Please note that Cypress is an Infineon Technologies Company.

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

Continuity of document content

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

Continuity of ordering part numbers

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

THIS SPEC IS OBSOLETE

Spec No: 002-00297

Spec Title: S6SAE101A00SA1002 - Solar-Powered Internet of Things (IoT) Device Kit User Guide

Replaced by: None

OBSOLETE

S6SAE101A00SA1002

Solar-Powered Internet of Things (IoT) Device Kit User Guide

Doc. No. 002-00297 Rev. *C

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
www.cypress.com

Contents

About This Document **4**

Handling and Use 4
Caution 4
Typographic Conventions 5
IoT Resources and Technical Support 5

1 Introduction **6**

1.1 Features 6
1.2 Applications 7

2 Kit Introduction **8**

2.1 Kit Contents 8

3 Software Installation **9**

3.1 Installing Software 9
3.2 Uninstalling Software 11
3.3 Using PSoC® Creator™ 11

4 Getting Started **12**

4.1 Solar-Powered BLE Beacon Operation 12
4.1.1 Installing USB Driver of BLE-USB Bridge 12
4.1.2 USB Driver Installation Failure 14
4.1.3 Installing USB Driver for Motherboard 16
4.1.4 Establishing BLE Connection 19
4.1.5 Vibration Energies Connection (Optional) 21
4.2 Solar-Powered Wireless Sensor Node (WSN) with BLE Beacon 22
4.2.1 Configuring Motherboard as a WSN 22
4.2.2 Validating WSN 25
4.3 Serial Command List 26

5 Program and Debug **31**

5.1 UART Bootloader (Program Only) 31
5.2 PSoC Creator with MiniProg3 (Program and Debug) 34
5.2.1 Program 34
5.2.2 Debug 38
5.2.2.1 Converting Application Type to Normal 39
5.2.2.2 Using Attach to Running Target Option 40

6 Example Project **43**

6.1 Flow Diagram 43
6.2 Function List 44
6.3 BLE Beacon Process 45
6.4 WSN with BLE Beacon Process 50
6.5 BLE Beacon Format 53
6.6 Sensor Transmitter Specification of WSN 55

7 Energy Harvesting PMIC (S6AE101A) **56**

7.1 Recommended Operating Conditions 56
7.2 DC Characteristics 56

OBSOLETE

8 Hardware 57

8.1 Energy Harvesting Motherboard 57
8.1.1 Board Detail 57
8.1.2 Input/Output Pin Description 58
8.1.3 Debug Connector Description 59
8.1.4 Jumper Description 60
8.1.5 Switch Description 60
8.1.6 LED Description 60
8.1.7 Circuit 61
8.1.8 BOM List 63
8.2 BLE-USB Bridge 64
8.2.1 Board Detail 64
8.2.2 Test Pin Description 65
8.2.3 Switch Description 65
8.2.4 LED Description 66
8.2.5 Circuit 67
8.2.1 BOM List 71

9 Ordering Information 73

Appendix A. Other Sample Projects 74

A.1 LED ONOFF Project 74
A.1.1 main.c 74
A.1.2 Flow of LED ONOFF Project 75
A.1.3 TopDesign.cysch 75
A.1.4 Process Steps 76
A.2 Simple BLE Project 77
A.2.1 main.c 77
A.2.2 Flow of Simple BLE Project 79
A.2.3 TopDesign.cysch 80
A.2.4 Process Steps 80

Appendix B. Using Extra Components 82

B.1 10-Ω Resistor for Current Measurement 82
B.2 Additional 220 µF Capacitor 85

Document Revision History 87

Worldwide Sales and Design Support 88

Products 88
PSoC® Solutions 88
Cypress Developer Community 88
Technical Support 88

OBSOLETE

About This Document

This manual explains how to use the S6SAE101A00SA1002 evaluation board. Make sure you read this manual before using the product.

Handling and Use

Handling and use of this product and notes regarding its safe use are described in this manual.

Follow the instructions in the manuals to use this product.

Keep this manual handy so that you can refer to it anytime during use of this product.

Caution

The following precautions apply to the product described in this manual.

⚠ **WARNING**	Indicates a potentially hazardous situation which could result in death or serious injury and/or a fault in the user's system if the product is not used correctly.

Electric shock, Damage	Before performing any operation described in this manual, turn off all the power supplies to the system. Performing such an operation with the power on may cause an electric shock or device fault.
Electric shock, Damage	Once the product has been turned on, do not touch any metal part of it. Doing so may cause an electric shock or device fault.

⚠ **CAUTION**	Indicates the presence of a hazard that may cause a minor or moderate injury, damages to this product or devices connected to it, or may cause the loss of software resources and other properties such as data, if the device is not used appropriately.

Cuts, Damage	Before moving the product, be sure to turn off all the power supplies and unplug the cables. Watch your step when carrying the product. Do not use the product in an unstable location such as a place exposed to strong vibration or a sloping surface. Doing so may cause the product to fall, resulting in an injury or fault.
Cuts	The product contains sharp edges that are left unavoidably exposed, such as jumper plugs. Handle the product with due care not to get injured with such pointed parts.
Damage	Do not place anything on the product or expose the product to physical shocks. Do not carry the product after the power has been turned on. Doing so may cause a malfunction due to overloading or shock.
Damage	Since the product contains many electronic components, keep it away from direct sunlight, high temperature, and high humidity to prevent condensation. Do not use or store the product where it is exposed to much dust or a strong magnetic or electric field for an extended period of time. Inappropriate operating or storage environments may cause a fault.
Damage	Use the product within the ranges given in the specifications. Operation over the specified ranges may cause a fault.
Damage	To prevent electrostatic breakdown, do not let your finger or other object come into contact with the metal parts of any of the connectors. Before handling the product, touch a metal object (such as a door knob) to discharge any static electricity from your body.
Damage	When turning the power ON or OFF, follow the relevant procedure as described in this document. Before turning the power on, in particular, be sure to finish making all the required connections. Furthermore, be sure to configure and use the product by following the instructions given in this document. Using the product incorrectly or inappropriately may cause a fault.
Damage	Because the product has no casing, it is recommended that it be stored in the original packaging. Transporting the product may cause a damage or fault. Therefore, keep the packaging materials and use them when re-shipping the product.

OBSOLETE

Typographic Conventions

Convention	Usage

`Courier New`	Courier New, 9pt.
	Displays file locations, user entered text, and source code examples:
	`C:\ ...cd\icc\`
	`CyDelay`

`Consolas`	Consolas 9pt. API and function names (when mentioned within body text) If there is not response to the `DOWNLOAD_MINIDRIVER` command, the device may be in autobaud mode.
Italics	Arial,9pt in body text; 8pt in table text. Displays file names and reference documentation: Read about the *sourcefile.hex* file in the *PSoC Designer User Guide*.
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**.

IoT Resources and Technical Support

Cypress provides a wealth of data at www.cypress.com/internet-things-iot to help you to select the right IoT device for your design, and quickly and effectively integrate the device into your design. Cypress provides customer access to a wide range of information, including technical documentation, schematic diagrams, product bill of materials, PCB layout information, and software updates. Customers can acquire technical documentation and software from the Cypress Support Community website (http://community.cypress.com/).

OBSOLETE

1 Introduction

The Solar-Powered IoT Device Kit provides an easy-to-use platform for the development of a solar-powered IoT device with BLE wireless connectivity. It includes the S6AE101A Energy Harvesting Power Management IC (PMIC) device, which is ideal for solar- or light-powered Energy Harvesting Systems (EHS) since it only consumes 250 nA. The S6AE101A also supports a hybrid EHS that uses a solar cell Energy Harvesting Device (EHD) along with a coin cell battery, and an optional vibration EHD with external diode bridge. The output voltage from the S6AE101A is configurable from 1.1 V to 5.2 V, supporting a broad range of device components for an IoT device. Also included in the kit is Cypress' EZ-BLE™ PRoC™ Module (CYBLE-022001-00), a fully integrated Bluetooth Low Energy (BLE) module solution that offers high flexibility for a wide variety of IoT device uses. A USB port is provided by Cypress' USB-UART LP Bridge Controller device (CY7C65213).

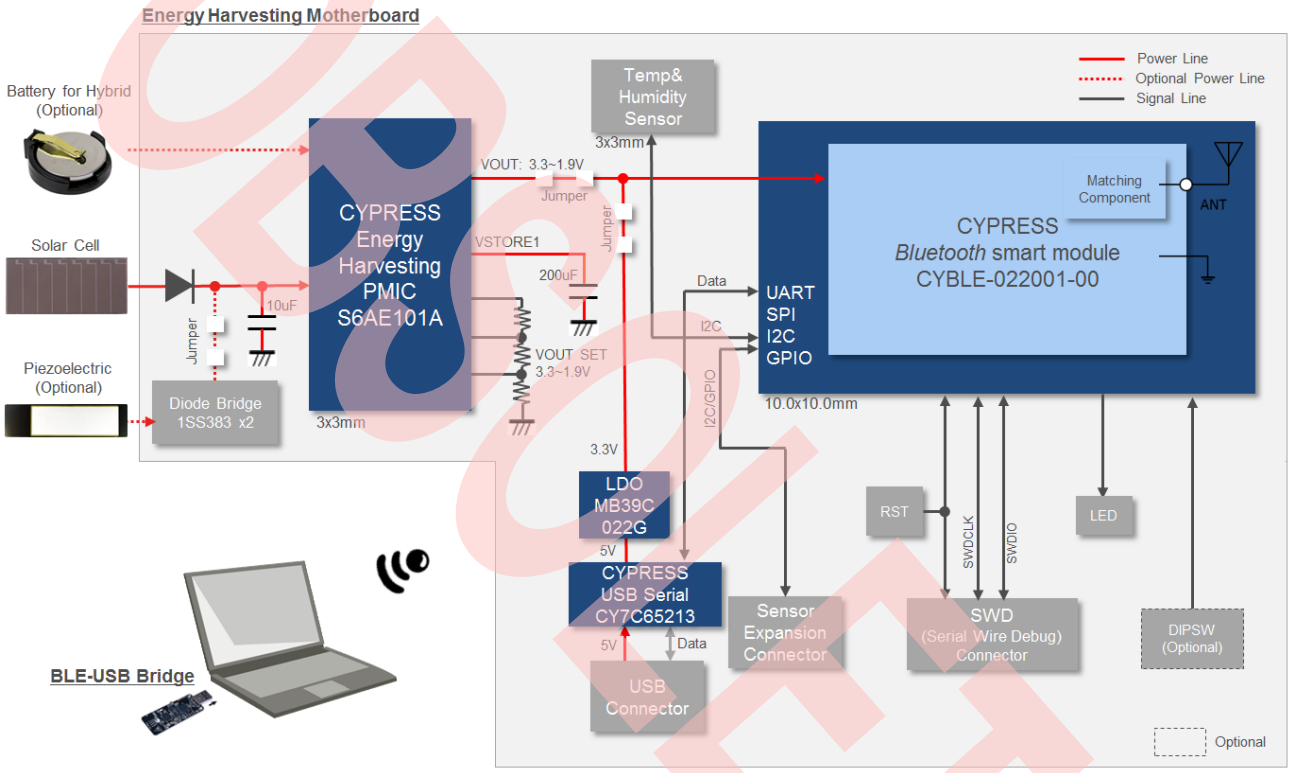


Figure 1-1. Block Diagram of Development Kit

1.1 Features

The Solar Powered IoT Device Kit provides everything needed to develop a light-powered sensor node that transmits sensor data using BLE:

- Operates using light (> 200 lux) energy harvested by the included solar cell
  - ☐ Supports the use of a vibration Energy Harvesting Device with an external diode bridge (not included)
  - ☐ Supports the use of a coin cell battery (optional, not supplied)
- Supports BLE communication with a PC through the provided BLE-USB Bridge that is pre-programmed with custom firmware for this kit
- Includes firmware that supports the following applications:
  - ☐ Bluetooth Low Energy (BLE) Beacon, transmitting data at 1.5 sec intervals with ambient light as low as 200 lx
  - ☐ Wireless Sensor Node (WSN), transmitting data at 6 sec intervals with ambient light as low as 200 lx
- Includes an expandable terminal on the Motherboard that can support the following:

OBSOLETE

  - ☐ Reset button for EZ-BLE Module
  - ☐ JTAG header to debug EZ-BLE Module
  - ☐ Expandable sensor interface (I2C/UART/SPI/GPIO)
  - ☐ DIP switch for future expansion (Not mounted)
  - ☐ LEDs for USB power and status
- Includes reference schematic, BOM list, and layout data for easy design
- Uses the following Cypress Devices:
  - ☐ S6AE101A ultra low power Energy Harvesting PMIC
  - ☐ CYBLE-022001-00 EZ-BLE PRoC Module
  - ☐ CY7C65213 USB-UART LP Bridge Controller
  - ☐ MB39C022G LDO

1.2 Applications

The following are the applications that use the Solar Powered IoT Device Kit:

- Battery-less WSN
- IoT device that monitors various sensors
- BLE Beacon
- Wearable device
- Building Energy Management System (BEMS)
- Home Energy Management System (HEMS)
- Factory Energy Management System (FEMS)
- Wireless lighting control
- Wireless HVAC sensor
- Security system

OBSOLETE

2 Kit Introduction

2.1 Kit Contents

Figure 2-1 shows the Solar-powered IoT Device kit. See the list below for a description of the numbered items.

Figure 2-1. Solar-Powered IoT Device Kit

1. Energy Harvesting Motherboard
2. BLE-USB Bridge
3. Solar Module (Panasonic AM-1801)
4. Two jumper wires
5. 220 µF Capacitor and 10Ω Resistor[1]
6. USB Standard-A to Mini-B cable
7. Quick Start Guide

[1] The 220 µF capacitor is an additional output capacitor. The 10Ω resistor is for current measurement. See Appendix B Using Extra Components for detailed information.

OBSOLETE

3 Software Installation

3.1 Installing Software

Follow these steps to install the S6SAE101A00SA1002 Solar-Powered IoT Device Kit software:

1. Download and install the Solar-Powered IoT Device Kit software from www.cypress.com/energy-harvesting.
   The Solar-Powered IoT Device Kit software is available in three different formats for download:
   - ☐ Solar-Powered IoT Device Kit Complete Setup: This installation package contains the files related to the kit. 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.
   - ☐ Solar-Powered IoT Device Kit Only Package: This executable file installs only the kit contents, which include code examples, hardware files, and user documents. This package can be used if all software prerequisites are installed on your computer.
   - ☐ Solar-Powered IoT Device Kit ISO: This file is a complete package, stored in a CD-ROM image format that can be used to create a CD, or extract using ISO extraction programs, such as WinZip or WinRAR. This file includes all the required software, utilities, drivers, hardware files, and user documents.
2. Run Install Solar-Powered IoT Device Kit to start the installation.
3. Select the folder to install the Solar-Powered IoT Device Kit-related files. Choose the directory and click **Next**.

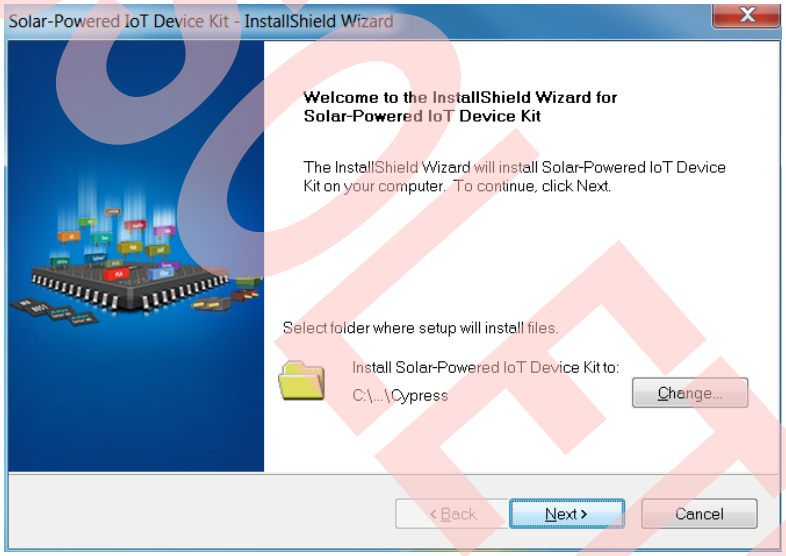


Figure 3-1. Selecting Directory

4. The Solar-Powered IoT Device Kit ISO installer automatically installs the required software. If the required software is not present in your computer., the Solar-Powered IoT Device Kit Setup installer directs you to download the required software from the Internet.
5. Choose the **Installation Type**, which can be Typical, Custom, or Complete in the Product Installation Overview window. Click **Next**.

OBSOLETE

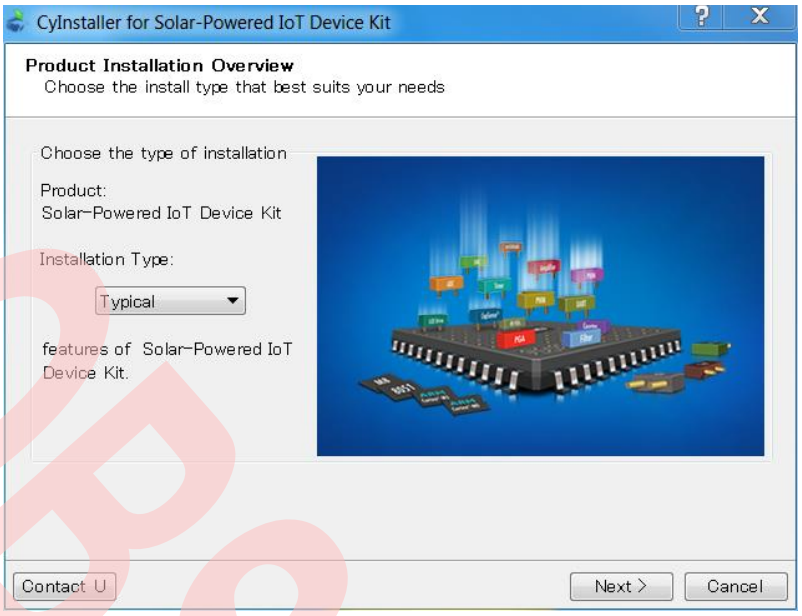


Figure 3-2. Selecting Installation Type

6. Read the Cypress License Agreement and accept the terms in the license agreement. Click **Next** to continue with the installation.

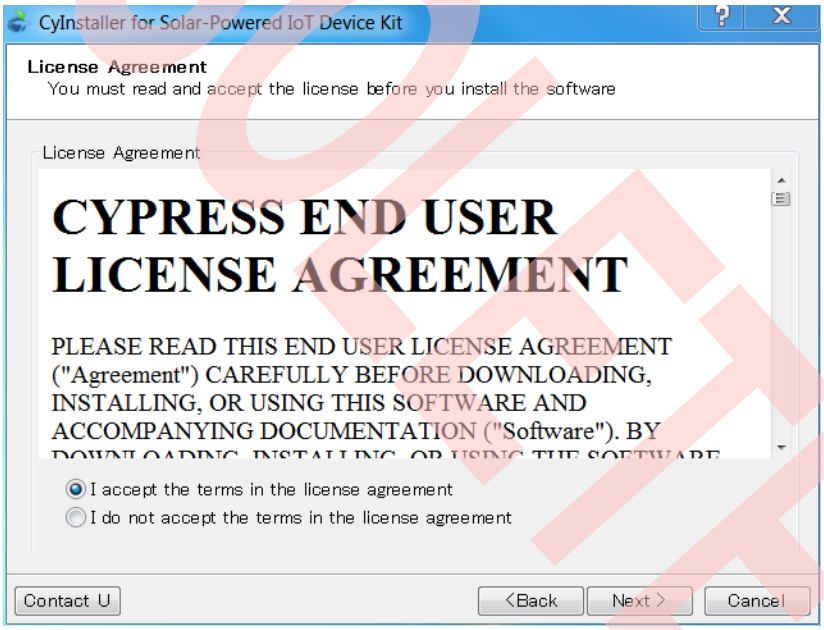


Figure 3-3. Accepting the License Agreement

7. When the installation begins, a list of packages appears on the installation page. A green check mark appears next to each package after successful installation. Click **Finish** to complete the Solar-Powered IoT Device Kit installation.
8. Enter your contact information or select the **Continue Without Contact Information** check box. Click **Finish** to complete the Solar-Powered IoT Device Kit installation.

OBSOLETE

After the installation is complete, the kit contents are available at:

<Install directory>\ Solar-Powered IoT Device Kit

The default locations, for Windows 7, are:

64-bit*: C:\Program Files (x86)\Cypress\ Solar-Powered IoT Device Kit*
32-bit*: C:\Program Files\Cypress\ Solar-Powered IoT Device Kit*

3.2 Uninstalling Software

To uninstall the Solar-Powered IoT Device Kit software, follow either of these methods (for Windows 7):

1. Go to **Start** > **All Programs** > **Cypress** > **Cypress Update Manager**; click the **Uninstall** button.
2. Go to **Start** > **Control Panel** > **Programs and Features**. Select the Solar-Powered IoT Device Kit program from the list and click **Uninstall/Change**.

3.3 Using PSoC® Creator™

PSoC Creator[2] is a state-of-the-art, easy-to-use integrated design environment (IDE). It is a revolutionary hardware and software co-design environment, powered by a library of pre-verified and pre-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 with integrated compiler chains and production programmers for PSoC devices.

Download the latest version from www.cypress.com/psoccreator.

For sample firmware information for this kit, see 6 Example Project.

[2] To develop firmware for the Solar-Powered IoT Device Kit, PSoC Creator 3.2 SP1 or newer version is required.

OBSOLETE

4 Getting Started

This section explains how to establish a connection between the Energy Harvesting Motherboard operating as a WSN and a PC with the BLE-USB Bridge. The connection confirms whether the Motherboard, BLE-USB Bridge, and the PC operate properly.

4.1 Solar-Powered BLE Beacon Operation

Establish a BLE connection to confirm that the Motherboard the BLE-USB Bridge operate properly. The motherboard contains all components of WSN, including:

- Energy harvesting PMIC S6AE101A
- Capacitors for energy storage
- An EZ-BLE PRoC module for transmitting data
- An I2C temperature and humidity sensor

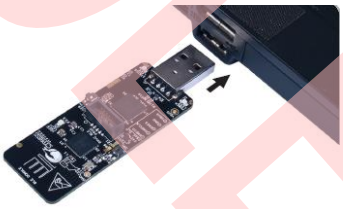
A USB to serial device is also included in the Motherboard to allow you to configure parameters such as the ID into the EZ-BLE module from a PC application. The motherboard comes with pre-loaded firmware to operate as a BLE Beacon. By connecting the Solar Module to the Motherboard and exposing the Solar Module to ambient light; the motherboard will power up and begin transmitting.

The BLE-USB Bridge is pre-configured to look for the transmission from the Motherboard operating as a BLE Beacon. By installing the BLE-USB Bridge on a Windows PC and using the provided software, you will be able to detect the Motherboard, and determine the distance between the Motherboard and the PC using BLE.

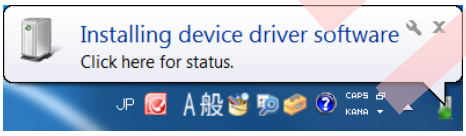
4.1.1 Installing USB Driver of BLE-USB Bridge

Follow these steps to install the USB driver of the BLE-USB Bridge:

1. Plug in the BLE-USB Bridge into your computer's USB port.



2. The driver installation starts automatically and the following message window will appear. Click the message window for status.



OBSOLETE

3. Confirm that the device driver installation has successfully completed (all components will be "Ready to use"). If the installation fails, proceed with the manual installation using the driver files in the *USB drivers* folder. See 4.1.2 USB Driver Installation Failure.

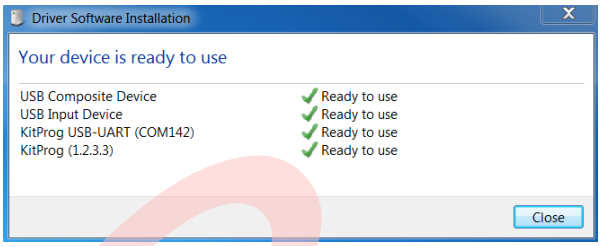


Figure 4-1. Installation Complete

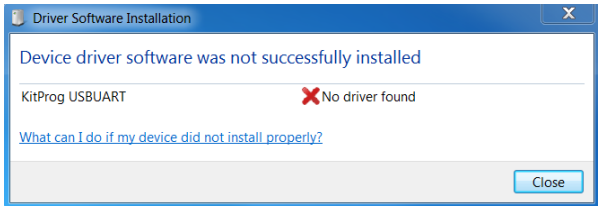
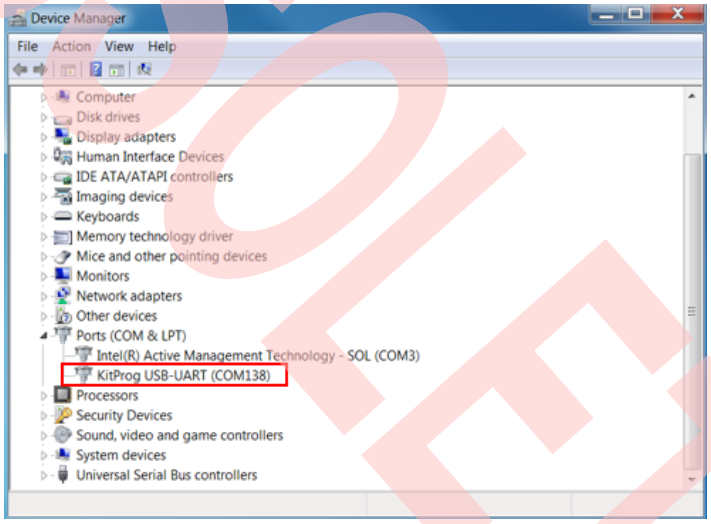


Figure 4-2. Installation Failure

4. After the successful installation of the device driver, check whether a new COM port KitProg USB-UART was added:
   a. Open the Device Manager:
      Windows 7: **Start** > **Control Panel** > **Device Manager**
      Windows 8/8.1/10: Right-click **Start** > **Device Manager**
   b. Under Ports (COM & LPT), check whether a COM port KitProg USB-UART was added. Note the COM number (COMxx).

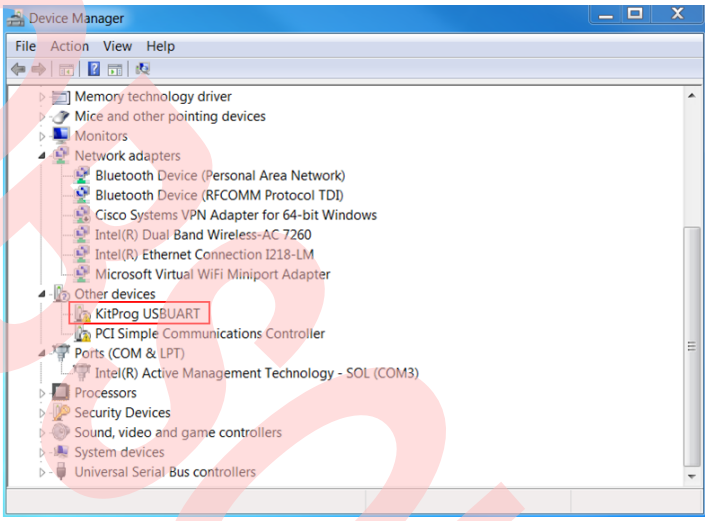


OBSOLETE

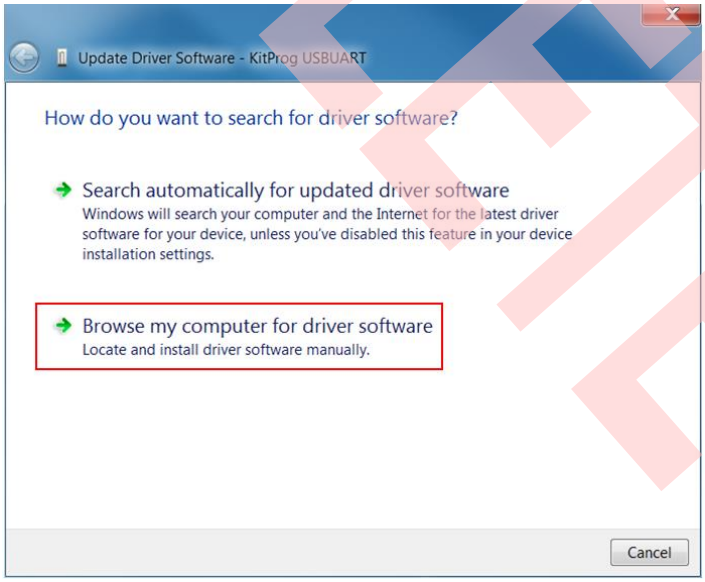
4.1.2 USB Driver Installation Failure

Follow these steps if the USB driver installation fails:

1. After the successful installation of the device driver, check whether a new COM port KitProg USB-UART was added:
   a. Open the Device Manager:
      Windows 7: **Start** > **Control Panel** > **Device Manager**
      Windows 8/8.1/10: Right-click **Start** > **Device Manager**
   b. Under Ports (COM & LPT), check whether a COM port KitProg USB-UART was added. Note the COM number (COMxx).



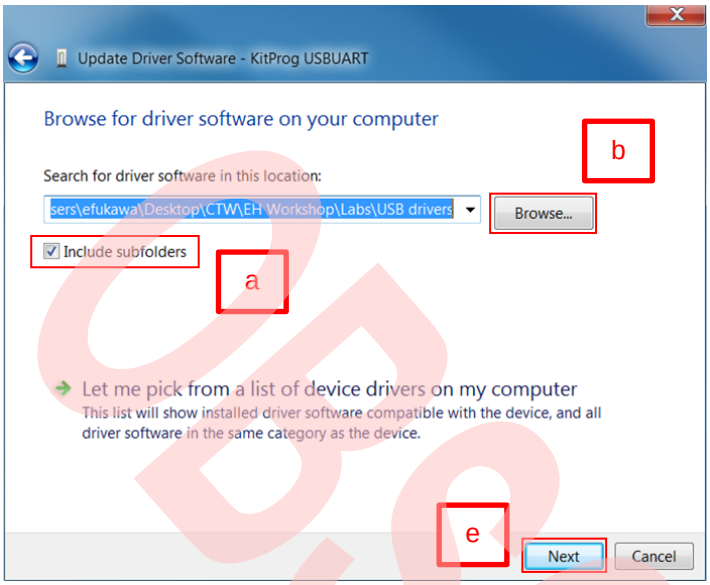
2. Update the USB driver software or the unconfigured KitProg USB-UART.
   a. Right-click **KitProg USB-UART**.
   b. Select **Update Driver Software…**
3. Select **Browse my computer for driver software**.

