

Chapter 1: Introduction

After completing this chapter, you will understand what ModusToolbox is, what tools are included, how to install the software, and how switch to different KitProg modes.

1.1	WHAT IS THIS CLASS?	2
1.2	WHAT IS MODUSTOOLBOX?	2
1.3	REFERENCE FLOW	2
1.4	PRODUCTS	3
1.4.1	TOOLS	3
1.4.2	FIRMWARE	3
1.5	SDK	4
1.6	SOFTWARE	4
1.7	DEVELOPMENT KITS	5
1.7.1	PSOC 6 KITS	5
1.7.2	WICED BLUETOOTH KIT	8
1.8	PSOC KITPROG PROGRAMMER	9
1.9	PSOC FIRMWARE LOADER	10
1.9.1	STARTING COMMAND LINE SHELL	10
1.9.2	RUNNING FIRMWARE LOADER	10
1.10	WICED BLUETOOTH PROGRAMMING	12
1.11	UARTS AND SERIAL TERMINALS	12
1.11.1	WINDOWS	13
1.11.2	MACOS	14
1.11.3	LINUX	17
1.12	EXERCISES	19
1.12.1	VERIFY THE SOFTWARE	19
1.12.2	NETWORK PROXY SETTINGS	20
1.12.3	RUN THE FW-LOADER TOOL	24
1.12.4	SWITCH KITPROG MODES AND VERIFY THE MODE	24
1.12.5	USE A TERMINAL PROGRAM TO ATTACH TO THE UART ON THE CY8CPROTO-062-4343W	24
1.12.6	USE A TERMINAL PROGRAM TO ATTACH TO THE UART ON THE CY8CKIT-062-WIFI-BT KIT	25
1.12.7	USE A TERMINAL PROGRAM TO ATTACH TO THE UART ON THE CYW920819EVB-02 KIT	25

1.1 What is this Class?

This class is a survey of the Cypress IoT development platform ModusToolbox 2.1. The learning objective is to introduce you to all the tools in ModusToolbox and help you develop some familiarity with using them. The class is “a mile wide and an inch deep.” This should enable you to understand the scope of the Cypress development ecosystem and teach you where to find “everything.”

This class will touch on PSoC 6, Wi-Fi, Bluetooth, FreeRTOS, Mbed OS, and Amazon Web Services, but it is not an in-depth study of any of those topics. You can learn more by taking PSoC 101, Cypress Wi-Fi 101 and Cypress Bluetooth 101.

To develop applications using these tools, you need to have good C-programming skills as most of the development effort with these types of chips is spent writing programs. Your skills should include:

- C Control Structure
- C Variables (Data)
- Multi-file programs
- Linking
- RTOS
- IoT Frameworks – MQTT, HTTP
- Bluetooth Low Energy

1.2 What is ModusToolbox?

ModusToolbox 2.1 is a set of Reference Flows, Products, and SDKs that enable an immersive development experience for customers creating converged MCU and Wireless systems. ModusToolbox leverages popular third-party networking solutions such as Mbed OS, Amazon FreeRTOS, AliOS Things, and Zephyr.

The guiding principles for ModusToolbox include:

- The customer experience is the top priority.
- The software is optimized for professional developers.
- Ease of use and getting started are critical to the success of ModusToolbox.
- ModusToolbox enables developers to use their development flow of choice.
- SDKs in ModusToolbox are built as simply as possible using normal C & C++ programming best principles and a consistent architecture.

1.3 Reference Flow

A Reference Flow is a Cypress documented, supported, and qualified **methodology** for a customer to use an SDK to create their product. It is a recipe, defining how to create projects, add middleware, configure devices, build, program and debug.

For example, a Reference Flow called the XYZ flow could be instructions for using Visual Studio Code and the PSoC 6 SDK to create some PSoC 6 product.

A Reference flow could be a path only through an SDK, or it might include other external tools that are not part of the SDK (or even ModusToolbox) e.g. Visual Studio Code, Sublime, XCODE, IAR, etc.

We understand that customers want to pick and choose the ModusToolbox Products they use, merge them into their own flows, and develop applications in ways we cannot predict. ModusToolbox treats Products as individual entities and thus enables such custom flows. Our customers must be able to “Program the way they want”.

1.4 Products

ModusToolbox Products are [Tools](#) and [Firmware](#) that can be used individually, or as a group, to develop connected applications for Cypress devices. Unlike previous Cypress software offerings, ModusToolbox is not a monolithic, IDE-centric software tool. Each Product is individually executable (for tools), buildable (for firmware), testable, portable, and deliverable. Products are distributed through multiple portals (for example mbed.com, github.com, and cypress.com) to enable users to work in their preferred environment.

1.4.1 Tools

Tools refer to programs and services that run on the developer’s host computer or in the cloud. For example:

- Eclipse-based IDE for ModusToolbox
- Compilers (GCC, ARM)
- Build System (make, Cygwin)
- Programming and Debug Tools (OpenOCD, PyOCD)
- Configurators and Tuners
- Project Creator
- Library Manager
- Firmware Loader
- JRE
- FTDI driver
- CyMcuElfTool

1.4.2 Firmware

Firmware refers to code that executes on the target device. This includes:

- BSP (Board Support Package)
- CSP (Chip Support Package) integrated into the BSP
- Libraries (e.g., RTOS, Network Stacks, Graphics, etc.)
- Customer or Cypress Application Firmware (i.e. their project or our code example)

1.5 SDK

A Software Development Kit (SDK) is a collection of ModusToolbox Products that support one or more specific Reference Flows. ModusToolbox 2.1 supports the following SDKs, which are covered in separate chapters in this training:

- Mbed SDK
- PSoC 6 SDK
- Amazon FreeRTOS SDK
- WICED Bluetooth SDK

1.6 Software

All this software needs to be installed prior to class. Refer to the Installation Instructions provided via email. See also exercise [1.12.1](#) to verify that the software is installed correctly.

Tool	Description
ModusToolbox 2.1	Cypress' kick ass new development toolbox
Mbed CLI	Python programs which allow you to use the ARM Mbed Command Line Interface.
Mbed Studio	An ARM GUI for writing code and building Mbed OS Projects
Visual Studio Code	The fastest growing code editor with debug extension on the market, which is quickly displacing Eclipse

The ModusToolbox 2.1 installer will install the tools, but not any firmware such as BSPs, code examples, middleware libraries, or even SDK files. The firmware is all hosted on the cloud (mainly on GitHub). The advantages are that you only get what you need when you need it, and you can update different pieces independently – you don't need a full new install every time a library is updated. Each item is developed and released on its own schedule.

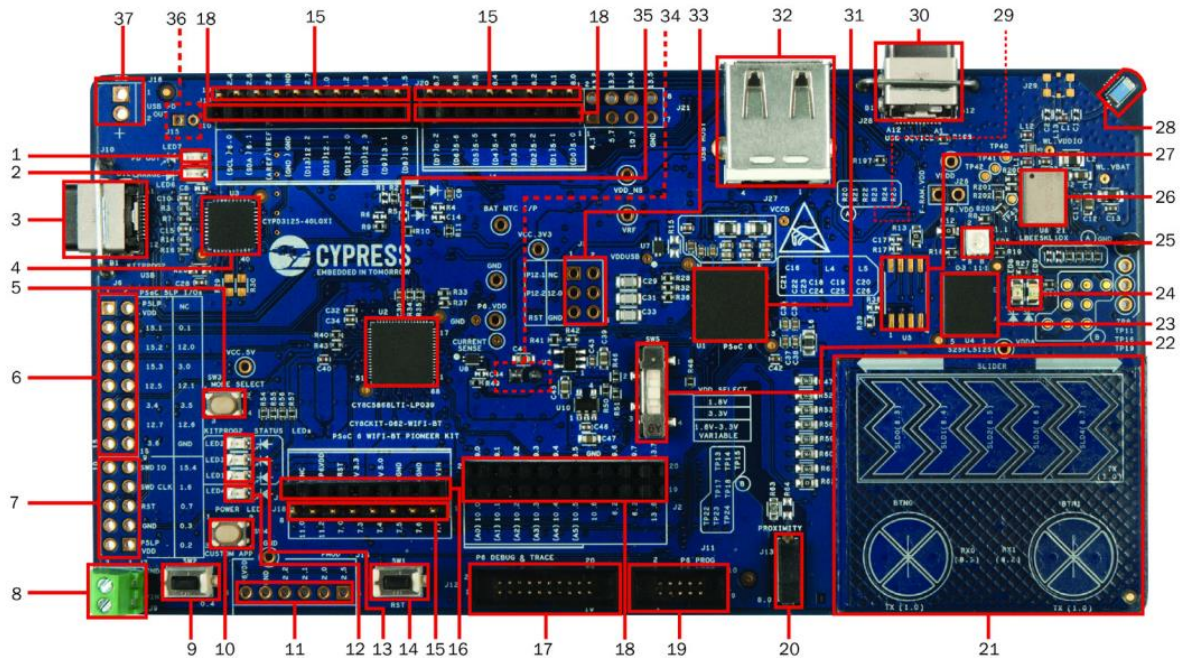
Note: You will need some type of code editor, such as Notepad++; however, ModusToolbox IDE and Visual Studio Code include code editors as well. You will also need some type of serial terminal, such a PuTTY or Serial. See section [1.11](#) for more details.

