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?

This class is a survey of the Cypress IoT development platform ModusToolbox 2.0. 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 2.0 is a set of Reference Flows, Products, and SDKs that enable an immersive development experience for customers creating converged MCU and Wireless systems. ModusToolbox 2.0 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.

1.5 Tools

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

- ModusToolbox Eclipse-based IDE
- 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
- CYMCUELF

1.6 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)

1.7 SDK

A Software Development Kit (SDK) is a collection of ModusToolbox Products that support one or more specific Reference Flows. ModusToolbox 2.0 supports the following SDKs, which are covered in separate chapters in this training:

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

1.8 Software

All this software needs to be installed prior to class, using the USB stick. See the Install the Software section later in this chapter for more details.

Tool	Description
ModusToolbox 2.0	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
Notepad ++	A popular simple lightweight code editor for Windows

1.9 PSoC KitProg Programmer

The programmer firmware on the PSoC 6 development kits is called Cypress KitProg3. 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 three variants that are all implemented in the KitProg firmware (Bulk, HID, and DAPLink).

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, UART and SPI. 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, fast ramping (~1 Hz), or really fast ramping (~2 Hz) to indicate the mode. See the following table.

Mode	Application	Mass Storage	Bridges	SDK	Description	LED
CMSIS-DAP BULK	MTB 2.0 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
CMSIS-DAP HID	MTB 2.0 IDE	No	UART I2C	PSoC 6 SDK & AFR SDK	The original protocol which uses USB HID – so it is slower.	1 Hz ramping
DAPLink CMSIS-DAP	Mbed Studio or Mbed CLI	Yes	UART	Mbed OS SDK	A modified version of CMSIS-DAP that enables web debugging	2 Hz ramping

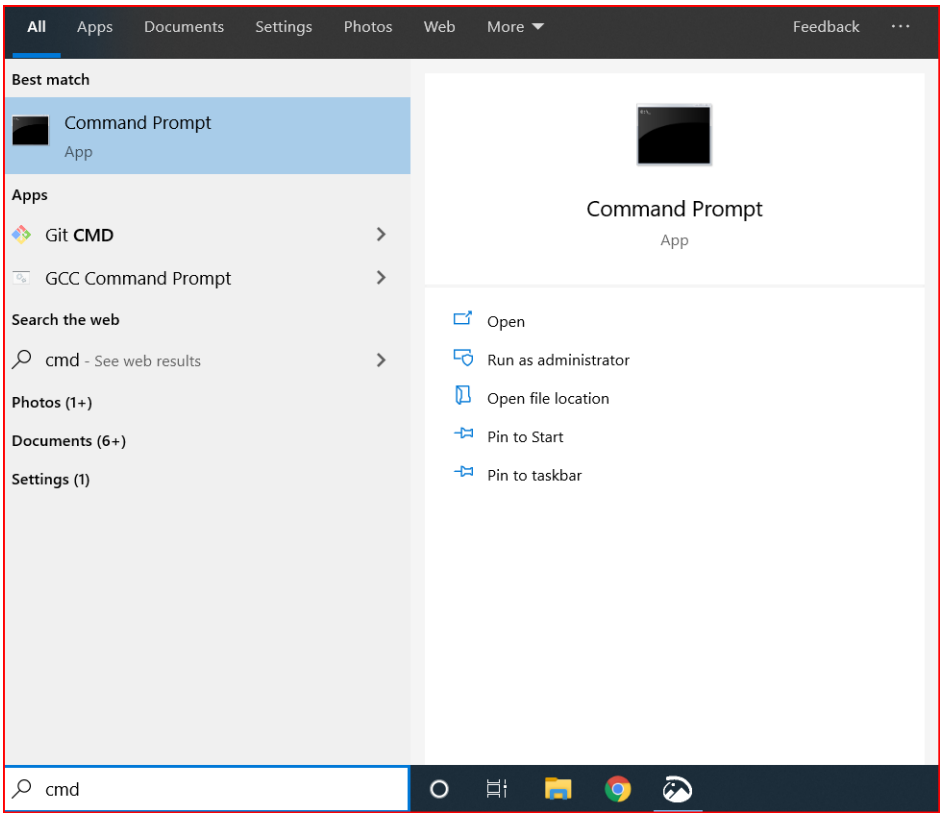
1.10 PSoC Firmware Loader

Firmware loader is a tool that we deliver as part of ModusToolbox 2.0. It is also available on GitHub at http://github.com/cypresssemiconductorco/Firmware-loader. This is a command line tool that allows you to install new KitProg firmware onto a PSoC 6 kit.

Command	Description
`fw-loader --device-list`	List all the KitProgs attached to your computer
`fw-loader --update-kp3`	Install the latest firmware onto your KitProg
`fw-loader --mode kp3-daplink`	Put the KitProg into DAPLink CMSIS-DAP mode
`fw-loader --mode kp3-hid`	Put the KitProg into CMSIS-DAP HID mode
`fw-loader --mode kp3-bulk`	Put the KitProg into CMSIS-DAP Bulk mode

1.10.1 Windows

On Windows, you can run the program from the Power Shell or command prompt. Go to the Windows search box and type `cmd`. When you see the "Command Prompt" App, click it.

