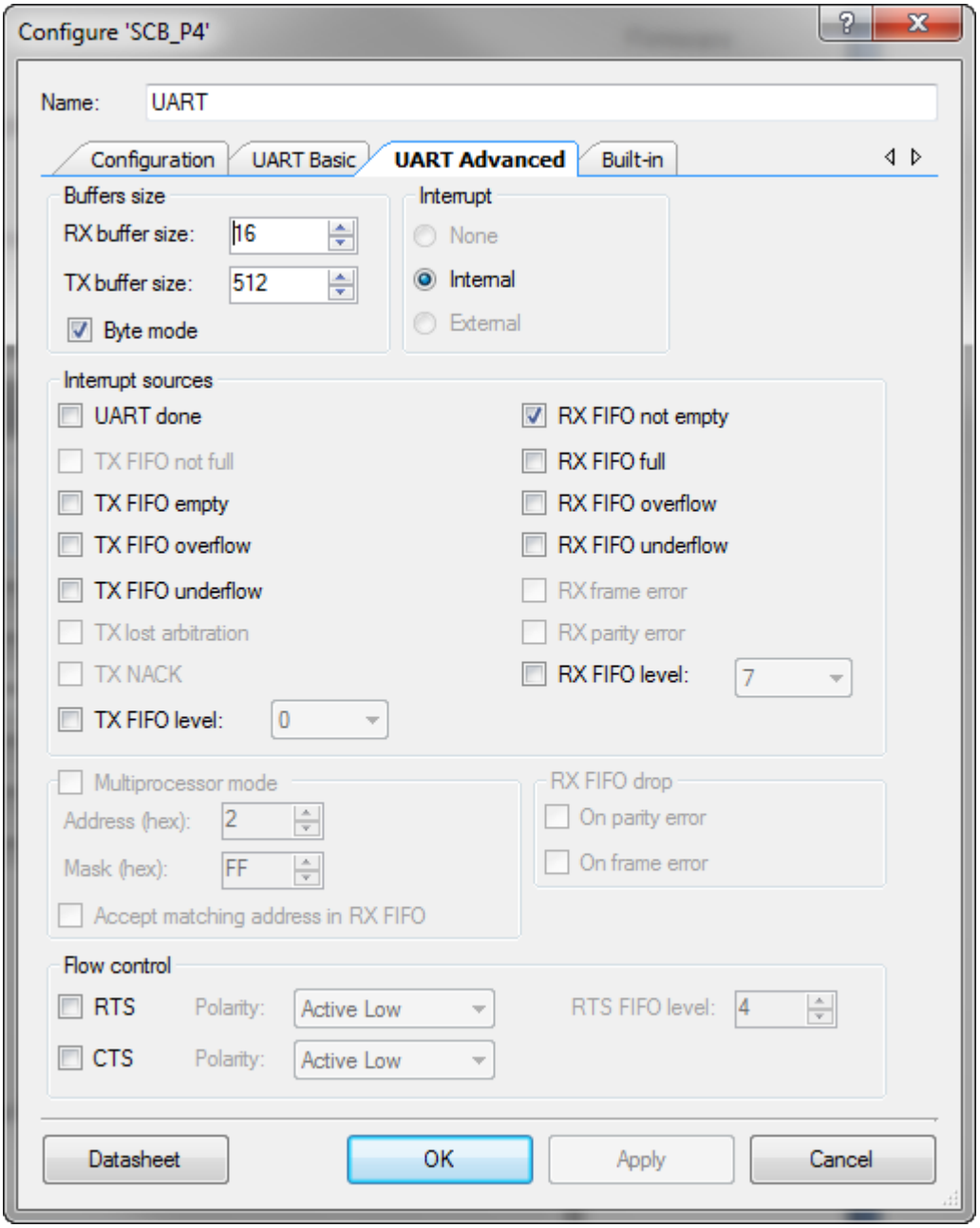


Figure 5-49. UART Component UART Advanced Tab



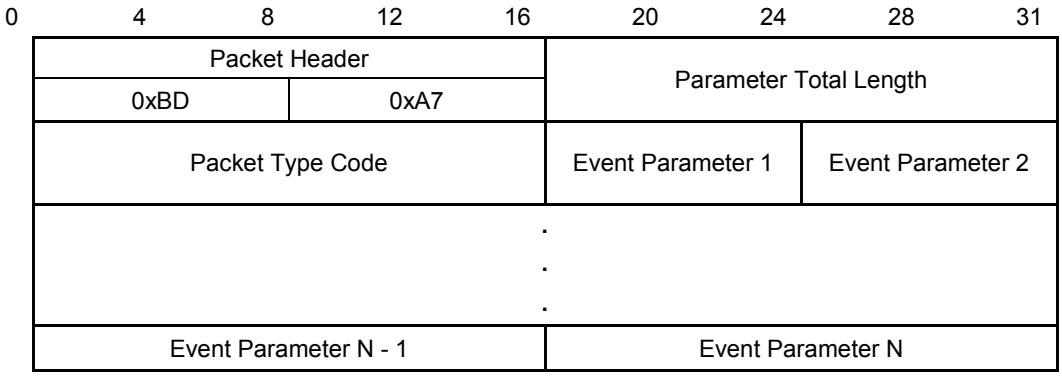
This subsystem has two modes of operation:

- HID host mode
- CySmart emulator mode

In the HID host mode, this subsystem transfers the formatted packets to UART. The BLE subsystem on PRoC BLE packetizes the received data notifications (mouse, keyboard, multimedia, battery, and power data) over the BLE link and sends them to the UART subsystem.

The UART packet structure shown in Figure 5-50 is decoded by the PSoC 5LP UART subsystem.

Figure 5-50. UART Packet Structure



Each row in Figure 5-50 corresponds to four bytes of data.

The packet type codes are predefined to differentiate among types of data to be sent to the PC:

```
MOUSE_REPORT = 0x0461u
KEYBOARD_REPORT = 0x0462u
```

In PSoC 5LP, the mouse endpoint is configured to use the report ID, whereas the keyboard endpoint doesn't use the report ID. See the USB Subsystem section.

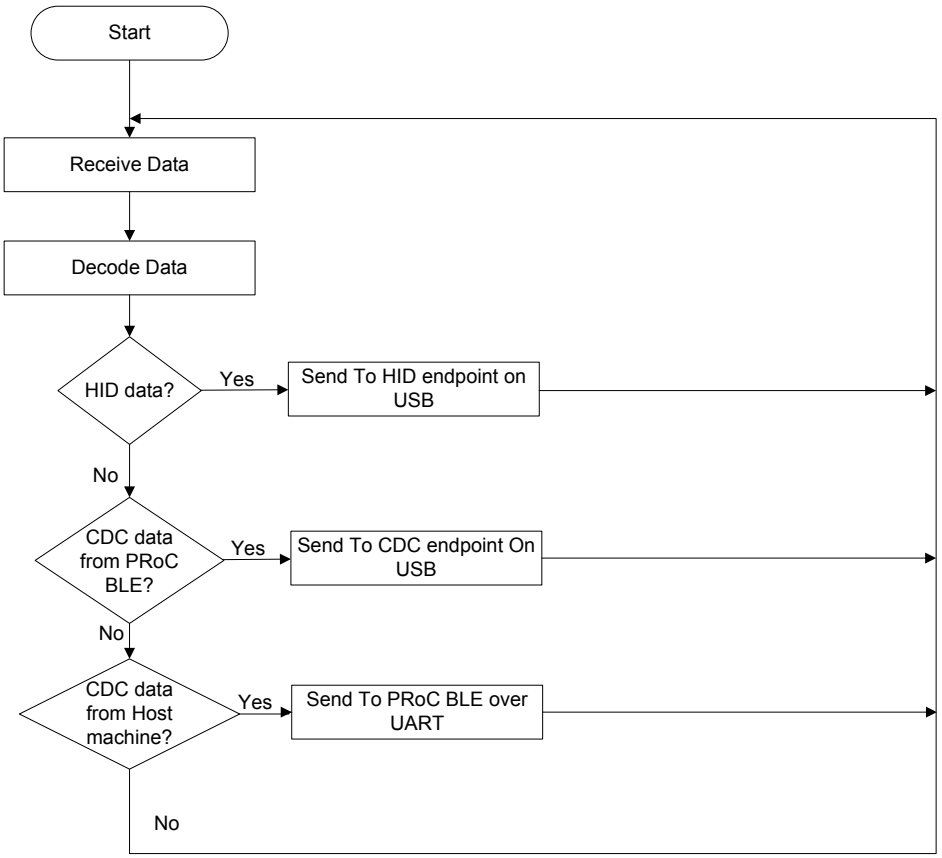
To match this configuration, the report ID number is appended to the mouse report notifications before sending it to the UART. Keyboard notifications will be sent only by adding the packet header fields.

In the CySmart emulator mode, the UART subsystem uses a custom command-event protocol to communicate with CySmart via a custom CDC interface.

5.2.5.2 PSoC 5LP UART

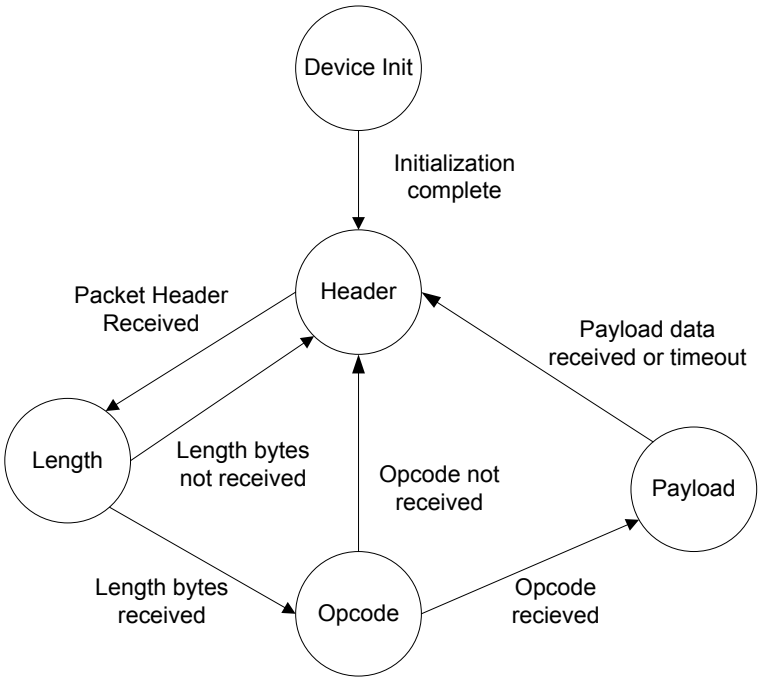
This subsystem decodes the data packets and sends them to the respective USB interface. The UART data is processed only if it follows the defined packet format. Figure 5-51 shows the UART to USB subsystem data flow.