1.7 Development Kits

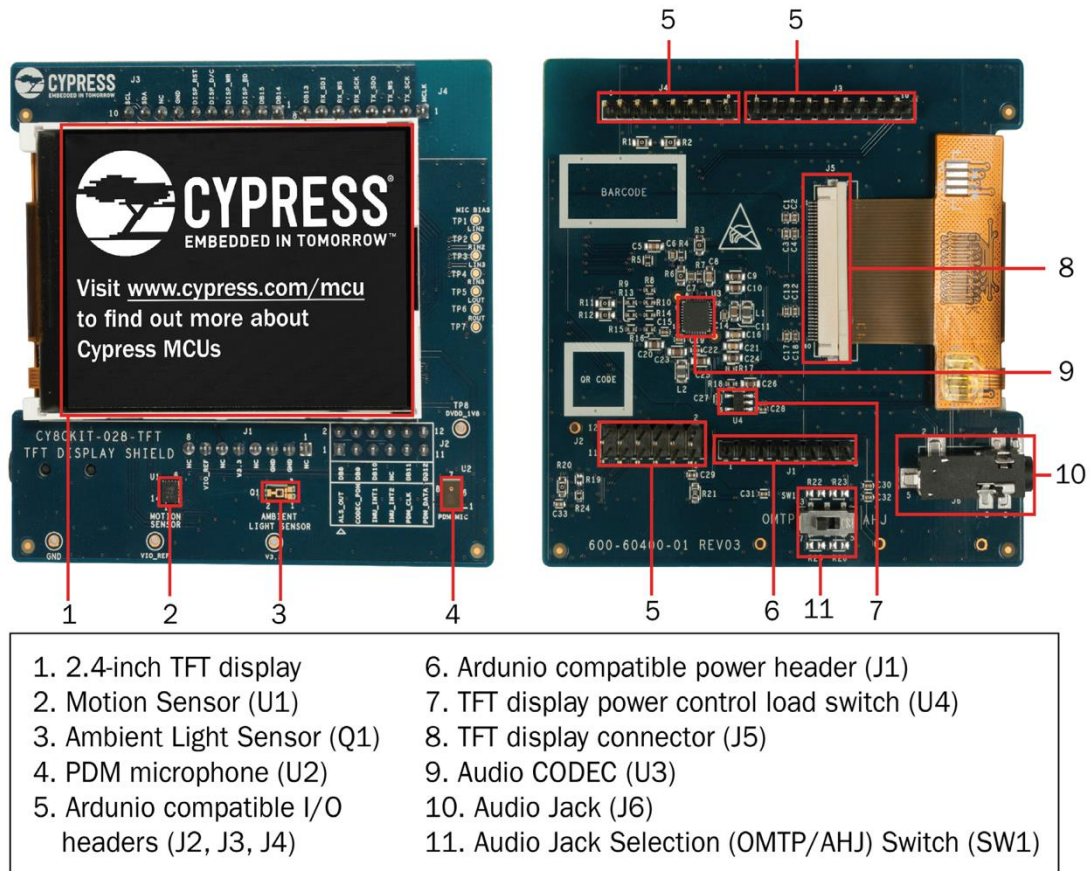
For this class, we will use the following development kits:

1.7.1 PSoC 6 Kits

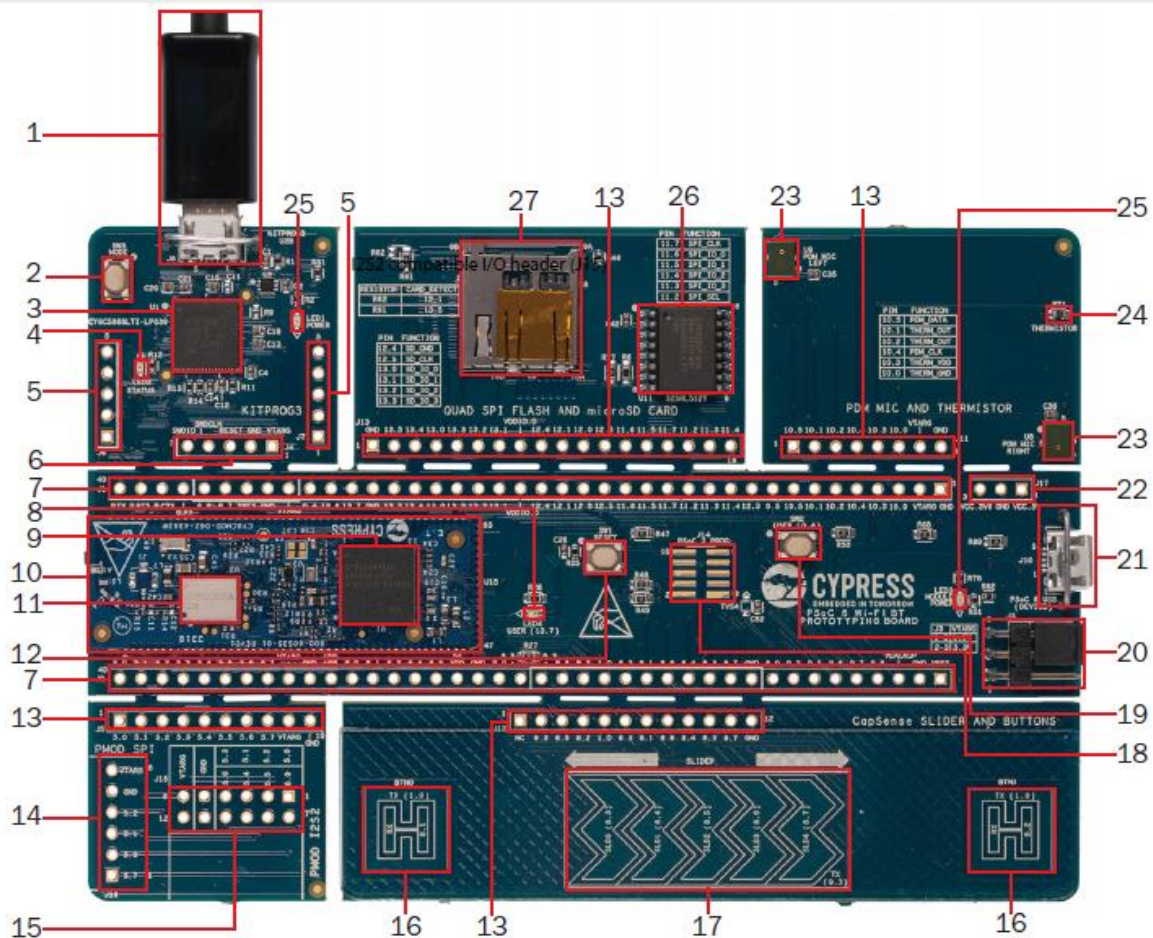
[CY8CKIT_062_WiFi_BT](#) – A PSoC 6-1M and a CYW4343W Bluetooth WiFi Combo