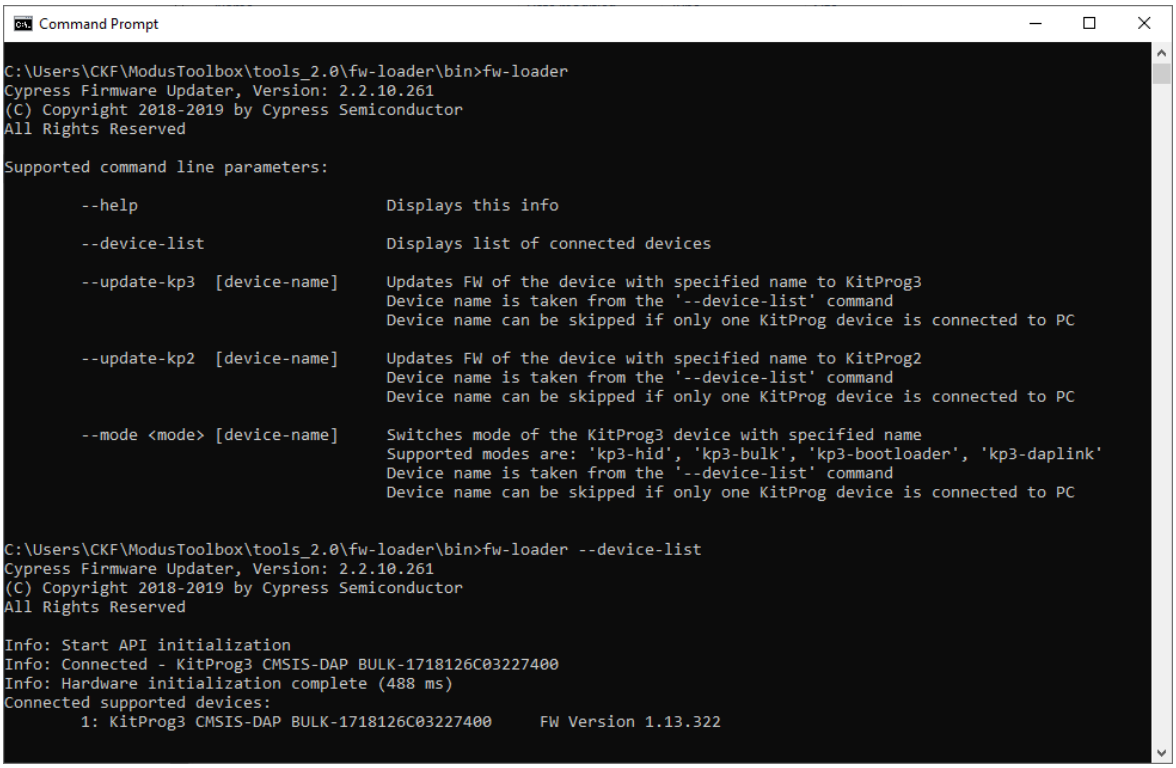


In the Command window, type the following to navigate to the installation directory:

```
cd C:\Users\<your account>\ModusToolbox\tools_2.0\fw-loader\bin
```

Then, run the fw-loader tool using the following command:

```
fw-loader
```



1.10.2 macOS

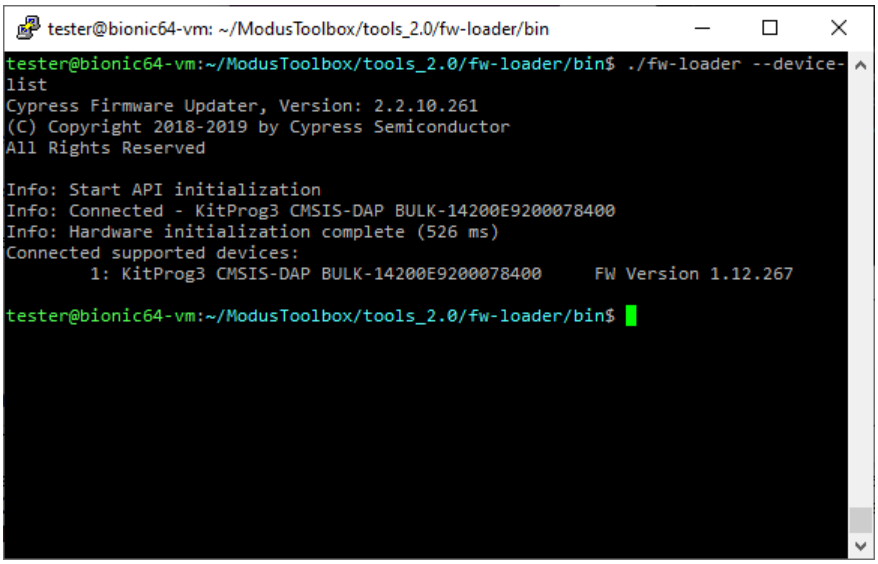
On macOS, start a terminal program and type the following:

```
cd /Applications/ModusToolbox/tools_2.0/fw-loader/bin
./fw-loader
```

1.10.3 Linux

On Linux, start a terminal program and type the following:

```
cd ~/ModusToolbox/tools_2.0/fw-loader/bin
./fw-loader
```



1.11 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.

1.12 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.

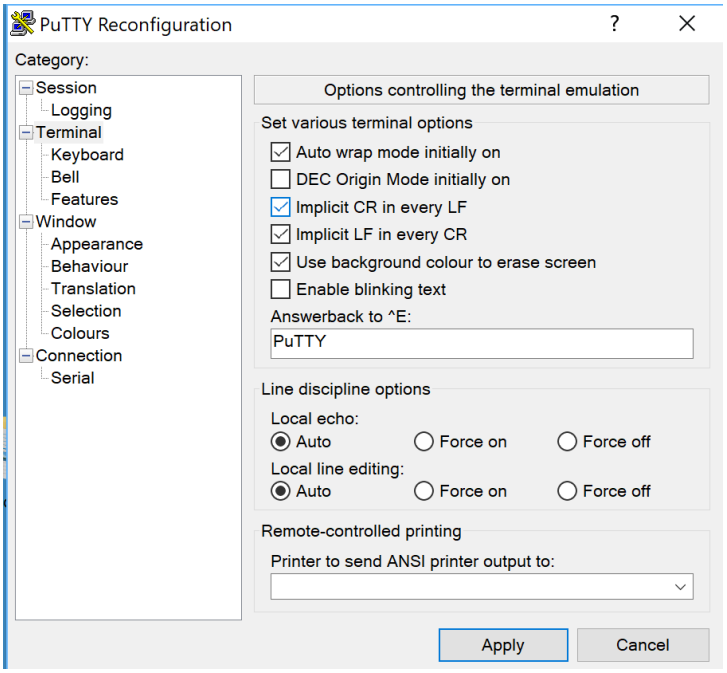
You will need to know the baud rate, which will depend on what YOU programmed into the firmware. It will probably be 9600 (for Mbed OS), 115200 (for PSoC 6), 115200 (for Amazon FreeRTOS), 115200 (Bluetooth PUART), 3000000 (Bluetooth HCI).

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.