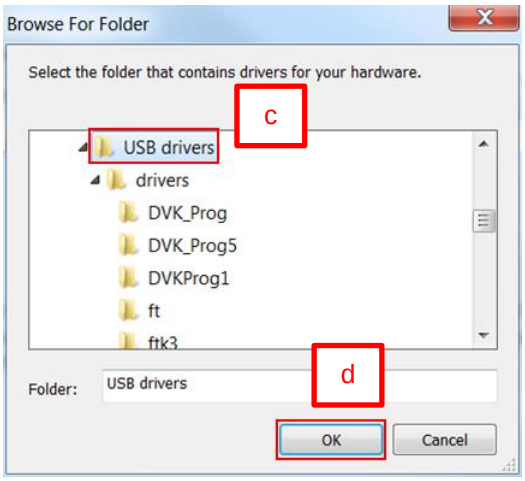


4. Search for USB driver in the *USB drivers* folder.
   a. Select the **Include subfolders** option.
   b. Click **Browse**.
   c. Select the **USB drivers** folder in the kit files.

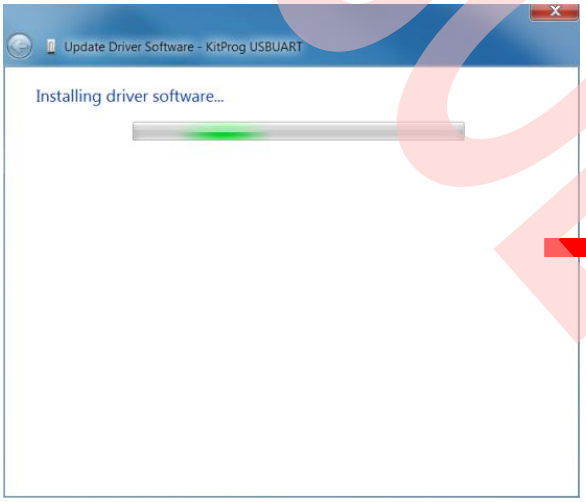
OBSOLETE

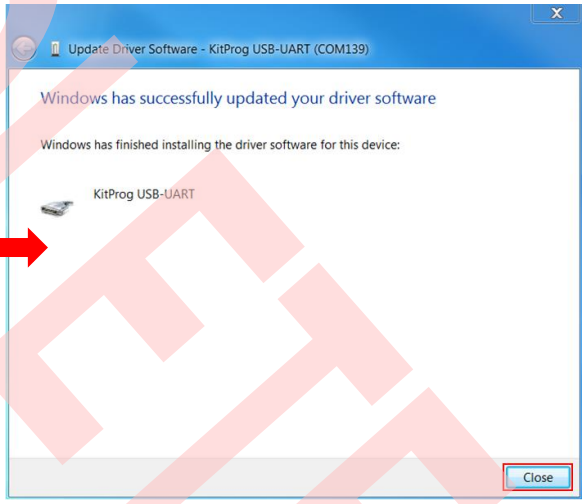
d. Click **OK**.
e. Click **Next**.





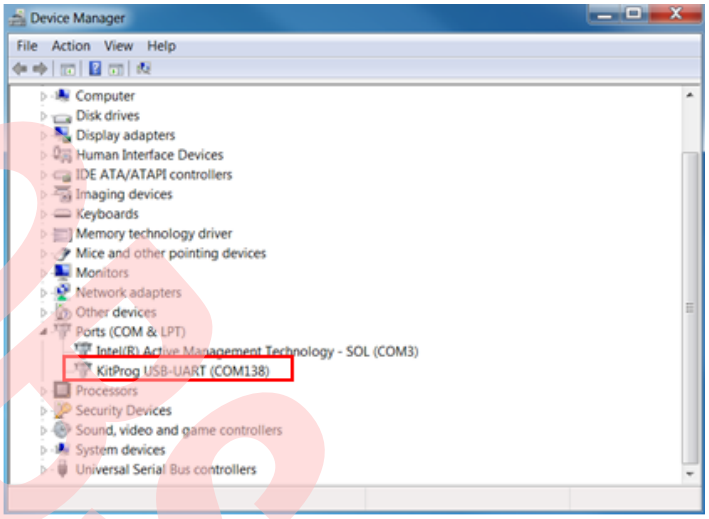
5. The installation of the USB driver begins. Click **Close** when the driver installation of the KitProg USB-UART finishes.





OBSOLETE

6. After the successful installation of the device driver, check whether a new COM port KitProg USB-UART was added:
   a. Open the Device Manager:
   b. Under Ports (COM & LPT), check whether a COM port KitProg USB-UART was added. Note the COM number (COMxx).



4.1.3 Installing USB Driver for Motherboard

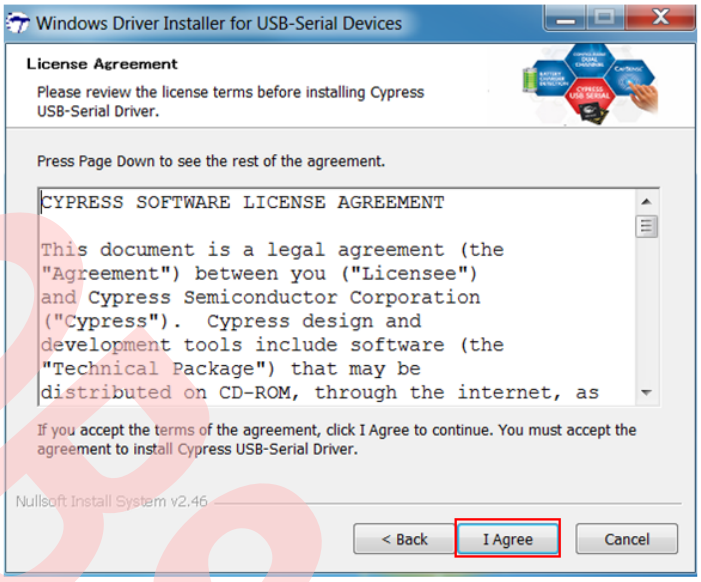
Follow these steps to install the USB driver for the Motherboard:

1. Run **CypressDriverInstaller.exe**, the installer for the USB serial device on the Motherboard. You can find the installer at *<Install directory>/Solar-Powered IoT Device Kit/1.0/USB drivers*.

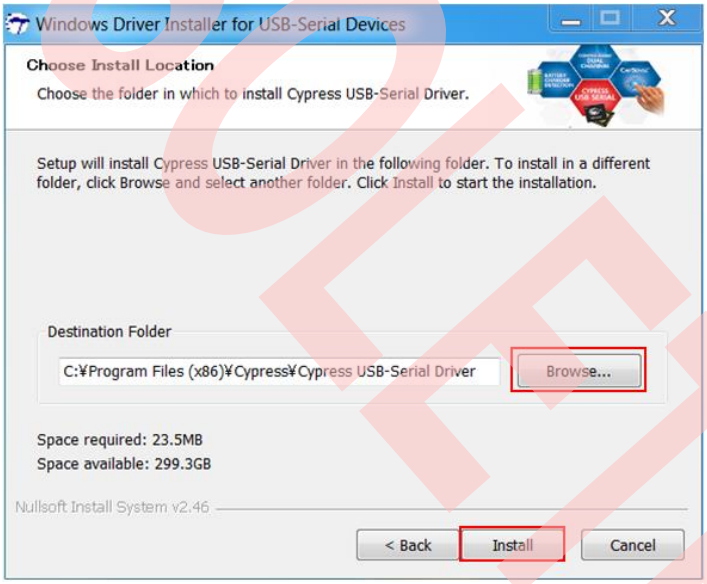
2. Click **Next** to continue.

OBSOLETE

3. Read the license agreement and click **I Agree**.

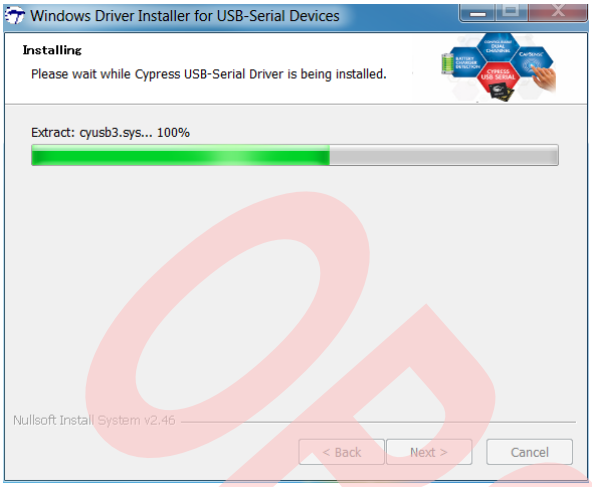


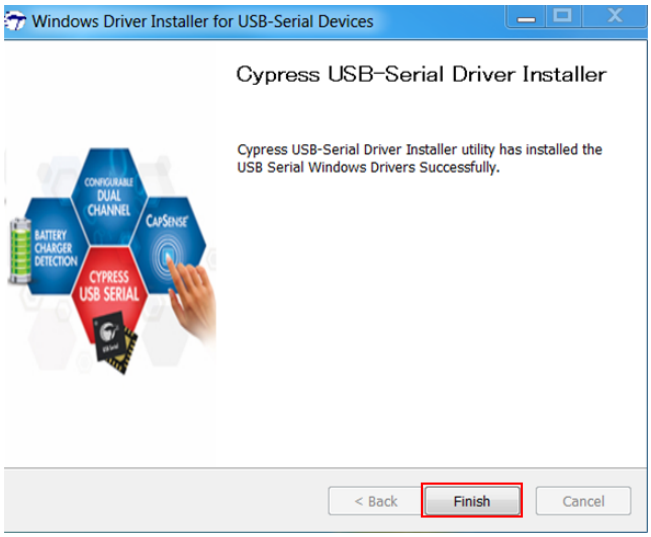
4. To install the driver at the default location, click **Install**. To change the Destination Folder, click **Browse** and choose a folder.



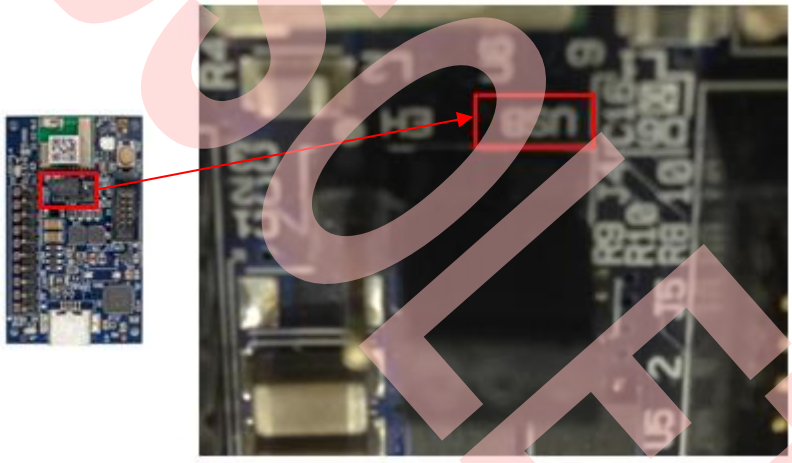
OBSOLETE

5. The installation of the USB driver begins. Click **Close** when the driver installation of the Motherboard finishes.

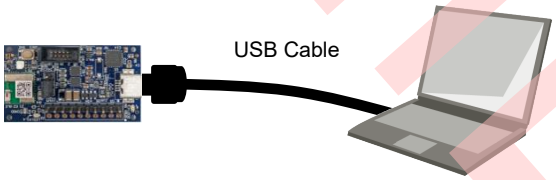




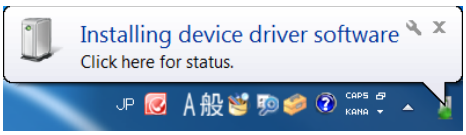
6. Configure the Motherboard to receive power from the USB port. Change jumper J4 to "USB" from "EH".”



7. Connect the Motherboard to your computer using a USB cable.

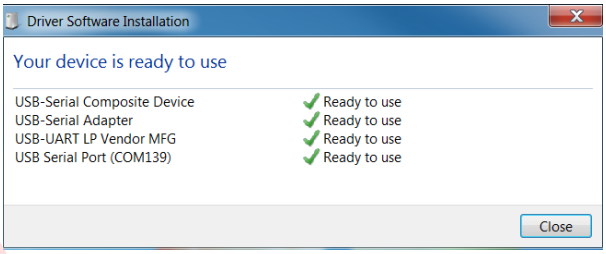


8. The driver installation starts automatically and the following message window will appear. Click the message window for status.



OBSOLETE

9. Confirm that the device driver installation has successfully completed (all components will be Ready to use).

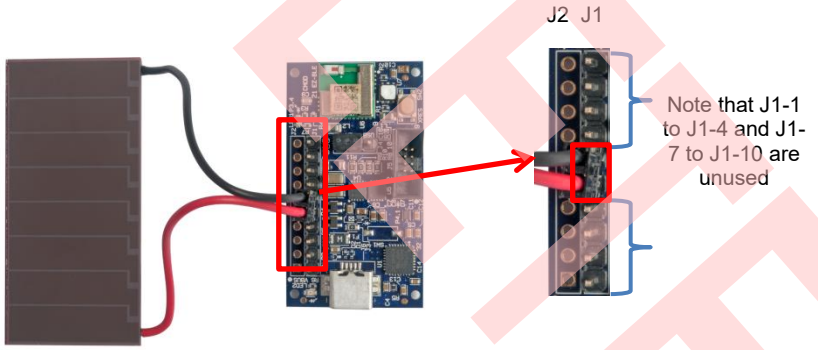


10. After the successful installation of the device driver, check whether a new COM port KitProg USB-UART was added:
    a. Open the Device Manager:
    b. Under Ports (COM & LPT), check whether a COM port KitProg USB-UART was added. Note the COM number (COMxx).
11. Finally, disconnect the USB cable, then reset the jumper J4 set in step 6 back to "EH" from "USB" to supply power from the Solar Module.

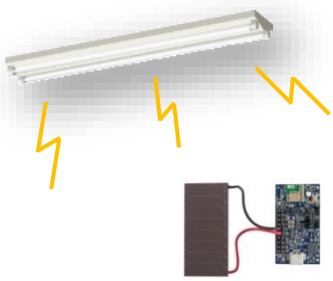
4.1.4 Establishing BLE Connection

Follow these steps to establish a BLE connection:

1. Connect the Solar Module (AM-1801, included in the kit) to the Energy Harvesting Motherboard. Plug the black wire (negative) to J1-6 and the red wire (positive) to J1-5 as shown below.



2. Place the Motherboard with Solar Module under an office light. The firmware to operate the Motherboard as a BLE Beacon is pre-loaded from the factory. After attaching the Solar Module and placing the Motherboard under a suitable light level (see Table 4-1), it will automatically power up and begin transmitting.

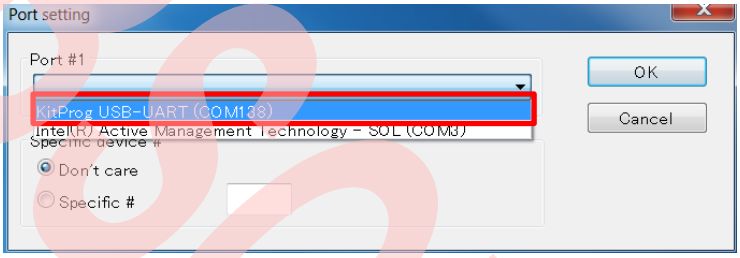


OBSOLETE

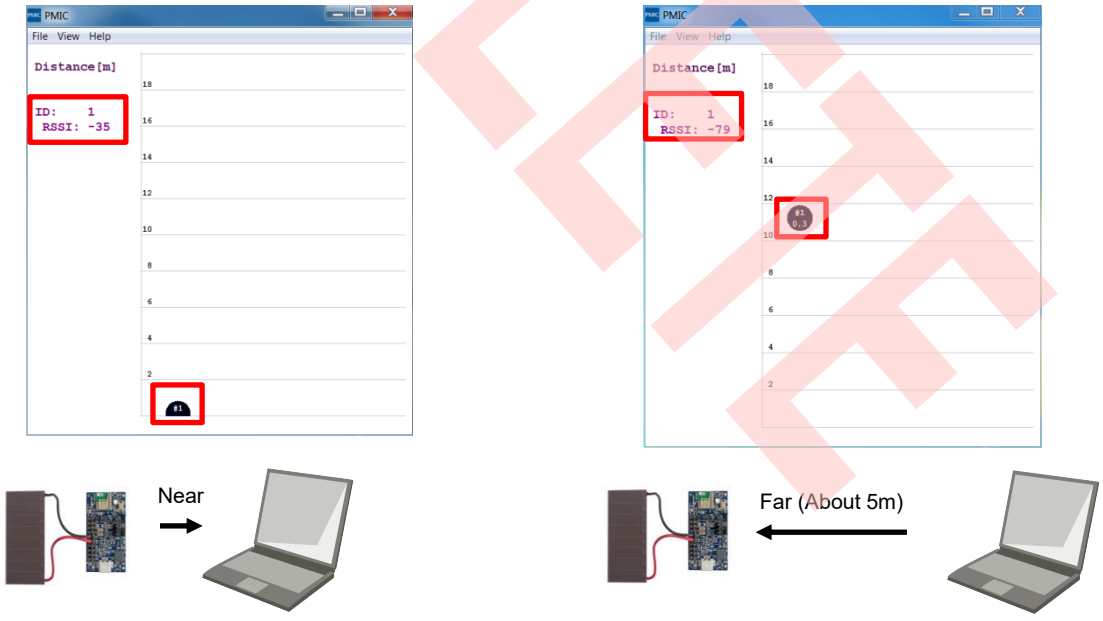
Typical Light Level	Environment	Time Interval of Beacon[3]
~1 lx	Moonlight	Does not work
50 lx~100 lx	Under street lighting	Does not work
200 lx~400 lx	At Museum	1.0 sec ~ 5.0 sec
400 lx~500 lx	Office lighting	0.6 sec ~ 1.0 sec
1000 lx	Shopping mall, Rainy day	0.4 sec ~ 0.6 sec

Table 4-1. Light Level versus Time Interval

3. Plug in the BLE-USB Bridge to your computer's USB port.
4. Run *PMIC.exe*, which is the Windows application used to view data received from the Motherboard. You can find the exe in the installation directory: *<Install directory>/Solar-Powered IoT Device Kit/1.0/PMIC Software*.
5. In the Port setting dialog, select **KitProg USB-UART (COMxx)** from the Port #1 drop-down list, where COMxx corresponds to the port that was confirmed in step 5. Retain the **Specific Device #** as **Don't care**. Click **OK**.



6. Find the MAJOR number of the Motherboard on the PMIC Software (see 4.3 Serial Command List). Then, move the Motherboard away from your computer. The Received Signal Strength Indicator (RSSI) value will change and the graphic will be updated.



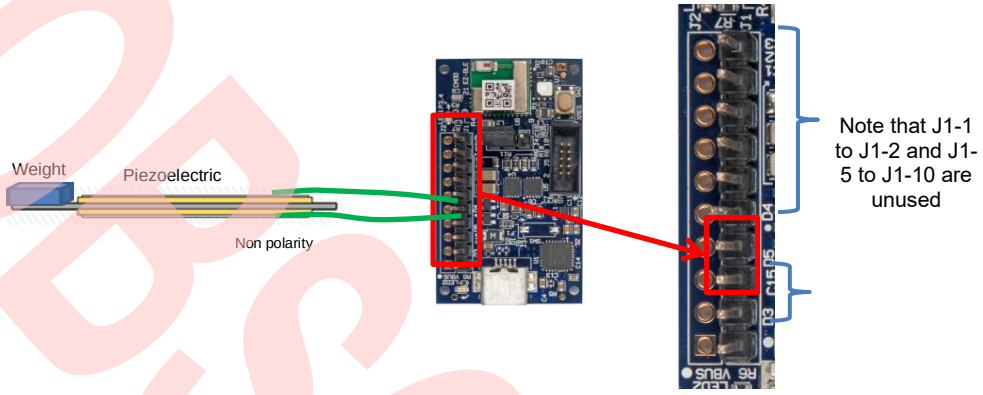
[3] The initial setting of time interval is set 1.5 sec. You need to configure an `ITRVL` command to change time interval. See 4.3 Serial Command List.

OBSOLETE

4.1.5 Vibration Energies Connection (Optional)

The Energy Harvesting Motherboard receives AC voltage from piezoelectric or electro-magnetic Energy Harvesting Devices (EHDs) that harvest vibration energy. To confirm this operation, a piezoelectric or electro-magnetic EHD is required (not supplied with Kit). To check the vibration energies connection:

1. Connect the piezoelectric or electro-magnetic EHD to the Motherboard. Plug the wires from the EHD to J1-3 and J1-4 as shown below. Note that there is no polarity.



2. Move the EHD to generate vibration energy.
3. Follow steps 3 to 5 in 4.1.4 Establishing BLE Connection to confirm that the Energy Harvesting Motherboard is operating. Figure 4-3 is sample waveform for the operation of vibration energies. If the Motherboard is not operating, you may have to increase the vibration energy. See the documentation for the EHD being used.

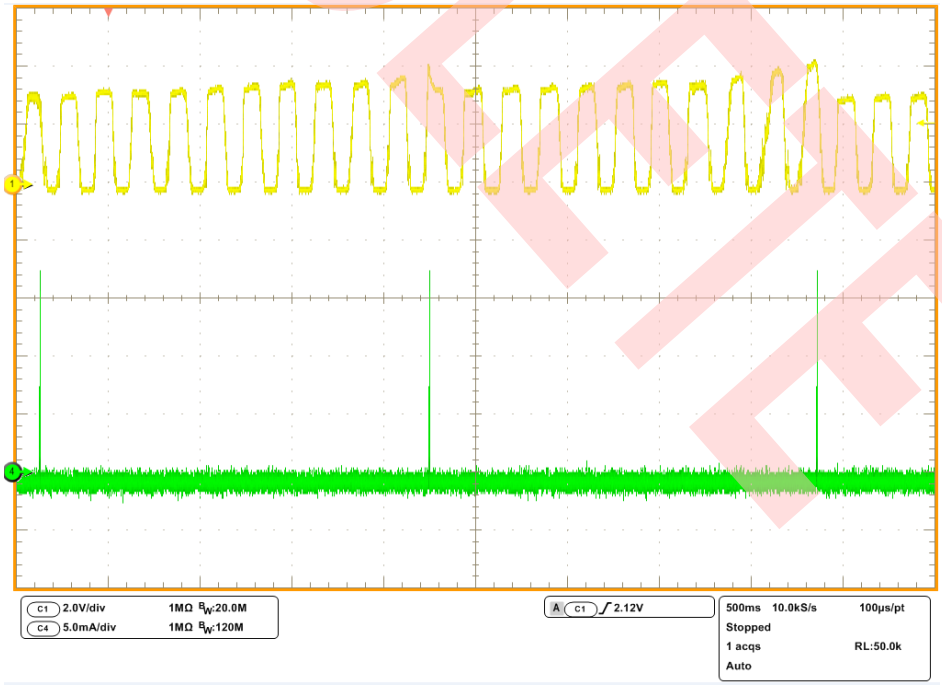


Figure 4-3. Sample Waveform for Operation of Vibration Energies

OBSOLETE

4.2 Solar-Powered Wireless Sensor Node (WSN) with BLE Beacon

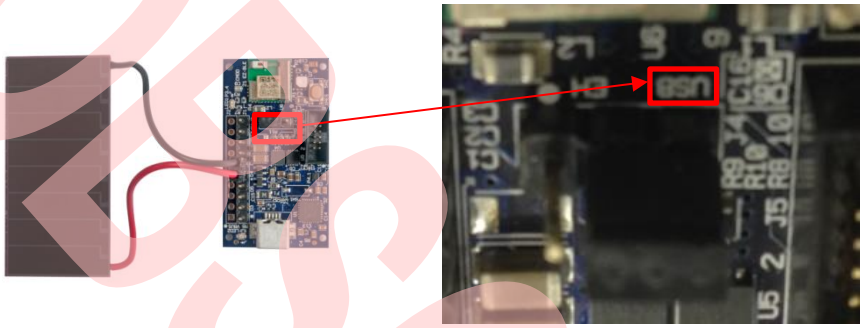
In this section, you will configure the Motherboard as a WSN by turning ON the temperature and humidity sensor. You will use a serial USB connection from your PC to send configuration commands to the Motherboard.

You will also check if the Motherboard is operating as a WSN by using the provided software on your PC to detect temperature and humidity changes.

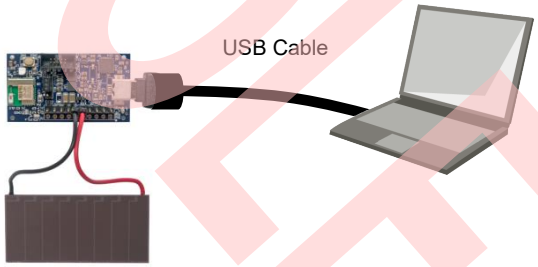
4.2.1 Configuring Motherboard as a WSN

Follow these steps to configure the Motherboard as a WSN:

1. Configure the Motherboard to receive power from the USB port by changing jumper J4 to "USB" from "EH".



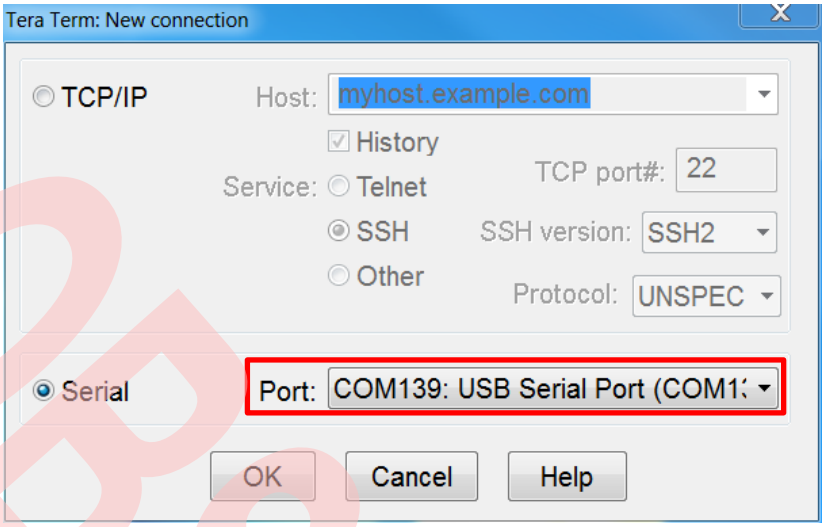
2. Connect the Energy Harvesting Motherboard to your computer using a USB cable.



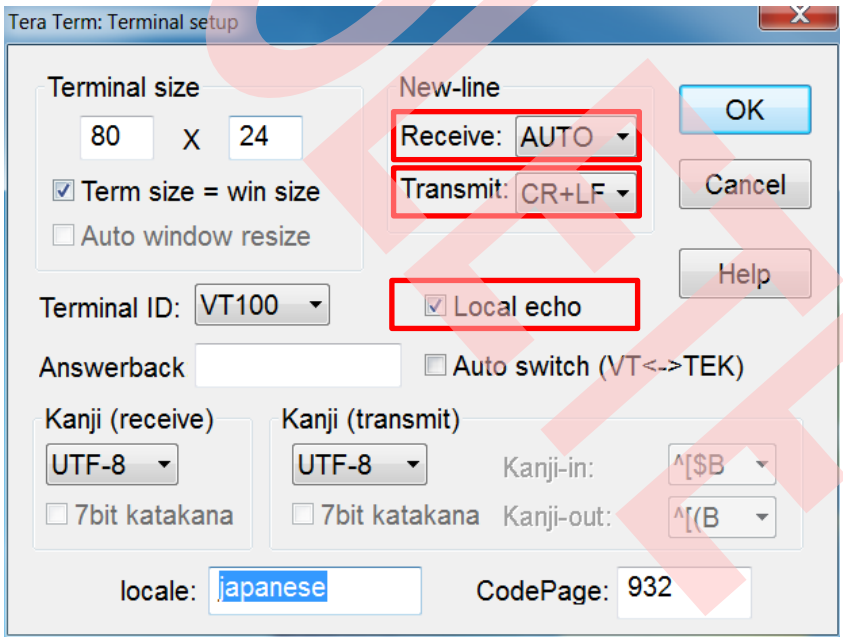
3. Confirm that a COM port (USB Serial Port) was added in the Windows Device Manager:
   a. Open the Device Manager:
   b. Under Ports (COM & LPT), check whether a USB Serial port was added. Note the COM number (COMxx).
4. Install Tera Term from the following location: *<Install directory>/Solar-Powered IoT Device Kit/1.0/PMIC Software/teraterm*.
5. After installing, run Tera Term:
   - ☐ Windows 7: **Start** > **All Programs** > **Tera Term**
   - ☐ Windows 8/8.1: Ctrl + Tab keys > All Apps > Tera Term
   - ☐ Windows 10: **Start** > **All Apps** > **Tera Term**

OBSOLETE

6. In Tera Term, follow the menu path **File** > **New Connection**. In the New Connection window, select the **Serial** option and select **COMxxx: USB Serial Port(COMxxx)** from the **Port** drop-down list. Click **OK**.

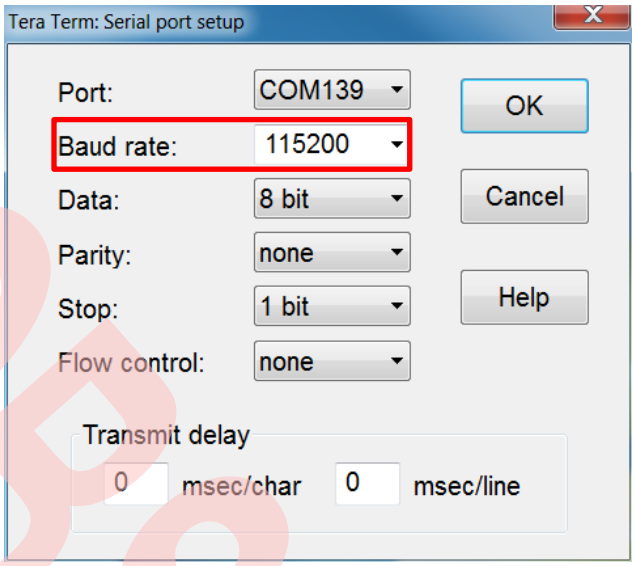


7. Follow the menu path **Setup** > **Terminal** and configure the following terminal settings and click **OK**:
   - ☐ **Receive**: AUTO
   - ☐ **Transmit**: CR+LF
   - ☐ **Local echo**: Check
   - ☐ **Other settings**: Default



OBSOLETE

8. Follow the menu path **Setup** > **Serial Port**. In the Serial Port setup window, set the **Baud rate** to **115200**. Click **OK**.