- | | |
|---|--|
| 1. USB PD output voltage availability indicator (LED7) | 21. CapSense slider and buttons |
| 2. Battery charging indicator (LED6) | 22. PSoC 6 VDD selection switch (SW5) |
| 3. KitProg USB Type-C connector (J10) | 23. Cypress 512-Mbit serial NOR Flash memory (S25FL512S, U4) |
| 4. Cypress EZ-PD™ CCG3 Type-C Port Controller with PD (CYPD3125-40LQXI, U3) | 24. PSoC 6 user LEDs (LED8 and LED9) |
| 5. KitProg programming mode selection button (SW3) | 25. RGB LED (LED5) |
| 6. KitProg I/O header (J6)1 | 26. WiFi/BT module (LBEE5KL 1DX, U6) |
| 7. KitProg programming/custom application header (J7)1 | 27. Cypress serial Ferroelectric RAM (U5)1 |
| 8. External power supply connector (J9) | 28. WiFi-BT Antenna |
| 9. PSoC 6 user button (SW2) | 29. VBACKUP and PMIC control selection switch (SW7)2 |
| 10. KitProg application selection button (SW4) | 30. PSoC 6 USB device Type-C connector (J28) |
| 11. Digilent® Pmod™ compatible I/O header (J14)1 | 31. Cypress PSoC 6 (CY8C6247BZI-D54, U1) |
| 12. Power LED (LED4) | 32. PSoC 6 USB Host Type-A connector (J27) |
| 13. KitProg status LEDs (LED1, LED2, and LED3) | 33. Arduino Uno R3 compatible ICSP header (J5)1 |
| 14. PSoC 6 reset button (SW1) | 34. PSoC 6 power monitoring jumper (J8)2 |
| 15. PSoC 6 I/O header (J18, J19 and J20) | 35. KitProg (PSoC 5LP) programmer and debugger (CY8C5868LTI-LP039, U2) |
| 16. Arduino™ Uno R3 compatible power header (J1) | 36. Battery connector (J15)1, 2 |
| 17. PSoC 6 debug and trace header (J12) | 37. USB PD output voltage (9V/12V) connector (J16)1 |
| 18. Arduino Uno R3 compatible PSoC 6 I/O header (J2, J3 and J4) | |
| 19. PSoC 6 program and debug header (J11) | |
| 20. CapSense proximity header (J13) | |



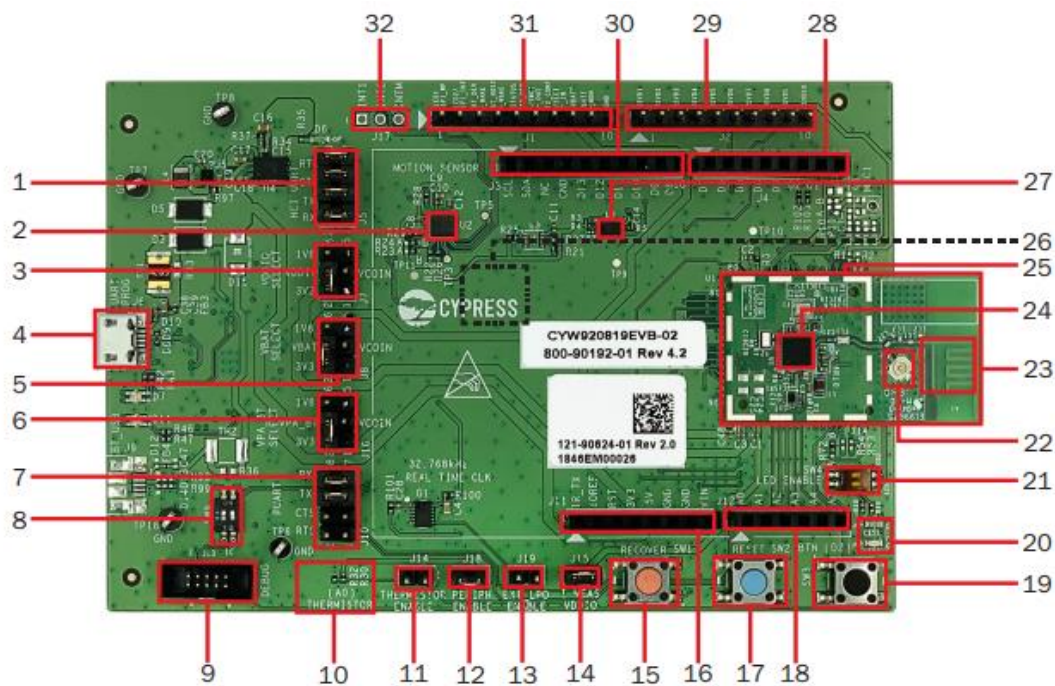
CY8CPROTO_062_4343W – A PSoC 6-2M and a CYW4343W Bluetooth WiFi Combo



- | | |
|--|---|
| 1. KitProg3 USB connector (J8) | 14. Digilent® Pmod™ SPI compatible I/O header (J16) |
| 2. KitProg3 programming mode selection button (SW3) | 15. Digilent® Pmod™ I2S2 compatible I/O header (J15) |
| 3. KitProg3 (PSoC 5LP) programmer and debugger (CY8C5868LTI-LP039, U1) | 16. CapSense buttons |
| 4. KitProg3 status LED (LED2) | 17. CapSense slider |
| 5. KitProg3 I/O headers (J6, J7) | 18. PSoC 6 MCU program and debug header (J14) |
| 6. KitProg3 5-pin programming header (J4) | 19. PSoC 6 MCU user button (SW2) |
| 7. PSoC 6 MCU I/O headers (J1, J2) | 20. Power selection jumper (J3) |
| 8. PSoC 6 MCU user LED (LED4) | 21. PSoC 6 USB device Connector (J10) |
| 9. PSoC 6 MCU (CY8C624ABZI-D44) | 22. External power supply connector (J17) |
| 10. Cypress PSoC 6 WiFi-BT Module (CY8CMOD-062-4343W, U15) | 23. PDM microphones (U8, U9) |
| 11. CYW4343W based Murata Type 1DX Module (LBEE5KL1DX) | 24. Thermistor (RT1) |
| 12. Reset button (SW1) | 25. Power LEDs (LED1, LED3) |
| 13. On-board peripheral headers (J5, J11, J12 and J13) | 26. Cypress 512-Mbit serial NOR flash memory (S25HL512T, U11) |
| | 27. microSD Card holder (J9) |

1.7.2 WICED Bluetooth Kit

[CYW920819EVB-02](#) – A single-chip Bluetooth evaluation kit.



- | | |
|---|---|
| 1. HCI UART Jumper (J5) | 17. Reset Button (SW2) |
| 2. Motion Sensor (U2) | 18. Arduino Header (J12) |
| 3. VDDIO Select Jumper (J7) | 19. User Button (SW3) |
| 4. USB Connector for Programming /USB-UART (J6) | 20. User LEDs (D1 and D2) |
| 5. VBATT Select Jumper (J8) | 21. LED Enable Switch (SW4) |
| 6. VPA_BT Select Jumper (J16) | 22. External Antenna Connector (U1.J3) |
| 7. PUART Enable Jumper (J10) | 23. PCB Antenna (U1.A1) |
| 8. SWD/GPIO Switch (SW9) | 24. CYW20819 (U1.U1) |
| 9. Debug Header (J13) | 25. Carrier Module (U1) |
| 10. Thermistor (R30) | 26. Coin Cell Holder (ZB1, bottom side) |
| 11. Thermistor Enable Jumper (J14) | 27. 8-Mb SPI Flash (U6) |
| 12. Peripheral Enable Jumper (J18) | 28. Arduino Header (J4) |
| 13. External LPO Enable Jumper (J19) | 29. WICED Header (J2) |
| 14. VDDIO Current Measurement Jumper (J15) | 30. Arduino Header (J3) |
| 15. Recovery Button (SW1) | 31. WICED Header (J1) |
| 16. Arduino Header (J11) | 32. Motion Sensor Interrupt Test Points (J17) |

1.8 PSoC KitProg Programmer

The programmer firmware on the PSoC 6 development kits is called Cypress KitProg3 (some kits ship with KitProg2 firmware, but we'll show you how to update it in a minute). It runs on a PSoC 5 chip also located on the kit. This firmware talks to your computer via USB and to the PSoC 6 target via a protocol called Serial Wire Debug (SWD). The host application on your computer needs to talk to the programmer to debug the PSoC 6 and to download firmware into the PSoC 6 flash. There are a bunch of different protocols out there for accomplishing this task. However, a few years ago Arm developed a standard called CMSIS-DAP, which has two variants that are implemented in the KitProg firmware (Bulk and DAPLink).

Note: Older versions of KitProg firmware also support HID mode, which we typically don't use anymore.

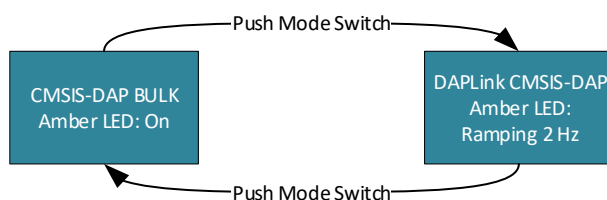
In addition to the CMSIS functions, there is also a function called "Mass storage". When the mass storage functionality is turned on, the programmer appears as a "flash drive" on your computer. You can copy – using the file manager – hex files to the flash drive, which will then be programmed. This function typically runs at the same time as the DAPLink functionality.

The programming firmware typically provides one or more communication bridge modes that allow the PSoC to talk to your PC via I2C and UART. These also typically run at the same time as the programming firmware.

The KitProg will appear to your computer to be multiple USB endpoints that implement each of the functions described in the previous paragraphs.