Figure 5-51. UART to USB Subsystem Data Flow Chart



The state machine shown in Figure 5-52 decodes the packet format and extracts the data as described in the following sections.

Figure 5-52. UART Packet State Machine



5.2.5.2.1 Device Init

In the Device Init state, the following activities are performed:

- All global variables are initialized.
- Power LED is turned on.
- ISR for the USB suspend command is started.
- ISR for the UART Component is started.
- The USBFS Component is started and the status LED is switched on (after successful start and enumeration).
- The UART Component is started.
- PRoC BLE is reset.

5.2.5.2.2 Header State

In this state, the application logic processes the data received by the UART and transitions to the Length state on receipt of a valid header.

5.2.5.2.3 Length State

In this state, the application logic receives the length bytes and transitions to the Opcode state on receipt of a valid length; otherwise, it transitions to the Header state.

5.2.5.2.4 Opcode State

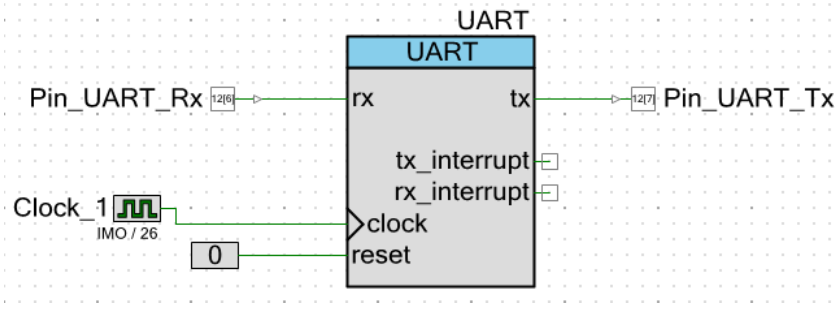
In this state, the application logic receives the opcode bytes and transitions to the payload state on receipt of a valid opcode; otherwise it transitions to the Header state.

5.2.5.2.5 Payload State

In this state, the application logic receives the payload bytes, transitions to the Header state on receipt of a valid data payload, and starts processing the received packet. On receipt of an invalid payload, the application logic transitions to the Header state and repeats the entire cycle.

This subsystem is built using the UART Component (Figure 5-53) provided in PSoC Creator.

Figure 5-53. UART Schematic View



The Component is configured as shown in Figure 5-54 and Figure 5-55.

Figure 5-54. UART Component Configure Tab

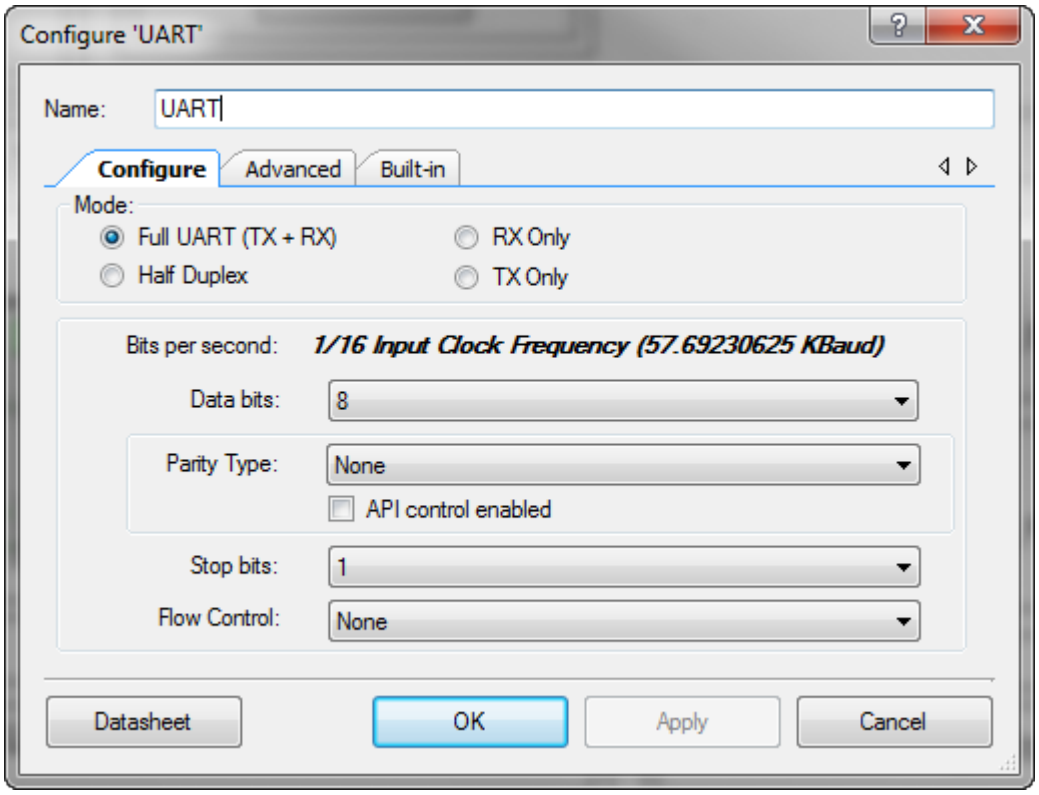
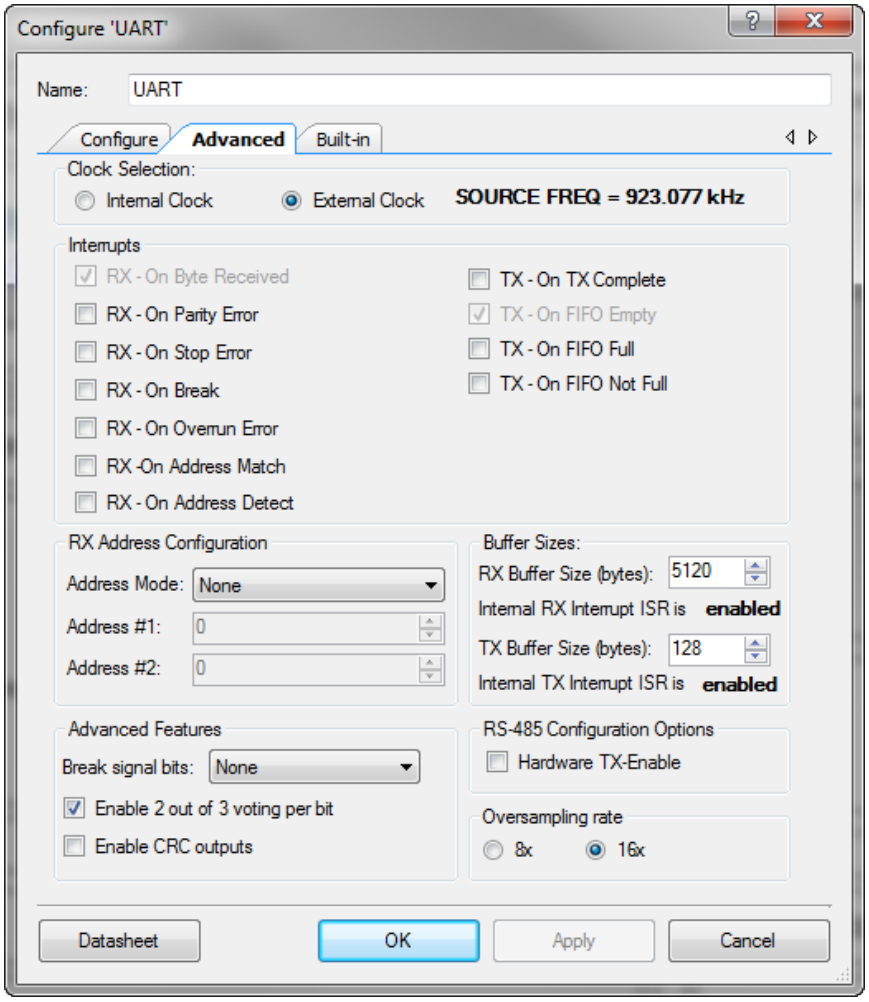


Figure 5-55. UART Component Advanced Tab



5.2.6 USB Subsystem

When connected to the host machine, the dongle enumerates as the following devices:

- HID keyboard device
- HID mouse device
- CDC COM port (custom CySmart interface)

The USB subsystem is built on the USBFS Component provided in PSoC Creator. The USBFS Component is configured with standard descriptors to comply with the USB device class, as shown in Figure 5-57. The USBFS Component is configured as shown in Figure 5-58, Figure 5-59, and Figure 5-60.