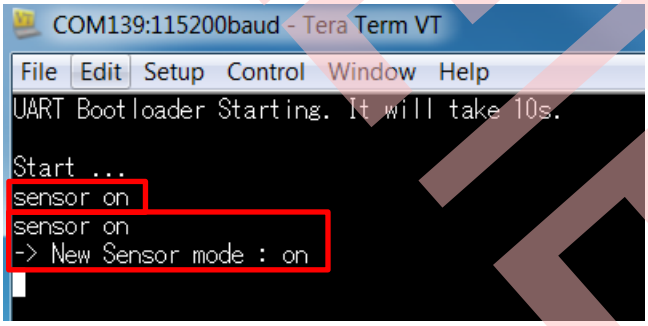


9. Enable the sensor on the Motherboard by placing the Motherboard into command mode and sending a command to the Motherboard from the PC.

   a. Push and release the XRES button on the Motherboard. Note that when the Motherboard is in the command mode, it stops transmitting BLE WSN data.

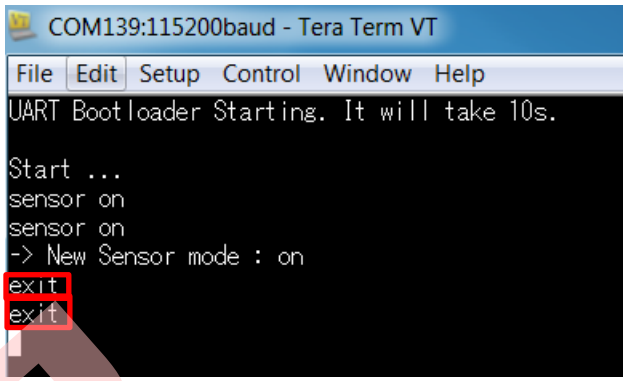
   b. Check whether the message “Start….” Appears on Tera Term. This confirms that the Motherboard is in the command mode and is ready to receive commands.

   c. On Tera Term, type “sensor on” and press **Enter**. The Motherboard responds with a confirmation message as shown below. See 4.3 Serial Command List for a list of all commands.

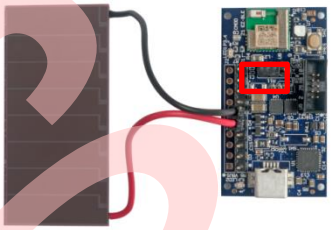


OBSOLETE

d. To leave the command waiting mode, type “exit” and press **Enter.** The board will acknowledge the command by responding with an “exit”. The board will retransmit the Bluetooth Beacon data.



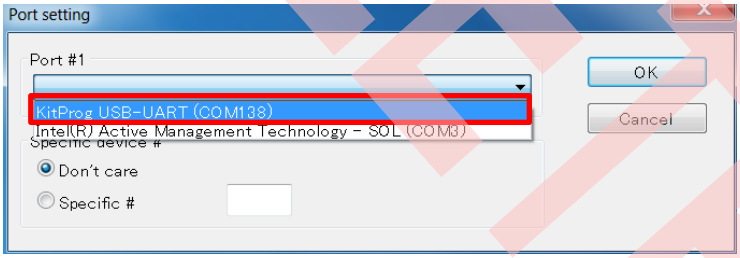
10. Finally, disconnect the USB cable Then, reset the jumper J4 set in step 1 to "EH" from "USB" to supply power from the Solar Module.



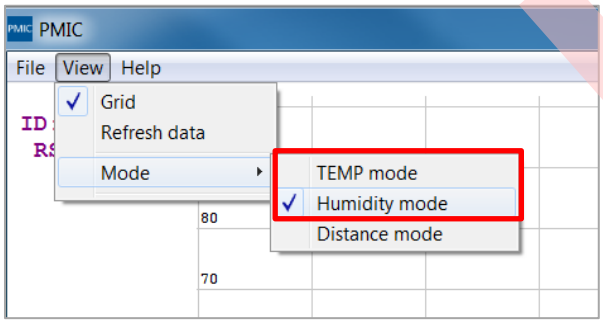
4.2.2 Validating WSN

Follow these steps to check whether WSN is operating:

1. Connect the BLE-USB Bridge to your computer's USB port.
2. Run *PMIC.exe* (*<Install directory>/Solar-Powered IoT Device Kit/1.0/PMIC Software*). In the Port setting dialog, select **KitProg USB-UART (COMxx)** from the Port #1 drop-down list, where COMxx corresponds to the port that was confirmed earlier. Retain the **Specific Device #** as **Don’t care**. Click **OK**.

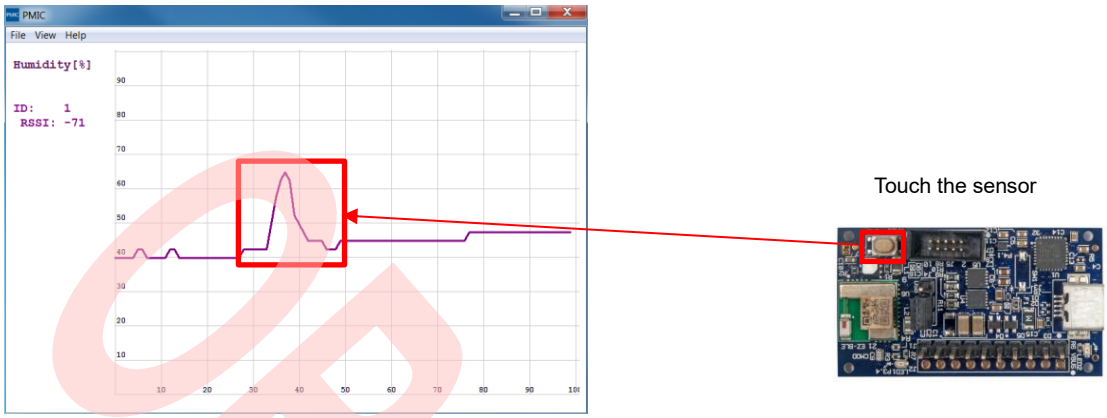


3. Follow the menu path **View** > **Mode** and select either **Humidity mode** or **TEMP mode**.



OBSOLETE

4. Place your finger on the sensor on the Motherboard. This raises the temperature and humidity from the indoor environment condition. You should see a corresponding change in humidity or temperature on your PC. When touching the board, be careful of static electricity.



If you need to operate by vibration, see 4.1.5 Vibration Energies Connection (Optional).

4.3 Serial Command List

Table 4-2 lists the serial commands which can be used to control the kit from your computer using the USB serial interface. See 4.2.1 Configuring Motherboard as a WSN for instructions on how to issue these commands via the USB serial interface. The commands are case sensitive.

No.	Commands Name	Description	Default
[1]	UUID	Read/Write of UUID[4]	00050001-0000-1000-8000-00805F9B0131 [hex]
[2]	MAJOR	Read/Write of MAJOR[4]	0x0001
[3]	MINOR	Read/Write of MINOR[1]	0x0001 (**Note**: Sensor data is sent after it is overwritten in the MINOR region when SENSOR command is ON)
[4]	TXPWR	Read/Write of Transmitter Power Strength	3 dBm
[5]	RSSI	Read/Write of Receiver Power Strength for distance 1m (RSSI)	-61 dBm
[6]	ITRVL	Read/Write of Advertise Interval for Beacon	1500 ms (**Note**: Time interval is fixed at six seconds when the SENSOR command is ON)
[7]	COID	Read/Write of Bluetooth Company	0x0131 (Cypress Semiconductor Corporation)
[8]	SENSOR	Overwrite sensor information in MINOR packet region and send it	OFF
[9]	ERASE	Default parameters	-
[10]	EXIT	Finish the command waiting mode, and then retransmit the Bluetooth beacon data	-
[11]	VER	Display Firmware Version	-

Table 4-2. Command List

[4] See 6.3 BLE Beacon Process for detailed information.

OBSOLETE

[1] Read/Write of UUID

[1-1] Read
Read UUID data. Default: 00050001-0000-1000-8000-00805F9B0131

Example
```
UUID↵
(echo)  UUID
(output) -> UUID: 00050001-0000-1000-8000-00805F9B0131
```

[1-2] Write
Write UUID data.

Example
```
UUID EEEEDDDD-CCCC-BBBB-AAAA-999988887777↵
(echo)  UUID EEEEDDDD-CCCC-BBBB-AAAA-999988887777
(output) -> New UUID: EEEEDDDD-CCCC-BBBB-AAAA-999988887777
```

[2] Read/Write of MAJOR

[2-1] Read
Read MAJOR. Default: 0x0001

Example
```
MAJOR↵
 (echo)  MAJOR
 (output) -> MAJOR: 0001
```

[2-2] Write
Write MAJOR.

Example
```
MAJOR 1A2F↵            <- Input HEX data
 (echo)  MAJOR 1A2F
 (output) -> New MAJOR: 1A2F
```

[3] Read/Write of MINOR

[3-1] Read
Read MINOR. Default: 0x0001

Example
```
MINOR↵
 (echo)  MINOR
 (output) -> MINOR: 0001
```

[3-2] Write
Write MINOR.

Example
```
MINOR 2C3D↵            <- Input HEX data
 (echo)  MINOR 2C3D
 (output) -> New MINOR: 2C3D
```

OBSOLETE

[4] Read/Write of Transmitter Power Strength

[4-1] Read
Read Power Strength. Default: 3 dBm

Example
```
TXPWR↵
 (echo)  TXPWR
 (output) -> TX power in dBm: 3
```

[4-2] Write
Set Power Strength. Set Value: -18, -12, -6, -3, -2, -1, 0, 3

Example
```
TXPWR -18↵
 (echo)  TXPWR -18
 (output) -> New TX power in dBm: -18
```

[5] Read/Write of Receiver Power Strength for distance 1m (RSSI)

[5-1] Read
Read RSSI. Default: -61dBm

Example
```
RSSI↵
 (echo)  RSSI
 (output) -> RSSI in dBm: -61
```

[5-2] Write
Set RSSI.

Example
```
RSSI -90↵
 (echo)  RSSI -90
 (output) -> New RSSI in dBm: -90
```

[6] Read/Write of Advertise Interval

[6-1] Read
Read Advertise Interval. Default: 1500ms

Example
```
ITRVL↵
 (echo)  ITRVL
 (output) -> Advertise Interval in msec:   1500
```

[6-2] Write
Set Advertise Interval. Set Value: 100~10240 ms

Example
```
ITRVL 10240↵
 (echo)  ITRVL 10240
 (output) -> New Advertise Interval in msec: 10240
```

OBSOLETE

[7] Read/Write of Bluetooth Company

[7-1] Read
Read Bluetooth Company. Default: 0x0131 (Cypress Semiconductor Corporation)

Example
```
COID↵
 (echo)  COID
 (output) -> Company ID: 0059
```

[7-2] Write
Write Bluetooth Company.

Example
```
COID 004C↵              <- Input HEX data
 (echo)  COID 004C
     (output) -> New Company ID: 004C
```

[8] Read/Write of sensor setting

[8-1] Read
Read sensor setting Default: OFF

Example
```
SENSOR↵
   (echo)  SENSOR
   (output) -> Sensor mode: OFF
```

[8-2] Write
Change sensor setting set value: ON or OFF

Example
```
SENSOR ON↵
(echo)  SENSOR ON
(output) -> New Sensor mode: ON
```

[9] ERASE
Erase the flash memory in MCU. After erase, all value will be default parameters.

Example
```
ERASE↵
(echo) ERASE
(output) Erase completed!
```

[10] EXIT
Finish the command waiting mode, and then retransmit the Bluetooth beacon data.

Example
```
EXIT↵
   (echo) EXIT
   (output) ---
```

OBSOLETE

[11] VER
Display Firmware Version.

Example

```
VER↵
    (echo) VER
    (output) -> S6SAE101A00SA1002 Sample Firmware, Version 1.0
```

[*] Input another command (Error Handling)
TEST

```
    (echo)  TEST
    (output) Command format error!!
```

OBSOLETE

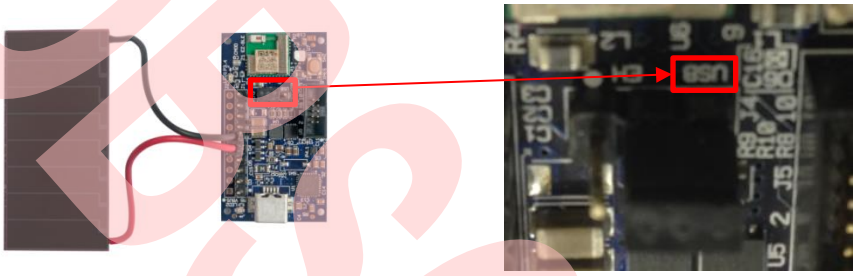
5 Program and Debug

The Solar-Powered IoT Device Kit can be programmed using UART Bootloader, and it can be programmed and debugged using PSoC Creater with MiniProg3. Before debugging the device, ensure that PSoC Creator is installed on the computer.

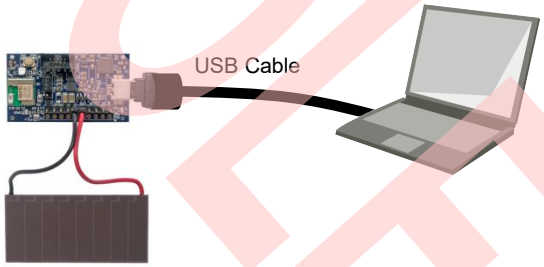
5.1 UART Bootloader (Program Only)

An UART Bootloader makes it possible for a product's firmware to be updated in the field. Bootloading is a process that allows you to upgrade your system firmware over UART. The UART Bootloader communicates with a host to get new application code or data, and writes it into the device's flash memory.

1. Configure the Motherboard to receive power from the USB port. Change jumper J4 to "USB" from "EH".

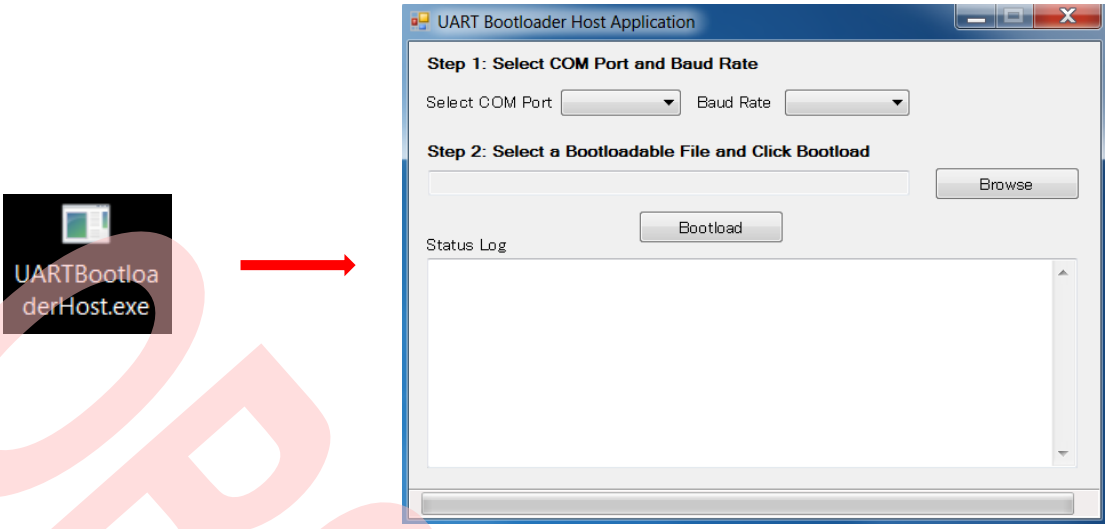


2. Connect the Energy Harvesting Motherboard to your computer using a USB cable.

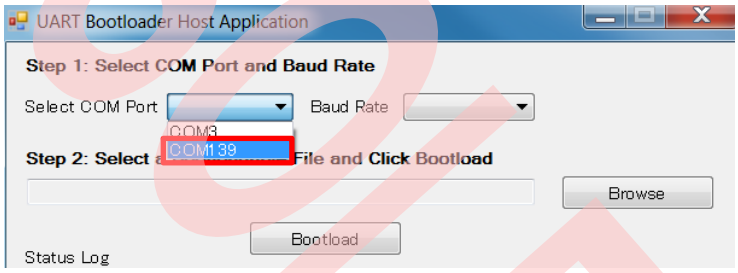


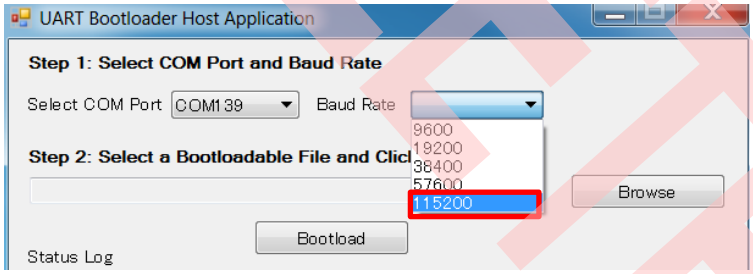
3. Confirm that a COM port called USB Serial Port was added:
   a. Open the Device Manager:
   b. Under Ports (COM & LPT), check whether a COM port USB Serial Port was added. Note the COM number (COMxx).
4. Run *UARTBootloaderHost.exe* from the installation directory: <*Install directory*>*\Solar-Powered IoT Device Kit\1.0\PMIC Software*.

OBSOLETE

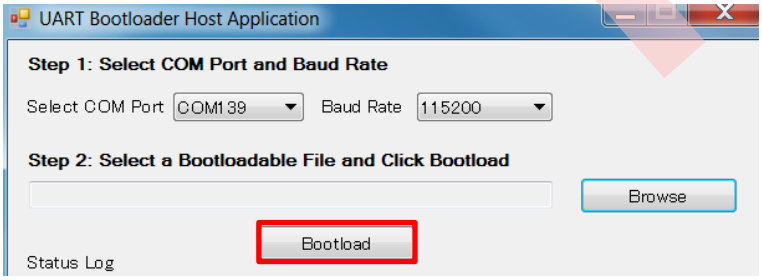


5. Select USB serial port where COMxx corresponds to the port that was confirmed in step 3. Select **115200** as the **Baud Rate**.

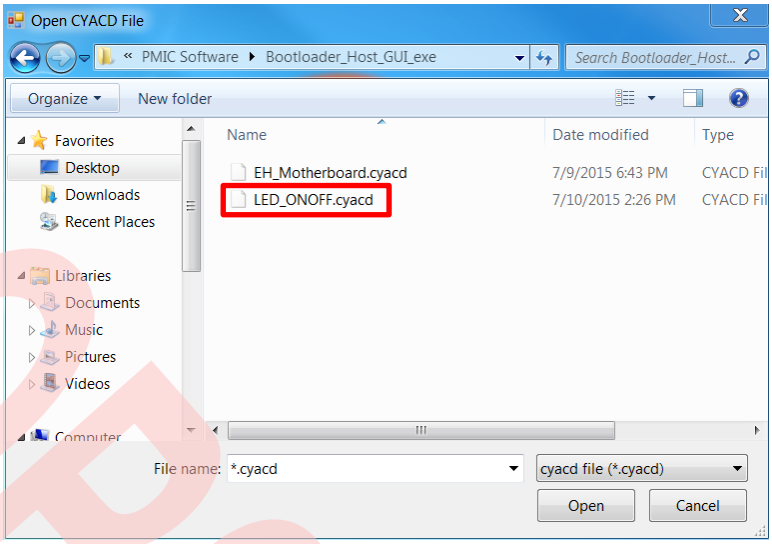




6. Click **Browse** and select *LED_ONOFF.cyacd* from *<Install directory>/Solar-Powered IoT Device Kit/1.0/PMIC Software/Bootloader_Host_GUI_exe*.

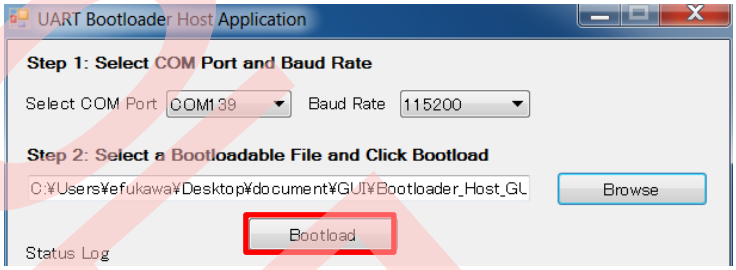


OBSOLETE

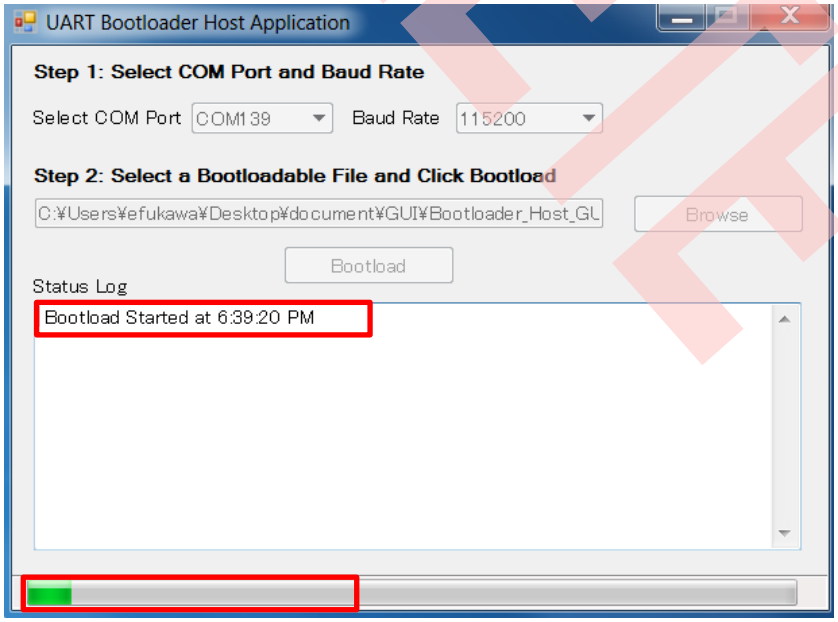


If you have already developed your own firmware using PSoC Creator, the *.cyacd* file is generated in the project folder: *[PSoC Creator Project Folder]/CortexM0/ARM_GCC_484/Debug (or Release)*.

7. Press the XRES button on the Energy Harvesting Motherboard, and then click **Bootload** in the UART Bootloader Host Application dialog.



A "Bootload Started at X:XX:XX" message will appear in the Status Log window, and then an indicator will advance.



The programing is completed when a "Bootload in successful !!" message appears in the Status Log window.

8. Disconnect and re-connect the USB cable, and then the status LED will blink at 1 sec interval. This sample firmware only supports LED blinking control as a demonstration (See A.1 LED ONOFF Project for detailed information), therefore, reprogram the *EH_Motherboard.cyacd* in the *Bootloader_Host_GUI_exe* folder to restore the BLE Beacon operation (See steps 4 to 7 above).

5.2 PSoC Creator with MiniProg3 (Program and Debug)

The MiniProg3[5] (not included in this kit) is the hardware/firmware block for onboard programming, debugging, and bridge functionality. It is a common reusable hardware/firmware block used across many Cypress kit platforms. The MiniProg3 communicates with PSoC Creator software to program/debug the target EZ-BLE PRoC Module over the SWD interface.

The following are the requirements for the preparation of program and debug:

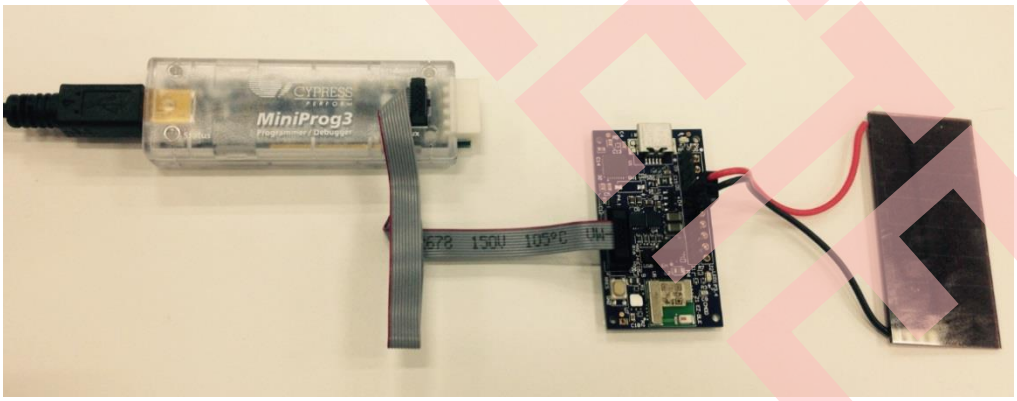
- Energy Harvesting Motherboard
- MiniProg3 Program and Debug Kit[5] (Not included in this kit)
- Windows PC for Programing and Debug
- PSoC Creator 3.2 SP1 or newer[6]

5.2.1 Program

Follow these steps to program the firmware:

Disconnect USB cable from the Energy Harvesting Motherboard.

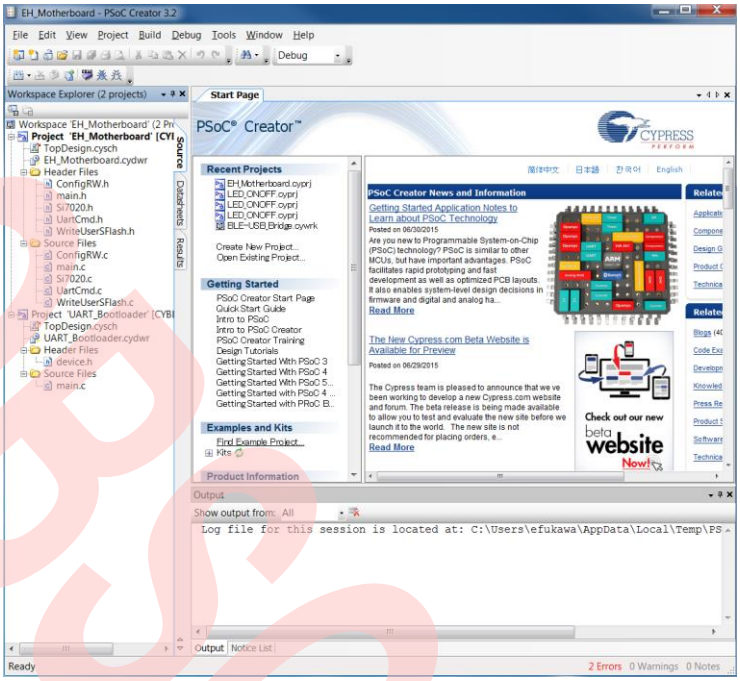
1. Set the J4 jumper socket to "EH" to supply the power from MiniProg3. A protection diode is mounted between the Solar Module and the S6AE101A, therefore, it is not necessary to remove the solar module when using the MiniProg3.
2. Connect the MiniProg3 to the Energy Harvesting Motherboard and your computer as shown below.



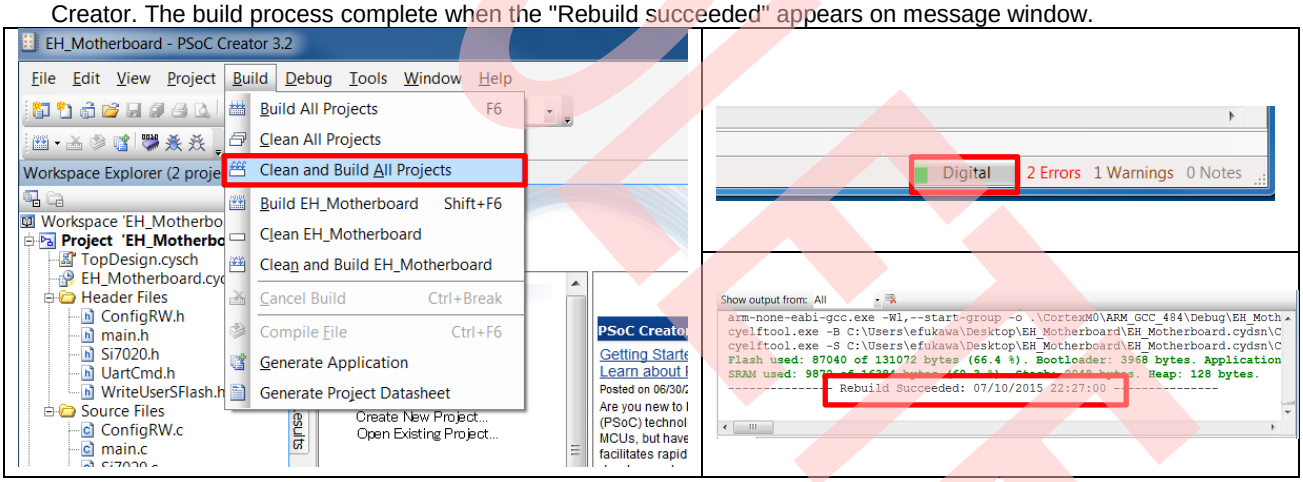
[5] To debug this board, a MiniProg3 Program and Debug Kit is required.

[6] The kit does not support PSoC Creator 3.2 or older. It needs to update PSoC Creator when you are using the 3.2 or older.

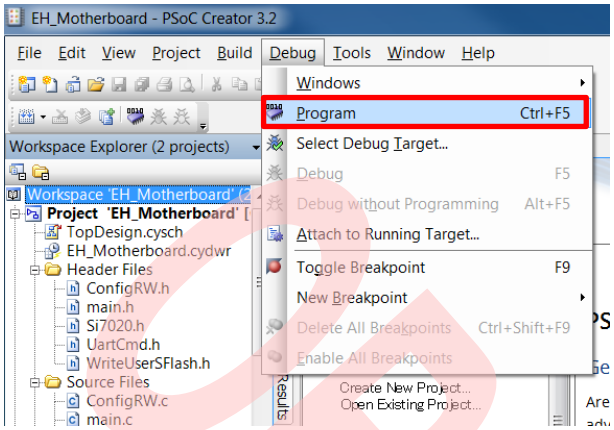
3. Open the sample project *EH_Motherboard.cypj* for this kit from *<Install directory>/Solar-Powered IoT Device Kit/1.0/Firmware/EH_Motherboard/EH_Motherboard.cydsn*.