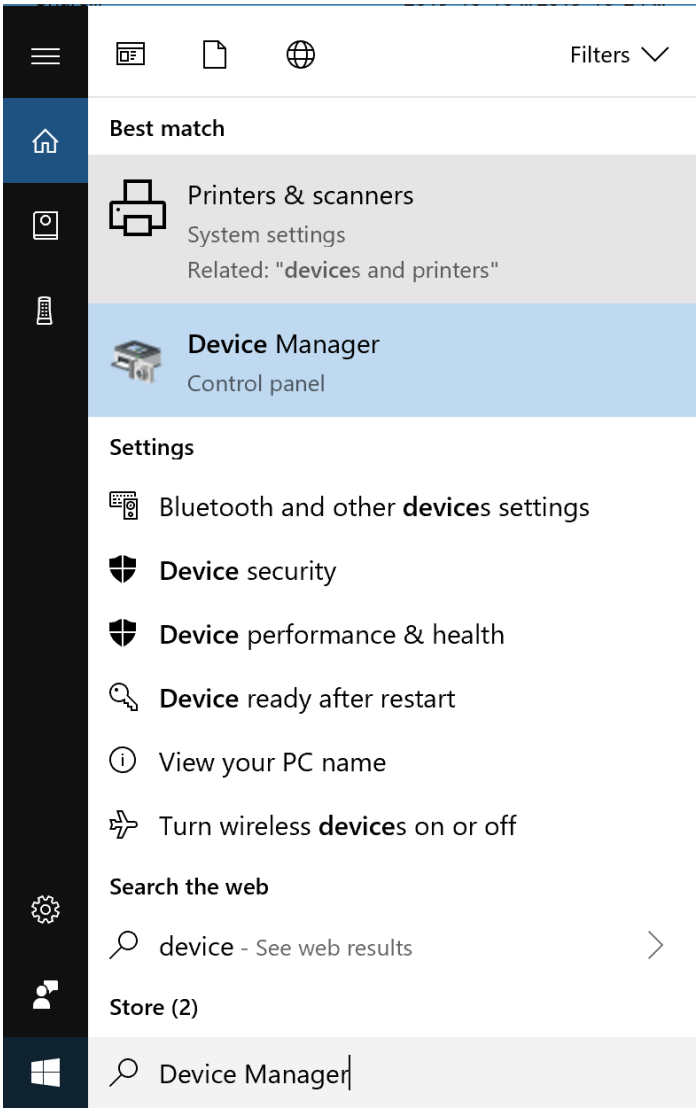
In addition, both the PSoC retarget-io and the Mbed OS printf have methods for doing this translation automatically. Here are the settings from PuTTY. Notice the option to “Implicit CR in every LF” (LF means line feed).



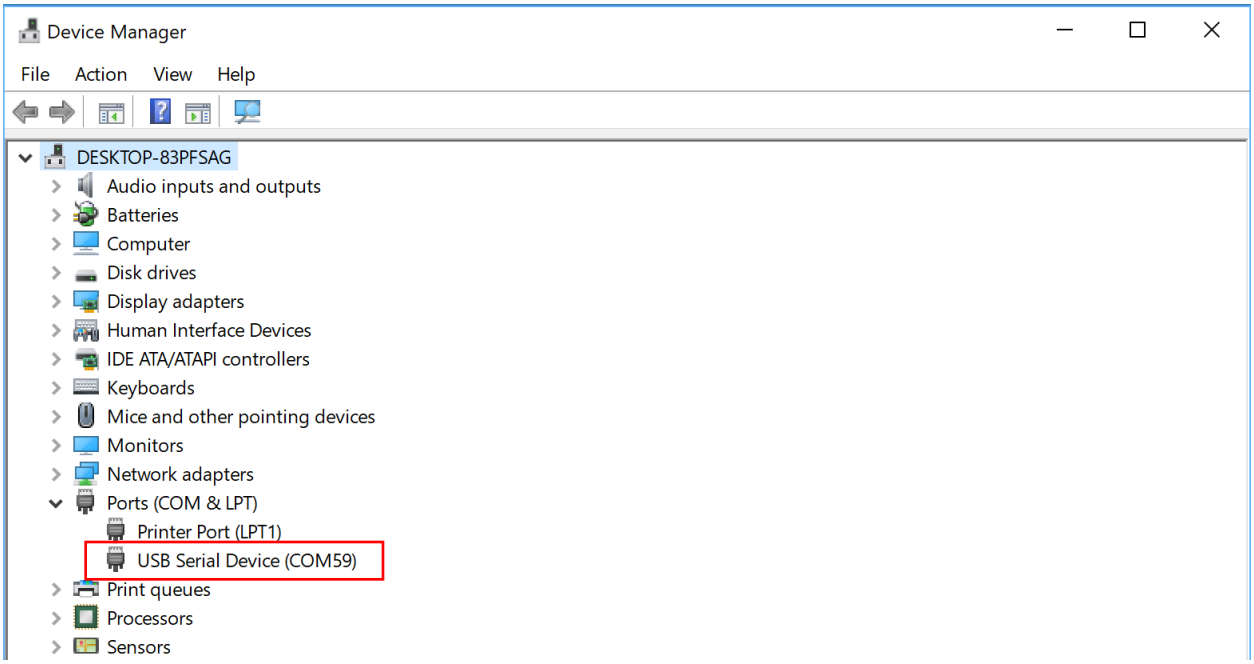
The Bluetooth SDK firmware has a wiced_printf() function that is a custom version of printf(); it is used with WICED_BT_TRACE (and other) macros for debug trace output to the debug output destination, usually either direct serial text output to PUART (displayable by standard serial emulators) or encoded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