Figure 5-56. USB Component of PSoC Creator

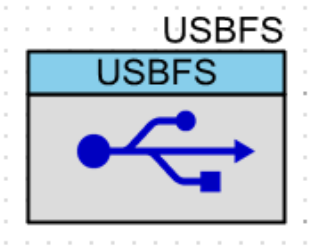


Figure 5-57. USBFS Component Device Configuration

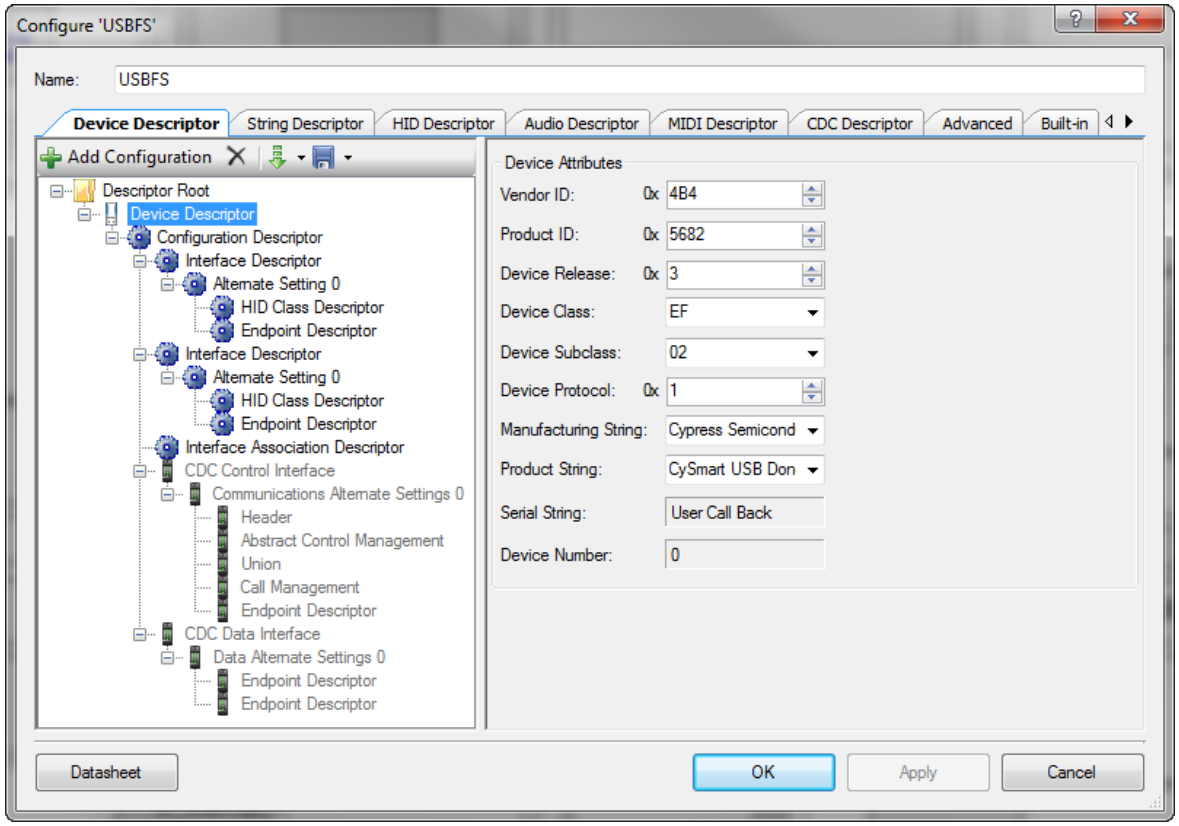


Figure 5-58. USBFS Component HID Descriptor Configuration

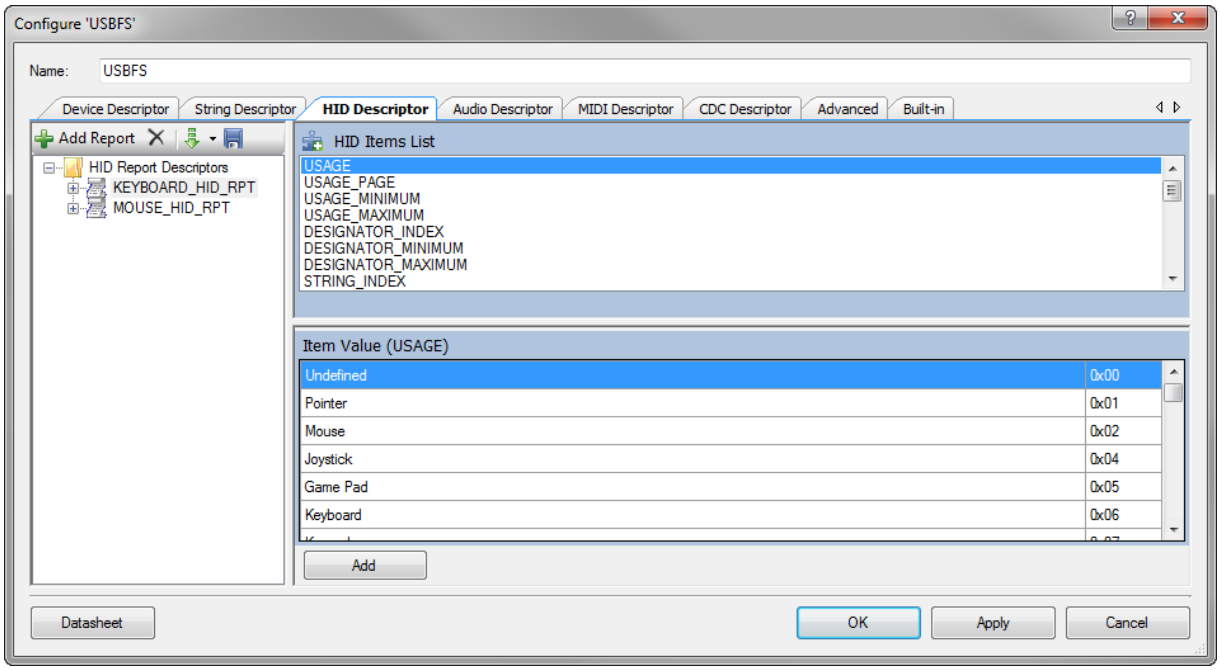


Figure 5-59. USBFS Component String Descriptor Configuration

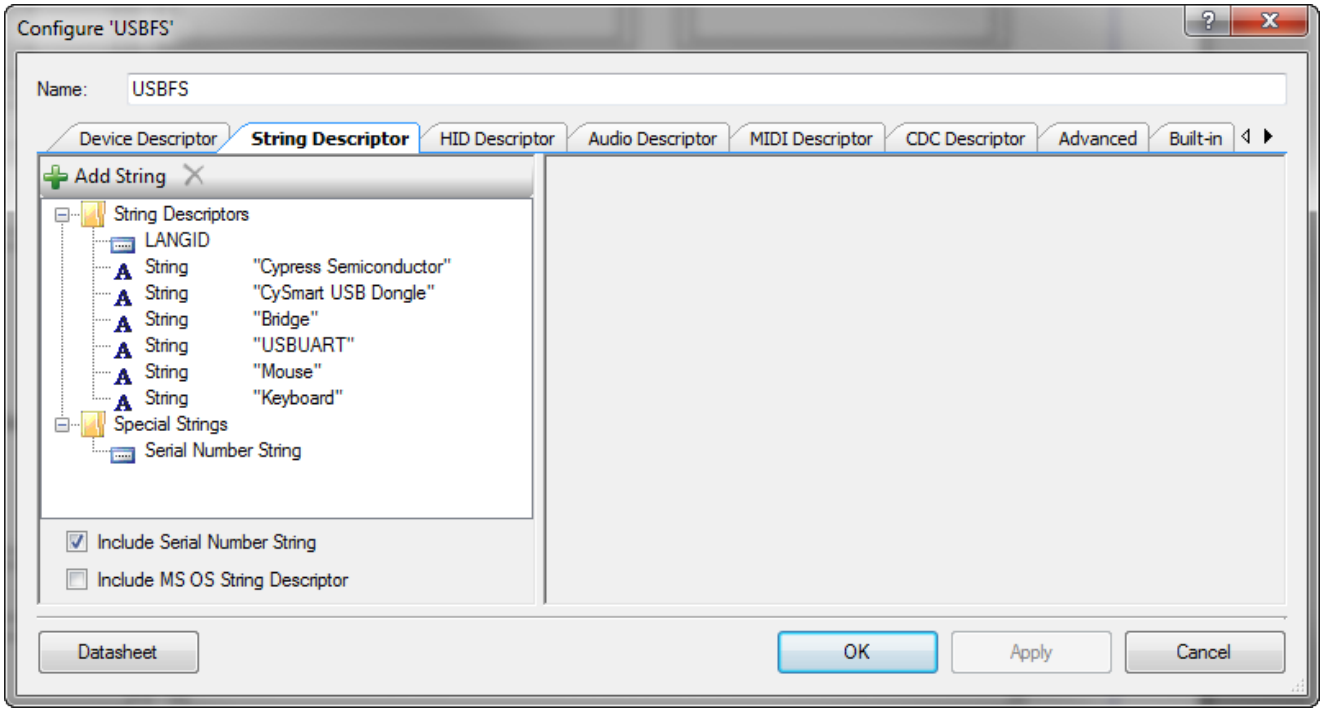
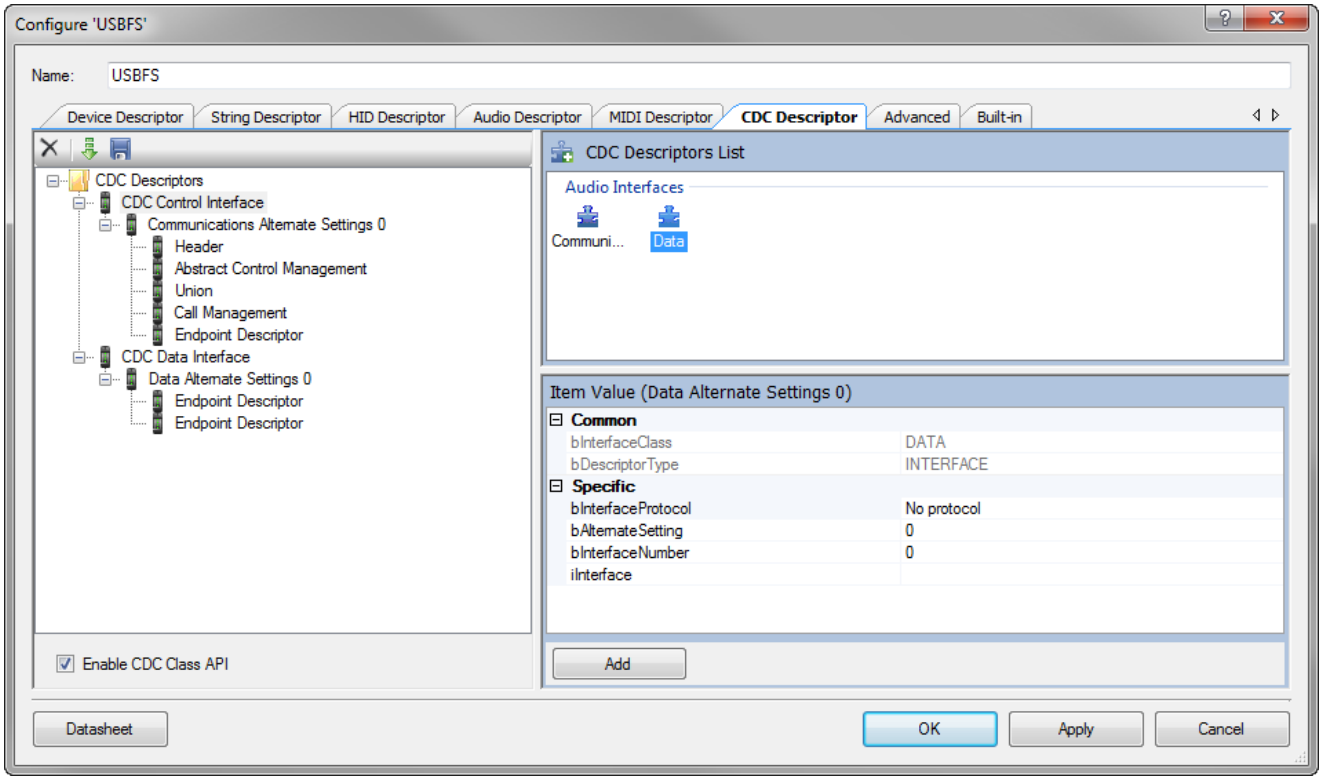