In order to program the PSoC, KitProg needs to be in the right mode – meaning the mode that has the functionality that works with your environment. You can switch modes by pressing the mode button on the development kit, or by using the [firmware loader program](#). Each PSoC 6 development kit has an LED that will be solid or ramping (~2 Hz) to indicate the mode. See the following table.

CMSIS Mode	Application	Mass Storage	Bridges	SDK	Description	LED
BULK	Eclipse IDE	No	UART I2C	PSoC 6 SDK & AFR SDK	The latest version of the protocol which uses USB bulk mode – by far the fastest.	Solid
DAPLink	Mbed Studio or Mbed CLI	Yes	UART	Mbed OS SDK	A modified version of CMSIS-DAP that enables web debugging	2 Hz Ramping



1.9 PSoC Firmware Loader

Firmware loader (fw-loader) is a tool that we deliver as part of ModusToolbox 2.1. It is a command-line tool that allows you to install new KitProg firmware onto a PSoC 6 kit, and switch modes programmatically.

You can find it in the following directory:

```
<install_dir>/ModusToolbox/tools_<version>/fw-loader/bin
```

It is also available on GitHub at <http://github.com/cypresssemiconductorco/firmware-loader>.

The following table shows the basic fw-loader commands:

Command	Description
<code>fw-loader --help</code> (or no argument)	Prints out help information.
<code>fw-loader --device-list</code>	List all the KitProgs attached to your computer
<code>fw-loader --update-kp3</code>	Install the latest firmware onto your KitProg
<code>fw-loader --mode kp3-daplink</code>	Put the KitProg into DAPLink CMSIS-DAP mode
<code>fw-loader --mode kp3-bulk</code>	Put the KitProg into CMSIS-DAP Bulk mode

You will practice using this tool in exercises [1.12.3](#) and [1.12.4](#).

1.9.1 Starting Command Line Shell

As part of the installation of ModusToolbox, you need to set up the environment to run command-line tools.

- For **Windows**, the installer provides a command-line utility. Navigate to the `modus-shell` directory and run `Cygwin.bat`. It is located in the following directory:

```
<install_path>/ModusToolbox/tools_2.1/modus-shell/
```

- For **macOS**, the installer will detect if you have the necessary tools. If not, it will prompt you to install them using the appropriate Apple system tools.
- For **Linux**, there is only a ZIP file, and you are expected to understand how to set up various tools for your chosen operating system.

1.9.2 Running Firmware Loader

After starting the appropriate shell for your operating system, navigate to the fw-loader executable:

- Windows (Cygwin.bat):**
`cd C:/Users/<your account>/ModusToolbox/tools_2.1/fw-loader/bin`
- macOS (bash or zsh):**
`cd /Applications/ModusToolbox/tools_2.1/fw-loader/bin`
- Linux (bash):**
`cd ~/ModusToolbox/tools_2.1/fw-loader/bin`

Then, run the fw-loader tool using the following command (this will list the help information):

```
./fw-loader
```

```
~/ModusToolbox/tools_2.1/fw-loader/bin
ckf@ORELP8JPO8MF ~/ModusToolbox/tools_2.1/fw-loader/bin
$ ./fw-loader
Cypress Firmware Updater, Version: 2.2.11.491
(C) Copyright 2018-2020 by Cypress Semiconductor
All Rights Reserved

Supported command line parameters:

--help                               Displays this info
--device-list                         Displays list of connected devices
--update-kp3 [device-name|all]       Updates FW of the device with specified name to KitProg3
                                     Device name is taken from the '--device-list' command
                                     Device name can be skipped if only one KitProg device is connected to PC
                                     To update FW of all the connected KitProg devices use 'all' specifier
--update-kp2 [device-name|all]       Updates FW of the device with specified name to KitProg2
                                     Device name is taken from the '--device-list' command
                                     Device name can be skipped if only one KitProg device is connected to PC
                                     To update FW of all the connected KitProg devices use 'all' specifier
--mode <mode> [device-name]         Switches mode of the KitProg3 device with specified name
                                     Supported modes are: 'kp3-hid', 'kp3-bulk', 'kp3-bootloader', 'kp3-daplink'
                                     Device name is taken from the '--device-list' command
                                     Device name can be skipped if only one KitProg device is connected to PC

ckf@ORELP8JPO8MF ~/ModusToolbox/tools_2.1/fw-loader/bin
$
```

You can list devices attached to your computer by using:

```
./fw-loader --device-list
```

```
~/ModusToolbox/tools_2.1/fw-loader/bin
ckf@ORELP8JPO8MF ~/ModusToolbox/tools_2.1/fw-loader/bin
$ ./fw-loader --device-list
Cypress Firmware Updater, Version: 2.2.11.491
(C) Copyright 2018-2020 by Cypress Semiconductor
All Rights Reserved

Info: Start API initialization
Info: Connected - KitProg3 CMSIS-DAP HID-1718126C03227400
Info: Hardware initialization complete (591 ms)
Connected supported devices:
1: KitProg3 CMSIS-DAP HID-1718126C03227400    FW Version 1.13.322

ckf@ORELP8JPO8MF ~/ModusToolbox/tools_2.1/fw-loader/bin
$
```

1.10 WICED Bluetooth Programming

For the Bluetooth kit, there is no KitProg and there is no mode button. The kit operates in USB bulk mode. This kit includes 2 UARTs: one for HCI and one for Peripheral (PUART).

- The HCI UART is used to download the application image, and to communicate between a host/MCU app and the embedded FW app running on the device. This UART uses the WICED HCI protocol.
- The PUART can be used by an application for any serial purpose. It is typically used to output trace messages.

The ChipLoad application takes a .hex file application image and downloads it over the WICED HCI UART.

1.11 UARTs and Serial Terminals

The KitProg firmware in some of the modes will enumerate as one or more UARTs (good old-fashioned serial ports). The WICED Bluetooth board also enumerates as two UARTs as discussed previously. They will appear differently based on the operating system that you are using.

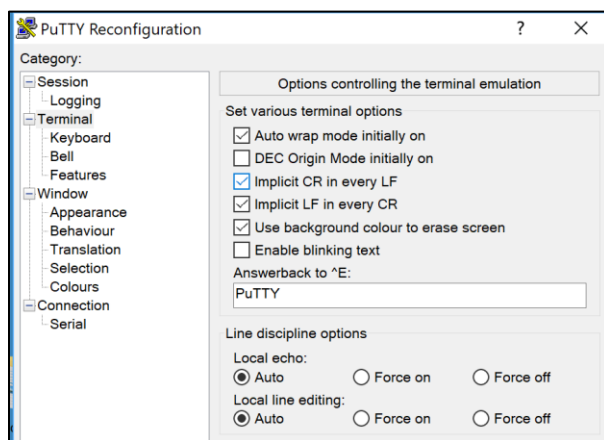
You will need to know the baud rate, which will depend on what YOU programmed into the firmware. It will probably be 9600 (for Mbed OS), 115200 (for PSoC 6), 115200 (for Amazon FreeRTOS), 115200 (Bluetooth PUART), 3000000 (Bluetooth HCI).

In the world of serial terminals there are two commonly used characters:

- “carriage return” symbolized in C code by ‘\r’
- “new line” symbolized in C code by ‘\n’

Oftentimes, developers only put new line in their code and rely on the terminal program to automatically supply a carriage return. As such, every terminal program out there has a function/option to automatically handle this for you.

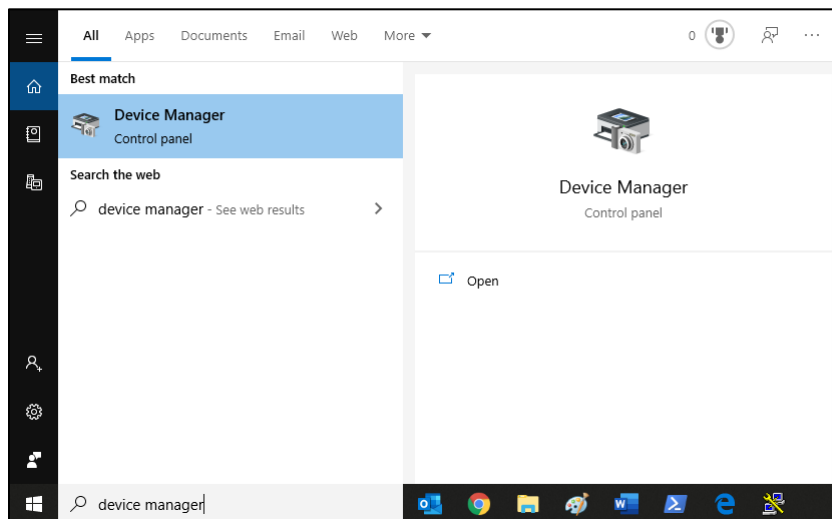
In addition, both the PSoC retarget-io and the Mbed OS printf have methods for doing this translation automatically. Here are the settings from PuTTY. Notice the option to “Implicit CR in every LF” (LF means line feed).