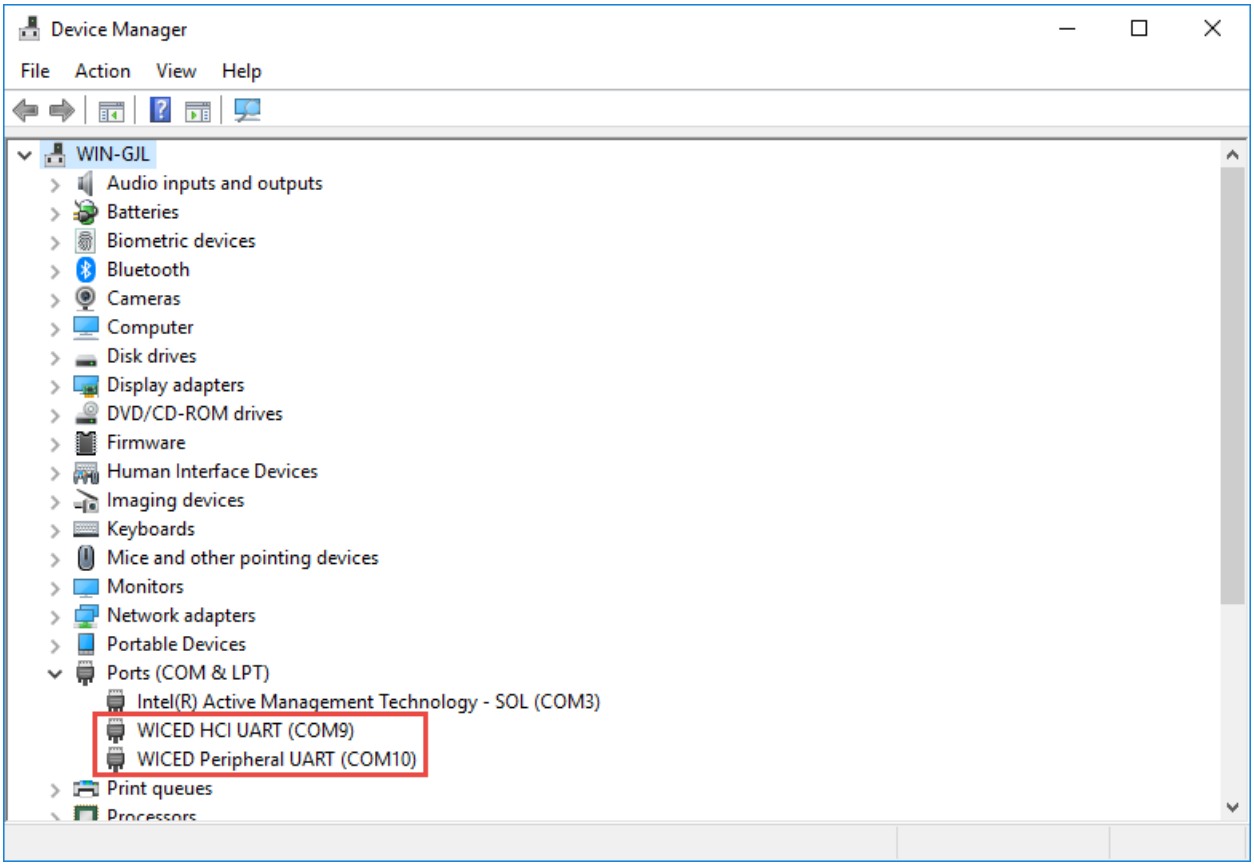
1.12.1 Windows 10

On Windows 10, 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.

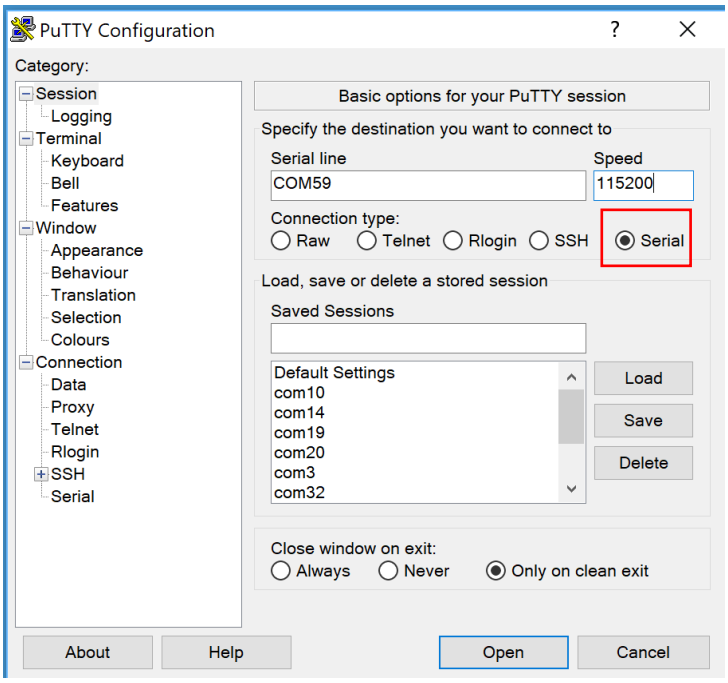


When you look under “Ports” you should see your device. In the PSoC 6 case it is “USB Serial Device”; for the Bluetooth case, you’ll see two ports: one for HCI and one for PUART.





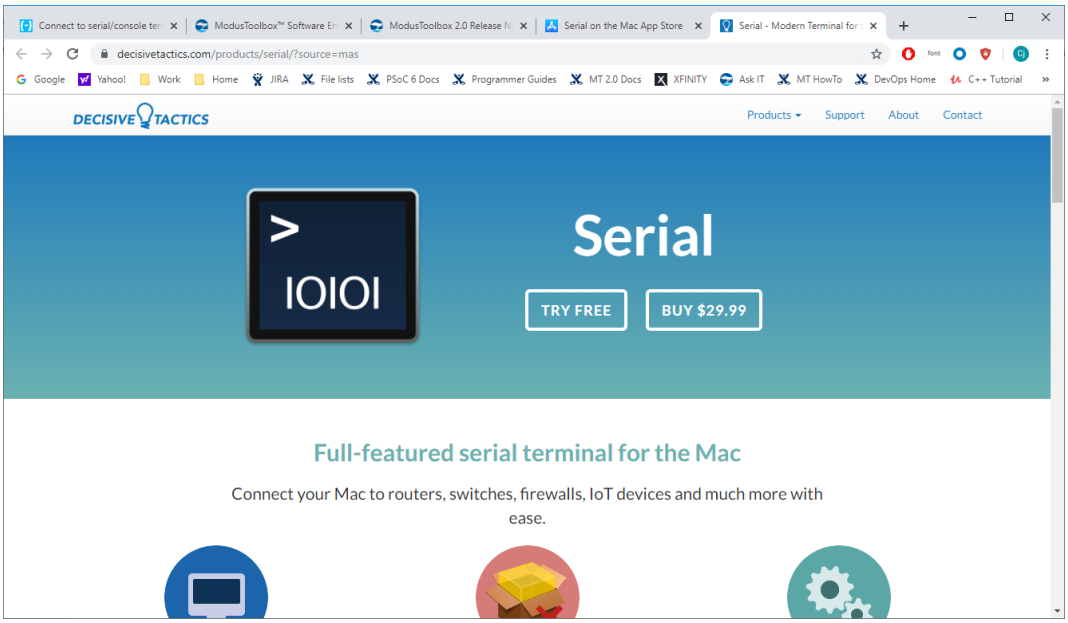
When you run PuTTY, click the “Serial” radio button and then type the COM port and baud rate. Use the appropriate COM port number for your system and device. Note that you can save the settings if you want for the next time.