Figure 5-60. USBFS Component CDC Descriptor Configuration



6. Advanced Topics

6.1 Connecting PRoC BLE Touch Mouse with Bluetooth Smart Ready Device

The PRoC BLE touch mouse can connect to any Bluetooth Smart Ready device that implements (or is backward compatible with) the Bluetooth 4.0 or Bluetooth 4.1 specification and that has an operating system that supports the HOGP specification. The touch mouse is proven to work with popular Bluetooth Smart Ready devices. Table 6-1 lists the Bluetooth Smart Ready devices and operating systems that the CY5682 touch mouse has been tested against, along with the mouse features supported on each.

Table 6-1. Touch Mouse Features Supported on Devices and OS

Bluetooth Smart Ready Device	Operating System	Operating System Version	Mouse Features Supported
HP 240 G3	Windows	8.1	All features* are supported
Samsung Note 3	Android	4.4.2	All features* except the following buttons Windows Button Side Button 1 Side Button 2
MacBook Pro	OS X Yosemite	10.10.3	
Acer C720	Chrome OS	Chrome Version 38.0.2125.110 Chrome OS release version 6158.70.0	

* Horizontal scroll will work only with applications that support USB HID 'AC Pan' application control feature.

If your PC/operating system does not meet these criteria, follow the steps outlined in Connecting PRoC BLE Touch Mouse with CySmart USB Dongle. The steps to connect the touch mouse with a Bluetooth Smart Ready device depend on the operating system running on the device. See the User Guide of the Bluetooth Smart Ready device as required.

6.1.1 Connect to a Windows 8.1 PC

For a PC running Windows 8.1, follow these steps:

1. Navigate to **PC Settings** > **PC and devices** > **Bluetooth**. Turn on Bluetooth to start scanning for devices.
2. Plug the two AAA batteries provided with the kit into the battery holder on the touch mouse.

   Note: The touch mouse can also work with a single AAA battery; however, using two AAA batteries is recommended to prolong battery life.
3. Set the ON/OFF switch on the touch mouse to the ON position.
4. Press the connect button on the touch mouse. The orange LED shows breathing effect to indicate that the touch mouse is now in advertising mode.
5. The "CY5682 Mouse RDK" device should appear in the list of available Bluetooth devices with a mouse icon next to it and a message below it stating "Ready to pair." Click on "CY5682 Mouse RDK" and click the **Pair** button that appears on the PC.
6. Wait while the CY5682 device is installed. A progress bar shows the status of the installation. Once the installation is complete, the "CY5682 Touch Mouse RDK" appears in the list with a message below it stating "Connected."

   Note: Installation time may vary based on the operating system, system configuration, and speed of the internet connection.

7. Place the touch mouse on a flat surface and move it around. If the connection is successful, the mouse cursor on the PC will follow the movement of the touch mouse. If a connection is not established within 30 seconds, the orange LED turns off. In this case, repeat the sequence from step 4 to connect the touch mouse with the PC.

6.1.2 Clear CY5682 Mouse RDK from a Windows 8.1 PC

1. Navigate to **PC Settings** > **PC and devices > Bluetooth**. Turn ON Bluetooth.
2. Locate “CY5682 Mouse RDK” in the list of Bluetooth devices and click on it.
3. Click on the “Remove device” button.
4. Click Yes on the window that appears with the message “Are you sure that you want to remove this device?” to remove the CY5682 Mouse RDK from list of Bluetooth devices.
5. Verify that “CY5682 Mouse RDK” is no longer seen in list of Bluetooth devices.

6.1.3 Connect to a Chromebook

For a Chromebook, follow these steps:

1. Navigate to **Settings** > **Show advanced settings…** > **Bluetooth**. Click on **Enable Bluetooth**.
2. Click on the **Add a device** button. A window appears that lists the available Bluetooth devices.
3. Insert two AAA batteries that are provided with the kit into the battery holder on the touch mouse.

   Note: The touch mouse can also work with a single AAA battery; however, using two AAA batteries is recommended to prolong battery life.
4. Set the ON/OFF switch on the touch mouse to the ON position.
5. Press the connect button on the touch mouse. The orange LED shows the breathing effect to indicate that the touch mouse is now in advertising mode.
6. The “CY5682 Mouse RDK” device should appear in the list of available Bluetooth devices. Click on “CY5682 Mouse RDK” and then click the **connect** button.
7. Wait while the CY5682 device is being installed. When the installation is complete, the “**CY5682 Touch Mouse RDK**” appears in the list with BOLD font.

   Note: Installation time may vary based on the operating system, system configuration, and the speed of the internet connection.
8. Place the touch mouse on a flat surface and move it around. If the connection is successful, the mouse cursor on the Chromebook will follow the movement of the touch mouse. If a connection is not established within 30 seconds, the orange LED turns OFF. In this case, repeat the sequence from step 5 to connect the touch mouse with the Chromebook.

6.1.4 Clear CY5682 Mouse RDK from Chromebook

1. Navigate to **Settings** > **Show advanced settings…** > **Bluetooth**. Click on **Enable Bluetooth.**
2. Locate “CY5682 Mouse RDK” in the list of Bluetooth devices and click on it.
3. Click on the “**x**” button beside the CY5682 Mouse RDK in the list of Bluetooth devices and wait for two seconds. Verify that “CY5682 Mouse RDK” font changes from bold to normal.
4. Click again on the “**x**” button to remove the device from list of Bluetooth devices.
5. Verify that “CY5682 Mouse RDK” is no longer seen in list of Bluetooth devices.

6.1.5 Connect to a MacBook Pro

For a MAC Book Pro, follow these steps:

1. Navigate to **System Preferences…** > **Bluetooth**. Click on the **Bluetooth** icon.
2. Insert two AAA batteries that are provided with the kit into the battery holder on the touch mouse.

   Note: The touch mouse can also work with a single AAA battery; however, using two AAA batteries is recommended to prolong battery life.
3. Set the ON/OFF switch on the touch mouse to the ON position.

4. Press the connect button on the touch mouse. The orange LED shows the breathing effect to indicate that the touch mouse is now in advertising mode.
5. The “CY5682 Mouse RDK” device should appear in the list of available Bluetooth devices. Click on “CY5682 Mouse RDK” and then click on the **Pair** button.
6. The “Connected” message appears below the “CY5682 Mouse RDK.”
7. Wait while the CY5682 device is being installed. When the installation is complete, the connected message disappears below “CY5682 Touch Mouse RDK.”

   Note: Installation time may vary based on the operating system, system configuration, and the speed of the internet connection.
8. Place the touch mouse on a flat surface and move it around. If the connection is successful, the mouse cursor on the MacBook Pro will follow the movement of the touch mouse. If a connection is not established within 30 seconds, the orange LED turns off. In this case, repeat the sequence from step 4 to connect the touch mouse with the MacBook Pro.

6.1.6 Clear CY5682 Mouse RDK from MacBook Pro

1. Navigate to **Terminal** and get in to root directory.
2. Change directory to Library/Preferences using command “cd Library/Preferences” and press Enter.
3. Enter the command `ls com.apple.Bluetooth*` and then press Enter.
4. Verify that “com.apple.Bluetooth.plist” is in the list
5. Enter the command `sudo rm com.apple.Bluetooth.plist` and then press Enter.
6. The request for password appears. Enter the system password and press Enter.
7. Enter the command `ls com.apple.Bluetooth*` and press Enter.
8. Verify that the message “No such file or directory” appears.
9. Now the Bluetooth list is cleared. Restart MacBook Pro and verify that there are no Bluetooth devices in the list.

6.1.7 Connect to an Android Device

For an Android device, follow these steps:

1. Navigate to **Settings** > **Bluetooth**. Turn on Bluetooth and click **SERACH FOR DEVICES** to start scanning for devices.
2. Insert two AAA batteries that are provided with the kit into the battery holder on the touch mouse.

   Note: The touch mouse can also work with a single AAA battery; however, using two AAA batteries is recommended to prolong battery life.
3. Set the ON/OFF switch on the touch mouse to the ON position.
4. Press the connect button on the touch mouse. The orange LED shows breathing effect to indicate that the touch mouse is now in advertising mode.
5. The “CY5682 Mouse RDK” device should appear in the list of available Bluetooth devices Click on “CY5682 Mouse RDK” and observe the message “Pairing…” under “CY5682 Mouse RDK”.
6. Wait while the CY5682 device is installed. The message “Connecting...” appears under CY5682 Mouse RDK” . Once the installation is complete, the “CY5682 Touch Mouse RDK” appears in the list with a message below it stating “Connected.”

   Note: Installation time may vary based on the operating system, system configuration, and the speed of the internet connection.
7. Place the touch mouse on a flat surface and move it around. If the connection is successful, the mouse cursor on the PC will follow the movement of the touch mouse. If a connection is not established within 30 seconds, the orange LED turns OFF. In this case, repeat the sequence from step 4 to connect the touch mouse with the Android device.

6.1.8 Clear CY5682 Mouse RDK from an Android Device

1. Navigate to **Settings** > **Bluetooth**.
2. Locate “CY5682 Mouse RDK” in the list of Bluetooth devices and tap on the settings icon beside it.
3. Locate **Unpair** button and tap on it. Verify that “CY5682 Mouse RDK” is removed from list of Bluetooth devices.