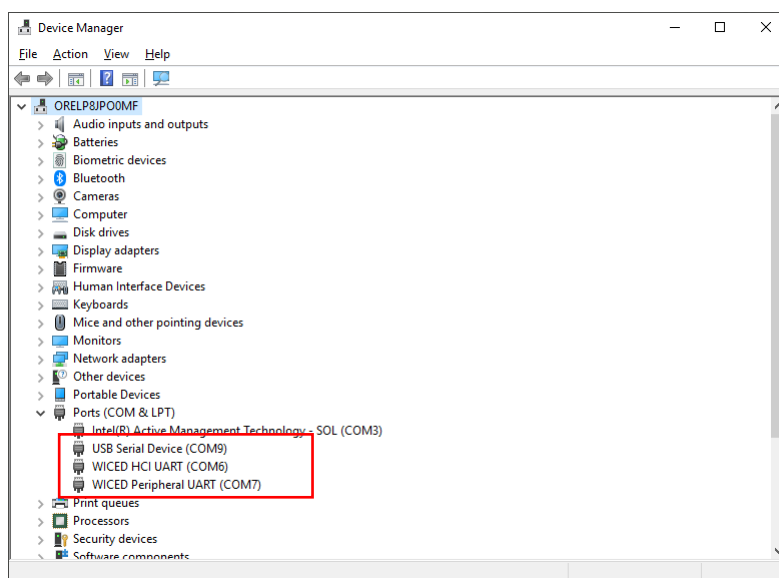
The Bluetooth SDK firmware has a `wiced_printf()` function that is a custom version of `printf()`; it is used with `WICED_BT_TRACE` (and other) macros for debug trace output to the debug output destination, usually either direct serial text output to PUART (displayable by standard serial emulators) or encoded output send to the `WICED_UART` port for decoding and display by BTSpy.

1.11.1 Windows

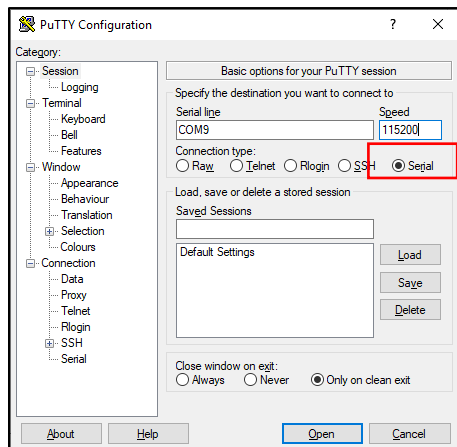
On Windows, we recommend that you use PuTTY to attach to the serial ports, but other serial terminals such as TerraTerm may also be used. In order to do that you will need to know which “COM” port to attach to. To find the port number, run the Device Manager by searching for the “Device Manager” on the Windows Start menu.



When you look under “Ports” you should see your device. In the PSoC 6 case it is “USB Serial Device”; for the Bluetooth case, you’ll see two ports: one for HCI and one for PUART.



When you run PuTTY, click the “Serial” radio button and then type the COM port and baud rate. Use the appropriate COM port number for your system and device. Note that you can save the settings if you want for the next time.

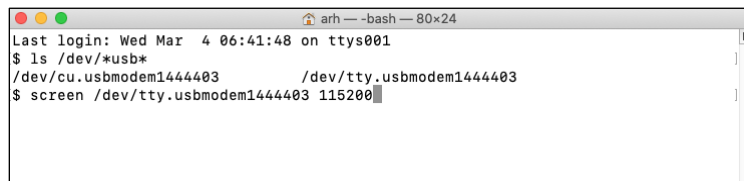


1.11.2 macOS

Screen Program

On macOS, you can also use the "screen" program to attach to a serial port. First, find the device of the tty by typing:

```
ls /dev/*usb*
```



Then attach using the device name and baud rate:

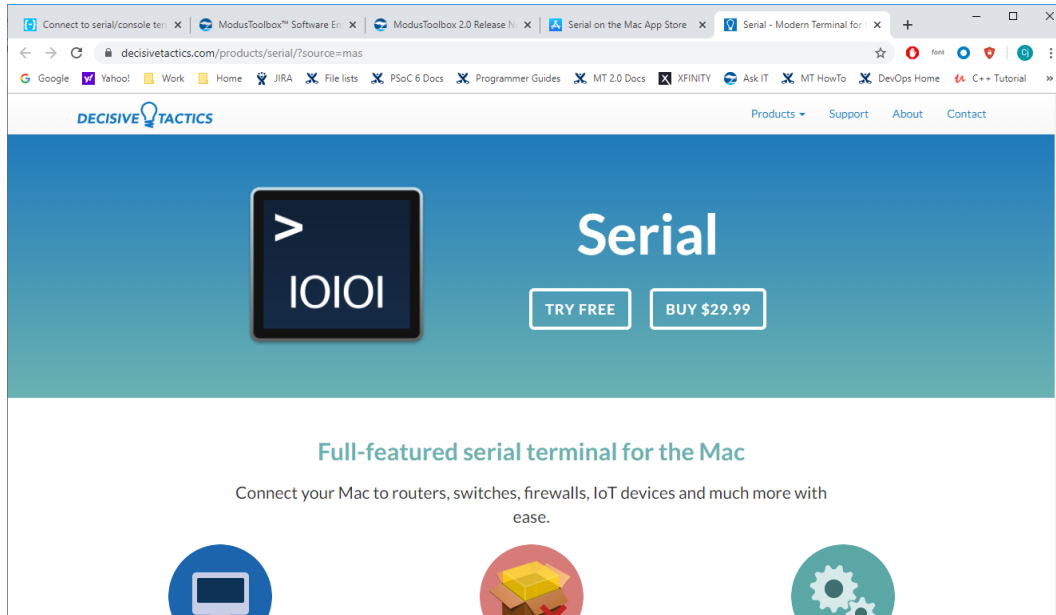
```
screen /dev/tty.usbmodem1444403 115200
```

You can then quit by pressing **Ctrl + A** then **Ctrl + D**.

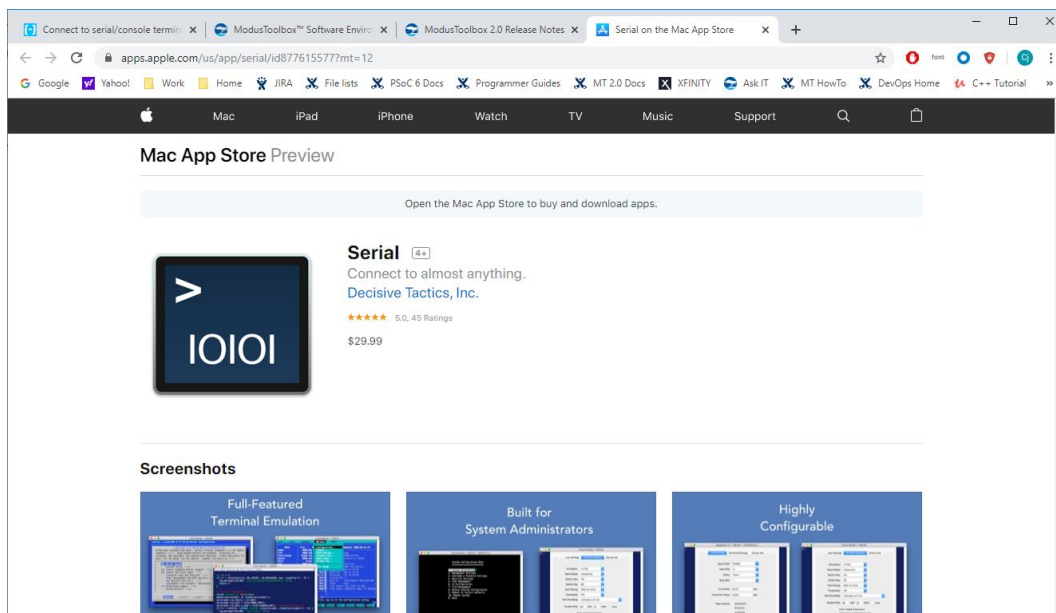
Serial App

On macOS, you can also use a program like the Serial.app to attach to a serial port. You can purchase it directly from the creator or on the Mac App Store. Here are the links:

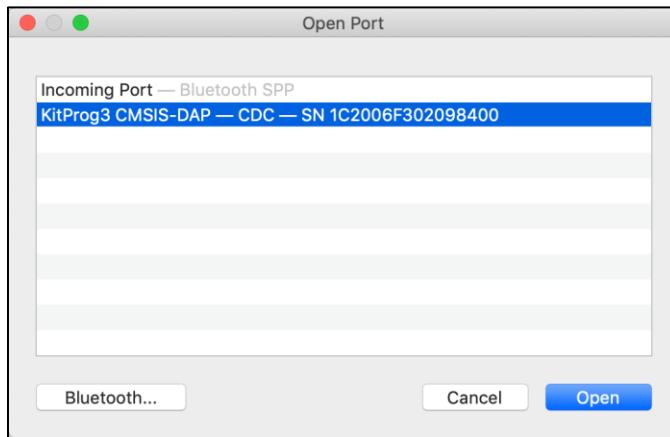
Creator's website: <https://www.decisivetactics.com/products/serial/>



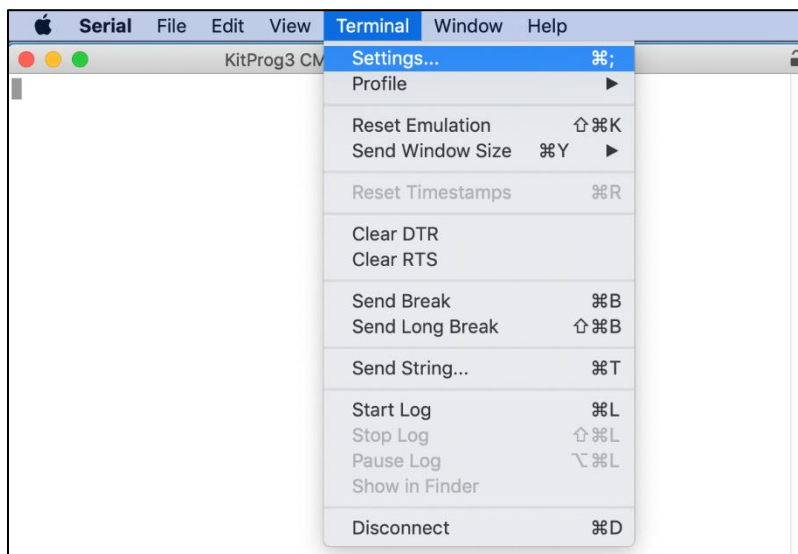
Mac App Store: <https://apps.apple.com/us/app/serial/id877615577?mt=12>



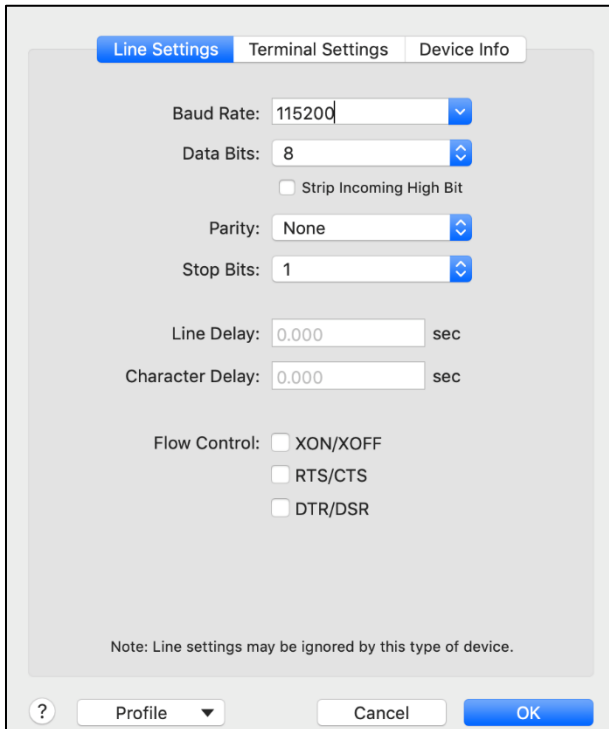
1. After installing it, run the Serial app to open the port selector. This shows the attached board (CY8CKIT-062-WIFI-BT) is set to CMSIS-DAP mode:



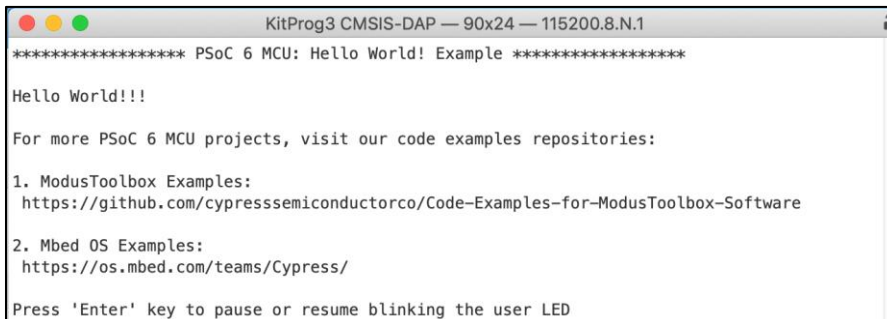
2. Select the kit and click **Open**.
3. Click on the **Terminal > Settings** menu to configure the serial interface:



4. Configure settings (same values as Windows):



5. Click **OK** to apply. On a brand-new kit, you should see something like this:



1.11.3 Linux

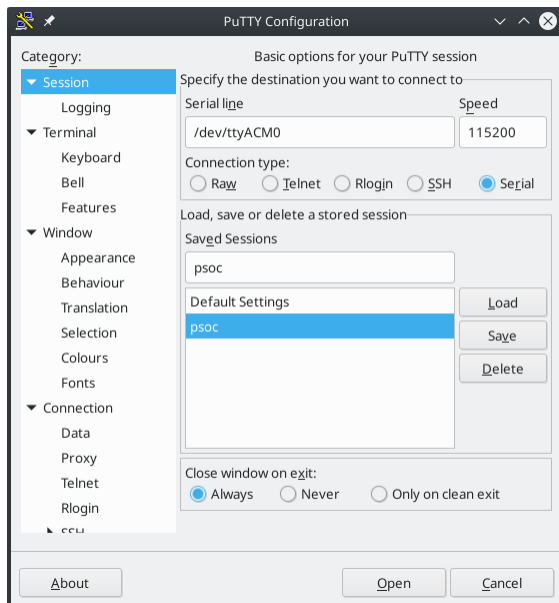
You can use PuTTY on Linux similar to Windows. The only difference is that the device name is in the form of `/dev/ttyACMn`, where `n` is usually a small number. To get a list of such devices, type the following on the command line:

```
ls /dev/ttyACM*
```

This should return something like this:

```
crw-rw----+ 1 root plugdev 166, 0 Dec  3 13:20 /dev/ttyACM0
```

Select the **Serial** option, and copy the device name into the PuTTY window:



Note On common Linux distributions, the serial UART ports belong to the root user and to the dialout and plugdev groups. Standard users are not allowed to access these devices. To fix this, you'll need to run this command:

```
sudo usermod -a -G dialout,plugdev $USER
```

1.12 Exercises

1.12.1 Verify the Software

Before class, you should have already downloaded and installed the following software. Refer to the Installation Instructions provided via email. Make sure the software is installed and configured correctly.



ModusToolbox 2.1 Installer

- Can you start the IDE?
- Can you create a workspace?
- Can you run ModusToolbox Command Line?
 - **For Windows:** Run `cygwin.bat` from the `ModusToolbox/tools_2.1/modus-shell` folder.
 - **For macOS/Linux:** Open a terminal window and type `make`.



Mbed Studio

- Can you start the IDE?



Mbed CLI

- Can you run the command `mbed` in Cygwin or a terminal window?

See “Troubleshooting” section in the Mbed OS chapter.