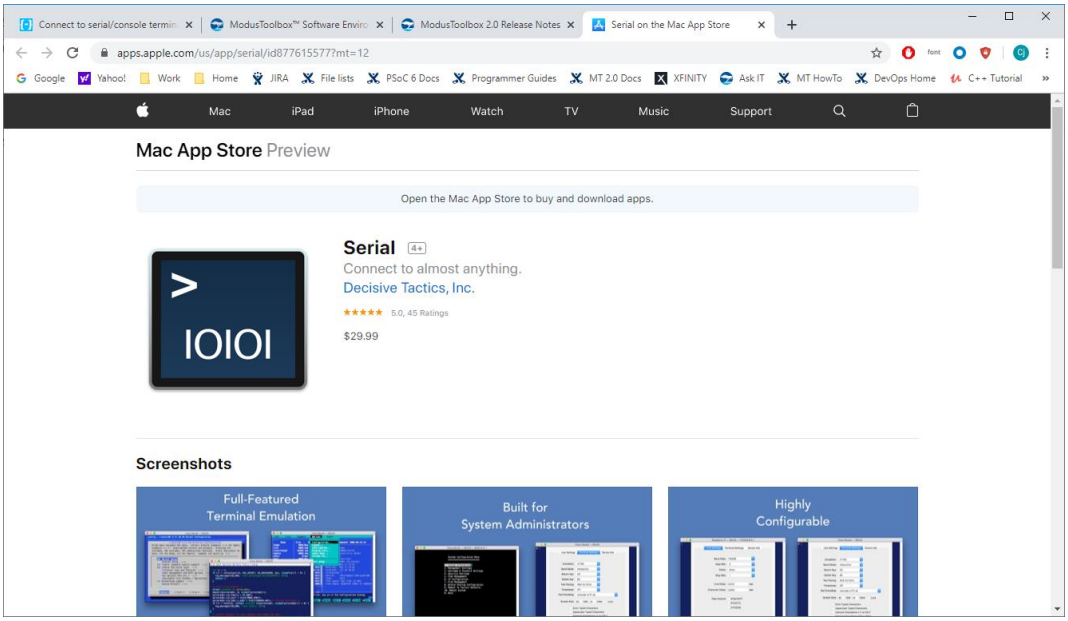
1.12.2 macOS

On macOS, you can 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:

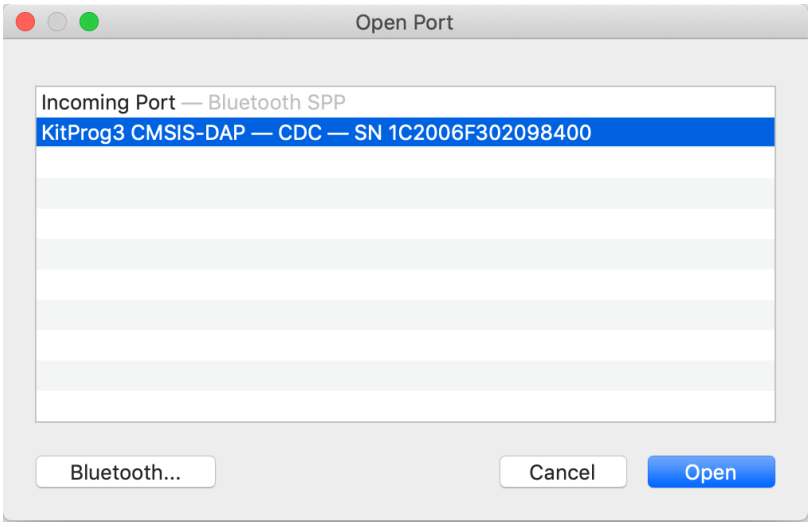
Creator's website: https://www.decisivetactics.com/products/serial/



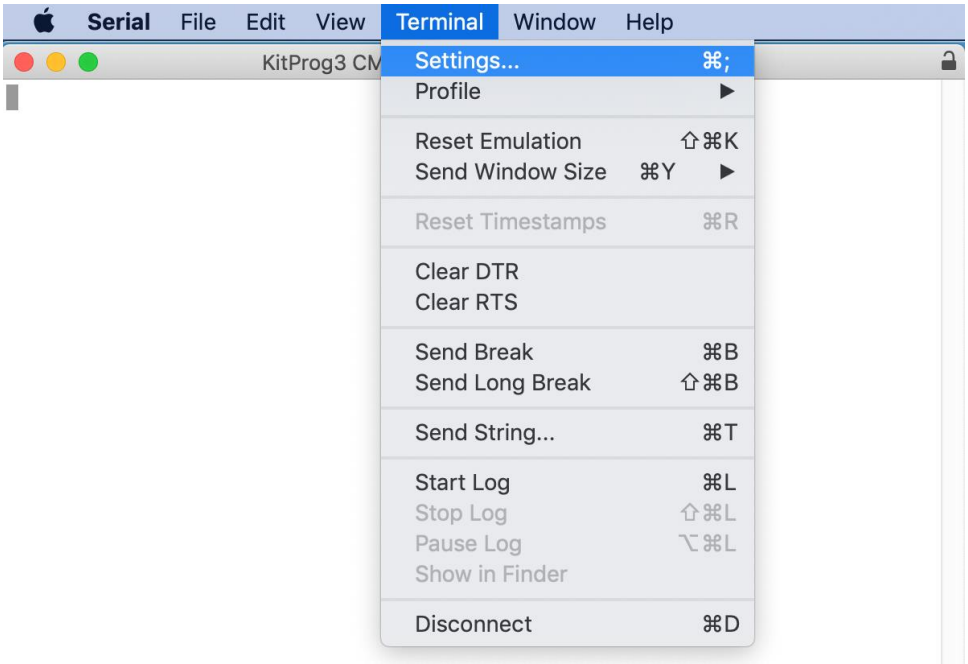
Mac App Store: https://apps.apple.com/us/app/serial/id877615577?mt=12