6.2 Current and Voltage Measurement

6.2.1 Current Measurement

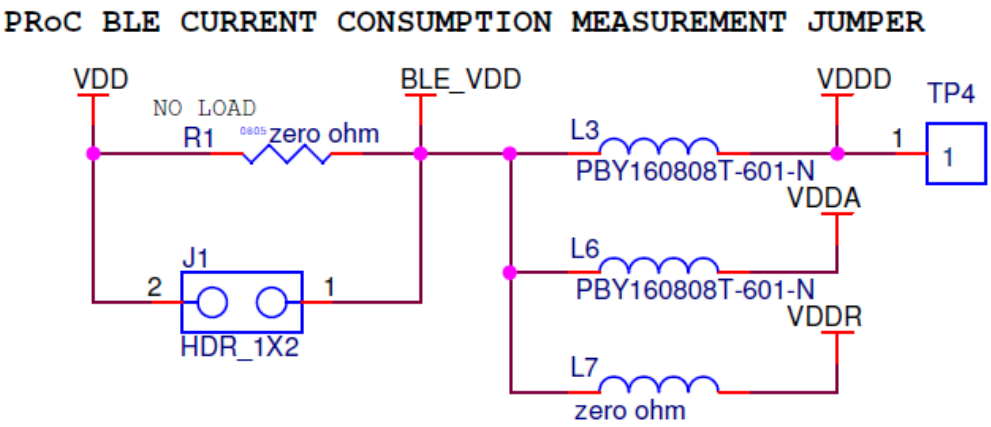
6.2.1.1 Measure System Current Consumption

The system current can be measured by removing the batteries and powering up the touch mouse using an external power supply with an ammeter in series.

6.2.1.2 Measure PRoC BLE Current Consumption

1. Remove the shunt on J1.
2. Remove the jumper installed on the 2-pin header J1 using the tweezers.
3. Connect an ammeter to J1 shown in Figure 6-1.

Figure 6-1. Current Measurement Circuitry



Following macros in the *platform.h* file in the firmware allows both BLE subsystem and MCU to enter lowest-power states in order to achieve the best current numbers.

```
#define ENABLE_BLE_LOW_POWER_MODE
```

This macro is enabled by default. It allows BLE subsystem to enter a low-power mode at every possible opportunity.

```
#define MCU_SLEEP_DISABLED
```

This macro is disabled by default. It allows the MCU to enter the Sleep state at every possible opportunity.

```
#define MCU_DEEP_SLEEP_DISABLED
```

This macro is disabled by default. It allows the MCU to enter the Deep Sleep state at every possible opportunity.

6.2.1.3 Battery Life

Table 6-2 lists the battery life for the touch mouse. This table also lists the current consumption of the system in each state.

Table 6-2. Battery Life Calculation

State	Current Measured at Battery Terminals (mA) (System Current)	Usage Time per Week (hours)	Battery Capacity Used (mAh)
Active with data transfer	14.1	5	70.5
Active without data transfer	1.4	1.25	1.75
Idle	0.189	27.6	5.21
Low Power	0.132	134.15	17.70
		Total battery capacity used per week (mAh)	95.17
Battery Life with a 2500-mAh battery capacity (Weeks)			26.27

The following assumptions are made for calculating the battery life summarized in the table above:

- Usage model
- Current measurement was made at constant 1.5 V
- Battery capacity is assumed to be 2500 mAh

6.2.2 Voltage Measurement

Test points for each power supply and ground are provided to measure the voltage. For details, refer to the Test Points section.

6.3 Accessing Debug Interfaces of the Touch Mouse

Figure 6-2 shows the fully assembled touch mouse.

Figure 6-2. Fully Assembled Touch Mouse

1. Remove the top cover by lifting it from the bottom portion of the casing, as shown in Figure 6-3. A groove is provided for this purpose. With the top cover removed, you can access the 10-pin programming/debug header, battery holder, and two-pin header, which allows PRoC BLE current measurement.

Figure 6-3. Removing the Top Cover of the Touch Mouse

Note: The three screws shown in Figure 6-4 are used to fix the PCBA to the mouse casing. Unscrewing these to disassemble the mouse is not recommended because the lens for the optical sensor, the plastic ON/OFF switch, and the plastic connect button may become dislodged while disassembling.