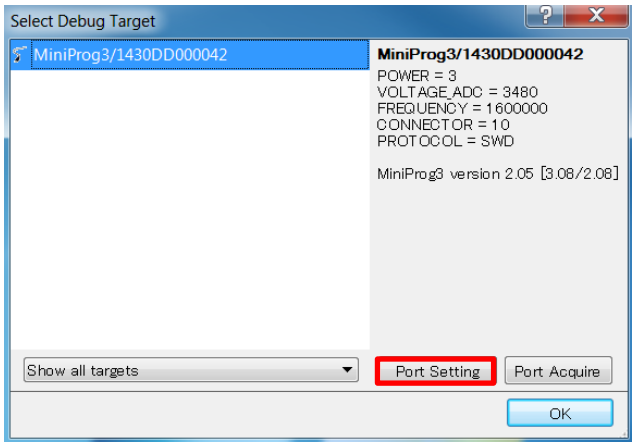


4. Follow the menu path **Build** > **Clean and Build All Projects**. The build status will appear on lower right side of PSoC Creator. The build process complete when the "Rebuild succeeded" appears on message window.



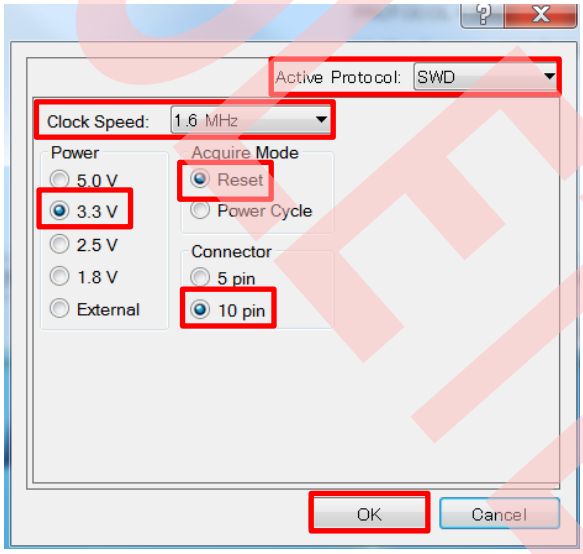
5. Follow the menu path **Debug** > **Program**. If the Select Debug Target window opens, click **Port Setting**.





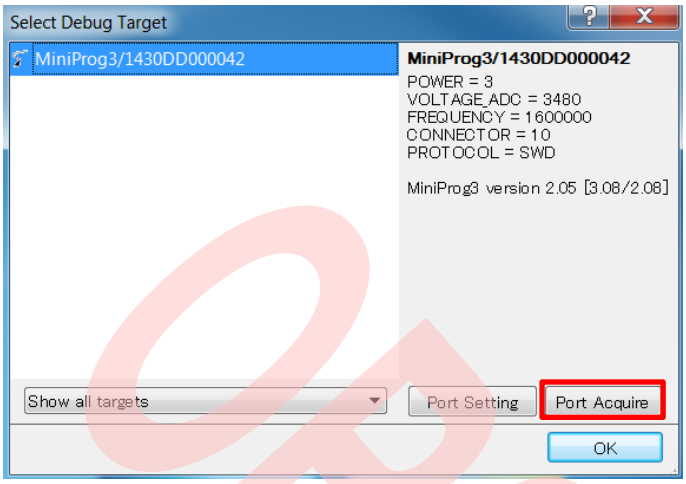
6. 7. In the pop-up window, set the following:
   - ☐ **Active Protocol** = SWD
   - ☐ **Clock Speed** = 1.6 MHz
   - ☐ **Power** = 3.3 V
   - ☐ **Acquire Mode** = Reset
   - ☐ **Connector** = 10 pin

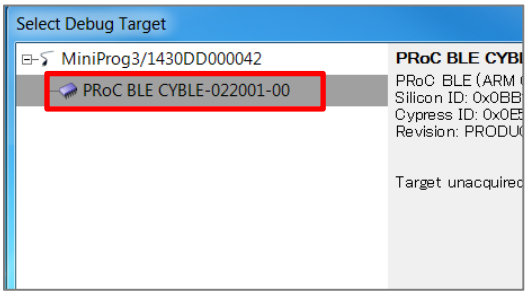
   Click **OK**.



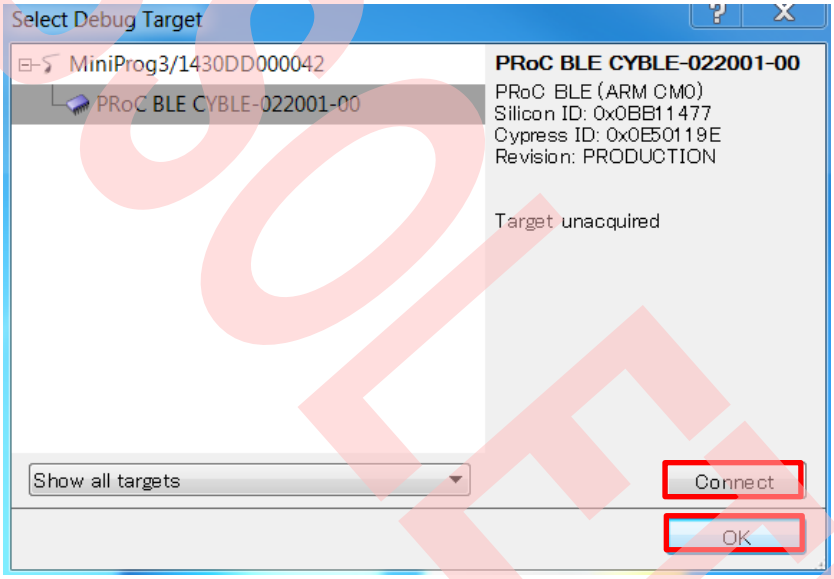
OBSOLETE

7. Click **Port Acquire**. **PRoC BLE CYBLE-022001-00** appears in the Select Debug Target window.



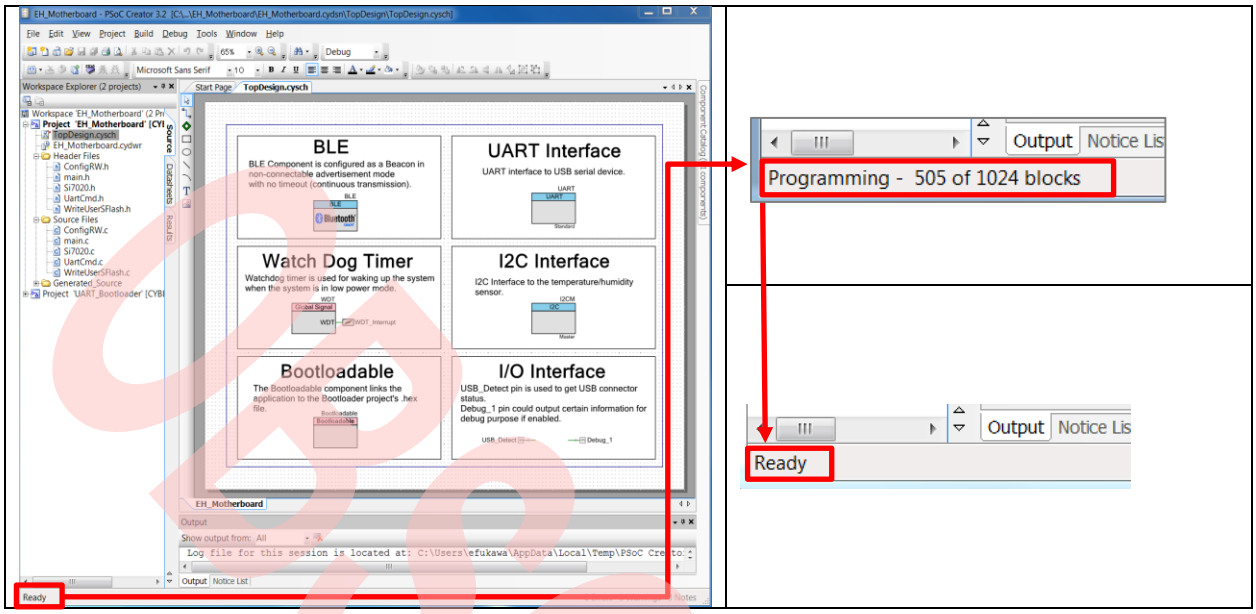


8. Click **Connect**, and then click **OK**.



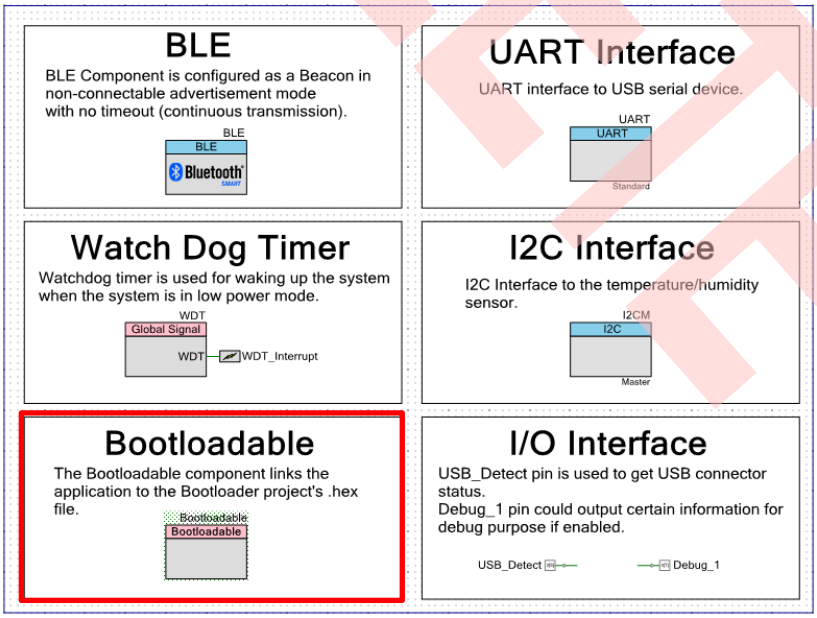
OBSOLETE

9. A "Programming - XXX of 1024 blocks" message will appear on lower left side of PSoC Creator, and then "Ready" appears once programming completes.



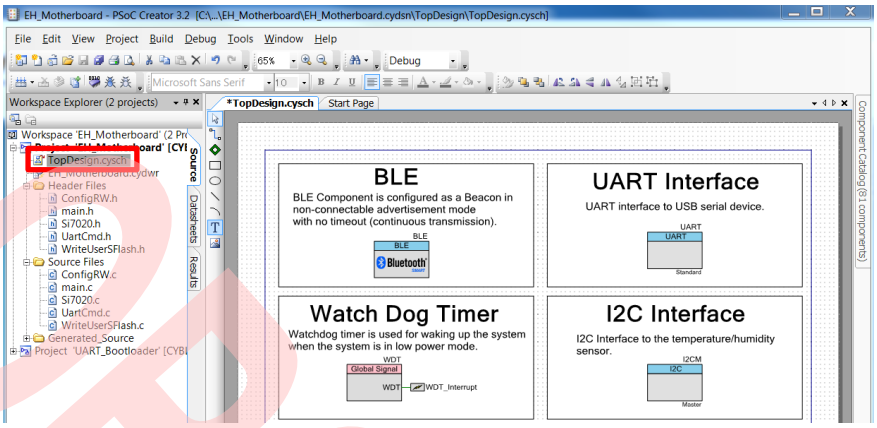
5.2.2 Debug

The sample project for this kit uses a bootloadable component for a UART Boot Loader. In the PSoC Creator bootloader system, the bootloader project executes first (at device reset) and then the bootloadable project. The jump from the bootloader to the bootloadable project is done through a software controlled device reset. This resets the debugger interface, which means that the bootloadable project cannot be run in the debugger mode. To debug a bootloadable project, convert the Application Type to Normal, debug it, and then convert it back to Bootloadable. Another option is to program the Bootloadable project *.hex* file onto the device and then use the **Attach to running target** option for debugging, while the bootloadable project is running. In this case, you can debug the bootloadable project only from the point where debugger is attached to the device. See AN68272 for more details.

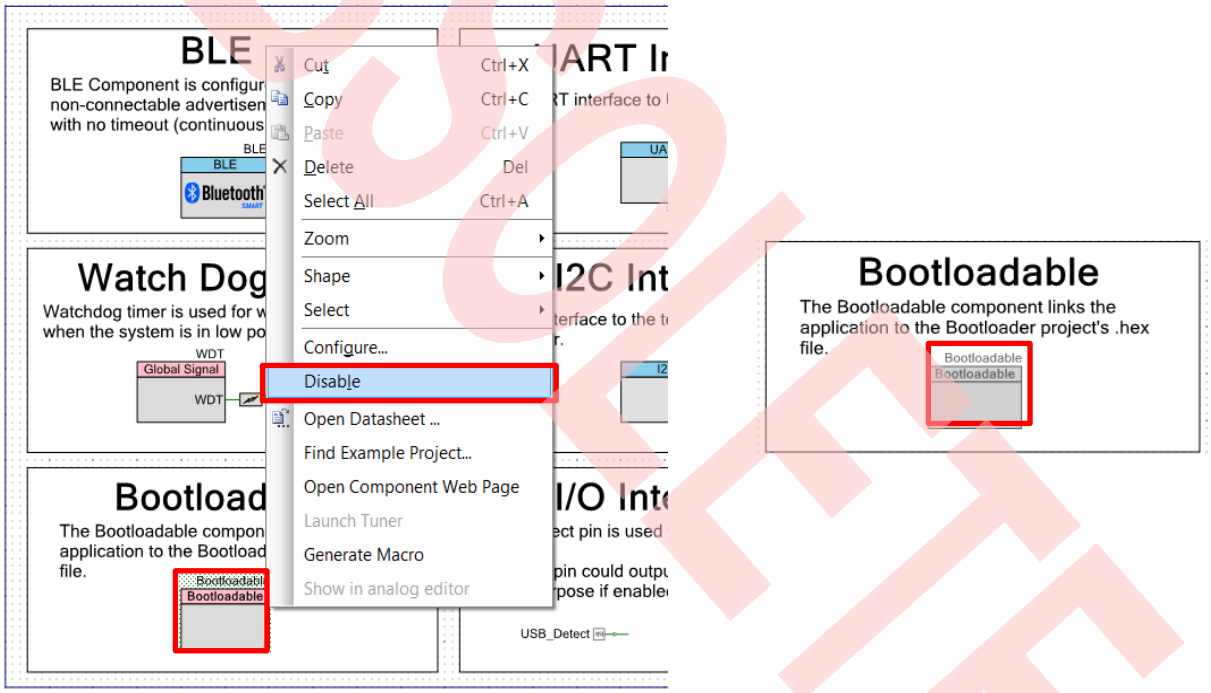


5.2.2.1 Converting Application Type to Normal

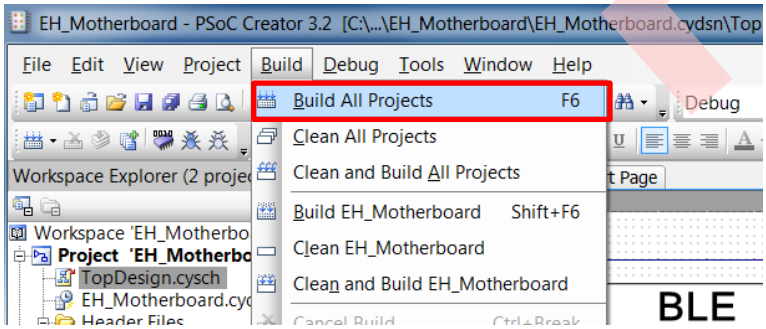
1. Open *TopDesign.cysch*.



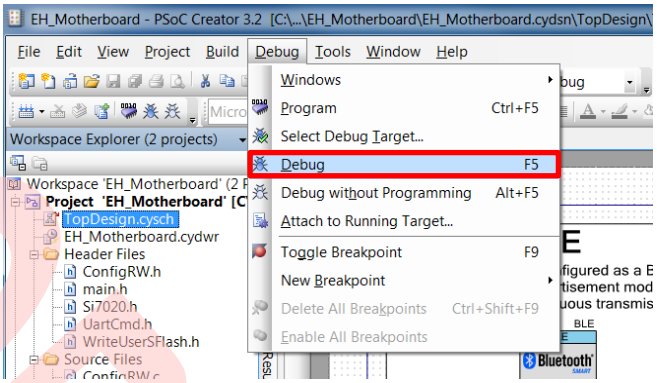
2. Right-click the Bootloadable Component and select **Disable**.



3. Follow the menu path **Build** > **Build All Projects** to rebuild the projects.



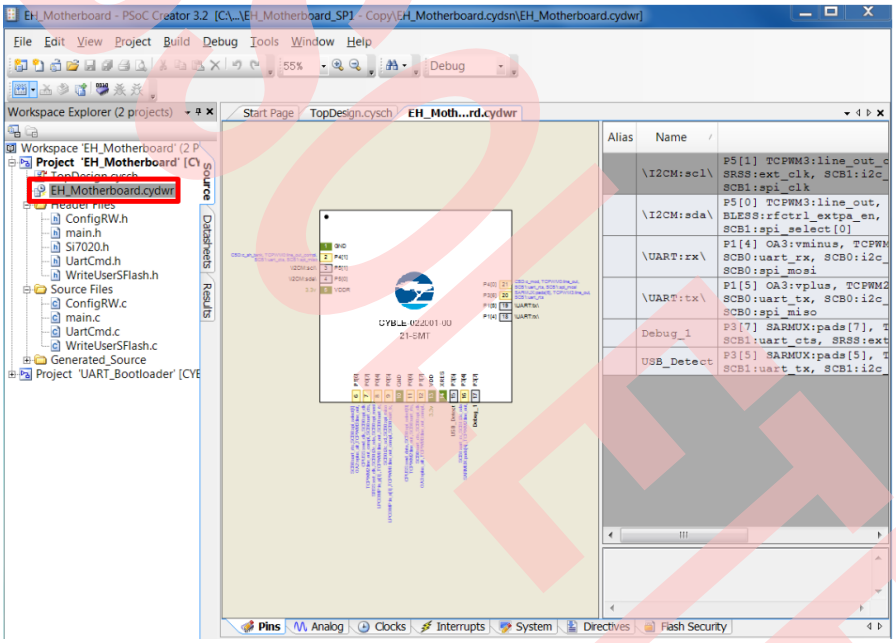
4. Follow the menu path **Debug** > **Debug**. See Help > PSoC. For more information on using the debugger, see **Help** > **PSoC Creator Help Topics** > **Using the Debugger**.



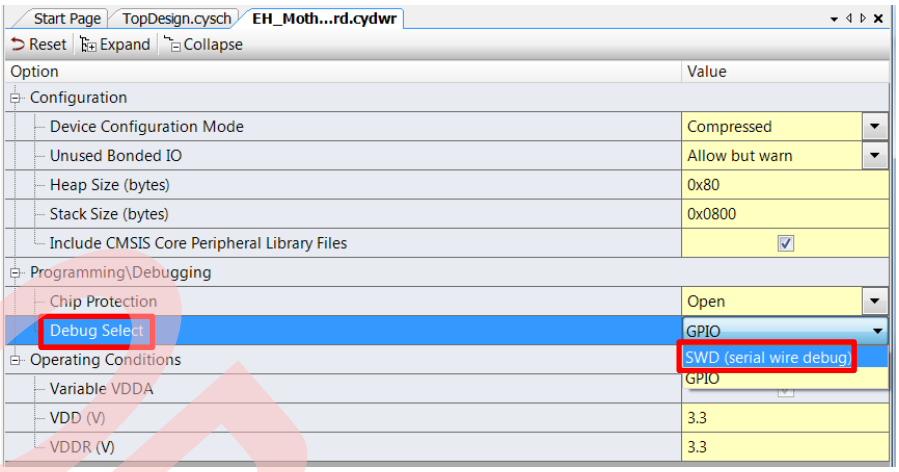
5. After debugging, right-click the Bootloadable Component and select **Enable**. Rebuild and reprogram the project.

5.2.2.2 Using Attach to Running Target Option

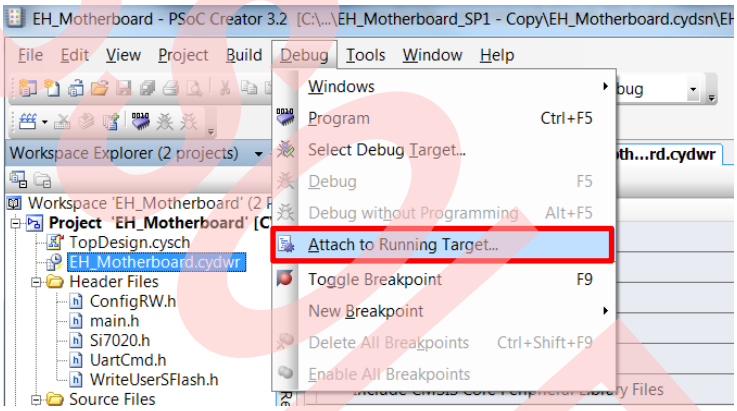
1. Open *EH_Motherboard.cydwr,* and then, go to the **System** tab.



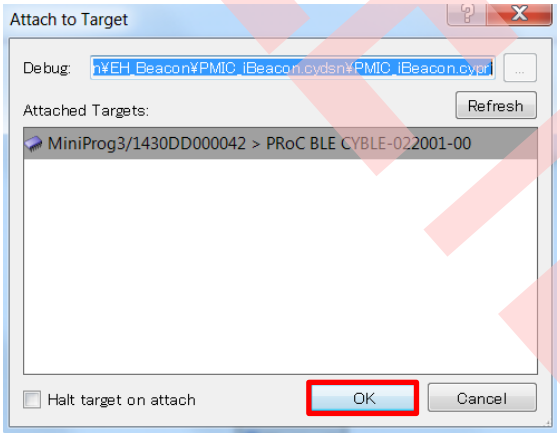
2. To debug, change **Debug Select** to SWD (serial wire debug). However, the consumption current will increase in SWD. So, after debugging, make sure to select **GPIO** to restore the Solar-Powered operation.



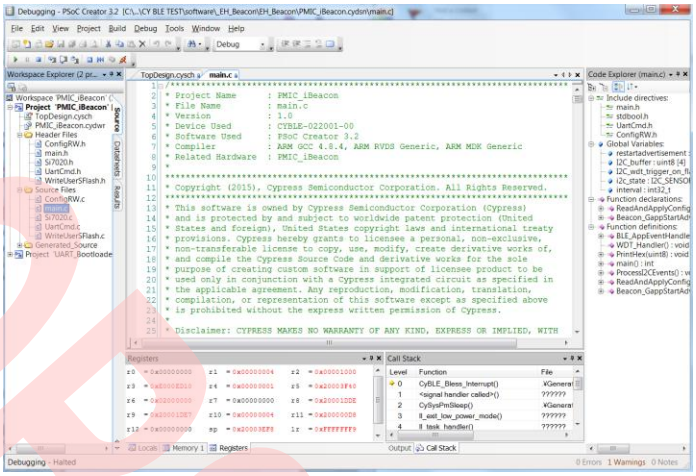
3. Run the Program. See 5.2.1 Program.
4. Follow the menu path **Debug** > **Attach to Running Target…**



5. In the Attach to Target window, click **OK**.



6. The window of PSoC Creator is changed to Debug mode automatically. For more information on using the debugger, see **Help** > **PSoC Creator Help Topics** > **Using the Debugger**.



7. After debugging, change **Debug Select** to **GPIO**. Rebuild and reprogram the project.

OBSOLETE

6 Example Project

This chapter introduces you to the initial provided firmware of the Solar-Powered IoT Device Kit. This section discusses the features such as the BLE Beacon Process and WSN with BLE Beacon Process. See Getting Started with EZ-BLE PRoC Module for standard reference design except energy harvesting.

6.1 Flow Diagram

Figure 6-1 is the flow diagram for the example project of the kit firmware.

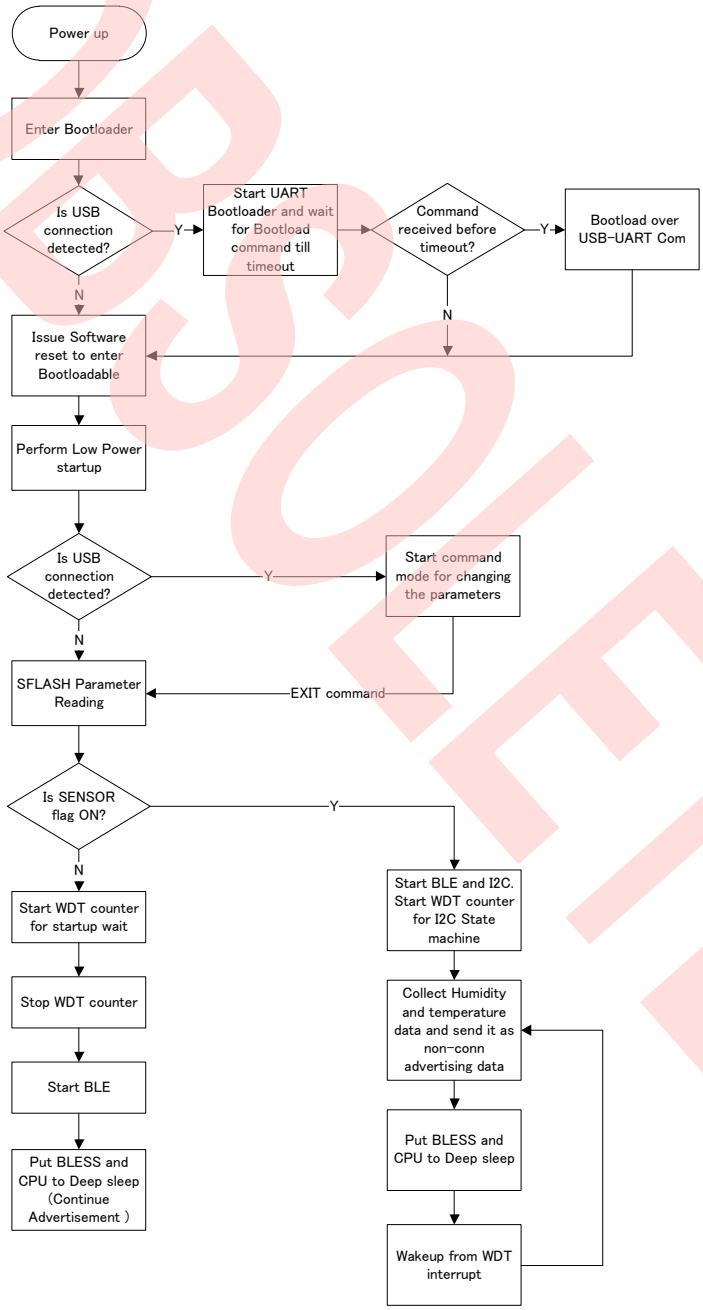


Figure 6-1. Flow of Example Project

OBSOLETE

6.2 Function List

Table 6-1 lists the functions for the example project of the kit firmware.

Function Name	Description
CyDelay	Blocks for milliseconds
CyDelayUs	Blocks for microseconds
USB_Detect_Read	Reads the current value on the pins (P3.5) of the Digital Port in right justified
UART_Start	Invokes SCB_Init() and SCB_Enable()
UART_UartPutString	Places a NULL terminated string in the transmit buffer to be sent at the next available bus time
CyBtldr_Start	Starts the entire bootload operation
Bootloader_SET_RUN_TYPE	Schedules Bootloadable to start after reset
CySoftwareReset	Forces a software reset of the device.
CySysClkWriteEcoDiv	Sets the divider for ECO
CySysClkEcoStart	Starts the External Crystal Oscillator (ECO)
CySysClkEcoStop	Stops the megahertz crystal
WDT_Interrupt_StartEx	Sets up the interrupt and enables
CySysClkWcoStart	Enables Watch Crystal Oscillator (WCO)
CySysClkWcoSetPowerMode	Sets the power mode for the 32 KHz WCO
CySysWdtLocked	Reports the WDT lock state
CySysClkSetLfclkSource	Sets the clock source for the LFCLK clock
CySysWdtEnable	Enables the specified WDT counters
CySysWdtLock	Locks out configuration changes to the Watchdog timer registers and ILO configuration register
CySysWdtUnlock	Unlocks the Watchdog Timer configuration register
CySysPmDeepSleep	Puts the part into the Deep Sleep state
CySysWdtEnable	Enables the specified WDT counters
CySysWdtDisable	Disables the specified WDT counters
ConfigRW_CheckSFlash	Checks whether there is configuration data in User SFlash
cmd_uart_sub	Processes UART commands
CyBle_Start	Initializes the BLE Stack, which consists of the BLE Stack Manager, BLE Controller, and BLE Host modules
CyBle_ProcessEvents	Checks the internal task queue in the BLE Stack, and pending operation of the BLE Stack
CyBle_EnterLPM	Requests the underlying BLE modules such as BLE Controller, BLE Host Stack, and BLE Stack manager to enter one of the supported low power modes
CyEnterCriticalSection	Does not allow interrupts while entering system low power modes
CySysPmDeepSleep	Puts the part into the Deep Sleep state
CySysClkWriteHfclkDirect	Selects the direct source for the HFCLK
CySysClkImoStart	Enables the IMO
CySysClkImoStop	Disables the IMO
CySysPmSleep	Puts the part into the Sleep state
ProcessI2CEvents	Handles I^2C events according to current state and update the state value
ProcessBeaconEvents	Handles Beacon events according to current state and update the state value
Si7020_Init	Initializes Temperature and Humidity sensor Si7020
Si7020_WriteRead	Sends conversion command and read Temperature and Humidity data
Beacon_GappStartAdvertisement	Starts the advertisement using the specified interval
CyBle_GappStopAdvertisement	Exits from discovery mode

Table 6-1. Function List

PRELIMINARY

6.3 BLE Beacon Process

This section explains the BLE Beacon Process in detail.

0. Complete BLE Beacon Process

The complete BLE Beacon process invokes `Beacon_GappStartAdvertisement()`. The waveform in Figure 6-2 shows the BLE Advertisement states.