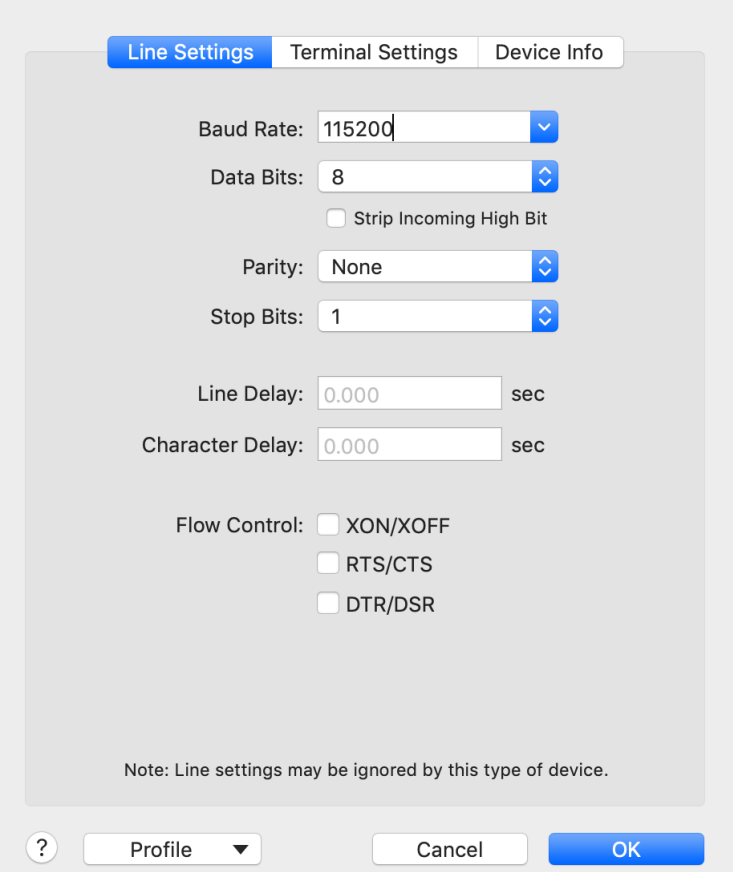
Open the port selector. This shows the attached board (CY8CKIT-062-WIFI-BT) is set to CMSIS-DAP mode:



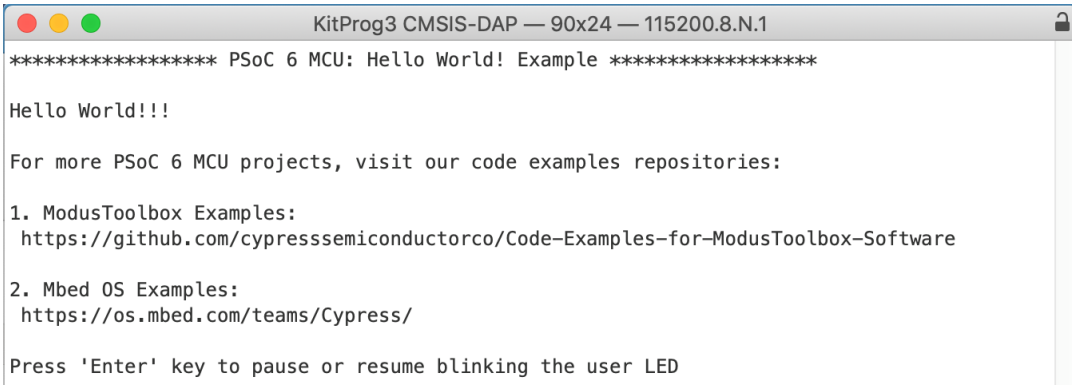
Open the **Terminal > Settings** menu to configure the serial interface:



Configure settings (same values as Windows):



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



1.12.3 Linux

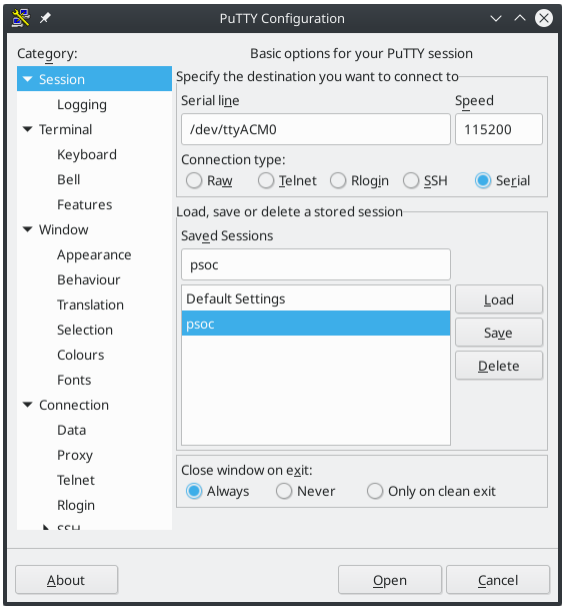
You can use PuTTY on Linux similar to Windows. The only difference is that the device name is in the form of /dev/ttyACMn, where n is usually a small number. To get a list of such devices, type the following on the command line:

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

This should return something like this:

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

Copy the device name into the PuTTY window:



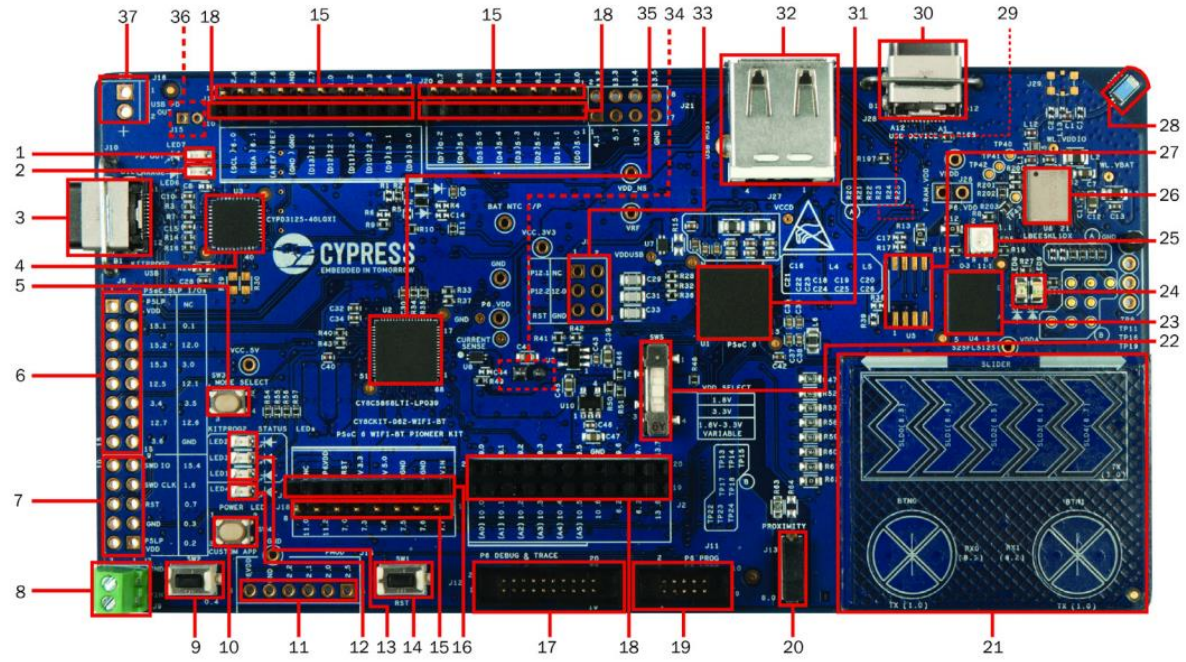
Note On common Linux distributions, the serial UART ports belong to the root user and to the dialout and plugdev groups. Standard users are not allowed to access these devices. To fix this, you'll need to run this command:

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

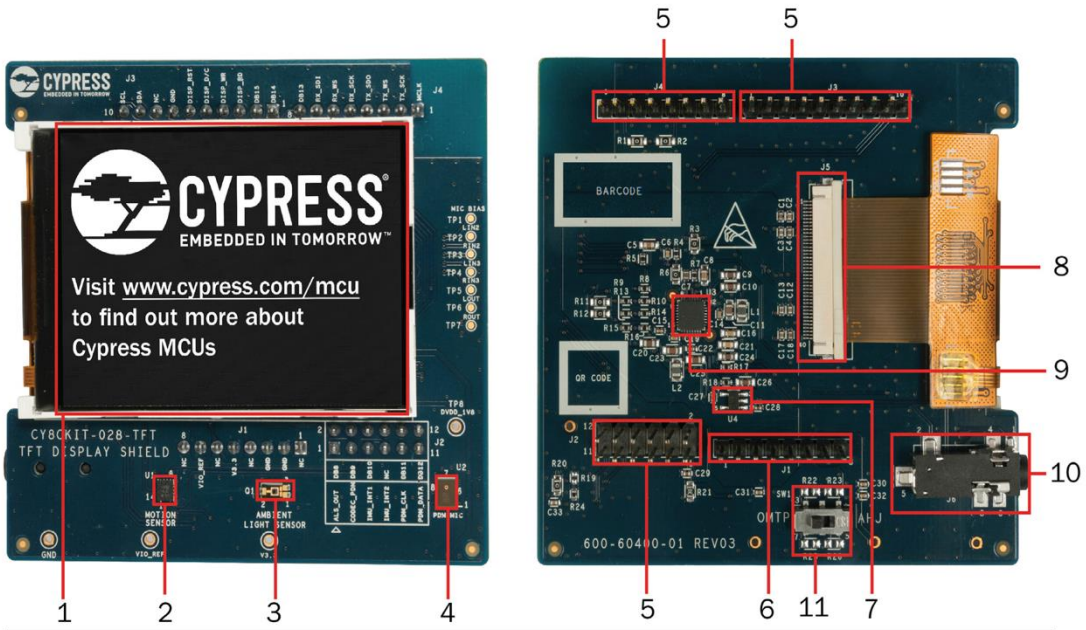
1.13 Development Kits

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

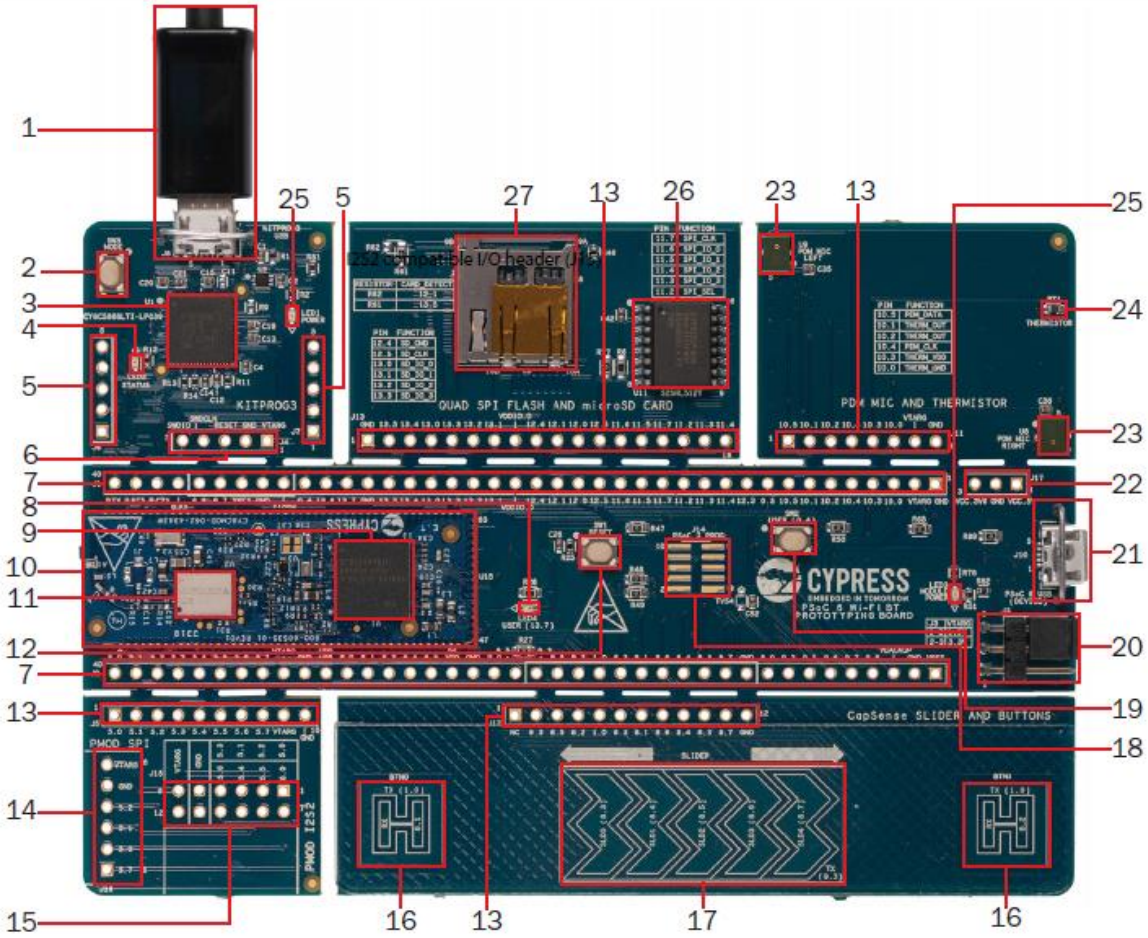
CY8CKIT_062_WiFi_BT – A PSoC 6-1M and a CYW4343W Bluetooth WiFi Combo