Figure 6-4. Touch Mouse with Screws

A. Appendix: Schematics and FAQ

A.1 Schematics

A.1.1 Touch Mouse

Figure A-1. Main Board Schematic – 1 of 4

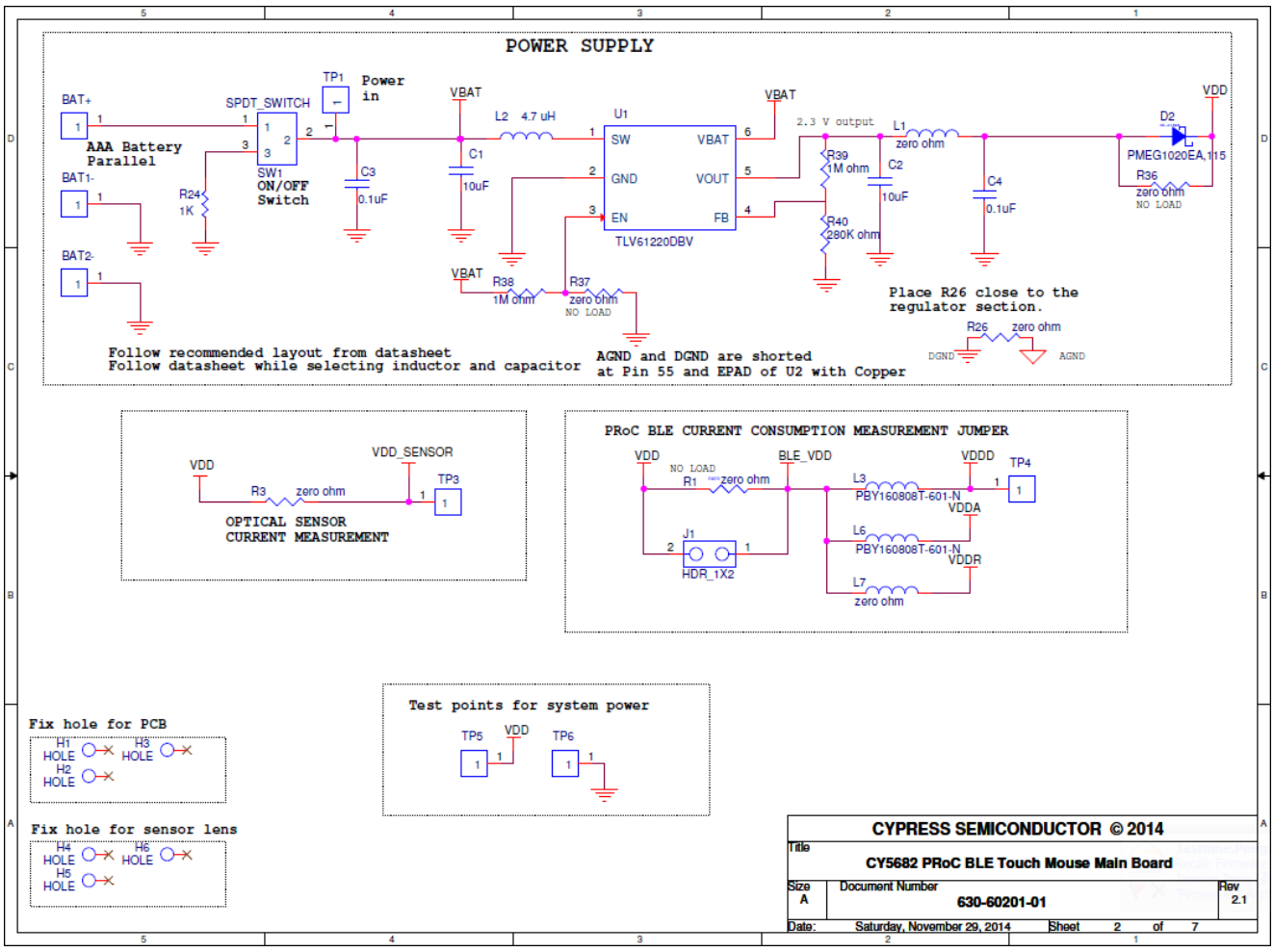


Figure A-2. Main Board Schematic – 2 of 4

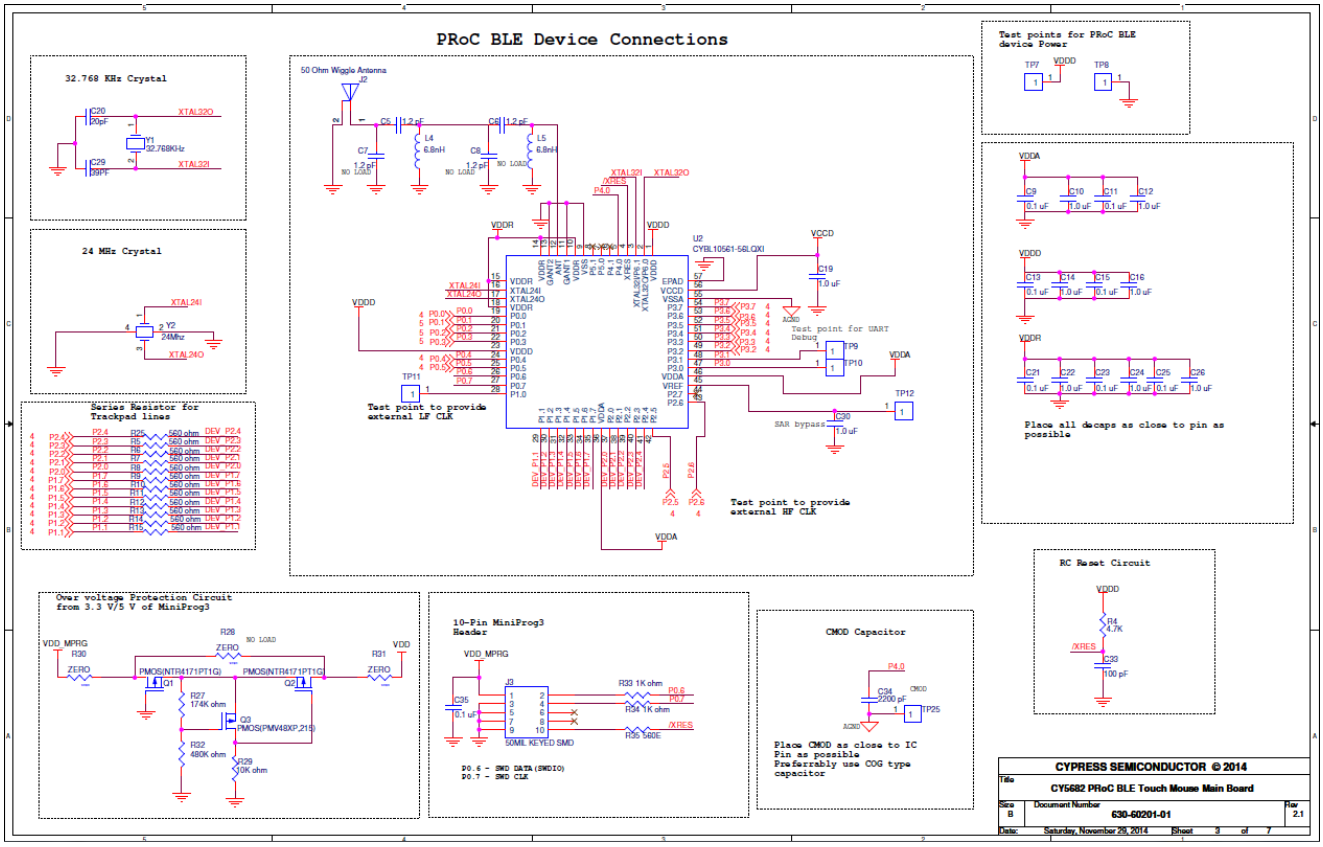


Figure A-3. Main Board Schematic – 3 of 4

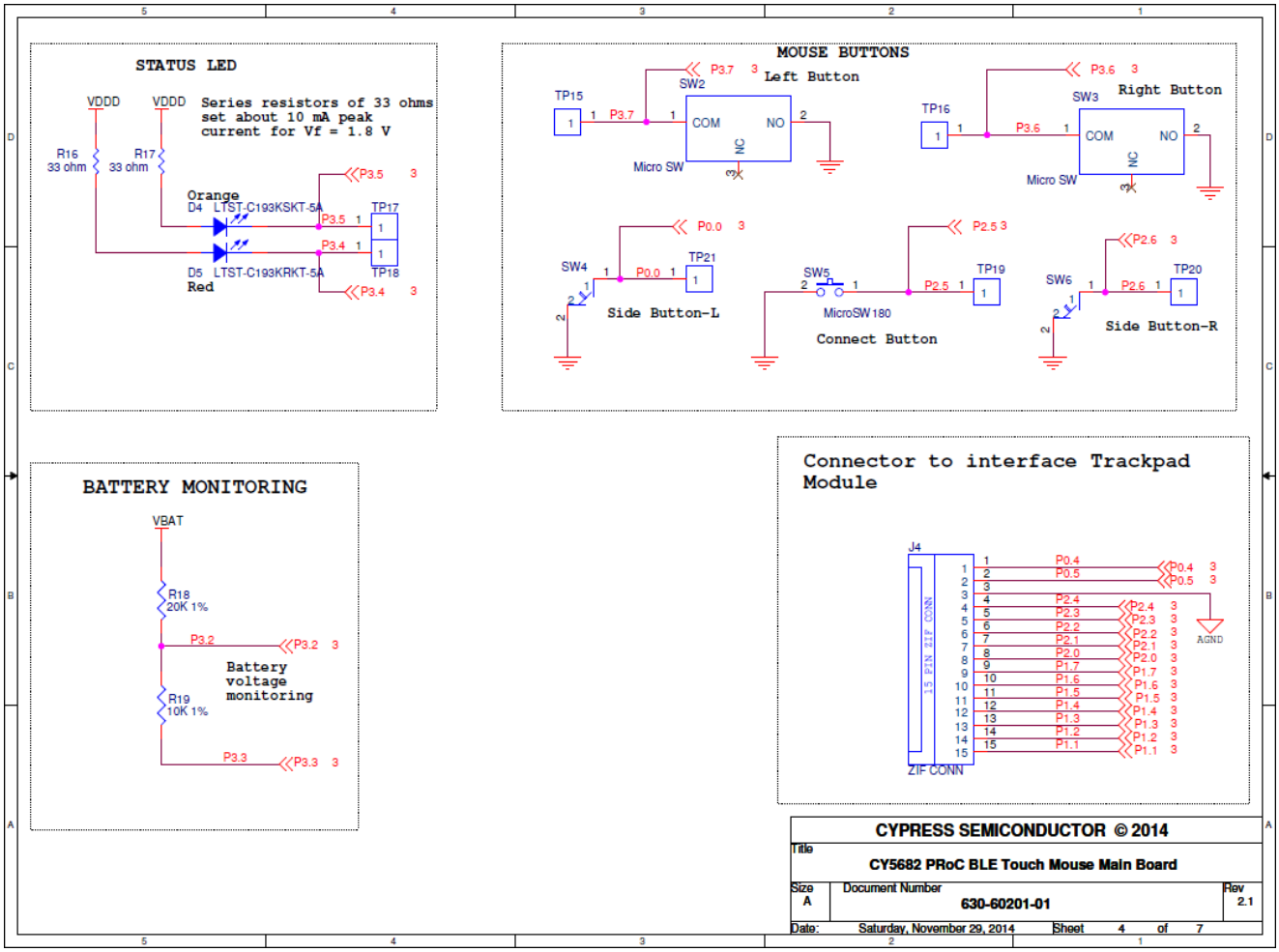


Figure A-4. Main Board Schematic – 4 of 4

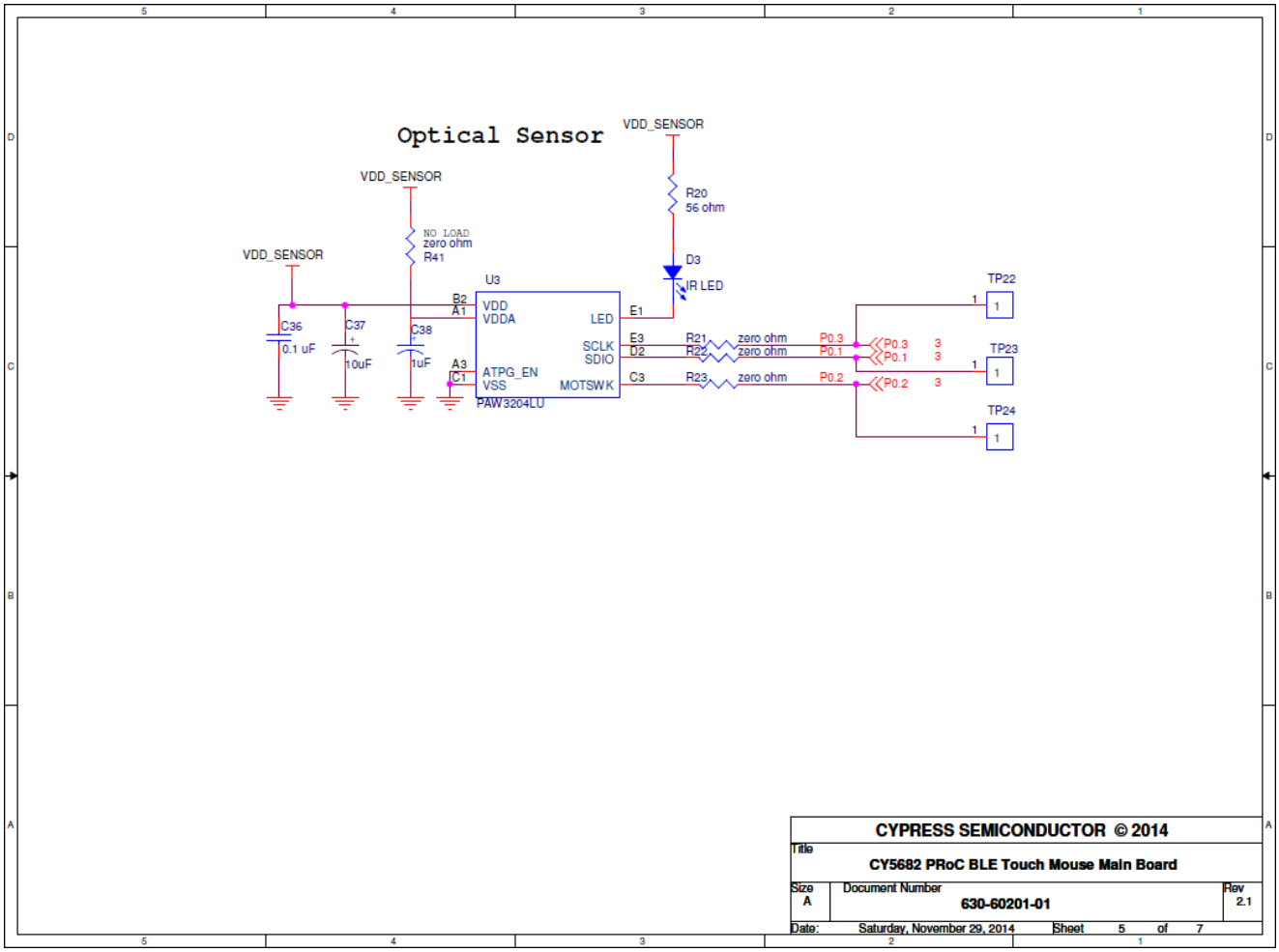
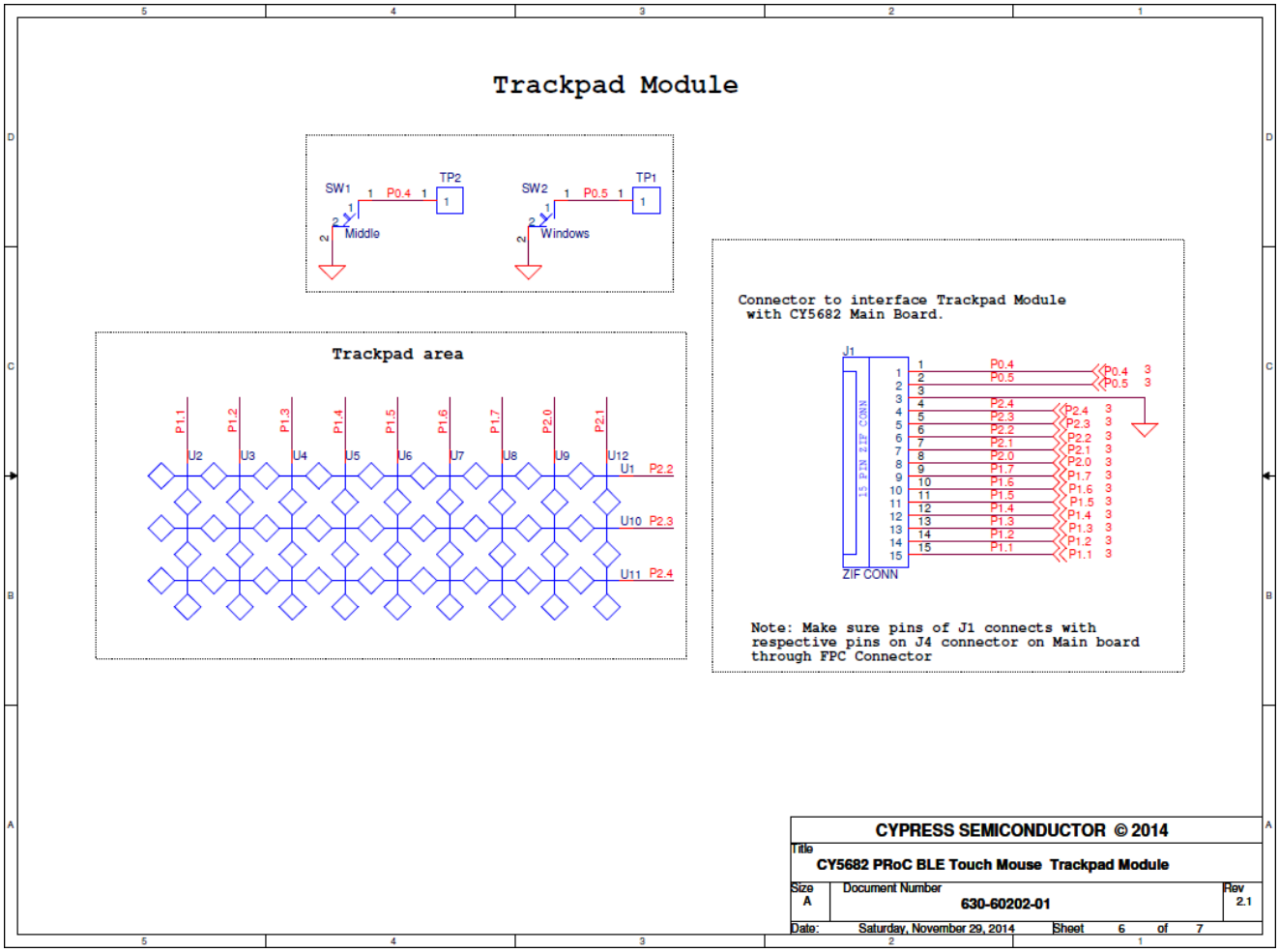