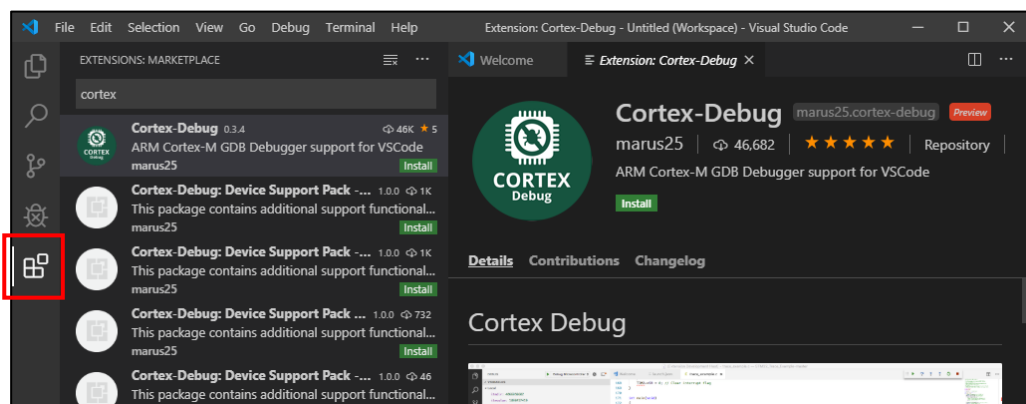
If using macOS, see also

<https://iotexpert.com/2020/03/02/stupid-python-tricks-install-mbed-cli-on-macos/>

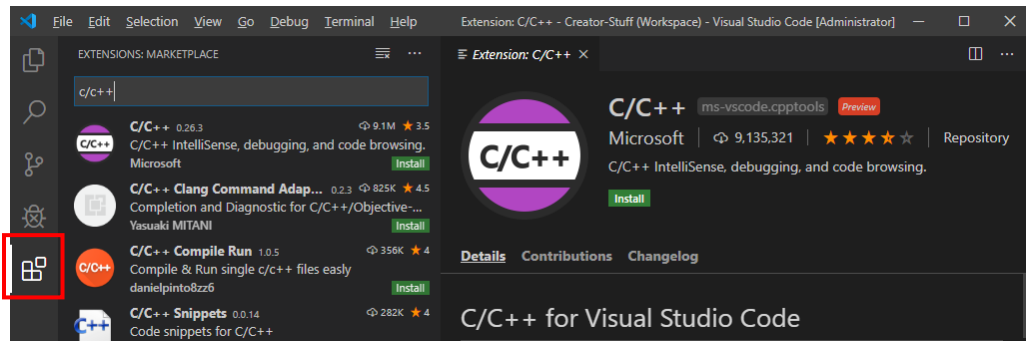


Visual Studio Code

- Can you start the IDE?
- Can you install the Cortex-Debug extension?



- Can you install the C/C++ tools extension?



1.12.2 Network Proxy Settings

Depending on the location of this class you may need to set up your network proxy settings. This will vary from site to site. When creating applications from inside a Cypress internal network, some proxy settings are necessary due to our security settings. This behavior will be improved in the future. The steps (for now) are:

Proxies by Site

Find the proxy host name and port number for your site. As of the writing of this manual, the proxies are as follows. There is a list at: <https://itsm.cypress.com/solutions/726105.portal>.

Site	Proxy Server (FQDN)	IP Address	Port
Japan	jpkwswwp03.cysemi.com	10.128.174.5	74
	jpkwswwp04.cysemi.com	10.128.174.6	74
Thailand	thbkkwwp01.cysemi.com	10.249.155.66	74
	thbkkwwp02.cysemi.com	10.249.155.67	74
Frankfurt	dceuwwp03.cysemi.com	10.244.7.219	74
	dceuwwp04.cysemi.com	10.244.63.131	74
Munich	eumunwwp02.cysemi.com	10.244.63.130	74
Malaysia	pngwwp01.cysemi.com	10.249.20.2	74
	pngwwp02.cysemi.com	10.249.20.3	74
Taiwan	tpewwp01.cysemi.com	10.240.112.66(Taipei)	74
	tpewwp01.cysemi.com	10.240.128.66(Hsinchu)	74
Israel	isr-mwg01.cysemi.com	10.168.251.60	74
	isr-mwg02.cysemi.com	10.168.251.61	74
Austin	corp-webw105.cysemi.com	10.248.251.78	74
	corp-webw106.cysemi.com	10.248.251.79	74
Korea	seowwp01.cysemi.com	10.240.64.58	74
San Jose	csj-mwg01.cysemi.com	10.198.170.5	74
	csj-mwg02.cysemi.com	10.198.170.6	74
KYCC	kycc-mwg01.cysemi.com	10.198.32.58	74
WADC	cywa-mwg01.cysemi.com	10.198.0.18	74
CCOS	ccos-wg.cysemi.com	192.168.75.12	74
INDC	indc-mwg01.cysemi.com	10.249.64.19	74
	indc-mwg02.cysemi.com	10.249.64.19	74
CHDC	chdc-mwg01.cysemi.com	10.240.160.18	74
	chdc-mwg02.cysemi.com	10.240.160.19	74

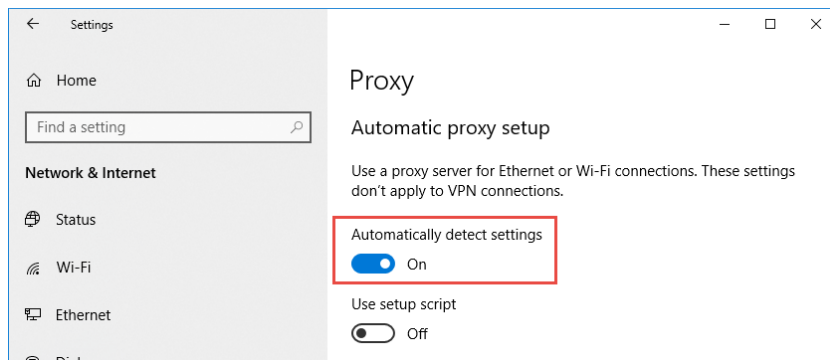
Site	Proxy Server (FQDN)	IP Address	Port
CML	cml-wg01.cysemi.com	10.249.224.231	74
	cml-wg02.cysemi.com	10.249.224.232	74
UKR	ukr-mwg01.cysemi.com	172.23.194.115	74
	ukr-mwg02.cysemi.com	172.23.194.116	74
Coresite	dmz-webw101.cypress.com	10.247.88.203	74
	dmz-webw102.cypress.com	10.247.88.204	74
MSK	jpmskwwp01.cysemi.com	10.128.207.66	74
	jpmskwwp02.cysemi.com	10.128.207.67	74
ORDC	bigbro.cysemi.com	172.23.172.33	74

Create OS specific proxy and variable updates:

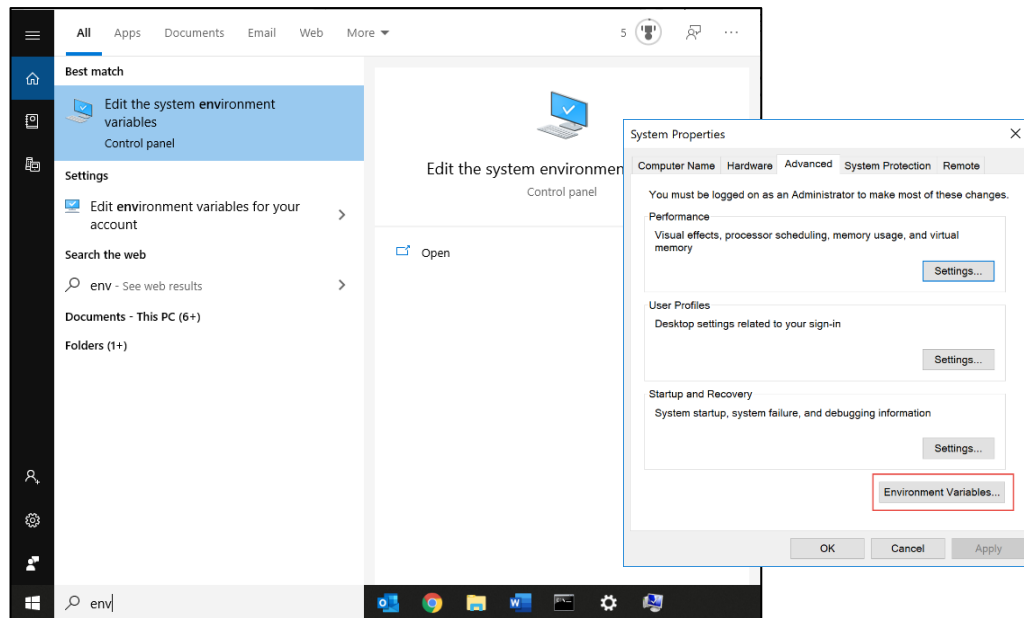
Windows:



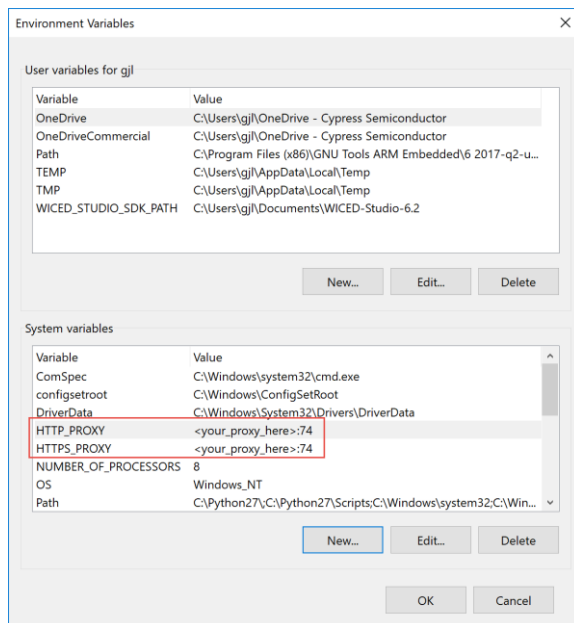
1. Set the Windows proxy settings (**Settings > Network & Internet > Proxy**) to “Automatically Detect Settings.”