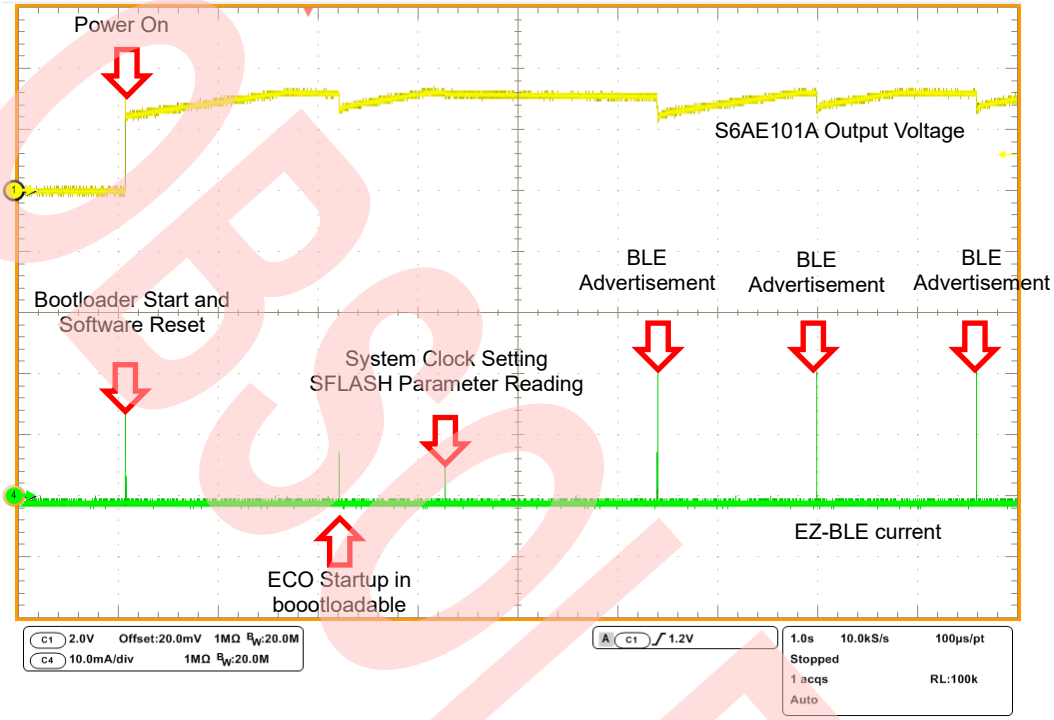


Figure 6-2. BLE Advertisement States

1. Power Up and Bootloader Start

The code starts executing from the bootloader which can be found in the UART_Bootloader project. The code is optimized to read the USB detect line (P3.5 of EZ-BLE) and immediately switch to Bootloadable code if the detect line was not detected.

OBSOLETE

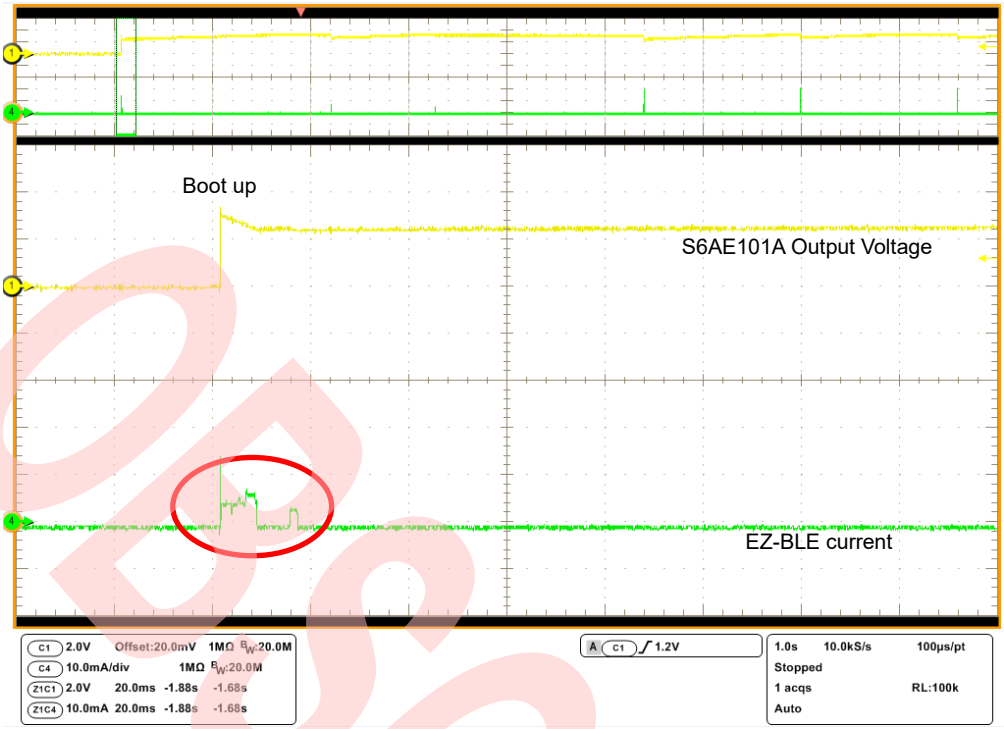


Figure 6-3. Power Up and Bootloader Start

```
CyGlobalIntEnable;
if(USB_DETECT_Read())
{
    UART_Start();
    CyDelay(100u);
    UART_UartPutString("UART Bootloader Starting. It will take 10s.\r\n");

 /* This API does the entire bootload operation. After a succesful
  * bootload operation, this API transfers program control to the
  * new application via a software reset */
    CyBtldr_Start();
}
else
{
  /* Schedule Bootloadable to start after reset */
    Bootloader_SET_RUN_TYPE(Bootloader_START_APP);
    CySoftwareReset();
}
```

OBSOLETE

2. System Initialization and Low Power Startup

Once the bootloader launches the bootloadable application, the code from the EH_Motherboard PSoC Creator project starts. This code begins with low power startup functions. This allows the system to conserve power during clock startup, especially the WCO which takes two seconds for startup. This is crucial to allow the project to survive on solar power.

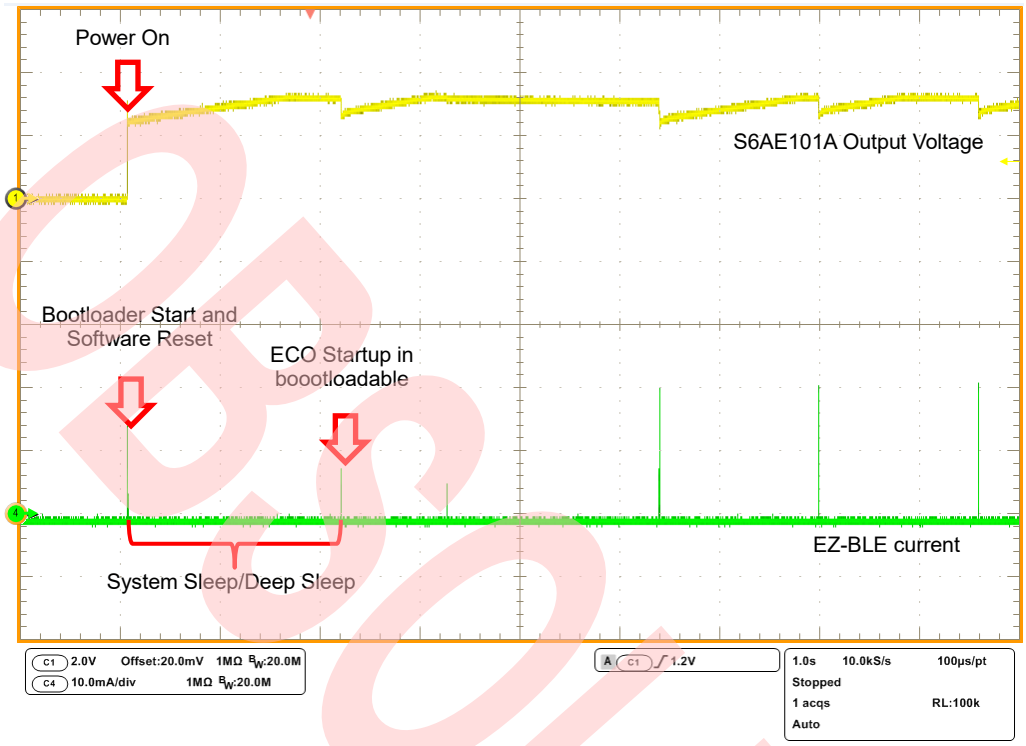


Figure 6-4. System Initialization and Low Power Startup

```
CyGlobalIntEnable;

/* Set the divider for ECO, ECO will be used as source when IMO is switched off to save
power, to drive the HFCLK */

CySysClkWriteEcoDiv(CY_SYS_CLK_ECO_DIV8);


/* If USE_WCO_FOR_TIMING is set, then do the following:

 * 1. Shut down the ECO (to reduce power consumption while WCO is starting)

 * 2. Enable WDT to wakeup the system after 500ms (WCO startup time).

 * 3. Configure PRoC BLE device in DeepSleep mode for the 500ms WCO startup time

 * 4. After WCO is enabled, restart the ECO so that BLESS interface can function */

#if USE_WCO_FOR_TIMING

CySysClkEcoStop();

WDT_Interrupt_StartEx(WDT_Handler);

CySysClkWcoStart();

CySysWdtUnlock(); /* Unlock the WDT registers for modification */

CySysWdtWriteMode(SOURCE_COUNTER, CY_SYS_WDT_MODE_INT);

CySysWdtWriteClearOnMatch(SOURCE_COUNTER, COUNTER_ENABLE);
```

OBSOLETE

```
CySysWdtWriteMatch(SOURCE_COUNTER, COUNT_PERIOD_WCO);
CySysWdtEnable(CY_SYS_WDT_COUNTER0_MASK);
CySysWdtLock();
CySysPmDeepSleep(); /* Wait for the WDT interrupt to wake up the device */
(void)CySysClkEcoStart(2000u);
CyDelayUs(500u);
(void)CySysClkWcoSetPowerMode(CY_SYS_CLK_WCO_LPM);     /* Switch to the low power mode
*/
CySysClkSetLfclkSource(CY_SYS_CLK_LFCLK_SRC_WCO);
CySysWdtUnlock();
CySysWdtDisable(CY_SYS_WDT_COUNTER0_MASK);
CySysWdtLock();
#endif
```

3. System Clock Setting

System is set with IMO at 12 MHz and ECO at 3 MHz.

4. SFLASH Parameter Reading

The user flash read is done at every start of the bootloadable to read the stored beacon data in SFLASH and then used for broadcasting. The flash read is done in `CYBLE_EVT_STACK_ON` to allow some breathing space between system on and flash read.

5. WDT Configuration and BLE Beacon Starting

WDT counter is configured for startup waiting. Then, `ProcessBeaconEvents()` is called to process BLE Beacon events.

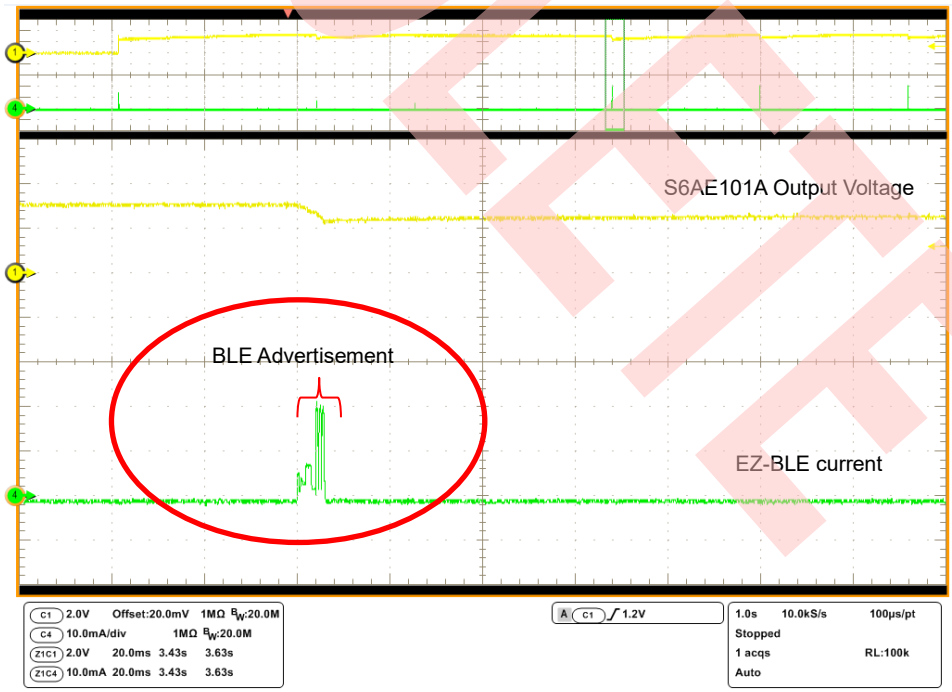


Figure 6-5. WDT Configuration and Starting of BLE Beacon

OBSOLETE

```
void ProcessBeaconEvents(void)
{/* If I2C_COUNTER triggers a new interrupt after another 3 seconds. */
    if(wdt_trigger_on_flag)
    {
        CYBLE_API_RESULT_T apiResult;
         switch(beacon_state)
        {
            case BEACON_WAIT:
       /* Wait two secounds to earn enough power to start advertising */
                beacon_state = BEACON_START;
            break;
            case BEACON_START:
                /* Stop I2C_COUNTER, and start advertisement */
                CySysWdtUnlock();
             /* Unlock the WDT registers for modification */
                CySysWdtDisable(CY_SYS_WDT_COUNTER1_MASK);
                CySysWdtLock();
                apiResult = Beacon_GappStartAdvertisement(interval);
                /* If fails to start advertisement, halt the processor. */
                if(apiResult != CYBLE_ERROR_OK)
                {
                    CYASSERT(0);
                }
                beacon_state = BEACON_RUN;
            break;
            default:
            break;
        }
        wdt_trigger_on_flag = false;
    }
}
```

OBSOLETE

6.4 WSN with BLE Beacon Process

This section provides detail information of WSN with BLE Beacon Process.

0. Complete WSN with BLE Beacon Process

The complete WSN with Beacon process involves running the I²C state machine on each WDT interrupt and to send new data as part of ADV packet. The waveform in Figure 6-6 shows the various states in rotation.

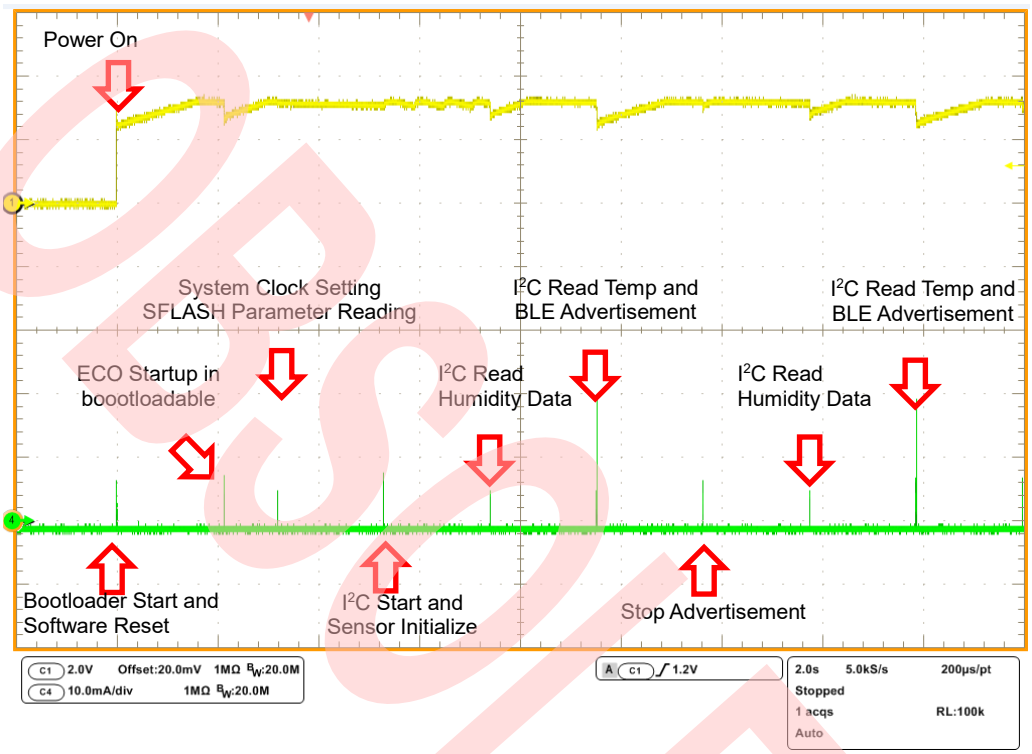


Figure 6-6. Various States of Rotation

1 to 4. Same Process as BLE Beacon

See process 1 to 4 of 6.3 BLE Beacon Process.

5. Components Start and WDT Configuration

Along with BLE start, WDT counter is configured for periodic interrupts to run the I²C state machine. This state machine initiates I²C and reads the data from the sensor to send it as part of the Advisement (ADV) packet. Also, `CyBle_ProcessEvents()` is called to process BLE events.

```
CySysWdtUnlock();
CySysWdtDisable(CY_SYS_WDT_COUNTER0_MASK);
CySysWdtWriteMode(I2C_COUNTER, CY_SYS_WDT_MODE_INT);
CySysWdtWriteClearOnMatch(I2C_COUNTER, COUNTER_ENABLE);
#if I2C_SENSOR_ENABLE
 /* If I2C sensor is enabled and sensor setting is on. */
 if(Sensor_Flag)
 {
   CySysWdtWriteMatch(I2C_COUNTER, I2C_COUNT_PERIOD);
 }
```

OBSOLETE

```
else
#endif
/* If I2C sensor is disabled or sensor setting is off. */
{
  CySysWdtWriteMatch(I2C_COUNTER, ((uint32)32767*3));
}
CySysWdtEnable(CY_SYS_WDT_COUNTER1_MASK);
CySysWdtLock();
```

6. I²C Read Humidity Data

After I2C is started, the next state reads the relative humidity value from I2C sensor and updates the ADV packet with new humidity data. The system then goes back to deep sleep.

Note: Si7020_WriteRead(I2C_buffer, 1, 2); 1 = number of write byte, 2= number of read byte

7. I²C Read Temperature Data and BLE Advertisement

At next WDT interrupt, the system wakes up and enters the I²C temperature state. Here the I2C data for temperature is read and ADV packet is updated with this new data. Also, the GAP Advertisement is started so that the new ADV packet can be transmitted by BLE.

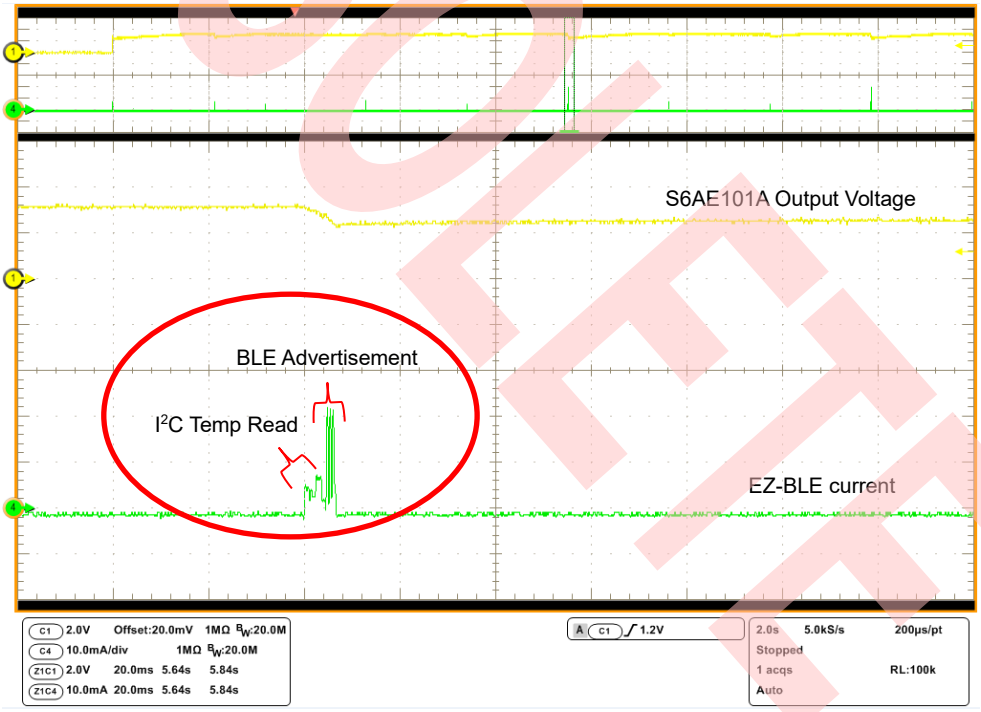


Figure 6-7. I²C Read Humidity Data

```
case I2C_READ_TEMP:
       /* Read Temperature data from I2C Sensor */
       I2C_buffer[2] = SI7020_READ_TEMP;
       Si7020_WriteRead(&I2C_buffer[2], 1, 2);
```

```
        /* Update Temperature index of ADV packet with new value */
        cyBle_discoveryData.advData[ADDR_TEM_OFFSET]= I2C_buffer[2];

         /* When sensor used, advertise interval is fixed to 10.24s. */
        apiResult = Beacon_GappStartAdvertisement(ITRVL_SENSOR_ON);
        /* If fails to start advertisement, halt the processor. */
         if(apiResult != CYBLE_ERROR_OK)
         {
            CYASSERT(0);
         }
         CyBle_ProcessEvents();

        /* Set next I2C state */
        i2c_state = I2C_STOP_ADV;
break;
```

8. Stop Current BLE Advertisement

At the next WDT interrupt, the stop advertisement API is called to stop the current advertisement. This is required before the ADV packet can be updated with new data in next I²C state.

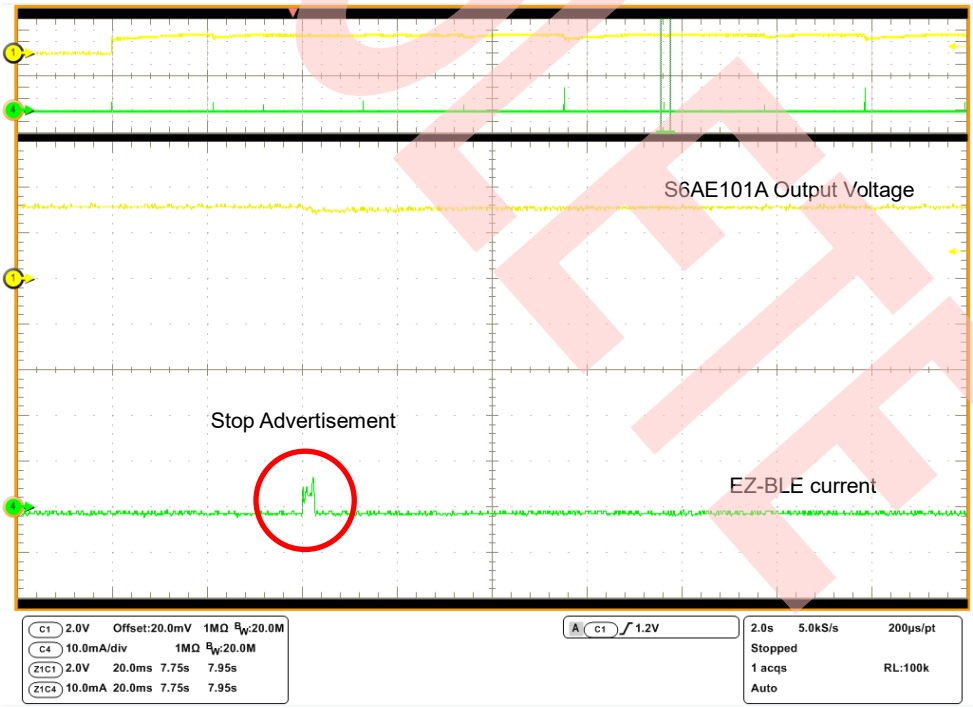


Figure 6-8. Stop Current BLE Advertisement

OBSOLETE

6.5 BLE Beacon Format

BLE Beacon is a one-way communication method that broadcasts at a regular interval. It consists small packets of data (30 bytes) of Advertisements.

Beacons that want to be discovered can be used for a variety of smartphone or computer applications to trigger push messages, app actions, and prompts.

Following is a link layer format of BLE for Advertising channel packet format. The link layer of BLE has Preamble, Access Address, Protocol Data Unite (PDU), and CRC. Note that following information are for Advertising channel packets format; the information does not include Data channel packets.

- "Preamble" must set "10101010b"
- "Access Address" must set "10001110100010011011111011010110b (0x8E89BED6)"
- "PDU" has "Header" and "Payload"

The packet structure of BLE Beacon belongs to "Advertising Data" in "Payload".

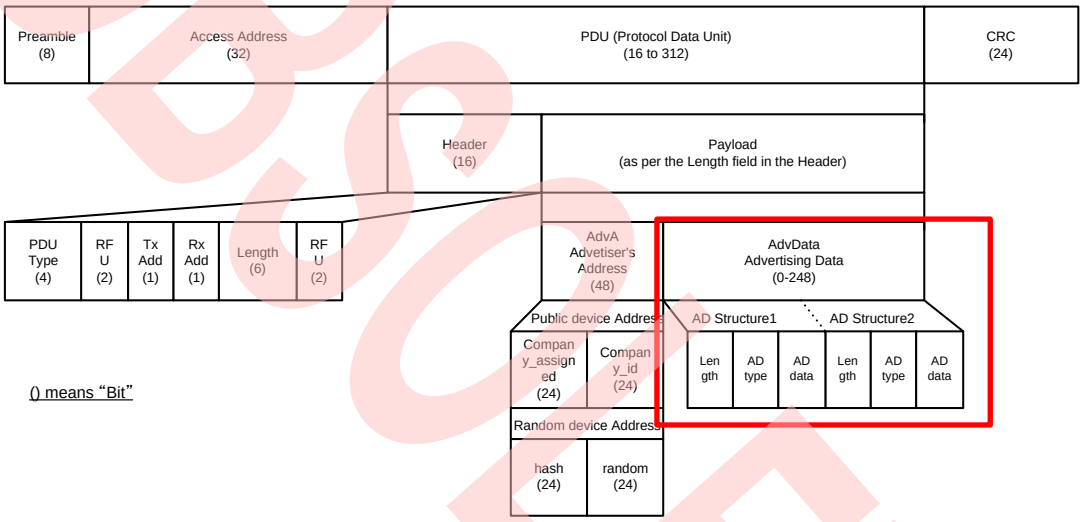


Figure 6-9.BLE Packet Format

Figure 6-10 shows the detail of BLE Beacon packet format for the kit.

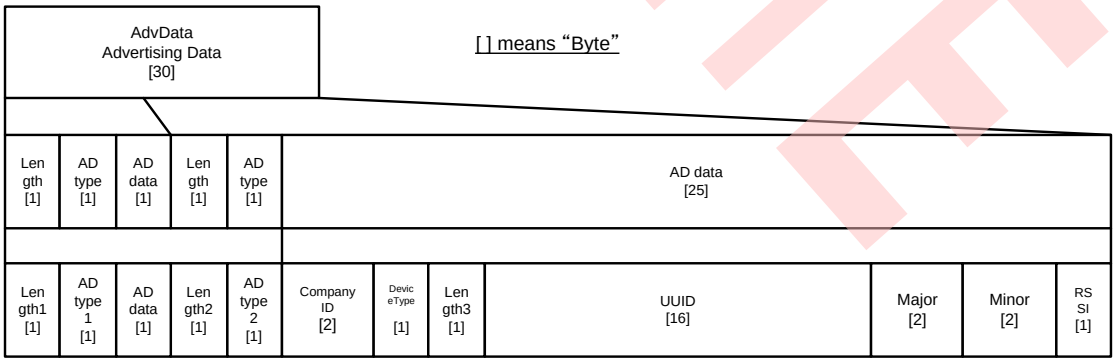


Figure 6-10. BLE Beacon Packet Format

Table 6-2 lists the initial values in the example project for Solar-Powered BLE Beacon Operation.

<BLE Beacon Format>	
Length	0x02
AD type 1	0x01
AD data	0x04
Length2	0x1A
AD type 2	0xFF
Company ID	0x0131 [Cypress Semiconductor Corporation]
Device type	0x02
Length3	0x15
UUID[7]	00050001-0000-1000-8000-00805F9B0131 [hex]
Major[8]	0x0001
Minor[9]	0x0001
RSSI[10]	0xC3 [-61dBm]
<Others>	
Transmitter power	3 dBm
Advertise interval	1500 ms

Table 6-2. Initial Values

The kit uses `ReadAndApplyConfig` function in *main.c* to update the BLE Beacon packet. Note that the kit does not use GAP Settings of Configure 'BLE' on *TopDesign.cysch*, because the `SFLASH` parameter is read and the BLE Beacon packet is updated when the Motherboards turns the power ON.

[7] This is a 16-byte string used to differentiate a large group of related beacons.

[8] This is a 2-byte string used to distinguish a smaller subset of beacons within the larger group.

[9] This is a 2-byte string meant to identify individual beacons.

[10] Received Signal Strength Indication. This is used to determine proximity (distance) from the beacon.

OBSOLETE

6.6 Sensor Transmitter Specification of WSN

When the “SENSOR” command is ON, the kit transmits the sensor data in the Beacon protocol packet where temperature and humidity sensor information are stored in the Minor region (2 bytes). Humidity data is stored in the upper byte and temperature data is stored in the lower byte.

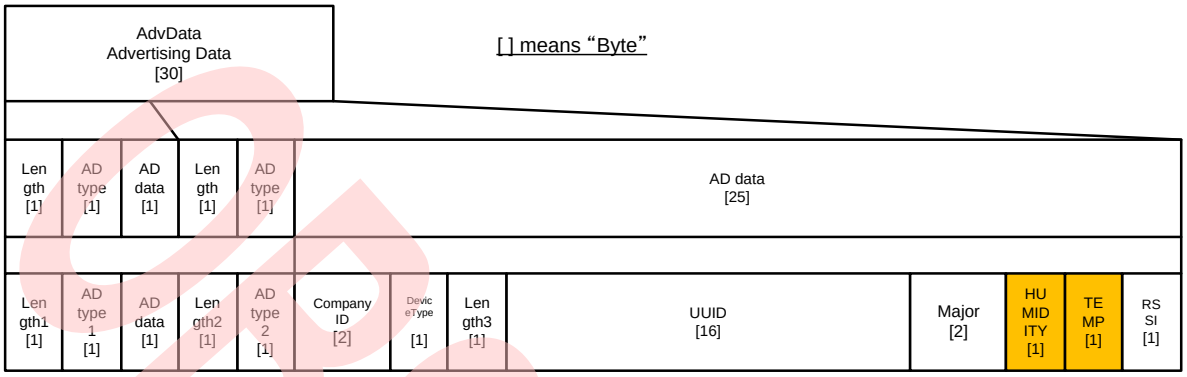


Figure 6-11. BLE Beacon Packet Format with Sensor Information

Meanwhile, whether temperature or humidity sensor information is written on Minor region is set by changing the ON/OFF status with the "SENSOR" command as described in 4.3 Serial Command List. If you set the SENSOR OFF, the board will send data based on the Beacon protocol excluding sensor information.

OBSOLETE

7 Energy Harvesting PMIC (S6AE101A)

This section provides the specification of the Energy Harvesting Power Management IC (S6AE101A) on the Energy Harvesting Motherboard. See the datasheet of S6AE101A (DS405-00026) for the latest information.

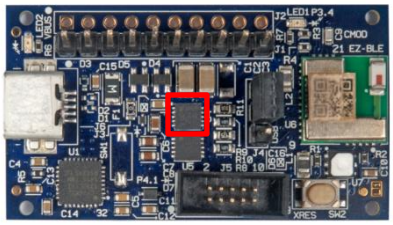


Figure 7-1. S6AE101A Energy Harvesting PMIC

7.1 Recommended Operating Conditions

Parameter	Symbol	Condition	Value			Unit
			Min	Typ	Max	
Power supply voltage 1 (*1)	V_{VDD}	VDD pin	2.0	3.3	5.5	V
Power supply voltage 2 (*1)	V_{VBAT}	VBAT pin	2.0	3.0	5.5	V
VOUT1 setting resistance	R_{VOUT}	Sum of R1, R2, R3	10	–	–	MΩ
VDD capacitance	C1	VDD pin	10	–	–	µF
VINT capacitance	C2	VINT pin	1	–	–	µF
VOUT maximum setting voltage	V_{SYSH}	VSTORE1 pin	1.25	–	5.2	V
VOUT minimum setting voltage	V_{SYSL}	VSTORE1 pin	1.1	–	V_{SYSH} × 0.9	V
Operating ambient temperature	Ta	–	−40	–	+85	°C

*1: When GND = 0V

Table 7-1. Recommended Operating Conditions

7.2 DC Characteristics

Parameter	Symbol	Condition	Value			Unit
			Min	Typ	Max	
Minimum Input power in start-up	W_{START}	VDD pin, Ta = +25°C, V_{VOUTH} setting =3V, By applying 0.4µA to VDD, when VOUT1 reaches 3V×95% after the point when VDD reaches 3V.	–	–	1.2	µW
Consumption current	I_{QIN1}	VDD pin input current, VDD=3V, Open VBAT pin, V_{VOUTH}=1.25V setting, Ta=+25°C, SET_VOUTFB resistance>100MΩ, VOUT1 Load=0mA	–	250	375	nA
OVP detection voltage	V_{OVPH}	VDD pin	5.2	5.4	5.5	V
OVP release voltage	V_{OVPL}		5.1	5.3	5.4	V
OVP detection hysteresis	V_{OVPHYS}		–	0.1	–	V
OVP protection current	I_{OVP}	VDD pin input current	6	–	–	mA

Table 7-2. DC Characteristics

OBSOLETE

8 Hardware

8.1 Energy Harvesting Motherboard

8.1.1 Board Detail

The Energy Harvesting Motherboard consists of the following blocks shown in Figure 8-1:

- Energy Harvesting Power Management IC (S6AE101A)
- EZ-BLE PRoC Module (CYBLE-022001-00)
- USB Serial Converter IC (CY7C65213)
- LDO for USB bus power (MB39C022G)
- Temperature and Humidity Sensor (Si7020-A10)
- Diode Bridge for the vibration energy input (1SS383 x2pcs)
- USB mini-B connector
- Jumpers with socket to select power supply
- MiniProg3 programing header for EZ-BLE
- 10-pin expansion header for Energy Harvesting Power Management IC
- 10-pin expansion header for EZ-BLE
- USB power LED, Status LED
- Push switch for EZ-BLE reset
- Pads of DIP switch for future expansion (optional)
- Pads for BK-885-coin battery holder (optional)
- Pads for AM-1801 or BCS4630B9 solar cell (optional)

OBSOLETE

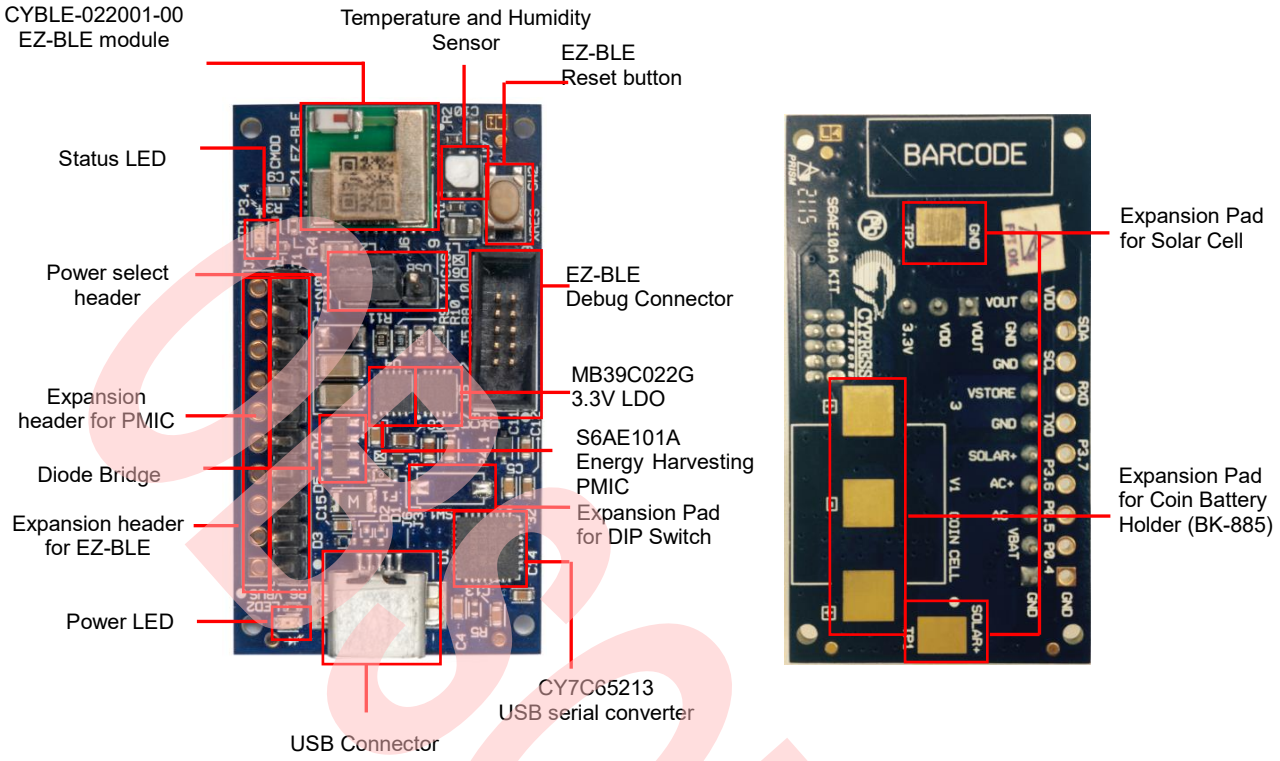


Figure 8-1. Energy Harvesting Motherboard

8.1.2 Input/Output Pin Description

Circuit Pin No.	Silk-Printed Name	I/O	Description
Expansion pins for S6AE101A Energy Harvesting PMIC (J1)			
J1-1	GND	-	GND pin
J1-2	VBAT	I	Primary battery input pin for Hybrid operation
J1-3	AC-	I	AC- voltage input for vibration energy
J1-4	AC+	I	AC+ voltage input for vibration energy
J1-5	SOLAR+	I	Solar cell input
J1-6	GND	-	GND pin
J1-7	VSTORE	O	Storage output pin
J1-8	GND	-	GND pin
J1-9	GND	-	GND pin
J1-10	VOUT	O	Output voltage pin
Expansion pins for EZ-BLE PRoC Module (J2)			
J2-1	GND	-	GND pin
J2-2	P0.4	I/O	GPIO P0.4 of EZ-BLE
J2-3	P0.5	I/O	GPIO P0.5 of EZ-BLE
J2-4	P3.6	I/O	GPIO P3.6 of EZ-BLE
J2-5	P3.7	I/O	GPIO P3.7 of EZ-BLE
J2-6	TXD	O	UART TXD of USB serial (RXD of EZ-BLE)
J2-7	RXD	I	UART RXD of USB serial (TXD of EZ-BLE)
J2-8	SCL	O	I^2C clock pin of EZ-BLE
J2-9	SDA	I/O	I^2C data pin of EZ-BLE
J2-10	VDD	-	Power Supply Input

Table 8-1. Input/Output Pin Description

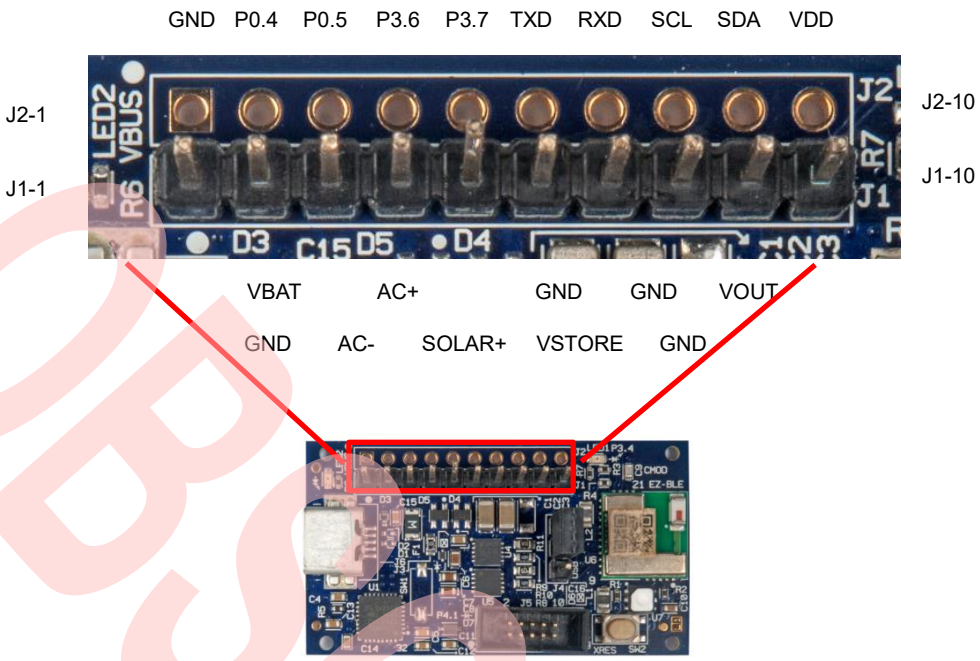


Figure 8-2.Pin Arrangement

8.1.3 Debug Connector Description

Circuit Pin No.	Silk-Printed Name	I/O	Description
J5-1	VDD	I	Power pin
J5-2	SWDIO	I/O	SWD data pin
J5-3	GND	-	GND pin
J5-4	SWDCLK	I/O	SWD clock pin
J5-5	GND	-	GND pin
J5-6	N.C	-	Non-connection
J5-7	GND	-	GND pin
J5-8	N.C	-	Non-connection
J5-9	GND	-	GND pin
J5-10	XRES	I	Reset pin of EZ-BLE

Table 8-2. Debug Connector Description

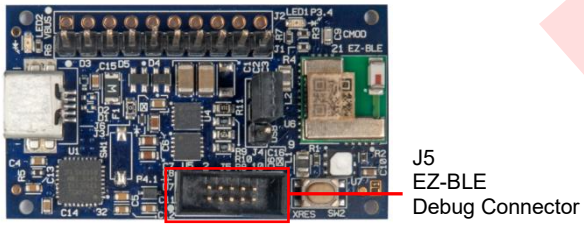


Figure 8-3. Position of Debug Connector

OBSOLETE

8.1.4 Jumper Description

Circuit Pin No.	Description	Default Settings
J3	0Ω resistor for bridge rectifier of vibration harvester	Short
J4	3pin jumper with socket for power source select of EZ-BLE EH: 3.3V to 1.9V Energy Harvesting PMIC VOUT USB: 3.3V LDO output via USB bus power	EH

Table 8-3. Jumper Description

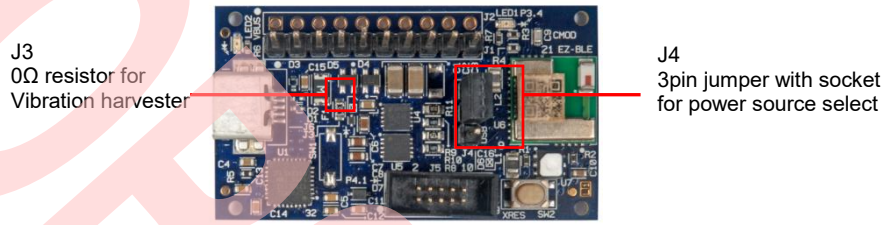


Figure 8-4. Position of Jumper

8.1.5 Switch Description

Circuit Pin No.	Silk-Printed Name	Description
SW1	SW1 (Not mounted)	DIP switch connection to GPIO P4.1 of EZ-BLE PRoC Module for future expansion
SW2	XRES SW2	Reset Button for BLE module

Table 8-4. Switch Description

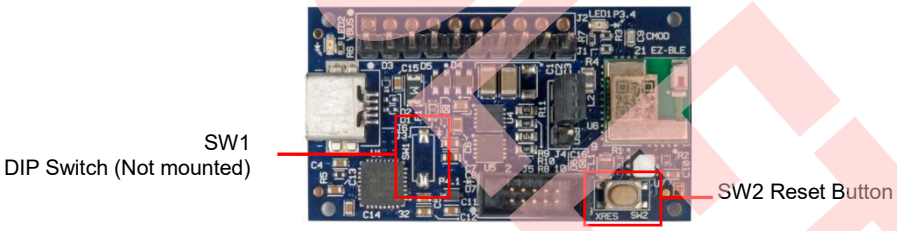


Figure 8-5. Position of Switch

8.1.6 LED Description

Circuit Pin No.	Silk-Printed Name	Description
LED1	LED1 P3.4	Status LED (GPIO P3.4 of EZ-BLE)
LED2	LED2 VBUS	USB power LED

Table 8-5. LED Description

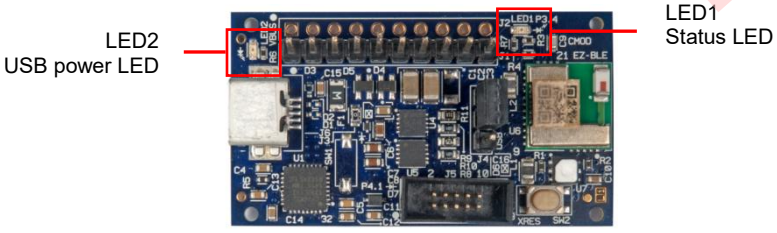
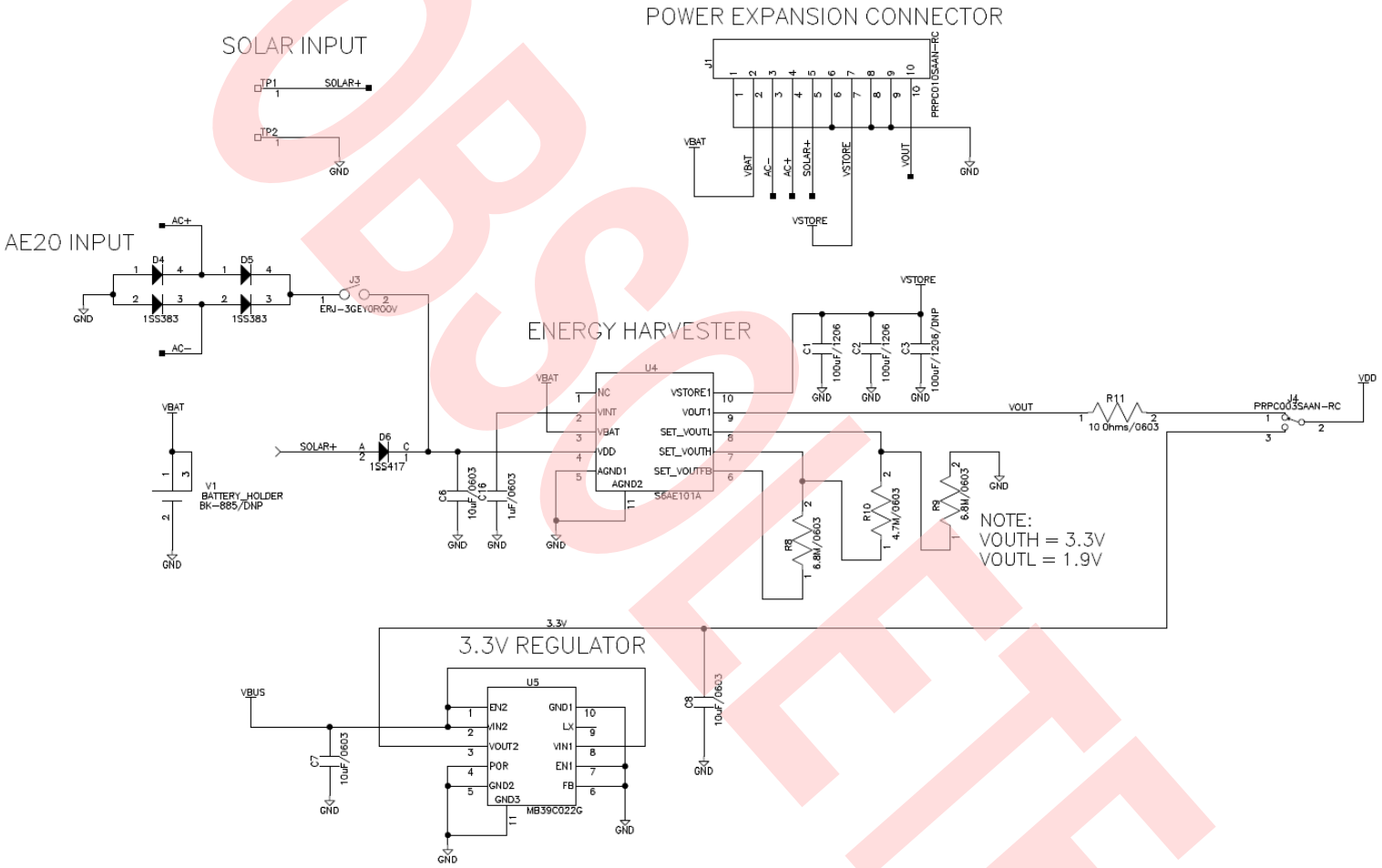
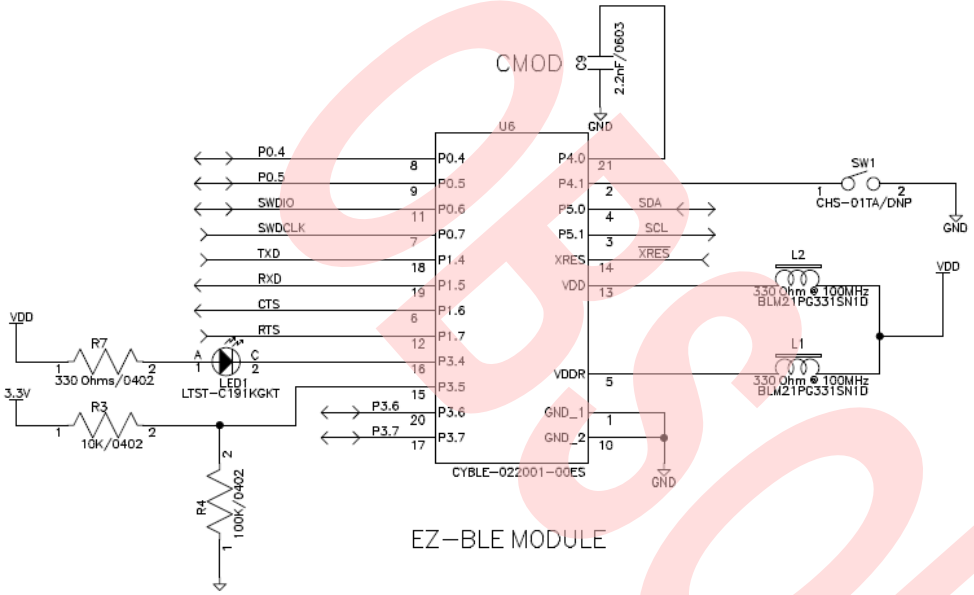


Figure 8-6. Position of LED

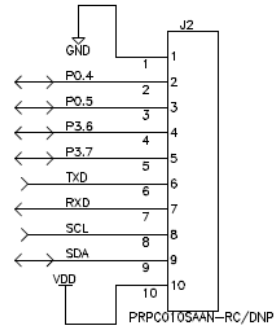
Obsolete

8.1.7 Circuit

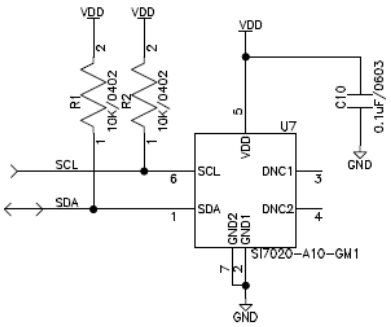

SOLAR INPUT
POWER EXPANSION CONNECTOR
AE20 INPUT
ENERGY HARVESTER
3.3V REGULATOR
NOTE:
VOUTH = 3.3V
VOUTL = 1.9V
OBSOLETE



EZ-BLE MODULE

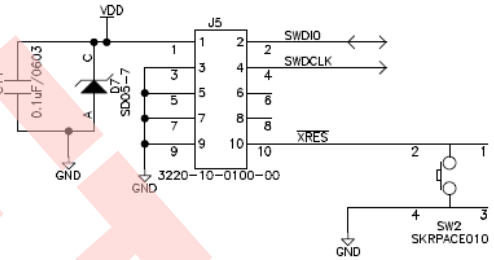


EZ-BLE EXPANSION CONNECTOR



TEMPERATURE & HUMIDITY SENSOR

EZ-BLE PROGRAM/DEBUG CONNECTOR



OBSOLETE

8.1.8 BOM List

No	Qty	Reference	Parts Number	Description	Manufacture	Note
1	2	C1, C2, (C3)	GRM31CR60J107ME39K	100 µF/1206	TDK Corporation	C3 Non-mount
2	3	C4, C5, C16	GRM188R61C105KA93D	1 µF/0603	TDK Corporation	
3	3	C6, C7, C8	GRM188R61C106MA73D	10 µF/0603	TDK Corporation	
4	1	C9	GRM1885C1H222JA01D	2.2 nF/0603	TDK Corporation	
5	6	C10, C11, C12, C13, C14, C15	GRM188R71C104KA01D	0.1 µF/0603	TDK Corporation	
6	3	D1, D2, D3	CG0402MLC-05LG	TVS/0402	Bourns, Inc.	
7	2	D4, D5	1SS383	DUAL DIODE	TOSHIBA CORPORATION	
8	1	D6	1SS417	DIODE	TOSHIBA CORPORATION	
9	1	D7	SD05-7	DIODE	Diodes Incorporated	
10	1	F1	1206L050/15YR	FUSE 500mA	Littelfuse	
11	2	L1, L2	BLM21PG331SN1D	330 Ω @ 100MHz	Murata Manufacturing Co., Ltd.	
12	2	LED1, LED2	LTST-C191KGKT	GREEN	Lite-On Inc	
13	1	U1	CY7C65213-32LTXI	USB-UART LP Bridge Controller	Cypress Semiconductor Corporation	
14	1	U4	S6AE101A	Energy Harvesting Power Management IC	Cypress Semiconductor Corporation	
15	1	U5	MB39C022G	1ch DCDC, 1ch LDO	Cypress Semiconductor Corporation	
16	1	U6	CYBLE-022001-00ES	Bluetooth Low Energy Module	Cypress Semiconductor Corporation	
17	1	U7	SI7020-A10-GM1	Temp & Humidity Sensors	Silicon Labs	
18	3	R1, R2, R3	ERJ-2GEJ103X	10 kΩ/0402	Panasonic Corporation	
19	1	R4	ERJ-2GEJ104X	100 kΩ/0402	Panasonic Corporation	
20	1	R5	ERJ-2GEJ472X	4.7 kΩ/0402	Panasonic Corporation	
21	1	R6	ERJ-2GEJ471X	470Ω/0402	Panasonic Corporation	
22	1	R7	ERJ-2GEJ331X	330Ω/0402	Panasonic Corporation	
23	2	R8, R9	CRCW06036M80FKEA	6.8 MΩ/0603	Vishay Dale	
24	1	R10	RC0603FR-074M7L	4.7 MΩ/0603	Yageo	
25	1	R11	MCR03ERTF10R0	10Ω/0603	Rohm Semiconductor	
26	0	(SW1)	CHS-01TA	1pin DIP Switch	Copal Electronics Inc	Non-mount
27	1	SW2	SKRPACE010	Push Switch	ALPS ELECTRIC CO., LTD	
28	0	(V1)	BK-885	Coin Battery Holder	-	Non-mount
29	1	J1, (J2)	PRPC010SAAN-RC	2.54mm 10pin Header	Sullins Connector Solutions	J2 Non-mount

CYPRESS CONFIDENTIAL

No	Qty	Reference	Parts Number	Description	Manufacture	Note
30	1	J3	ERJ-3GEY0R00V	0Ω/0603	Panasonic Corporation	
31	1	J4	PRPC003SAAN-RC	2.54mm 3pin Header with socket	Sullins Connector Solutions	
32	1	J5	3220-10-0100-00	1.27mm 10pin Box Header	CNC Tech	
33	1	J6	UX60SC-MB-5ST	USB Mini Connector	Hirose Electric Co., Ltd	
34	2	TP1, TP2	TEST_PAD_3.0x4.0	-	-	

8.2 BLE-USB Bridge

8.2.1 Board Detail

The BLE-USB Bridge consists of the blocks shown in Figure 8-7. The initial firmware programmed into the BLE-USB Bridge does not support the CySmartTM Software Utility. Instead, the firmware is a custom version used for demonstrating the features of this kit.

- CYBL10162-56LQXI PRoC BLE device
- CY8C5868LTI-LP039 PSoC 5LP programmer and debugger
- Antenna matching network (AMN)
- Wiggle antenna
- 24-MHz crystal
- 32.768-kHz Crystal (bottom side)
- PRoC BLE external programming header
- PSoC 5LP Programming test points
- PRoC BLE reset button
- User button
- Power LED
- Status LED
- User LED
- Pads for P1_4 (PRoC BLE)
- Pads for P1_5 (PRoC BLE)
- USB plug

OBSOLETE

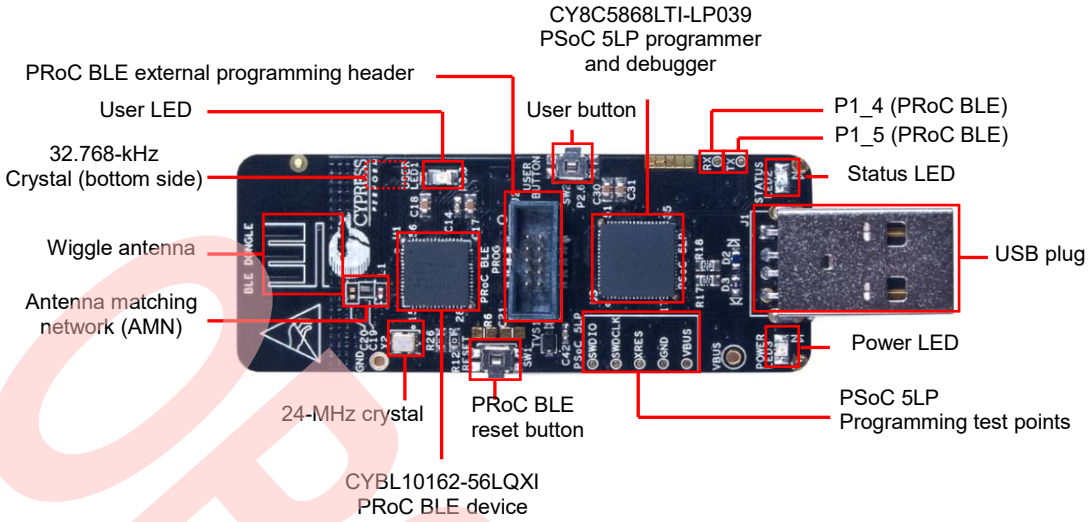


Figure 8-7. BLE-USB Bridge

8.2.2 Test Pin Description

Circuit Pin No.	Circuit Name	I/O	Description
TP1	BLE_TEST1	I/O	GPIO P3.0 of PRoC BLE
TP2	BLE_TEST2	I/O	GPIO P0.4 of PRoC BLE
TP3	VDDA	-	Power Pin of PRoC BLE
TP4	VDDR	-	Power Pin of PRoC BLE
TP5	VDDD	-	Power Pin of PRoC BLE
TP6	VBUS	-	Power Pin of USB Bus
TP7	GND	-	GND pin
TP8	P5LP_SWDIO	I/O	SWD data pin of PSoC 5LP
TP9	P5LP_SWDCLK	I/O	SWD clock pin of PSoC 5LP
TP10	GND	-	GND pin
TP11	P5LP_XRES	I	Reset pin of PSoC 5LP
TP12	P5LP15_4	I/O	GPIO P15_4 of PSoC 5LP
TP13	VBUS	-	Power pin of USB Bus
TP14	BLE_TX	O	UART TXD of USB serial (RXD of PSoC 5LP)
TP15	BLE_RX	I	UART RXD of USB serial (TXD of PSoC 5LP)

Table 8-6. Test Pin Description

8.2.3 Switch Description

Circuit Pin No.	Silk-Printed Name	Description
SW1	RESET	Hardware Reset Button
SW2	USER BUTTON	User Button of PRoC BLE

Table 8-7. Switch Description

OBSOLETE

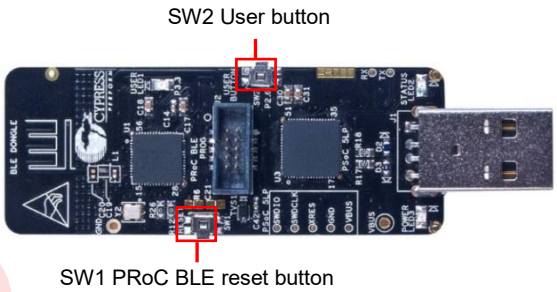


Figure 8-8. Position of Switch

8.2.4 LED Description

Circuit Pin No.	Silk-Printed Name	Description
LED1	USER LED1	GPIO P3.3 of PRoC BLE
LED2	STATUS LED2	GPIO P3_1 of PSoC 5LP
LED3	POWER LED3	Power LED of PSoC 5LP

Table 8-8. LED Description

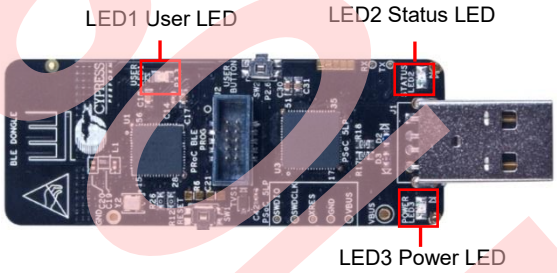
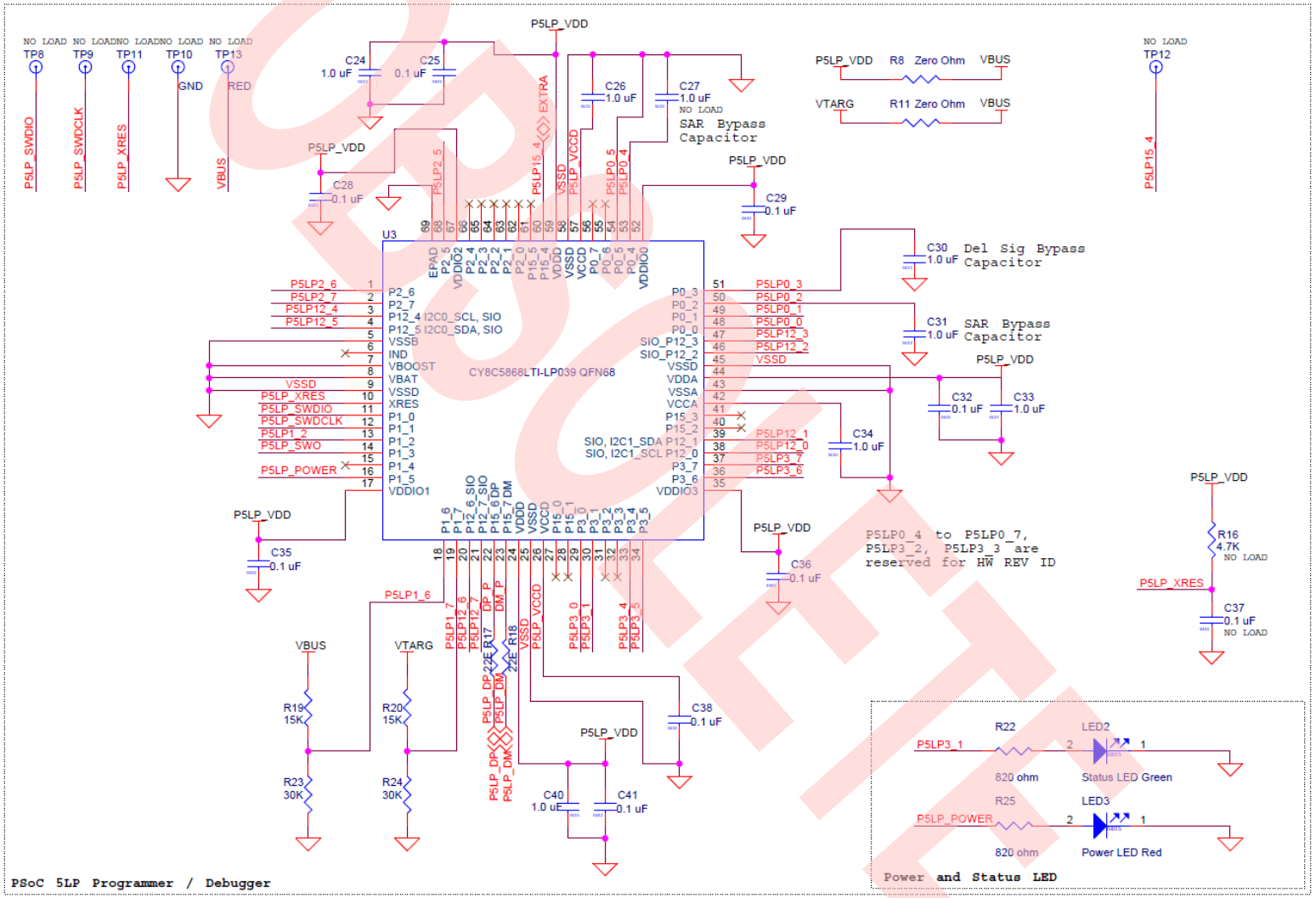
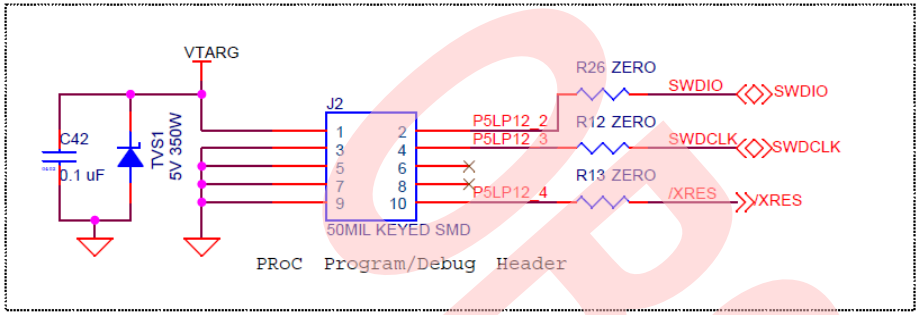


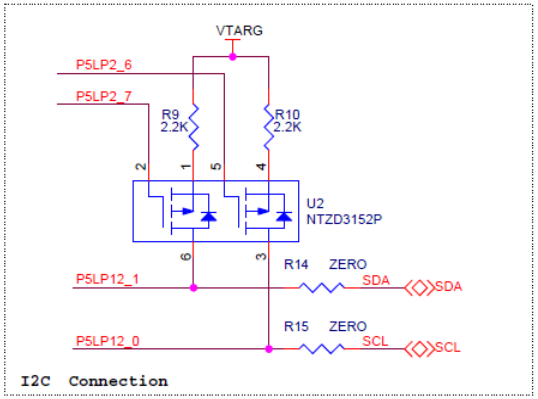
Figure 8-9. Position of LED

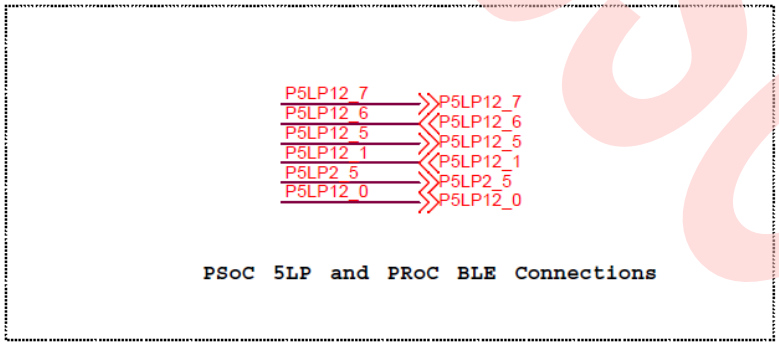
OBSOLETE

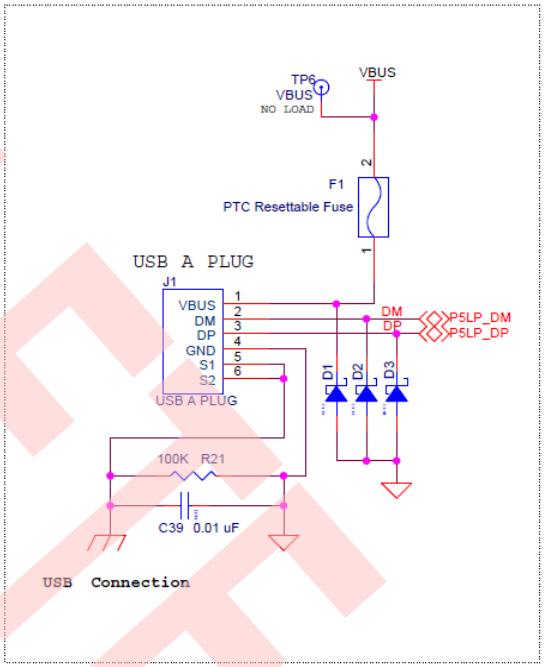
8.2.5 Circuit

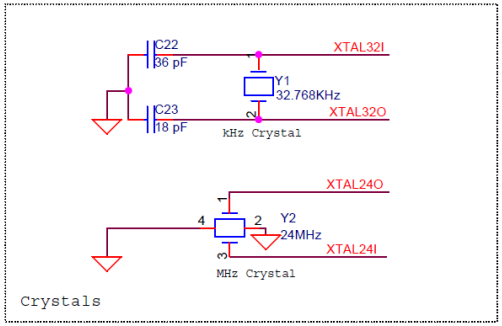

P5LP_VDD
CY8C5868LTI-LP039 QFN68
U3
SAR Bypass Capacitor
Del Sig Bypass Capacitor
SAR Bypass Capacitor
P5LP0_4 to P5LP0_7, P5LP3_2, P5LP3_3 are reserved for HW REV ID
PSoC 5LP Programmer / Debugger
Power and Status LED
Status LED Green
Power LED Red

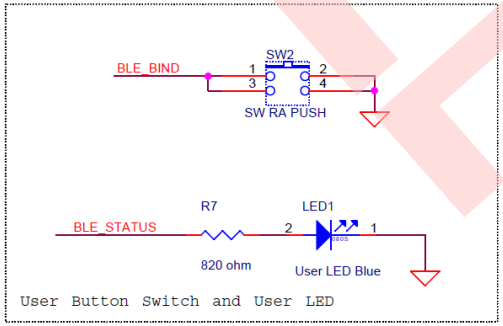

VTARG
C42
0.1 uF
TVS1
5V 350W
J2
R26 ZERO
R12 ZERO
R13 ZERO
P5LP12_2
P5LP12_3
P5LP12_4
SWDIO
SWDCLK
/XRES
SWDIO
SWDCLK
XRES
50MIL KEYED SMD
PRoC Program/Debug Header


VTARG
P5LP2_6
P5LP2_7
R9
2.2K
R10
2.2K
U2
NTZD3152P
P5LP12_1
P5LP12_0
R14 ZERO
R15 ZERO
SDA
SCL
I2C Connection

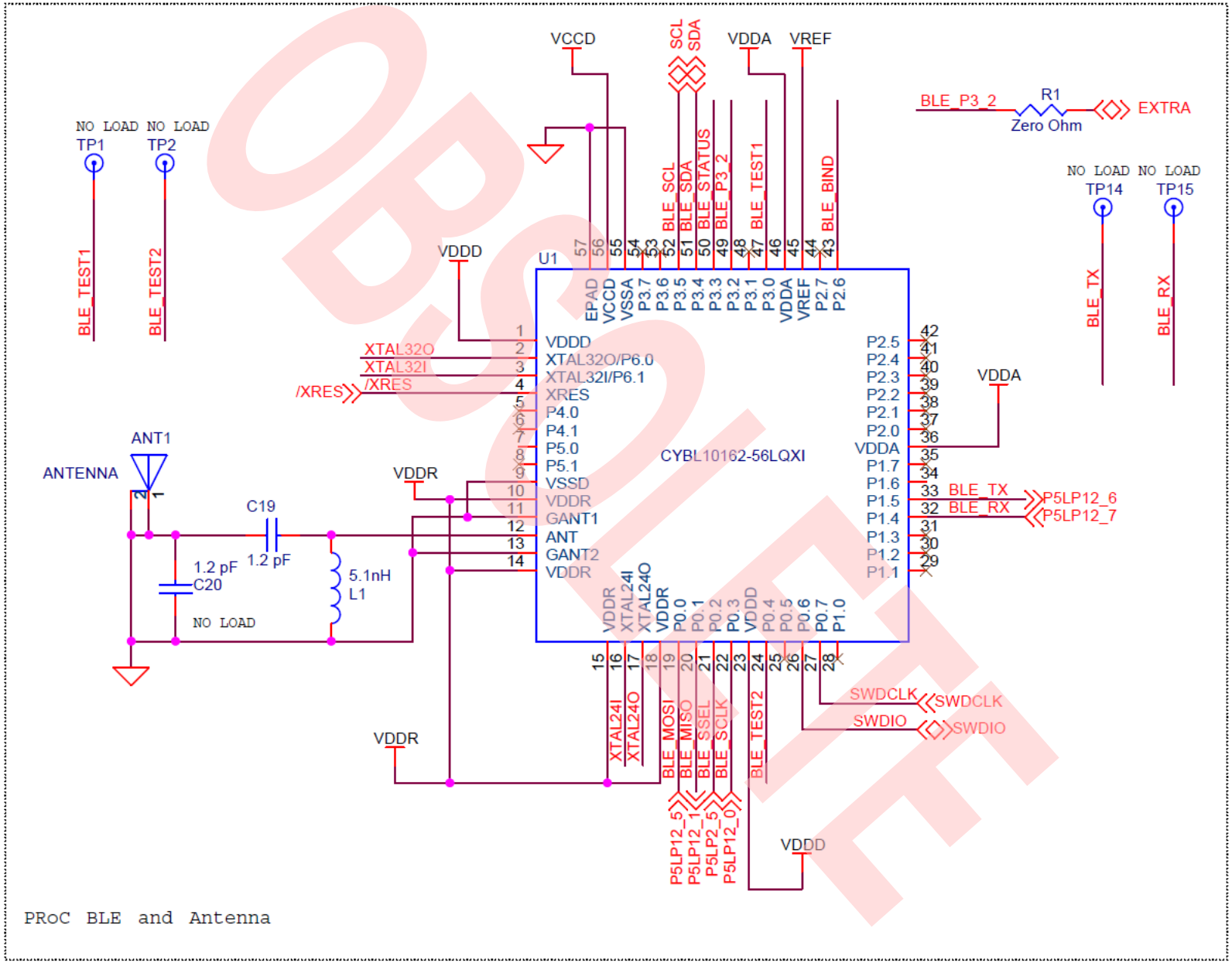

P5LP12_7
P5LP12_6
P5LP12_5
P5LP12_1
P5LP2_5
P5LP12_0
PSoC 5LP and PRoC BLE Connections


TP6
VBUS
NO LOAD
F1
PTC Resettable Fuse
USB A PLUG
J1
VBUS
DM
DP
GND
S1
S2
USB A PLUG
D1
D2
D3
100K R21
C39 0.01 uF
USB Connection

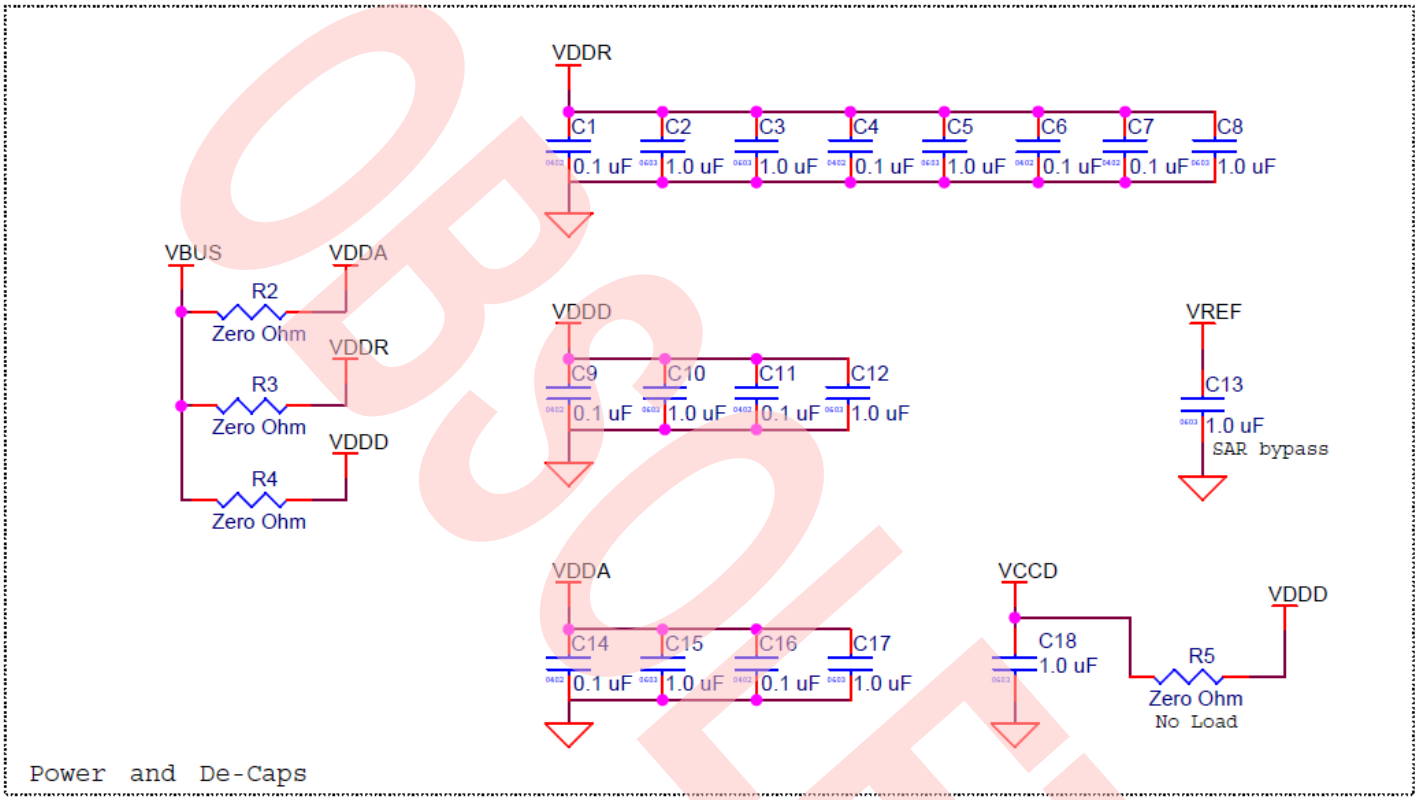

C22
36 pF
C23
18 pF
Y1
32.768KHz
kHz Crystal
XTAL32I
XTAL32O
XTAL24O
Y2
24MHz
XTAL24I
MHz Crystal
Crystals

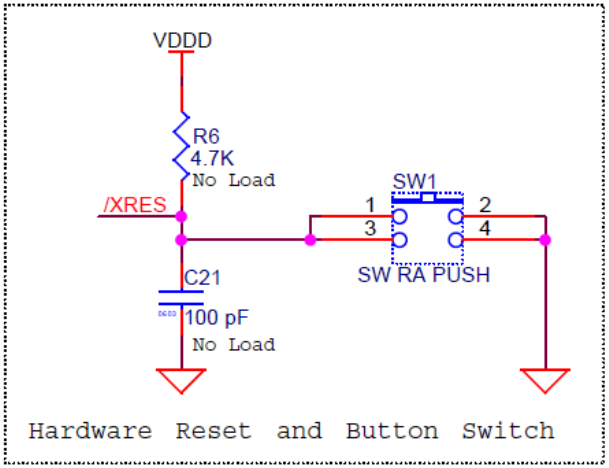

BLE_BIND
SW2
SW RA PUSH
BLE_STATUS
R7
820 ohm
LED1
User LED Blue
User Button Switch and User LED

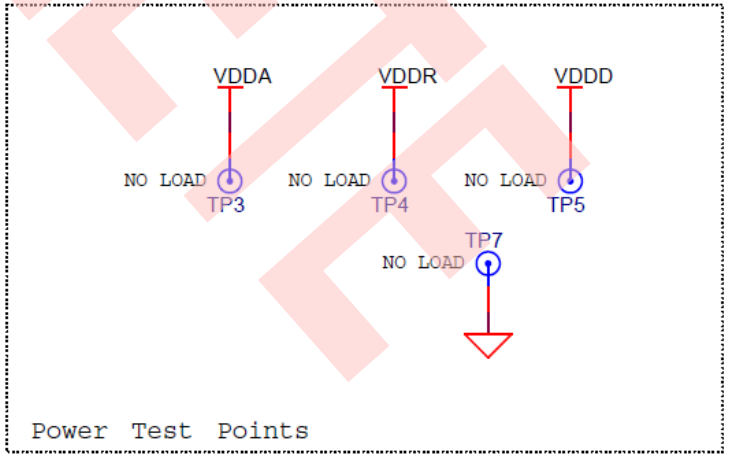
OBSOLETE


VCCD
SCL
SDA
VDDA
VREF
BLE_P3_2
R1
Zero Ohm
EXTRA
NO LOAD NO LOAD
TP1
TP2
BLE_TEST1
BLE_TEST2
NO LOAD NO LOAD
TP14
TP15
BLE_TX
BLE_RX
VDDD
U1
BLE_SCL
BLE_SDA
BLE_STATUS
BLE_P3_2
BLE_TEST1
BLE_BIND
EPAD
VCCD
VSSA
P3.7
P3.6
P3.5
P3.4
P3.3
P3.2
P3.1
P3.0
VDDA
VREF
P2.7
P2.6
XTAL32O
XTAL32I
/XRES
VDDD
XTAL32O/P6.0
XTAL32I/P6.1
XRES
P4.0
P4.1
P5.0
P5.1
VSSD
VDDR
GANT1
ANT
GANT2
VDDR
CYBL10162-56LQXI
P2.5
P2.4
P2.3
P2.2
P2.1
P2.0
VDDA
P1.7
P1.6
P1.5
P1.4
P1.3
P1.2
P1.1
VDDA
BLE_TX
BLE_RX
P5LP12_6
P5LP12_7
ANT1
ANTENNA
C19
1.2 pF
1.2 pF
C20
NO LOAD
5.1nH
L1
VDDR
VDDR
VDDR
XTAL24I
XTAL24O
VDDR
P0.0
P0.1
P0.2
P0.3
VDDD
P0.4
P0.5
P0.6
P0.7
P1.0
XTAL24I
XTAL24O
BLE_MOSI
BLE_MISO
BLE_SSEL
BLE_SCLK
BLE_TEST2
SWDCLK
SWDIO
SWDCLK
SWDIO
P5LP12_5
P5LP12_1
P5LP2_5
P5LP12_0
VDDD
PRoC BLE and Antenna

OBSOLETE


VDDR
C1
0.1 uF
C2
1.0 uF
C3
1.0 uF
C4
0.1 uF
C5
1.0 uF
C6
0.1 uF
C7
0.1 uF
C8
1.0 uF
VBUS
VDDA
R2
Zero Ohm
VDDR
R3
Zero Ohm
VDDD
R4
Zero Ohm
VDDD
C9
0.1 uF
C10
1.0 uF
C11
0.1 uF
C12
1.0 uF
VREF
C13
1.0 uF
SAR bypass
VDDA
C14
0.1 uF
C15
1.0 uF
C16
0.1 uF
C17
1.0 uF
VCCD
C18
1.0 uF
R5
Zero Ohm
No Load
VDDD
Power and De-Caps


VDDD
R6
4.7K
No Load
/XRES
SW1
1
2
3
4
SW RA PUSH
C21
100 pF
No Load
Hardware Reset and Button Switch


VDDA
VDDR
VDDD
NO LOAD
TP3
NO LOAD
TP4
NO LOAD
TP5
TP7
NO LOAD
Power Test Points

OBSOLETE

8.2.1 BOM List

No	Qty	Reference	Parts Number	Description	Manufacture	Note
1	17	C1, C4, C6, C7, C9, C11, C14, C16, C25, C28, C29, C32, C35, C36, C38, C41, C42	C1005X5R1A104K050BA	0.1 µF/0402	TDK Corporation	
2	17	C2, C3, C5, C8, C10, C12, C13, C15, C17, C18, C24, C26, C30, C31, C33, C34, C40	TMK107BJ105KA-T	1 µF/0603	Taiyo Yuden	
3	1	C19	500R07S1R2BV4T	1.2 pF/0402	Johanson Technology Inc	
4	1	C22	GRM1555C1H360JA01D	36 pF/0402	Murata Manufacturing Co., Ltd.	
5	1	C23	GRM1555C1H180FA01D	18 pF/0402	Murata Manufacturing Co., Ltd.	
6	1	C39	C1005X7R1C103K050BA	0.01 µF/0402	TDK Corporation	
7	3	D1, D2, D3	CG0603MLC-05LE	ESD diode	Bourns, Inc.	
8	1	F1	MF-MSMF050-2	FUSE	Bourns, Inc.	
9	1	J1	480370001	USB Type-A PLUG	Molex Inc	
10	1	J2	20021521-00010T1LF	50MIL KEYED SMD	FCI	
11	1	LED1	LTST-C171TBKT	Status LED Blue	LiteOn Inc	
12	1	LED2	CMD17-21VGC/TR8	Status LED Green	Chicago Miniature	
13	1	LED3	LTST-C170KRKT	Power LED Red	LiteOn Inc	
14	1	L1	L-07C5N1SV6T	5.1 nH/0402	Johanson Technology Inc	
15	2	R8, R11	ERJ-6GEY0R00V	0Ω/0805	Panasonic Corporation	
16	1	R7	ERJ-3GEYJ821V	820Ω/0603	Panasonic Corporation	
17	2	R22, R25	ERJ-6GEYJ821V	820Ω/0805	Panasonic Corporation	
18	2	R9, R10	ERJ-3GEYJ222V	2.2 kΩ/0603	Panasonic Corporation	
19	9	R1, R2, R3, R4, R12, R13, R14, R15, R26	ERJ-3GEY0R00V	0Ω/0603	Panasonic Corporation	
20	2	R17, R18	ERJ-3EKF22R0V	22Ω/0603	Panasonic Corporation	
21	1	R21	ERJ-2GEJ104X	100 kΩ/0402	Panasonic Corporation	
22	2	R19, R20	ERJ-3GEYJ153V	15 kΩ/0603	Panasonic Corporation	
23	2	R23, R24	ERJ-3GEYJ303V	30 kΩ/0603	Panasonic Corporation	
24	2	SW1, SW2	EVQ-P3401P	Push Switch	Panasonic Corporation	
25	1	TVS1	SD05-7	5V 350W	Diodes Inc.	
26	1	U1	CYBL10162-56LQXI	PRoC BLE, Programmable Radio on Chip	Cypress Semiconductor Corporation	
27	1	U2	NTZD3152PT1G	DUAL PMOS	ON Semiconductor	
28	1	U3	CY8C5868LTI-LP039	PSoC 5LP Programmable System on Chip	Cypress Semiconductor Corporation	
29	1	Y1	ECS-.327-12.5-34B	32.768 kHz	ECS Inc	
30	1	Y2	ECS-240-8-36CKM	24 MHz	ECS Inc	
31	0	(C20)	500R07S1R2BV4T	1.2 pF/0402	Johanson Technology Inc	Non-mount
32	0	(C21)	C0603C101K5RACTU	100 pF/0603	Kemet Corporation	Non-mount
33	0	(C37)	C1005X5R1A104K050BA	0.1 µF/0402	TDK Corporation	Non-mount
34	0	(C27)	TMK107BJ105KA-T	1 µF/0603	TAIYO YUDEN CO., LTD.	Non-mount
35	0	(R5)	1623094-1	0Ω/0603	TE Connectivity Ltd.	Non-mount

OBSOLETE

36	0	(R6), (R16)	ERJ-3GEYJ472V	4.7 kΩ/0603	Panasonic Corporation	Non-mount
37	15	TP1, TP2, TP3, TP4, TP5, TP6, TP7, TP8, TP9, TP10, TP11, TP12, TP13, TP14, TP15	-	-	-	

OBSOLETE

9 Ordering Information

Part Number	Version	Note
S6SAE101A00SA1002	Rev 1.0	

OBSOLETE

Appendix A. Other Sample Projects

A.1 LED ONOFF Project

The LED ONOFF project, available in *<Install directory>/Solar-Powered IoT Device Kit/1.0/Firmware//Other/LED_ONOFF/LED_ONOFF.cydsn*, demonstrates the simple LED control.

This sample firmware only supports LED blinking control using USB bus power; therefore, reprogram the EH_Motherboard project in the *Firmware* folder to restore the Solar-Powered BLE Beacon operation.

This section explains the LED_ONOFF project.

A.1.1 main.c

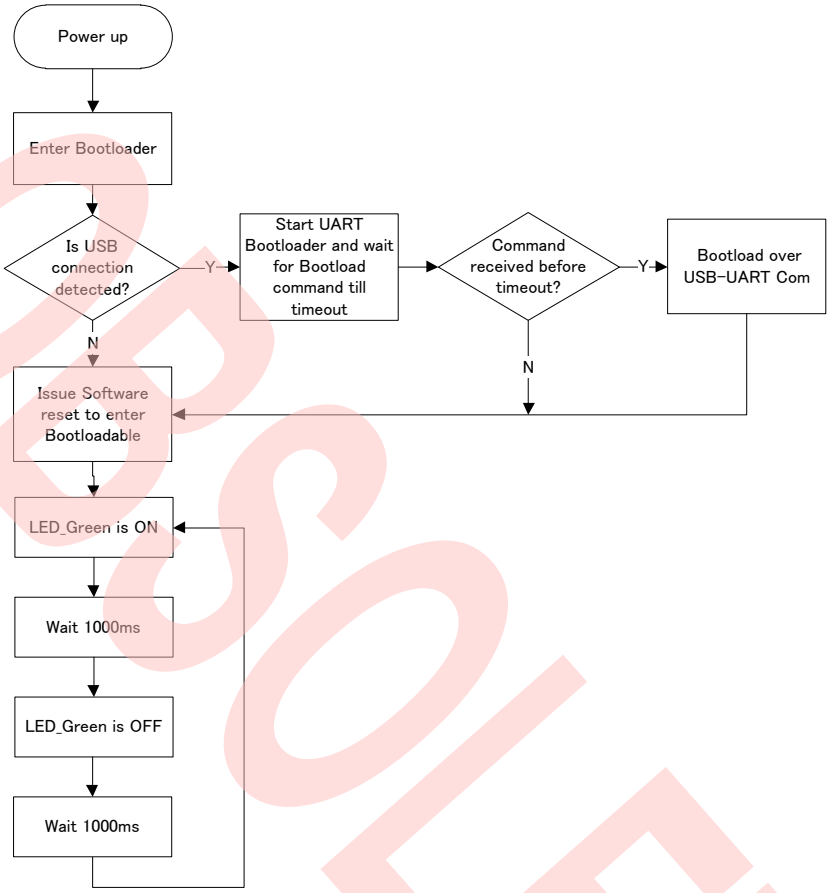
This file contains the main function, which only controls the turning ON and OFF of the LED. The P3.4 of EZ-BLE module controls a status LED on Energy Harvesting Motherboard. The blinking interval is set 1000 ms.

```
#include <project.h>


#define ON                  (1u)        /* P3.4 is high      */
#define OFF                 (0u)        /* P3.4 is low       */
#define BLINK_INTERVAL (1000)     /* 1000 ms           */


int main()
{
    while(1)
    {
        LED_Green_Write(ON);
        CyDelay(BLINK_INTERVAL);
        LED_Green_Write(OFF);
        CyDelay(BLINK_INTERVAL);
    }
    return 0;
}
```

OBSOLETE

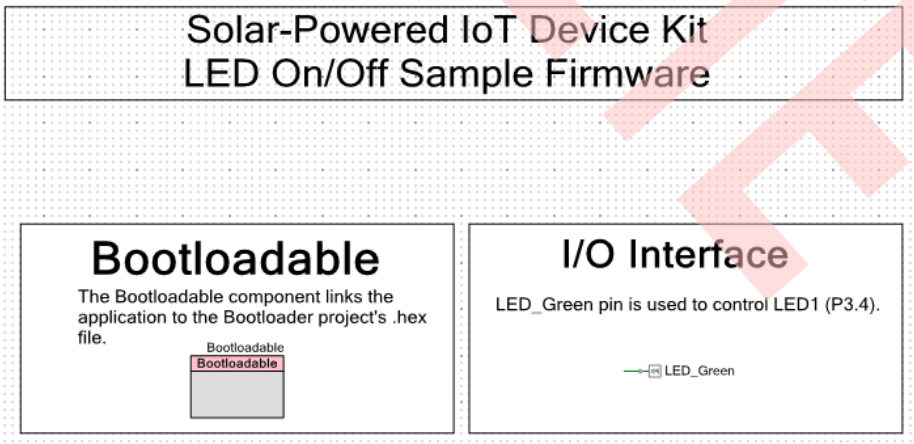
A.1.2 Flow of LED ONOFF Project

The following figure shows the flow of the LED ONOFF project.