Figure A-5. Trackpad Module Schematic



A.1.2 Dongle

Figure A-6. Dongle Board Schematic – 1 of 2

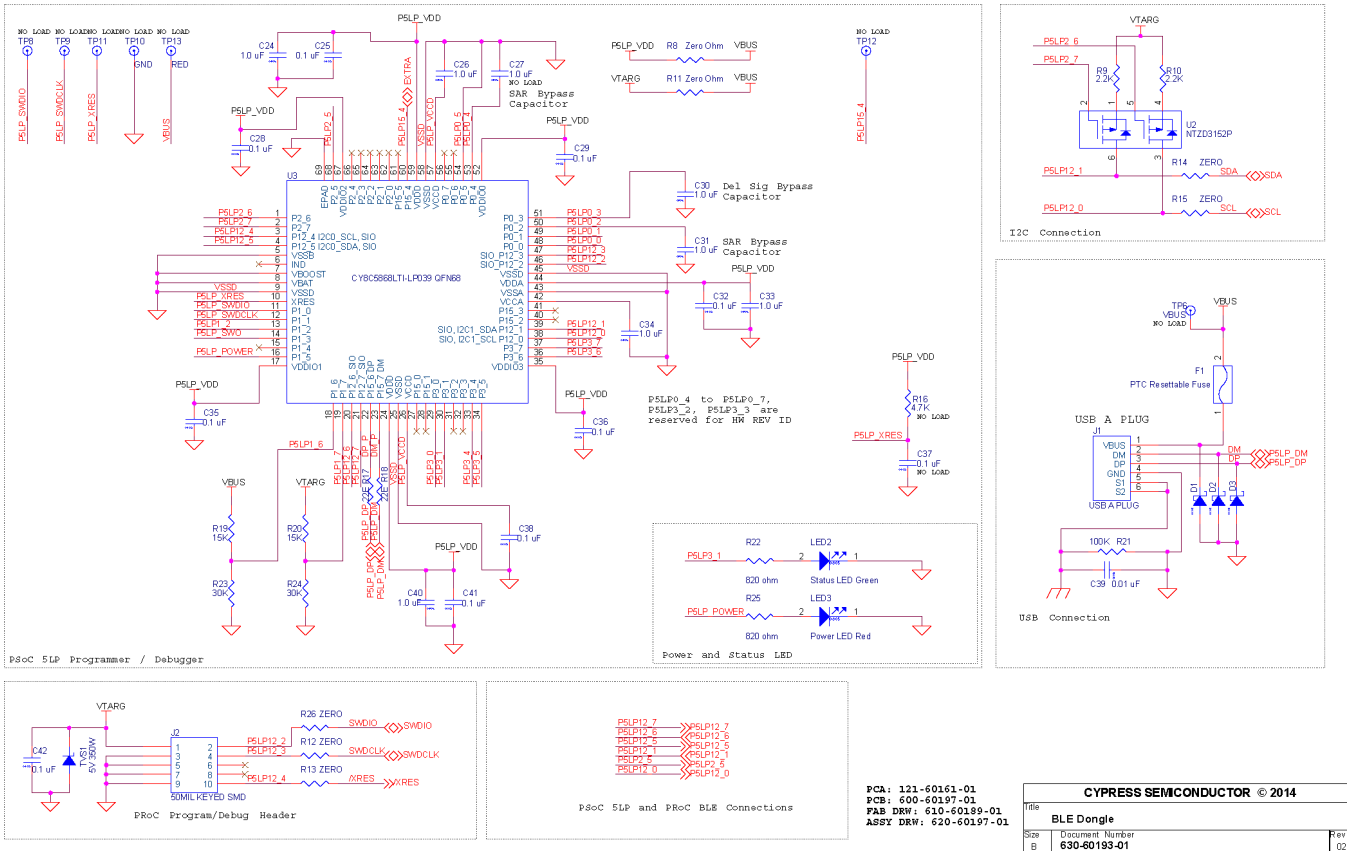
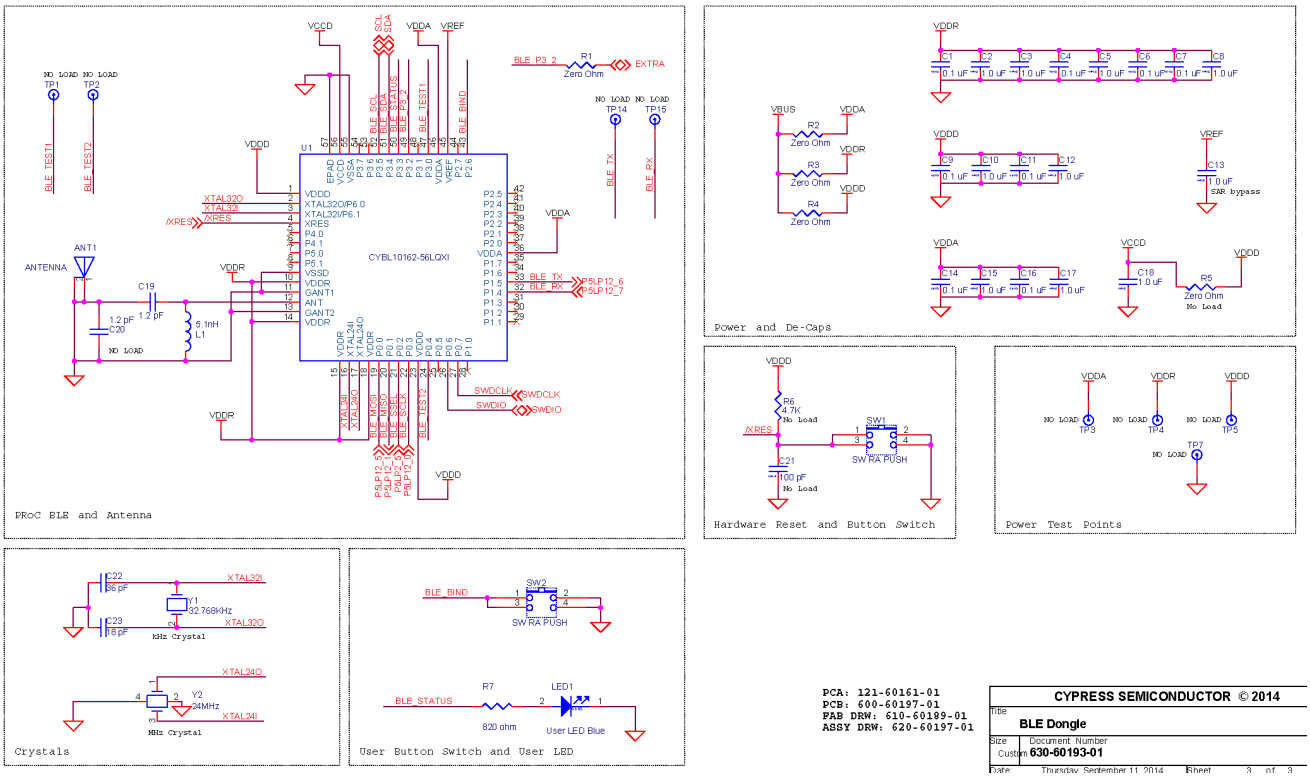