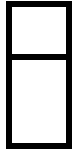


2. Open the Environment Variables dialog.



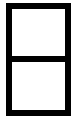
3. Create two Windows system variables (HTTP_PROXY and HTTPS_PROXY) using the appropriate proxy name and port number (see previous table):



macOS:

1. Set **System Preferences > Network > Advanced > Proxies > Auto Proxy Discovery**.
2. Open a terminal and check if the variables (http_proxy, https_proxy) are set in the terminal. For example:

```
$ export | grep proxy
declare -x http_proxy="bigbro.ore.cypress.com:74/"
declare -x https_proxy="bigbro.ore.cypress.com:74/"
```

Linux:

1. Set the "Network Proxy" to "Manual".
2. Open a terminal and check if the variables are set in the terminal. For example:

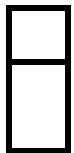
```
$ export | grep proxy
declare -x http_proxy="bigbro.ore.cypress.com:74/"
declare -x https_proxy="bigbro.ore.cypress.com:74/"
```



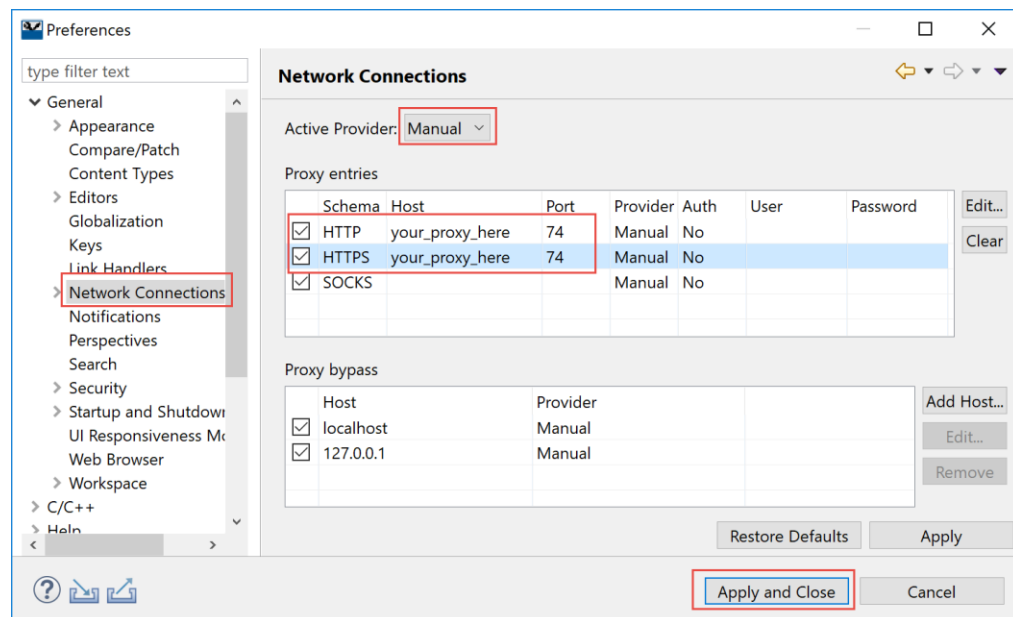
3. Setup Cypress CA Certificate.

This is Cypress-specific; see

<https://confluence.cypress.com/pages/viewpage.action?spaceKey=MIDDLEWARE&title=Development+environment#Developmentenvironment-SetupCypressCACertificates>.

Start the Eclipse IDE

1. Open menu item **Window > Preferences > General > Network Connections**.
2. Set "Active Provider" to "Manual" and set HTTP and HTTPS to the correct proxy host and port for your location.



1.12.3 Run the fw-loader Tool

See section [1.9 PSoC Firmware Loader](#) for details.



- Check the Kitprog firmware version on the kits.

```
./fw-loader --device-list
```



- Does a device show up? What mode is it in? Update to the latest version.

```
./fw-loader --update-kp3
```

1.12.4 Switch KitProg Modes and Verify the Mode

See section [1.9 PSoC Firmware Loader](#) for details.



- Switch between all the KitProg modes using the Mode Switch button.



- Switch between the modes using the fw-loader tool.

```
./fw-loader --mode <mode>
```



- Verify that you know “what you are doing” by using the fw-loader tool.

```
./fw-loader --device-list
```



- When you are done, put the kit into CMSIS-BULK mode

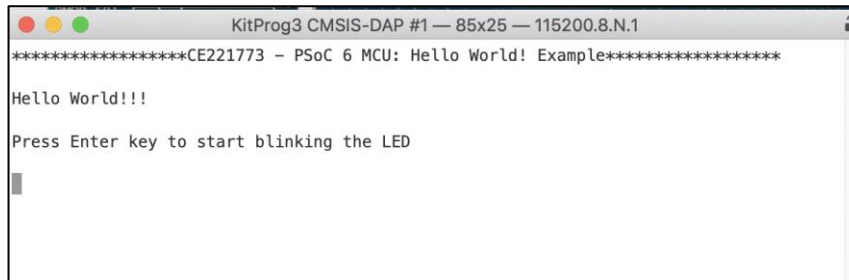
```
./fw-loader -mode kp3-bulk
```

The LED should be solid ON.

1.12.5 Use a terminal program to attach to the UART on the CY8CPROTO-062-4343W



- On a brand-new kit, you should see something like this (baud rate 115200):



1.12.6 Use a terminal program to attach to the UART on the CY8CKIT-062-WIFI-BT kit



- On a brand-new kit, you should see something like this (baud rate 115200):

```

Cypress KitProg3 (CMSIS-DAP, Mass Storage) — 80x24 — 115200.8.N.1

Starting WICED vWiced_006.001.000.0066
Platform CY8CKIT_062 initialised
Started ThreadX v5.8
Initialising NetX_Duo v5.10_sp3
Creating Packet pools
WLAN MAC Address : A0:C9:A0:46:91:E7
WLAN Firmware : wl0: Nov 8 2017 20:47:26 version 7.45.98.40 (r677271 CY) FWI
D 01-85d87950
WLAN CLM : API: 12.2 Data: 9.10.39 Compiler: 1.29.4 ClmImport: 1.36.3 Cr
eation: 2017-11-08 20:36:12
*****
Using another device connect to the following WiFi network
SSID : WICED Config
Password: 12345678
Open a web browser and go to http://192.168.0.1
On the page click the Wi-Fi Setup button. Select your WiFi network
type in the password, press connect
*****
IPv4 network ready IP: 192.168.0.1
Setting IPv6 link-local address
IPv6 network ready IP: FE80:0000:0000:0000:A2C9:A0FF:FE46:91E7
  
```

1.12.7 Use a terminal program to attach to the UART on the CYW920819EVB-02 kit



- On a brand-new kit, you should see something like this (baud rate 115200):

```

PuTTY (inactive)

-----
CE226300 BLE Environmental Sensing Service Application
-----
This application measures voltage on the selected DC channel
every 5000 milliseconds (configurable) and displays
the measured temperature via UART.
-----

Discover this device with the name: "Thermistor"
Bluetooth Device Address: 20 81 9a 14 f6 34

Bluetooth Management Event:      BTM_ENABLED_EVT

GATT status:      WICED_BT_GATT_SUCCESS || WICED_BT_GATT_ENCRYPED_MITM
ds1:0x00501400, len:0x0003dc00
ds2:0x0053f000, len:0x00001000
Active DS:501400 vs1:500400 vs2:501400

Bluetooth Management Event:      BTM_BLE_ADVERT_STATE_CHANGED_EVT

Advertisement state changed to BTM_BLE_ADVERT_UNDIRECTED_HIGH
Starting undirected BLE advertisements successful

Temperature (in degree Celsius)      22.49
This device is not connected to any BLE central device

Temperature (in degree Celsius)      22.47
This device is not connected to any BLE central device

Bluetooth Management Event:      BTM_BLE_ADVERT_STATE_CHANGED_EVT

Advertisement state changed to BTM_BLE_ADVERT_UNDIRECTED_LOW

Temperature (in degree Celsius)      22.47
  
```