A.1.3 TopDesign.cysch

The following figure shows the top design for LED_ONOFF project. The Bootloadable component links the application to the Bootloader project's *hex* file. A LED_Green of I/O Interface is used to control the status LED1 (P3.4).

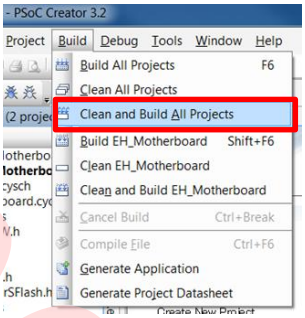


OBSOLETE

A.1.4 Process Steps

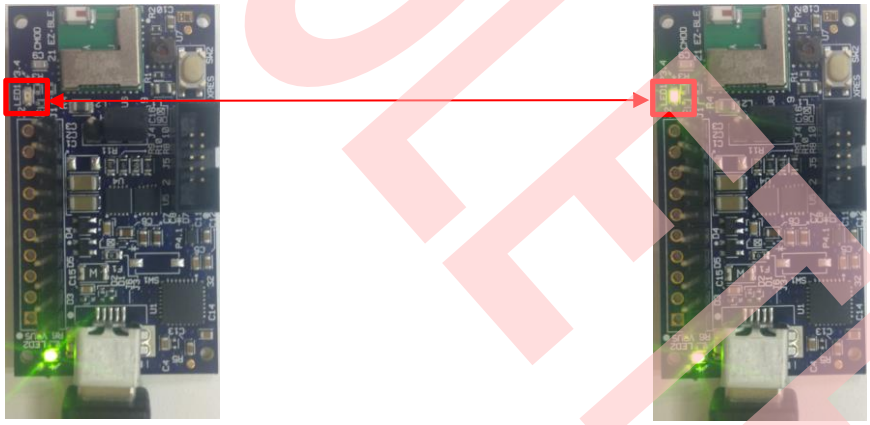
1. Open the sample project (*LED_ONOFF.cypj*) for this kit available at: (*<Install directory>/Solar-Powered IoT Device Kit/1.0/Firmware//Other/LED_ONOFF*).
2. Follow the menu path **Build** > **Clean and Build All Projects**. The build status will appear at the lower right side of PSoC Creator. The build process is complete when the message “Rebuild succeeded” appears on the status bar.

.



3. Program the LED_ONOFF project.
   a. If you use the UART Bootloader, see 5.1 UART Bootloader (Program Only). The *cyacd* file is generated in the project folder of PSoC Creator:
   <Install directory>/Solar-Powered IoT Device Kit/1.0/Firmware//Other/LED_ONOFF/LED_ONOFF.cydsn/CortexM0/ARM_GCC_484/Debug/LED_ONOFF.cyacd
   b. If you use the MiniProg3, see steps 6 to 10 of 5.2 PSoC Creator with MiniProg3 (Program and Debug).

The status LED on Energy Harvesting Motherboard will turn ON and OFF for a second using USB bus power.



OBSOLETE

A.2 Simple BLE Project

The simple BLE project, available in *<Install directory>/Solar-Powered IoT Device Kit/1.0/Firmware//Other/Simple_BLE/Simple_BLE.cydsn,* demonstrates the simple BLE Beacon transmitter.

This sample firmware only supports the BLE Beacon and Bootloader with no other frills (such as I²C, flash read/write, UART serial configuration). This project has one source file (*main.c*).

This section explains the Simple BLE project.

A.2.1 main.c

This file contains the main function, which only transmits the BLE Beacon as shown below. This firmware flow is same as BLE Beacon Process except the SFLASH parameter reading. All parameters such as UUID, MAJOR, and MINOR are fixed by the BLE Component on *TopDesign.cysch*.

```
int main()
{
    CyGlobalIntEnable;
    /* Set the divider for ECO, ECO will be used as source when IMO is * switched off
    to save power, to drive the HFCLK */
    CySysClkWriteEcoDiv(CY_SYS_CLK_ECO_DIV8);

    /* Start WCO & ECO in low power mode */
    LowPower_WCO_ECO_Start();
    CyBle_Start(BLE_AppEventHandler);
    CyBle_ProcessEvents();

    for(;;)
    {
        CYBLE_LP_MODE_T pwrState;
        CYBLE_BLESS_STATE_T blessState;
        uint8 intStatus = 0;
       /* BLE stack processing state machine interface */
        CyBle_ProcessEvents();
        /* Configure BLESS in Deep-Sleep mode */
        pwrState  = CyBle_EnterLPM(CYBLE_BLESS_DEEPSLEEP);
        /* No interrupts allowed while entering system low power modes
         */
        intStatus = CyEnterCriticalSection();
        blessState = CyBle_GetBleSsState();
        /* Make sure BLESS is in Deep-Sleep before configuring system in * Deep-Sleep
         mode */
        if(pwrState == CYBLE_BLESS_DEEPSLEEP)
        {
            /* If BLESS is in Deep Sleep or is in the process of waking * up from Deep
           Sleep, put system in Deep Sleep mode */
           if(blessState == CYBLE_BLESS_STATE_ECO_ON || blessState ==
           CYBLE_BLESS_STATE_DEEPSLEEP)
```

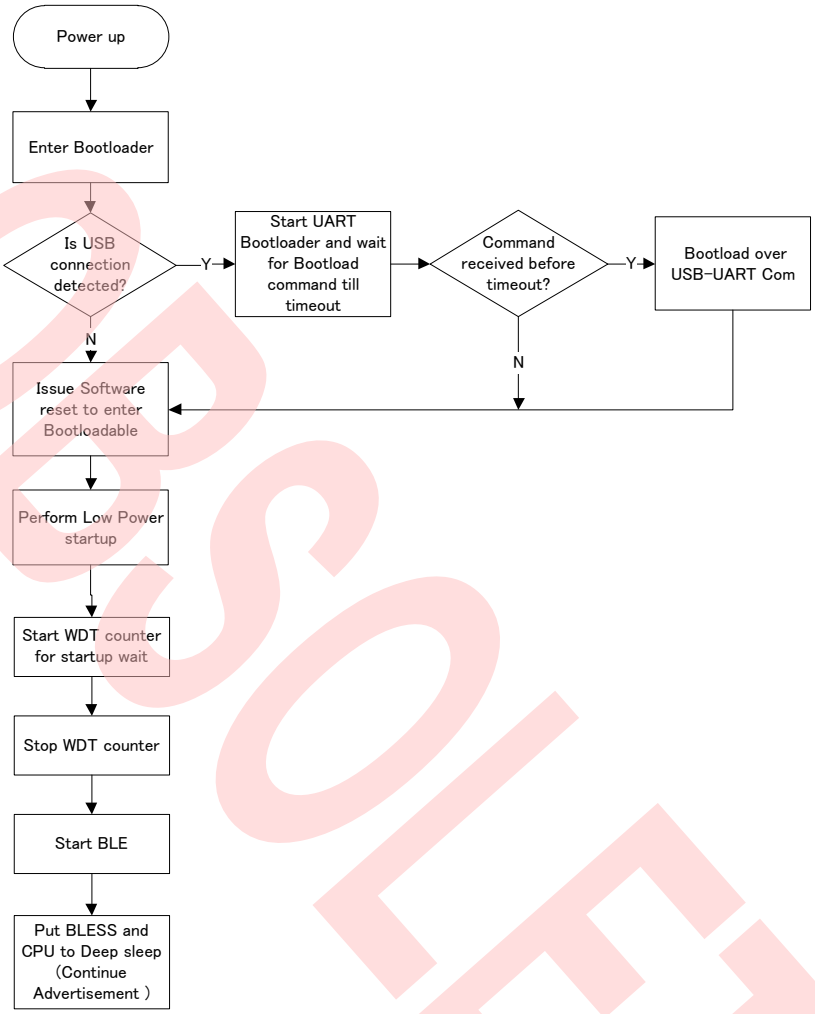
OBSOLETE

```
            {
                CySysPmDeepSleep(); /* System Deep-Sleep. 1.3uA mode */
            }
        }
        /* If BLESS is in Active state,
          * and if BLESS Tx/Rx Event is not complete, stop IMO and put CPU * to Sleep
        */
        else if (blessState != CYBLE_BLESS_STATE_EVENT_CLOSE)
        {
            /* Change HF clock source from IMO to ECO, as IMO can be stopped * to save
           power */
            CySysClkWriteHfclkDirect(CY_SYS_CLK_HFCLK_ECO);
            /* Stop IMO for reducing power consumption */
            CySysClkImoStop();
            /* Put the CPU to Sleep. 1.1mA mode */
            CySysPmSleep();
          /* Starts execution after waking up, start IMO */
            CySysClkImoStart();
            /* Change HF clock source back to IMO */
            CySysClkWriteHfclkDirect(CY_SYS_CLK_HFCLK_IMO);
        }
        CyExitCriticalSection(intStatus);
        /* If restartadvertisement flag is set, which means it's the first * time cpu
       goes here. */
        if(restartadvertisement)
        {
            restartadvertisement = false;
            CySysWdtUnlock();
            CySysWdtDisable(CY_SYS_WDT_COUNTER0_MASK);
            CySysWdtWriteMode(SOURCE_COUNTER, CY_SYS_WDT_MODE_INT);
            CySysWdtWriteClearOnMatch(SOURCE_COUNTER, COUNTER_ENABLE);
            CySysWdtWriteMatch(SOURCE_COUNTER, COUNT_PERIOD_1S);
            CySysWdtEnable(CY_SYS_WDT_COUNTER0_MASK);
            CySysWdtLock();
            wdt_trigger_on_flag = true;
        }
        ProcessBeaconEvents();
    }
}
```

OBSOLETE

A.2.2 Flow of Simple BLE Project

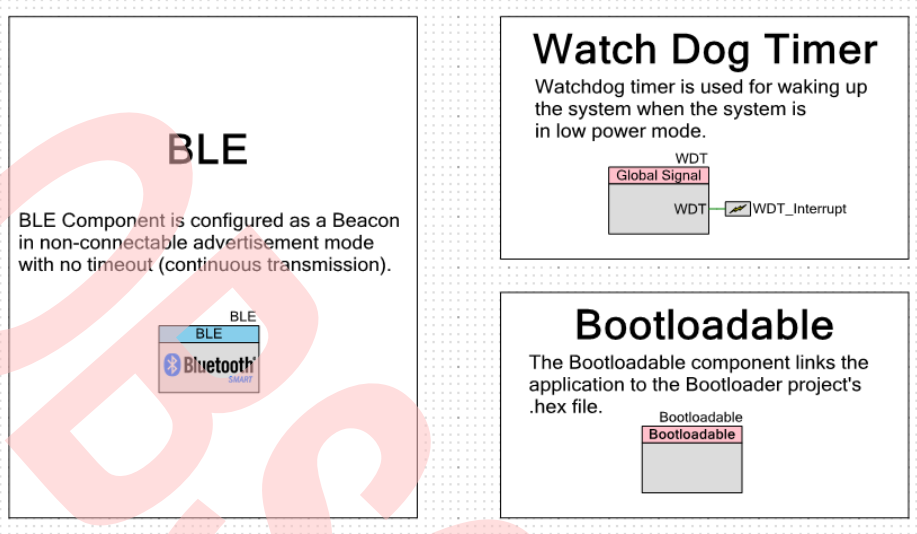
The following figure shows the flow of the Simple BLE project.



OBSOLETE

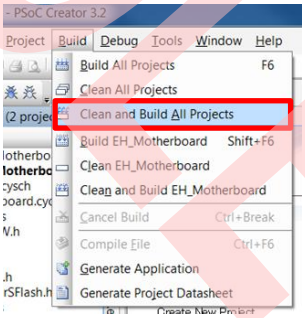
A.2.3 TopDesign.cysch

The following figure shows the top design for SIMPLE_BLE project. BLE Component is configured as a Beacon with fixed parameter. The Watch Dog Timer is used for waking up the system when the system is in low power mode. The Bootloadable Component links the application to the Bootloader project's *hex* file.



A.2.4 Process Steps

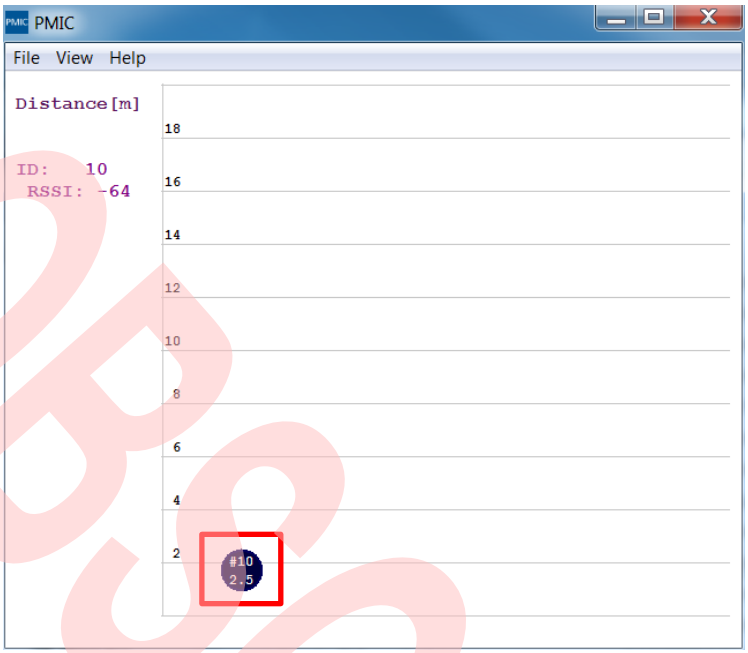
1. Open the sample project (*Simple_BLE.cyprj*) for this kit available at: (*<Install directory>/Solar-Powered IoT Device Kit/1.0/Firmware/Other/Simple_BLE/Simple_BLE.cydsn/ Simple_BLE.cyprj*).
2. Follow the menu path **Build** > **Clean and Build All Projects**. The build status will appear at the lower right side of PSoC Creator. The build process is complete when the message "Rebuild succeeded" appears on the status bar.



3. Program the Simple_BLE project.
   a. If you use the UART Bootloader, see 5.1 UART Bootloader (Program Only). The *cyacd* file is generated in the project folder of PSoC Creator:
   <Install directory>/Solar-Powered IoT Device Kit/1.0/Firmware//Other/Simple_BLE/Simple_BLE.cydsn/CortexM0/ARM_GCC_484/Debug/ Simple_BLE.cyacd
   b. If you use the MiniProg3, see steps 6 to 10 of 5.2 PSoC Creator with MiniProg3 (Program and Debug).

OBSOLETE

4. Run *PMIC.exe*. The MAJOR number 10 (fixed value by firmware) of the Motherboard is displayed on the PMIC Software (see 4.1.4 Establishing BLE Connection for more details). Then, move the Motherboard further away from your computer. The Received Signal Strength Indicator (RSSI) value will change and the graphic will be updated.



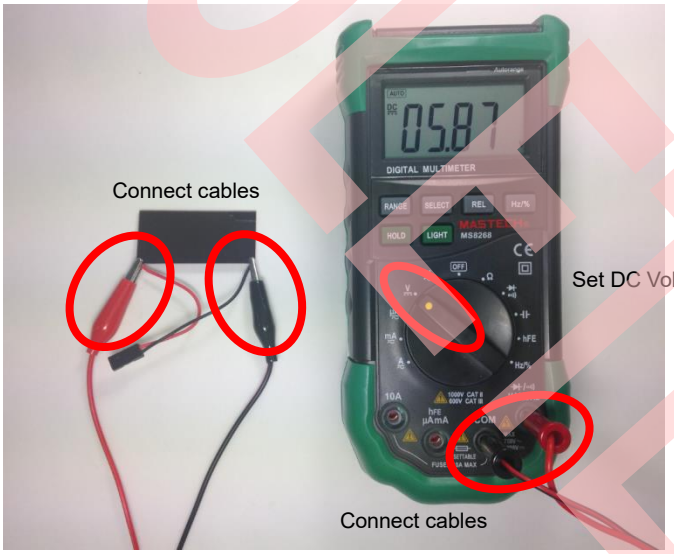
OBSOLETE

Appendix B. Using Extra Components

B.1 10-Ω Resistor for Current Measurement

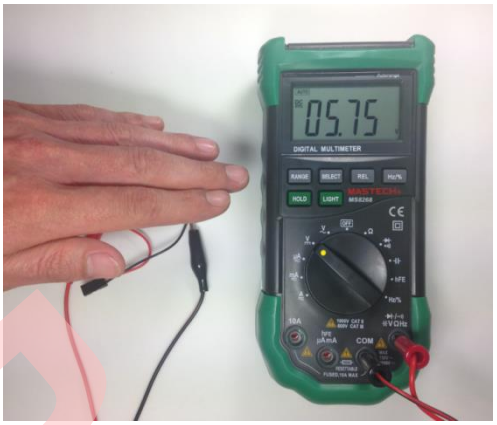
This chapter explains the Solar Module, 10-Ω resistor, and a Multimeter (MASTECH: MS8268, not included in this kit.) to measure the voltage and current for the solar module using 10-Ω resistor that is included in the Solar-Powered IoT Device Kit. First, you will take measurements under constant ambient light. Then, you will vary the available light and observe the change in the energy output of the solar module.

1. Set the Multimeter to measure DC Voltage. Connect the Multimeter to the Solar Module as shown in the figure, and observe the DC voltage (V1).

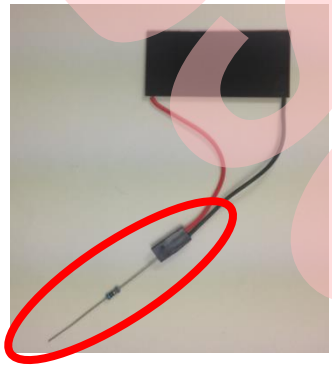


OBSOLETE

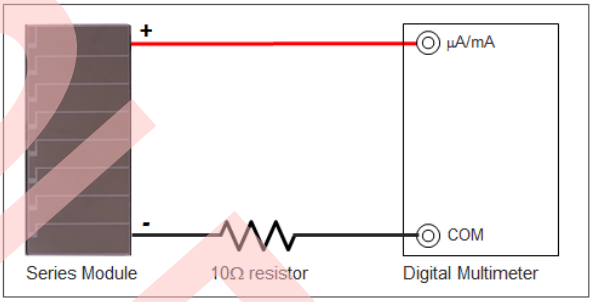
2. Place your hand about 2 cm above the Solar Module to block most of the light and observe the DC voltage (V2).



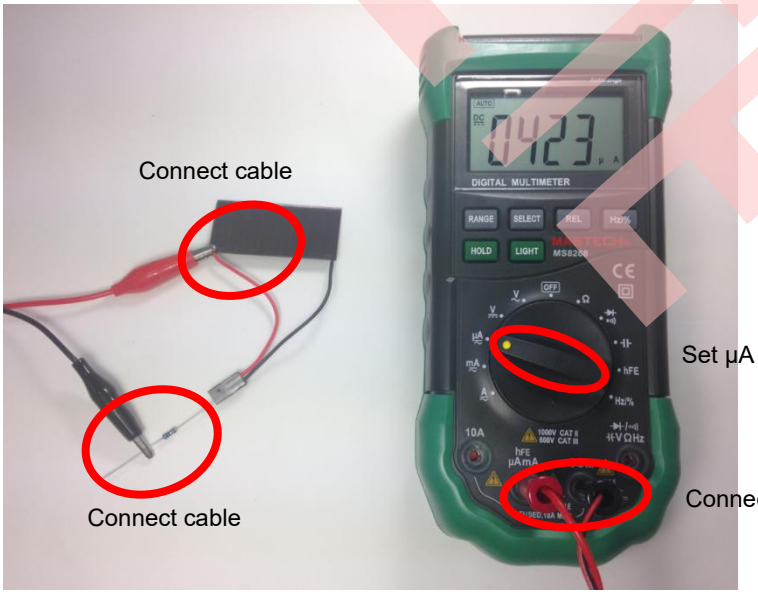
3. Now, you measure the current produced by the Solar Module. Connect a 10-Ω resistor to the black wire on the Solar Module as shown in the figure.



Measurement Block

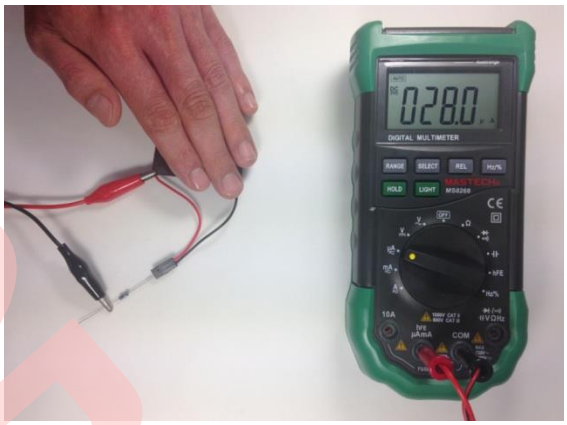


4. Set the Multimeter to measure Current in the microamp range. Connect the positive cable of the Multimeter to the positive lead of the Solar Module, and the negative cable to the 10-Ω resistor. Observe the current (I1).



OBSOLETE

5. Place your hand about 2 cm above the Solar Module to block most of the light and observe the DC voltage (I2).



6. Calculate the energy produced by the Solar Module. Calculate the Maximum Power Point (MPP[11]) energy of the Solar Module.

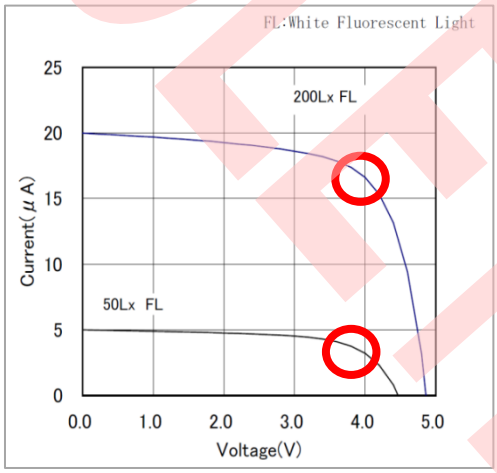
Under constant ambient light (Example 400 Lux)

E1 = (0.8 x V1) x (0.8 x I1) = (0.8 x 5.87V) x (0.8 x 42.3uA) = 158.91 [uW]

Covering Solar Module with hand (Example 250 Lux)

E2 = (0.8 x V2) x (0.8 x I2) = (0.8 x 5.75V) x (0.8 x 28.0uA) = 103.04 [uW]

I-V Characteristics of Solar Module (AM-1801)



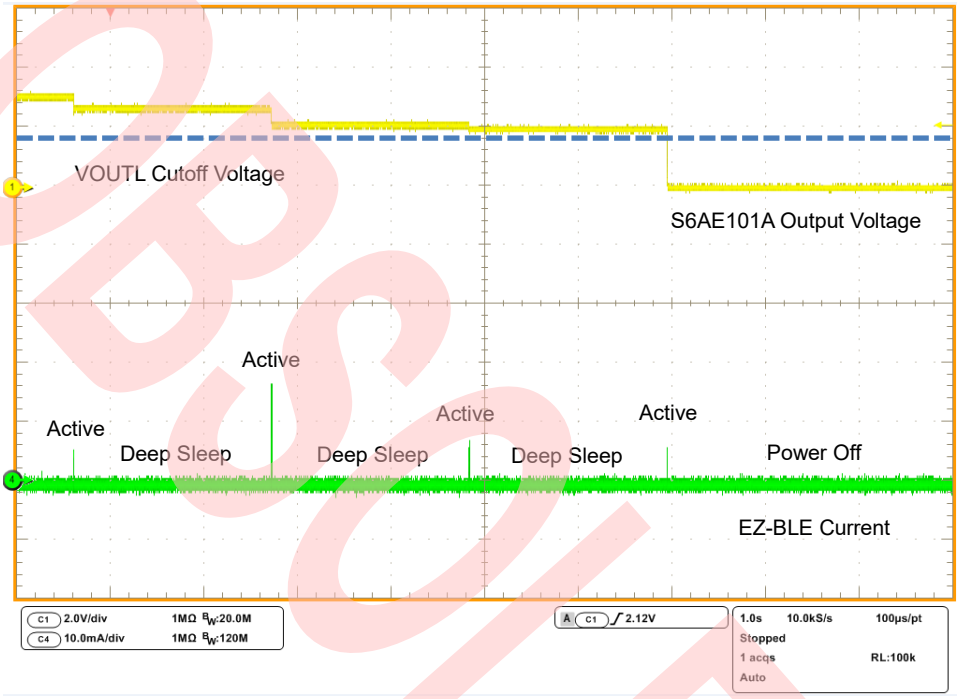
[11] The MPP of a typical solar cell is 80% of the maximum voltage and maximum current.

OBSOLETE

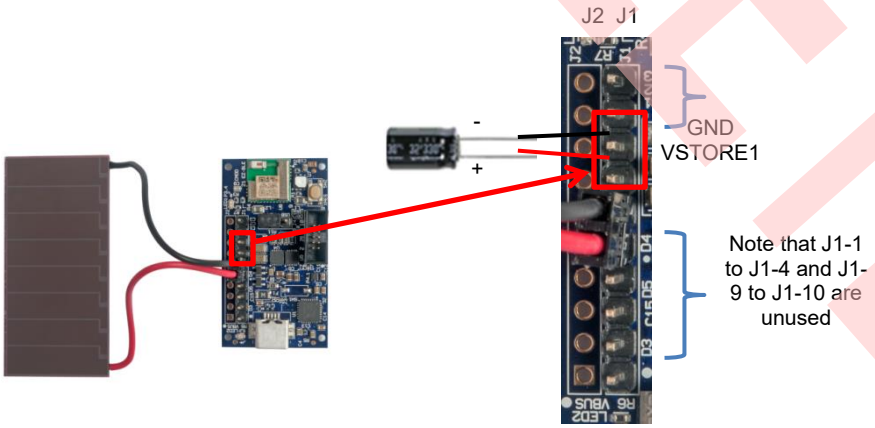
B.2 Additional 220 µF Capacitor

EHS comprises an EHD such as a Solar Module, an Energy Harvesting PMIC, and a Storage Device (such as a capacitor). The purpose of EHS purpose is to supply power to the EHS Load. When designing an EHS, consider voltage, current, and time in three contexts: power supplied by the EHD, power stored in the Storage Device, and power required to operate the system.

Following is example of failed waveform for EHS. The output voltage of the S6AE101A slowly drops due to consumption, eventually shutting down the output completely.



One way to resolve this failure is to connect the additional 220 µF capacitor to VSTORE1 of S6AE101A..

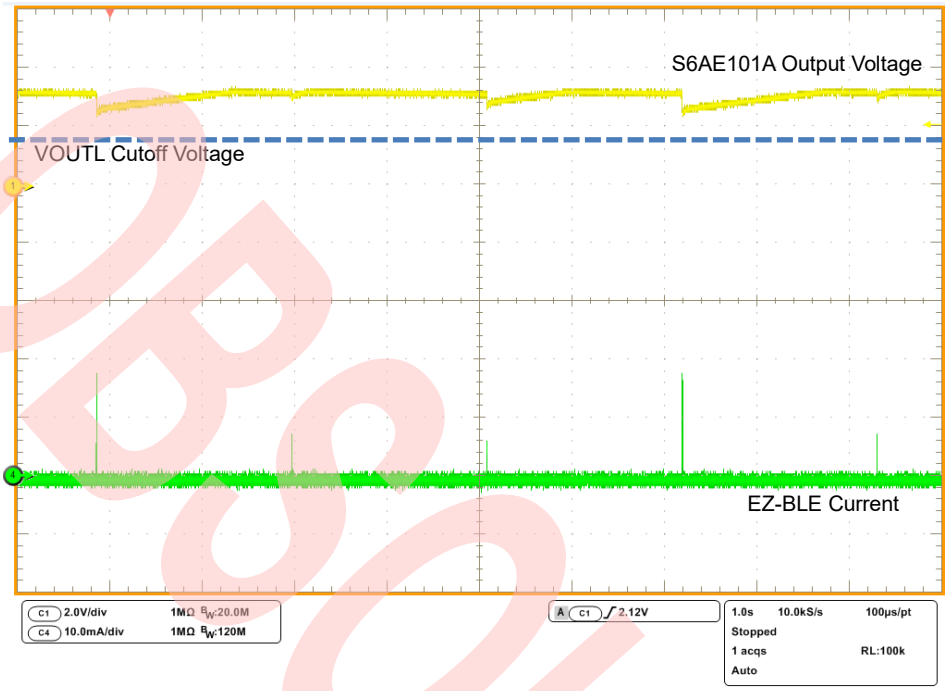


OBSOLETE

The waveform for EHS is completed to add the extra capacitor to VSTORE1 pin. Note that value of capacitor depends on the consumption energy. You can calculate the energy of capacitor using following formula.

Calculation formula for energy of capacitor

```
ECAP = 0.5 × C × (VOUTH2 – VOUTL2)
```



OBSOLETE

Document Revision History

Document Title: Solar-Powered Internet of Things (IoT) Device Kit User Guide

Document Number: 002-00297

Revision	ECN	Issue Date	Description of Change
**	08/25/2015	EIFU	New Kit Guide
*A	05/12/2017	GNKK	Updated the Cypress logo and copyright information.
*B	08/27/2018	EIFU	Sunset review Completed content review
*C	07/12/2022	EIFU	Obsolete document Completing Sunset review

OBSOLETE

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

Arm® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6 MCU

Cypress Developer Community

Community | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

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

© Cypress Semiconductor Corporation, 2015-2022. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. No computing device can be absolutely secure. Therefore, despite security measures implemented in Cypress hardware or software products, Cypress does not assume any liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. In addition, the products described in these materials may contain design defects or errors known as errata which may cause the product to deviate from published specifications. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.