Figure A-7. Dongle Board Schematic – 2 of 2



A.2 Troubleshooting and FAQ

- **My touch mouse does not work with the CySmart USB dongle out of the box. How can I resolve this issue?**
  This may happen very rarely due to human error while packaging the kit. You can easily resolve this issue by following the steps to connect your CySmart USB dongle to your touch mouse in the Connecting PRoC BLE Touch Mouse with CySmart USB Dongle section.
- **Why does the Programming and Debugging PRoC BLE on Touch Mouse and CySmart USB Dongle section recommend programming at 3.3 V?**
  PRoC BLE has an operating voltage range of 1.8 V to 5.5 V. However, the optical sensor on the touch mouse can tolerate only voltages up to 3.6 V. Hence, Cypress recommends programming at 3.3 V for the touch mouse. Note that the touch mouse incorporates an overvoltage protection circuit as described in the Power Supply section to prevent damage to the circuit due to programming at high voltages.
- **How do I open the touch mouse to insert batteries?**

  Step 1 in the Accessing Debug Interfaces of the Touch Mouse section provides instructions to remove the top cover of the touch mouse. Once the top cover is removed, you will be able to see the two AAA battery holder.
- **Do's and Don'ts**
  - Avoid applying high voltages to the test points provided on the touch mouse. This may cause damage to the circuit due to overvoltage or reverse voltage.
  - Avoid spilling any kind of liquid on the touch mouse or the CySmart USB dongle. They are not waterproof and may be damaged.
- **Can I use the CY5682 touch mouse with a Windows 7 host?**

  A Windows 7 host does not support BLE (Bluetooth 4.0). Hence, using the touch mouse directly with a Windows 7 host is not recommended. You should instead use the supplied CySmart dongle when using the CY5682 with a Windows 7 computer.
- **My mouse cursor is not moving smoothly or I see the cursor jumping while moving the mouse.**

  Bluetooth Low Energy operates in the same frequency band (2.4 GHz) as a number of other technologies like Wi-Fi and Bluetooth Classic. These technologies cause interference, which may result in mouse data packets being dropped. If you are using the mouse in close proximity to such an interferer (such as a Wi-Fi router), you should move your mouse/laptop at least 20 cm away from the interferer. **Note that the trackpad on the mouse does not work after the mouse is inactive for greater than a minute.**

  The mouse is designed to maximize battery life by minimizing power consumption. After a minute of inactivity, the mouse goes into sleep state and can only wake-up when moved (such as due to the optical sensor) or when a button is pressed. The trackpad is not scanned during the sleep state. Therefore, move the mouse or press a button on the mouse before using the trackpad.
- **What does the Keyboard report and Mouse report structure look like for CY5682 Touch Mouse?**

  Keyboard input:

  The touch mouse supports the following keyboard keys : [**Windows**], [**Windows**] [**Ctrl**] [**Backspace**], and [**Windows**] [**C**].

  The keyboard structure for the touch mouse is shown below:.

```
typedef struct _Keyboard_Report_
{
    uint8 mkey;                          /* Modifier key */
    uint8 keylength_used;                /* Number of key used in the keycode*/
    uint8 keycode[NUMBER_OF_KEYCODES];   /* key codes */
}Keyboard_Report;
```

  Mouse input :

  The touch mouse supports left-click, right-click, middle-click, delta X, delta Y, H-wheel(horizontal scroll), and Z-wheel(vertical scroll).

  The Mouse Report for the touch mouse is shown below:

```
typedef struct _Mouse_Report_
{
    uint8 click;                         /* Click button status.
```

```
                                        * 0th bit is used for Left click,
                                        * 1st bit is used for Right Click and
                                        * 2nd bit is used for middle click*/
    int8 x;                            /* Delta x movement */
    int8 y;                            /* Delta y movement */
    int8 zwheel;                       /* Delta Vertical movement */
    int8 hwheel;                       /* Delta Horizontal movement */
}Mouse_Report;
```

- **What are the HID keycodes for each of the function touch mouse?**

  Table 6-3 lists the HID key codes sent by the touch mouse for each feature.

Table 6-3. Touch Mouse HID Key Codes

Touch Mouse Function	HID Key Code
Left button	Left-click (0th bit in first byte of mouse report)
Right button	Right-click (first bit in first byte of mouse report)
Middle button	Middle-click (second bit in first byte of mouse report)
X movement	Delta X (second byte of the mouse report)
Y movement	Delta Y (third byte of the mouse report)
Vertical scroll	Z-wheel (fourth byte of the mouse report)
Horizontal scroll	H-wheel (fifth byte of the mouse report)
Windows button	Modifier key = 0x08 -- Windows key(0x08) keylength = 0 keyboard key = {0, 0, 0, 0, 0, 0}
Side button 1	Modifier key = 0x81 -- Windows key (0x08) + Ctrl key (0x01) keylength = 1 Keyboard key = {0x2A, 0, 0, 0, 0, 0} -- Backspace (0x2A)
Side button 2	Modifier key = 0x80 keylength = 0 keyboard key = {0x06, 0, 0, 0, 0, 0} -- keyboard key ‘c’ (0x06)

A.3 Troubleshooting and FAQ for Interoperability Issues with Bluetooth Smart Ready Devices

- **The mouse cannot connect back to Chromebook when Bluetooth settings panel is open.**

  Chrome OS does not allow Bluetooth devices to connect using directed advertisement if the Bluetooth settings panel is open. To connect the mouse, close the Bluetooth settings panel by clicking "< Bluetooth" at the bottom of the panel or by clicking on the desktop. Now move the mouse to connect to the Chromebook.

- **Mouse does not connect back to Chromebook after going out-of-range and then coming back in-range.**

  In order to reconnect to the Chromebook, the touch mouse does directed advertisement, which needs to be recognized and responded to by the Chromebook. It is observed that ChromeOS (6158.70.0) on an Acer C720 Chromebook does not respond to directed advertisement in this scenario. Do the following to connect back to the Chromebook:

  1. On Chromebook, click on the lower-left status area. A panel appears.
  2. Click on "**Bluetooth enabled >**" to go to Bluetooth settings.
  3. Click on "**CY5682 Mouse RDK**". The device name should now change to "**CY5682 Mouse RDK: Connecting…**"
  4. Move the mouse.

  Mouse cursor should now start moving and a "✓" mark will appear next to the device name ("CY5682 Mouse RDK") in the Bluetooth settings panel.

- **Moving the mouse or pressing a button on the mouse does not wake up the Chromebook**

  The Acer C720 Chromebook supports two sleep modes - display sleep after 10 minutes of inactivity and computer sleep after 30 minutes of inactivity. The mouse can wake the system up from display sleep; however, it cannot wake the Chromebook up from computer sleep. Press any key on your Chromebook's keyboard or slide your finger on the Chromebook's trackpad to wake the system up from computer sleep and then use the mouse.

- **Lag and drift in mouse cursor movement on Chromebook**

  It is observed that the cursor movement in ChromeOS (6158.70.0) on an Acer C720 shows lag or drift. In addition, the Chromebook may temporarily stop receiving data from the touch mouse for a few connection intervals when the touch mouse is moved away from the Chromebook. This results in the touch mouse sending queued data in one connection interval, which can cause lag and drift in cursor movement on the Chromebook. To avoid or improve this behavior, ensure that the touch mouse is brought as close to Chromebook as possible.

- **Mouse does not connect back to MacBook Pro after going out of range and then coming back in range.**

  To connect back to a MacBook Pro, the touch mouse does directed advertisement, which needs to be recognized and responded to by MacBook Pro. MacOS(10.10.1) on MacBook Pro does not respond to directed advertisement in this scenario. If the mouse is unable to connect back automatically, turn MacBook Pro's Bluetooth OFF and then turn it back ON, and then move the mouse. The touch mouse is reconnected to MacBook Pro automatically and you will be able to see cursor movement on MacBook Pro.

- **Mouse stops working when used with MacBook Pro. The orange LED on the mouse blinks on user activity.**

  If the connection with MacBook Pro breaks, the touch mouse does directed advertisement to connect back to MacBook Pro. MacBook Pro should recognize and respond to the touch mouse. On rare occasions, MacOS (10.10.1) on MacBook Pro does not respond to directed advertisement in this scenario. To resolve this issue, do the following:

  - Turn MacBook Pro's Bluetooth OFF and then turn it back ON.
  - If the mouse is still unable to connect back, restart the MacBook Pro. Turn ON Bluetooth and then move the mouse. The touch mouse connects back automatically; you will be able to see cursor movement on MacBook Pro.

- **Mouse stops working with a PC running Windows 8.1. The orange LED on the Mouse blinks on user activity**

  On rare occasions, the BLE link between the mouse and the PC running Windows 8.1 may get disconnected. To connect back to the PC on Windows 8.1, the touch mouse does directed advertisement, which Windows 8.1 must recognize and respond to. Windows 8.1 does not respond to directed advertisement in this scenario.

  To resolve this issue, download and install the latest BLE driver from Microsoft website or from the PC manufacturer's website such as http://support.lenovo.com/us/en/downloads/ds037314. for a Lenovo laptop.

  If this issue still occurs, you can restore the BLE link and resume usage. Do the following:

  1. Navigate to PC Settings > PC and devices > Bluetooth.
  2. Move the mouse; it will connect back automatically. You will be able to see cursor movement on the PC.

- **Mouse may not work with Samsung Note 3 after going out of range and then coming back in-range**

  To resume the operation of the Touch Mouse, encryption needs to be turned ON in the Samsung Note 3 after reconnection. It was found that sometimes Samsung Note 3 running Android 4.4.2 does not complete the encryption process. If the mouse is unable to connect back automatically, turn Samsung Note 3's Bluetooth OFF and then turn it back ON, and then move the mouse. The touch mouse is reconnected to the Samsung Note 3 automatically and you will be able to see cursor movement on Samsung Note 3.

Revision History

Document Revision History

Document Title: CY5682 PRoC™ BLE Touch Mouse Reference Design Kit Guide Document Number: 001-94177			
Revision	**Issue Date**	**Origin of Change**	**Description of Change**
**	02/18/2015	UTSV / DEJO/SKUV	First version of this kit guide
*A	03/12/2015	UTSV / DEJO/SKUV	Fixed bookmarks Edits throughout the document
*B	04/22/2015	UTSV / DEJO/SKUV	Updated names of flash APIs in Table 5-1
*C	07/16/2015	SELV	Updated Table 6-1 Added a troubleshooting note to the Troubleshooting and FAQ section
*D	09/08/2015	SELV	Updated 'GAP Settings' (Figure 5-11) with 'Slave latency' value of 0 during connection establishment