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

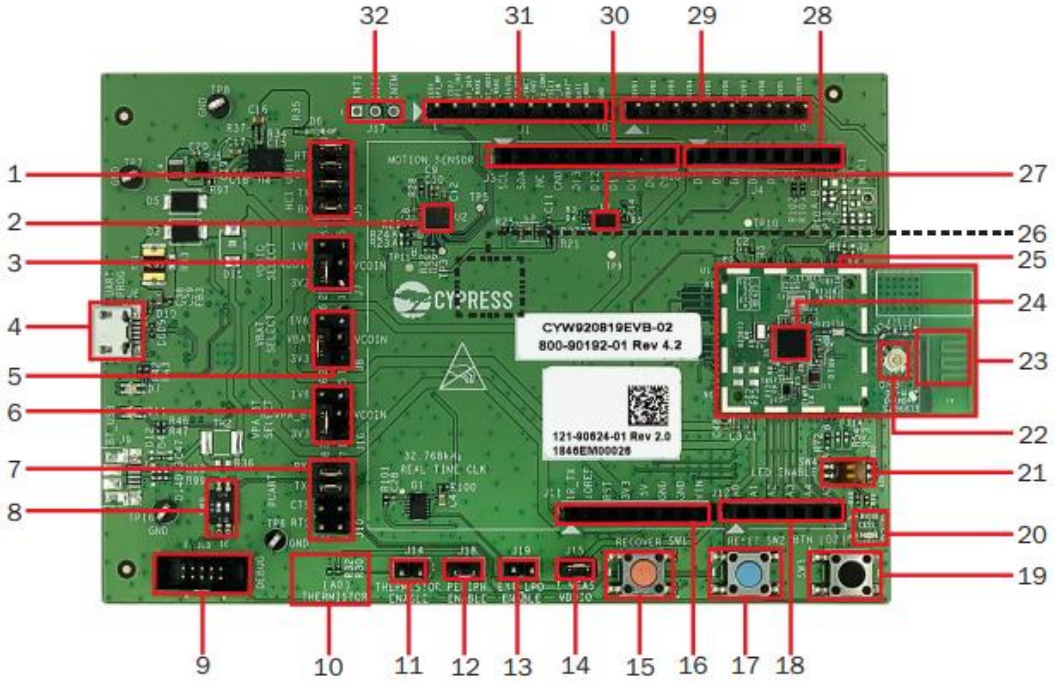


1. 2.4-inch TFT display
2. Motion Sensor (U1)
3. Ambient Light Sensor (Q1)
4. PDM microphone (U2)
5. Ardunio compatible I/O headers (J2, J3, J4)
6. Ardunio compatible power header (J1)
7. TFT display power control load switch (U4)
8. TFT display connector (J5)
9. Audio CODEC (U3)
10. Audio Jack (J6)
11. Audio Jack Selection (OMTP/AHJ) Switch (SW1)



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

CYW920819EVB-02 – A single-chip Bluetooth evaluation kit.



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

1.14 Exercises

1.14.1 Install the Software

Before class, install and verify the following software using the provided USB Stick:

ModusToolbox 2.0 Installer

See instructions at http://www.cypress.com/ModusToolboxInstallGuide

- 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_2.0/modus-shell folder.
    - For macOS/Linux: Type `make`.

Mbed Studio

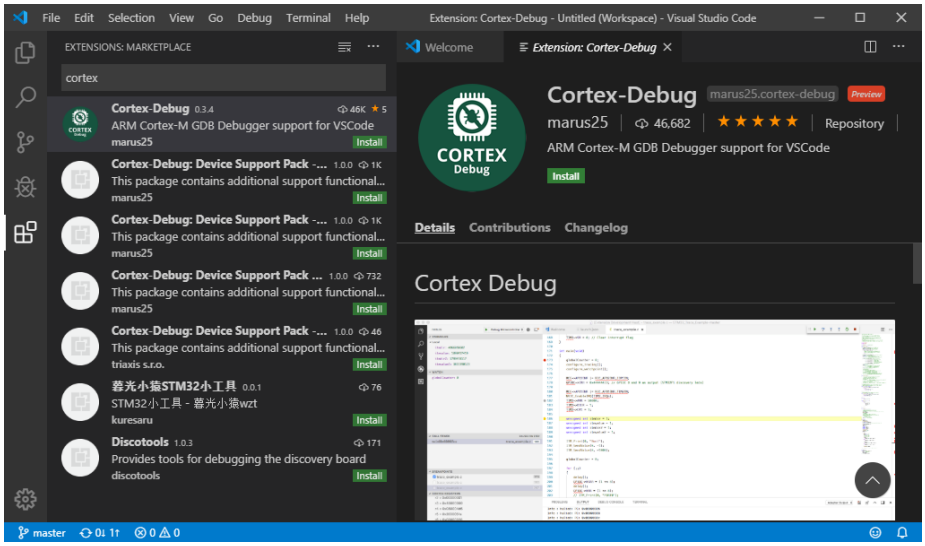
See instructions at https://os.mbed.com/docs/mbed-studio/0.5/introduction/index.html

- Can you start the IDE?

Visual Studio Code

See instructions at https://code.visualstudio.com/docs

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



Mbed CLI

See instructions at https://os.mbed.com/docs/mbed-os/v5.14/quick-start/offline-with-mbed-cli.html

- What happens when you run the command `mbed` in a terminal window?

1.14.2 Run the fw-loader Tool

- [ ] 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
```

1.14.3 Switch KitProg Modes and Verify the Mode

- [ ] Switch between all the KitProg modes using the Mode Switch button.
- [ ] Switch between the modes using the fw-loader tool.

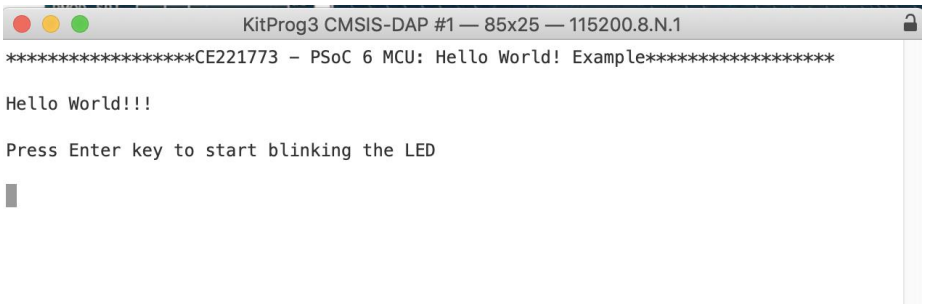
```
fw-loader --mode <mode>
```

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

```
fw-loader --device-list
```

1.14.4 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:



1.14.5 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:

1.14.6 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: