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 IoT development platform ModusToolbox 2.2. 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 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 one of our courses such as PSoC 101, Wi-Fi 101, and 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 2.2 is a set of Reference Flows, Products, and Solutions 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.
- Solutions in ModusToolbox are built as simply as possible using normal C & C++ programming best principles and a consistent architecture.

1.3 What is IoT-AdvantEdge™?

Building intelligent products for the IoT is not easy. Providing security and integrating cloud applications is difficult, power management is tough, and getting wireless and embedded systems to work together can be a nightmare. IoT-AdvantEdge gets products to market faster and more cost-effectively with the confidence of proven solutions that bring together the essential building blocks of the IoT.

Much more information about IoT-AdvantEdge and how all the pieces fit together can be found at: https://www.cypress.com/solutions/iot-advantedge.

1.4 Reference Flows

A Reference Flow is a documented, supported, and qualified **methodology** for a customer to use a solution 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 solution to create some PSoC 6 product. A Reference flow could be a path only through a solution, or it might include other external tools that are not part of the solution (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".

1.5 Products

ModusToolbox Products are Tools and Firmware that can be used individually, or as a group, to develop connected applications for our devices. Unlike previous 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.

1.5.1 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
- FTDI driver

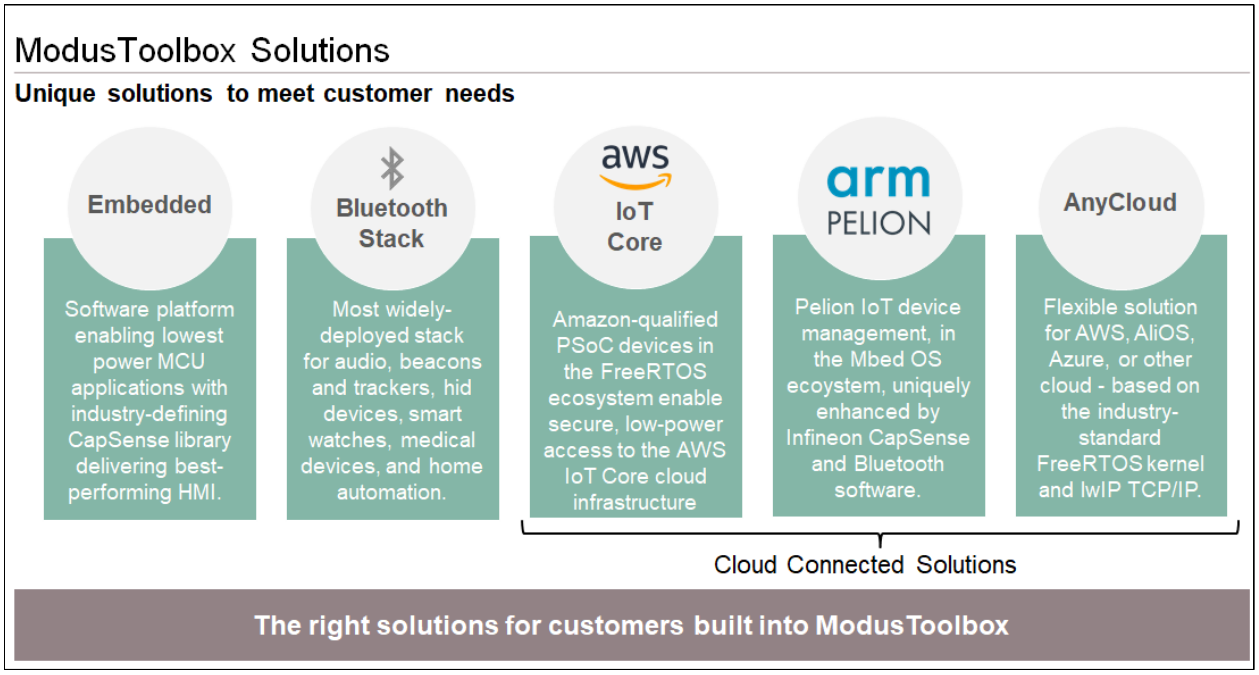
1.5.2 Firmware

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 Code Example Application Firmware (i.e. their project or our code example)

1.6 Solutions

A solution is a collection of ModusToolbox Products that support one or more specific Reference Flows. A solution may be released as static collection of products or it may be created by dynamically pulling in what is necessary for a given application. ModusToolbox 2.2 supports the following cloud solutions, which are covered in separate chapters in this training:


ModusToolbox Solutions
Unique solutions to meet customer needs
Embedded
Software platform enabling lowest power MCU applications with industry-defining CapSense library delivering best-performing HMI.
Bluetooth Stack
Most widely-deployed stack for audio, beacons and trackers, hid devices, smart watches, medical devices, and home automation.
aws IoT Core
Amazon-qualified PSoC devices in the FreeRTOS ecosystem enable secure, low-power access to the AWS IoT Core cloud infrastructure
arm PELION
Pelion IoT device management, in the Mbed OS ecoystem, uniquely enhanced by Infineon CapSense and Bluetooth software.
AnyCloud
Flexible solution for AWS, AliOS, Azure, or other cloud - based on the industry-standard FreeRTOS kernel and lwIP TCP/IP.
Cloud Connected Solutions
The right solutions for customers built into ModusToolbox

- PSoC 6 solution for AnyCloud
- Mbed solution for Pelion
- FreeRTOS solution for Amazon IoT Device
- WICED Bluetooth solution (BT SDK)

1.7 Software

All this software needs to be installed prior to class. Refer to the Installation Instructions provided in the file MTB2-x-SW-Install-Instructions.pdf. See also exercise 1.12.1 to verify that the software is installed correctly.

Tool	Description
ModusToolbox	Our kick butt development toolbox.
Mbed CLI	Python programs which allow you to use the ARM Mbed Command Line Interface. Along with Mbed CLI, you will install Git, Mercurial, and GCC which are required. Mbed CLI will be installed in a Python virtual environment using pipenv.
Mbed Studio	An ARM GUI for writing code and building Mbed OS Projects.
Python	Python is used by ModusToolbox and Mbed.
Visual Studio Code	The fastest growing code editor with debug extension on the market, which is quickly displacing Eclipse.
IAR Embedded Workbench (optional)	Full featured IDE with native support for PSoC 6.
Keil µVision (optional)	Full featured IDE from Arm with native support for PSoC 6.

The ModusToolbox 2.2 installer will install the tools, but not any firmware such as BSPs, code examples, middleware libraries, or even solution 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.

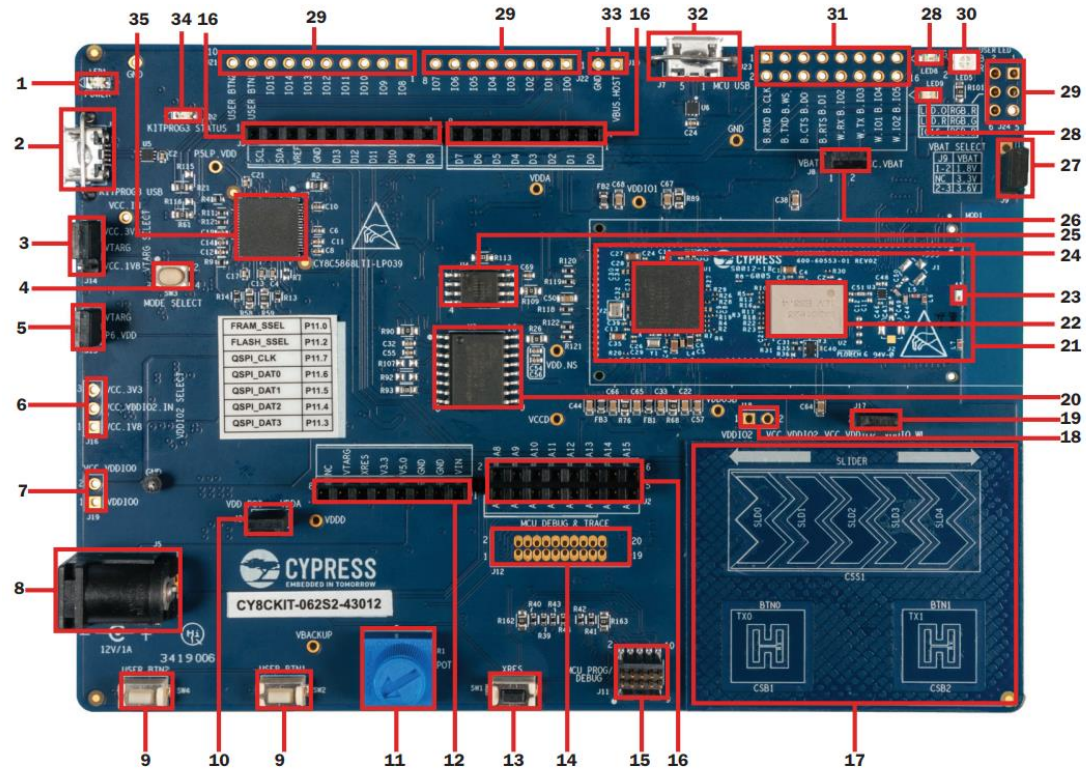
Note: You will need some type of code editor, such as Notepad++; however, the Eclipse IDE for ModusToolbox and Visual Studio Code include code editors as well. You will also need some type of serial terminal, such a PuTTY or Serial. See section 1.11 for more details.

1.8 Development Kits

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

1.8.1 PSoC 6 Kits

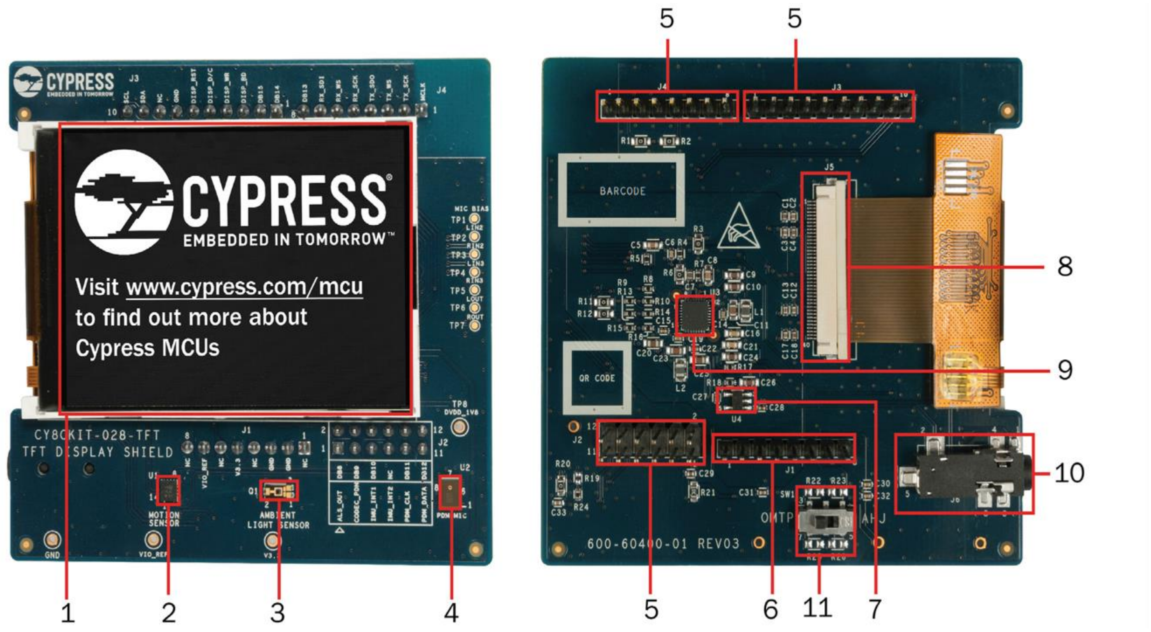
CY8CKIT-062S2-43012 – A PSoC 6-2M and a CYW43012 WiFi + BT Combo



1. Power LED (LED1)
2. KitProg3 USB connector (J6)
3. PSoC 6 MCU VDD power selection jumper (J14)
4. KitProg3 programming mode selection button (SW3)
5. PSoC 6 MCU VDD current measurement jumper (J15)
6. PSoC 6 MCU VDDIO2 and CYW43012 VDDIO power selection jumper (J16)
7. PSoC 6 MCU VDDIO0 current measurement jumper (J19)
8. External power supply VIN connector (J5)
9. PSoC 6 MCU user buttons (SW2 and SW4)
10. Potentiometer connection jumper (J25)
11. Potentiometer (R1)
12. Arduino-compatible power header (J1)
13. PSoC 6 MCU reset button (SW1)
14. PSoC 6 MCU debug and trace header (J12)
15. PSoC 6 MCU program and debug header (J11)
16. Arduino Uno R3-compatible I/O headers (J2, J3, and J4)
17. CapSense slider (SLIDER) and buttons (BTN0 and BTN1)
18. PSoC 6 MCU VDDIO2 current measurement jumper (J18)
19. CYW43012 VDDIO current measurement jumper(J17)
20. Cypress serial NOR flash memory (S25FL512S, U3)
21. Cypress PSoC 6 (2M) with CYW43012 Carrier Module (CY8CMOD-062S2-43012, MOD1)
22. CYW43012 based Murata Type 1LV module
23. Wi-Fi/BT antenna
24. PSoC 6 MCU
25. Cypress serial Ferroelectric RAM (CY15B104QSN, U4)
26. CYW43012 VBAT current measurement jumper (J8)
27. CYW43012 VBAT power selection jumper (J9):
28. PSoC 6 MCU user LEDs (LED8 and LED9)
29. PSoC 6 I/O header (J21, J22, J24)
30. RGB LED (LED5)
31. Wi-Fi/BT GPIO header (J23)
32. PSoC 6 USB device connector (J7)
33. Optional USB Host power supply header (J10)
34. KitProg3 status LED (LED2)
35. KitProg3 (PSoC 5LP) programmer and debugger (CY8C5868LTI-LP039, U2)
36. microSD Card holder (J20) (on back of board)

1.8.2 Arduino Compatible Shield

CY8CKIT-028-TFT



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)

1.9 PSoC KitProg Programmer

The programmer firmware on the PSoC 6 development kits is called KitProg3 (some kits ship with KitProg2 firmware, but we'll show you how to update it later). 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.

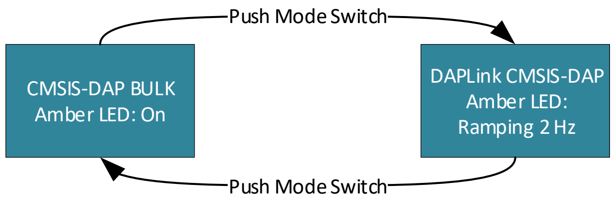
In addition to the CMSIS functions, there is also a function called "Mass storage". When the mass storage functionality is turned on, the programmer appears as a "flash drive" on your computer. You can copy – using the file manager – hex files to the flash drive, which will then be programmed. This function 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	Solution	Description	LED
BULK	Eclipse IDE	No	UART I2C	PSoC 6 & AFR	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	A modified version of CMSIS-DAP that enables web debugging	2 Hz Ramping



1.10 PSoC Firmware Loader

Firmware loader (fw-loader) is a tool that we deliver as part of ModusToolbox 2.2. 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 some of the basic fw-loader commands:

Command	Description
`fw-loader --help` (or no argument)	Print out help information.
`fw-loader --device-list`	List all the KitProg devices 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-bulk`	Put the KitProg into CMSIS-DAP Bulk mode.
`fw-loader --mode kp3-hid`	Put the KitProg into CMSIS-DAP HID mode (not typically used).
`fw-loader --mode kp3-bootloader`	Put the KitProg into bootloader mode (not typically used).

You will practice using this tool in exercises 1.12.2 and 1.12.3.

1.10.1 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_<version>/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.

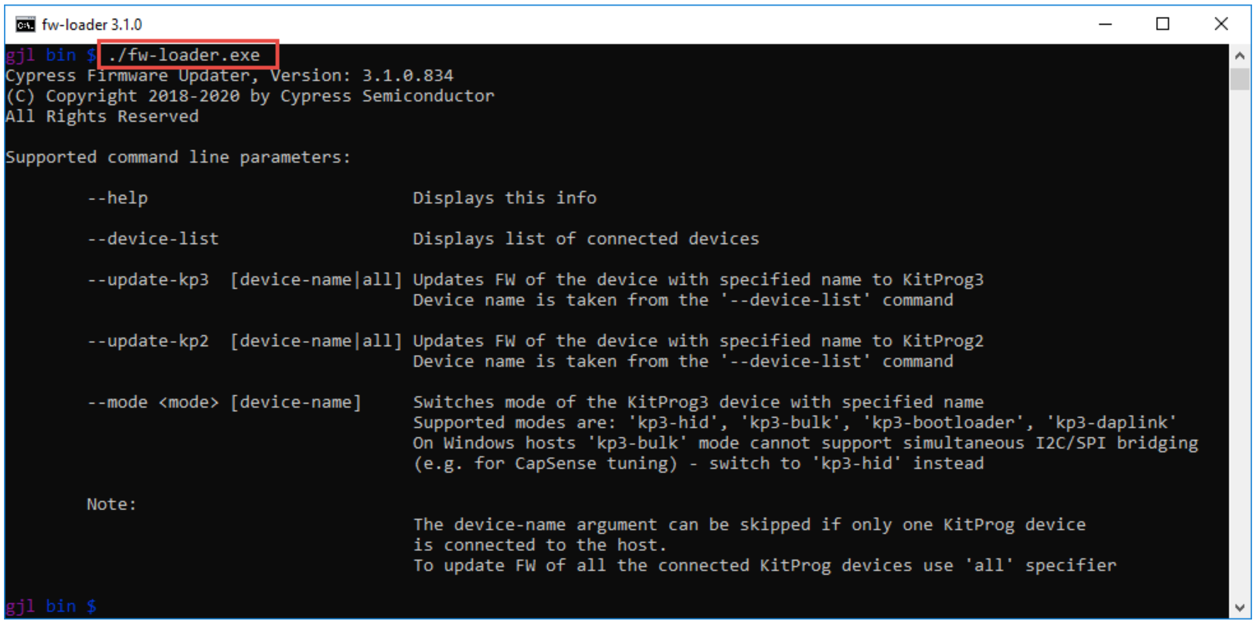
1.10.2 Running Firmware Loader

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

- Windows (modus-shell):
  ```
  cd ~/ModusToolbox/tools_2.2/fw-loader/bin
  ```
- macOS (bash or zsh):
  ```
  cd /Applications/ModusToolbox/tools_2.2/fw-loader/bin
  ```
- Linux (bash):
  ```
  cd ~/ModusToolbox/tools_2.2/fw-loader/bin
  ```

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

```
./fw-loader.exe
```

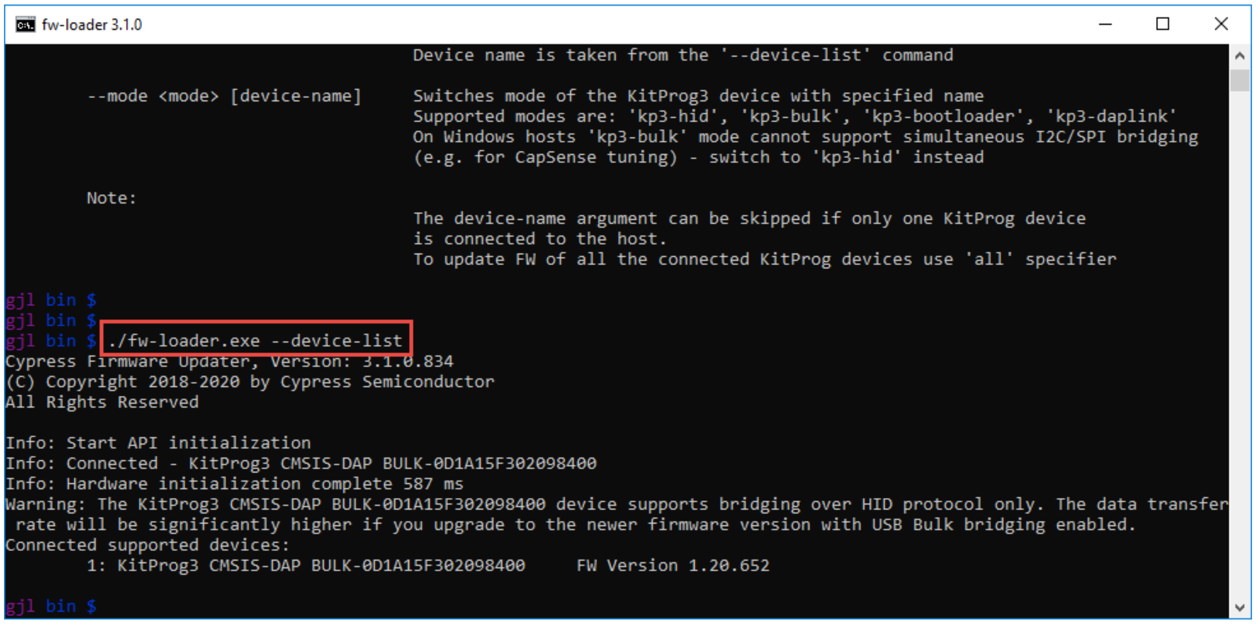


fw-loader 3.1.0

```
gjl bin $ ./fw-loader.exe
Cypress Firmware Updater, Version: 3.1.0.834
(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

        --update-kp2  [device-name|all] Updates FW of the device with specified name to KitProg2
                                       Device name is taken from the '--device-list' command

        --mode <mode> [device-name]    Switches mode of the KitProg3 device with specified name
                                       Supported modes are: 'kp3-hid', 'kp3-bulk', 'kp3-bootloader', 'kp3-daplink'
                                       On Windows hosts 'kp3-bulk' mode cannot support simultaneous I2C/SPI bridging
                                       (e.g. for CapSense tuning) - switch to 'kp3-hid' instead

        Note:
                                       The device-name argument can be skipped if only one KitProg device
                                       is connected to the host.
                                       To update FW of all the connected KitProg devices use 'all' specifier

gjl bin $
```

You can list devices attached to your computer by using:

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



fw-loader 3.1.0

```
                                       Device name is taken from the '--device-list' command

        --mode <mode> [device-name]    Switches mode of the KitProg3 device with specified name
                                       Supported modes are: 'kp3-hid', 'kp3-bulk', 'kp3-bootloader', 'kp3-daplink'
                                       On Windows hosts 'kp3-bulk' mode cannot support simultaneous I2C/SPI bridging
                                       (e.g. for CapSense tuning) - switch to 'kp3-hid' instead

        Note:
                                       The device-name argument can be skipped if only one KitProg device
                                       is connected to the host.
                                       To update FW of all the connected KitProg devices use 'all' specifier

gjl bin $
gjl bin $
gjl bin $ ./fw-loader.exe --device-list
Cypress Firmware Updater, Version: 3.1.0.834
(C) Copyright 2018-2020 by Cypress Semiconductor
All Rights Reserved

Info: Start API initialization
Info: Connected - KitProg3 CMSIS-DAP BULK-0D1A15F302098400
Info: Hardware initialization complete 587 ms
Warning: The KitProg3 CMSIS-DAP BULK-0D1A15F302098400 device supports bridging over HID protocol only. The data transfer
 rate will be significantly higher if you upgrade to the newer firmware version with USB Bulk bridging enabled.
Connected supported devices:
        1: KitProg3 CMSIS-DAP BULK-0D1A15F302098400     FW Version 1.20.652

gjl bin $
```

Note: In Windows, you can launch the fw-loader tool from the **Start** menu or by doing an application search for "fw-loader".

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). They will appear differently based on the operating system that you are using.

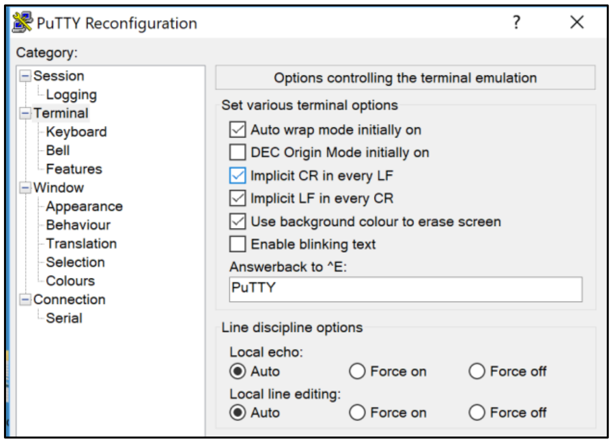
You will need to know the baud rate, which will depend on what YOU programmed into the firmware. It will probably be 9600 (for Mbed OS), 115200 (for PSoC 6), and 115200 (for Amazon FreeRTOS).

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).

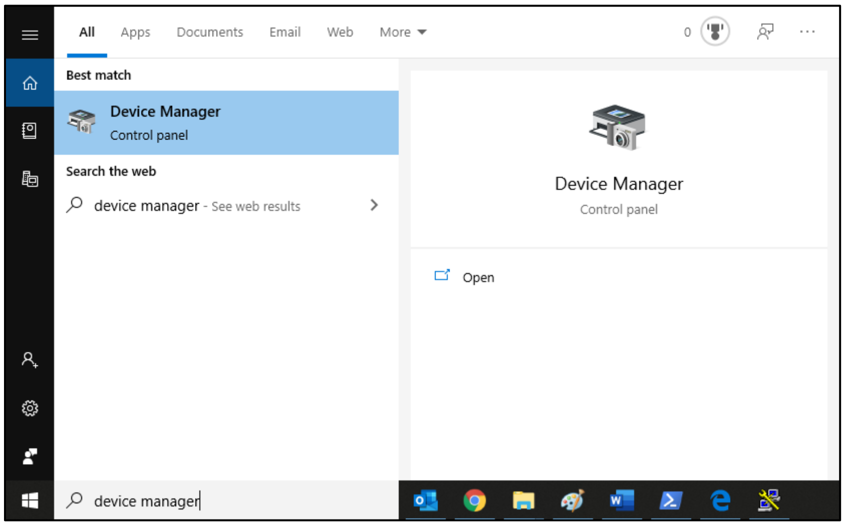


Set various terminal options
[✓] Auto wrap mode initially on
[ ] DEC Origin Mode initially on
[✓] Implicit CR in every LF
[✓] Implicit LF in every CR
[✓] Use background colour to erase screen
[ ] Enable blinking text
Answerback to ^E: PuTTY

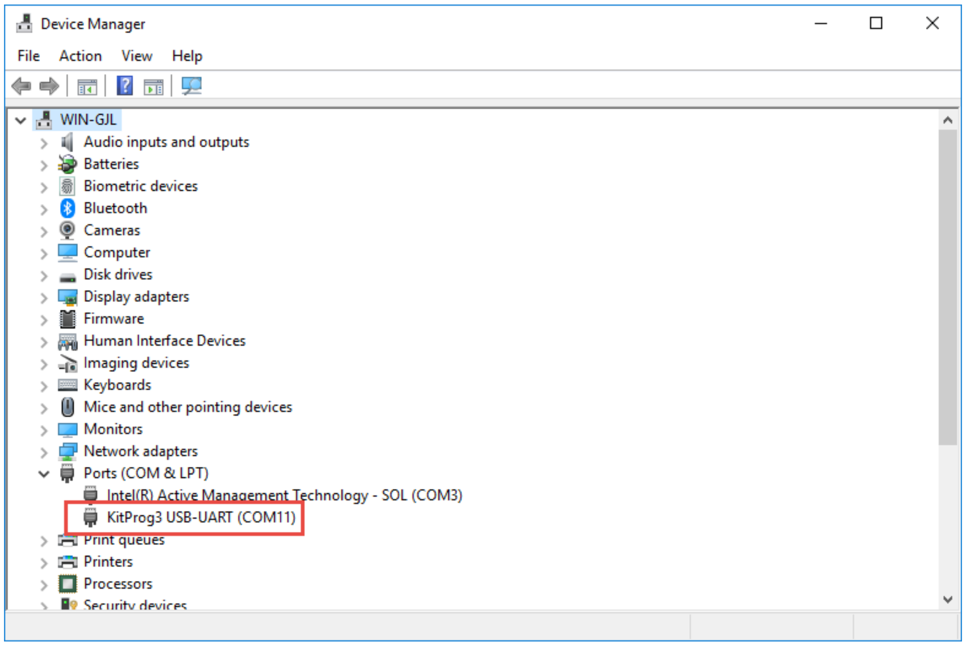
Line discipline options
Local echo: [✓ Auto] Force on Force off
Local line editing: [✓ Auto] Force on Force off

1.11.1 Windows

On Windows, we recommend that you use PuTTY to attach to the serial ports, but you can use other serial terminals such as Tera Term. 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 Start menu.

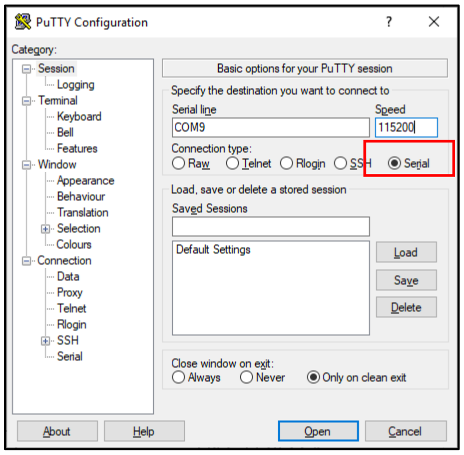


When you look under "Ports" you should see your device.



WIN-GJL
- Audio inputs and outputs
- Batteries
- Biometric devices
- Bluetooth
- Cameras
- Computer
- Disk drives
- Display adapters
- Firmware
- Human Interface Devices
- Imaging devices
- Keyboards
- Mice and other pointing devices
- Monitors
- Network adapters
- Ports (COM & LPT)
  - Intel(R) Active Management Technology - SOL (COM3)
  - KitProg3 USB-UART (COM11)
- Print queues
- Printers
- Processors
- Security devices

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.



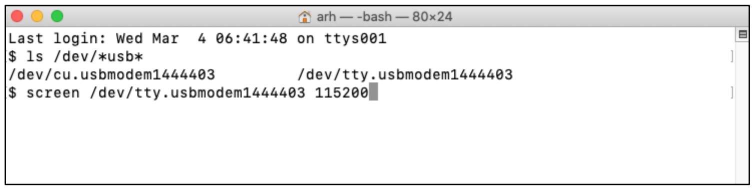
Serial line: COM9
Speed: 115200
Connection type: Raw Telnet Rlogin SSH [✓ Serial]

1.11.2 macOS

Screen Program

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

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



```
Last login: Wed Mar  4 06:41:48 on ttys001
$ ls /dev/*usb*
/dev/cu.usbmodem1444403         /dev/tty.usbmodem1444403
$ screen /dev/tty.usbmodem1444403 115200
```

Then attach using the device name and baud rate:

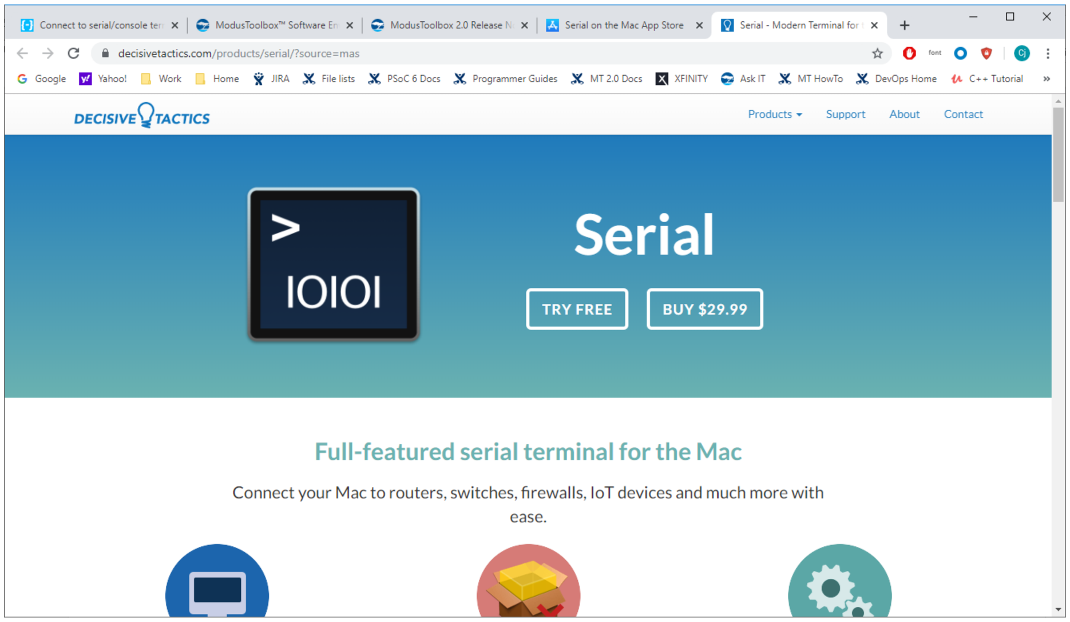
```
screen /dev/tty.usbmodem1444403 115200
```

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

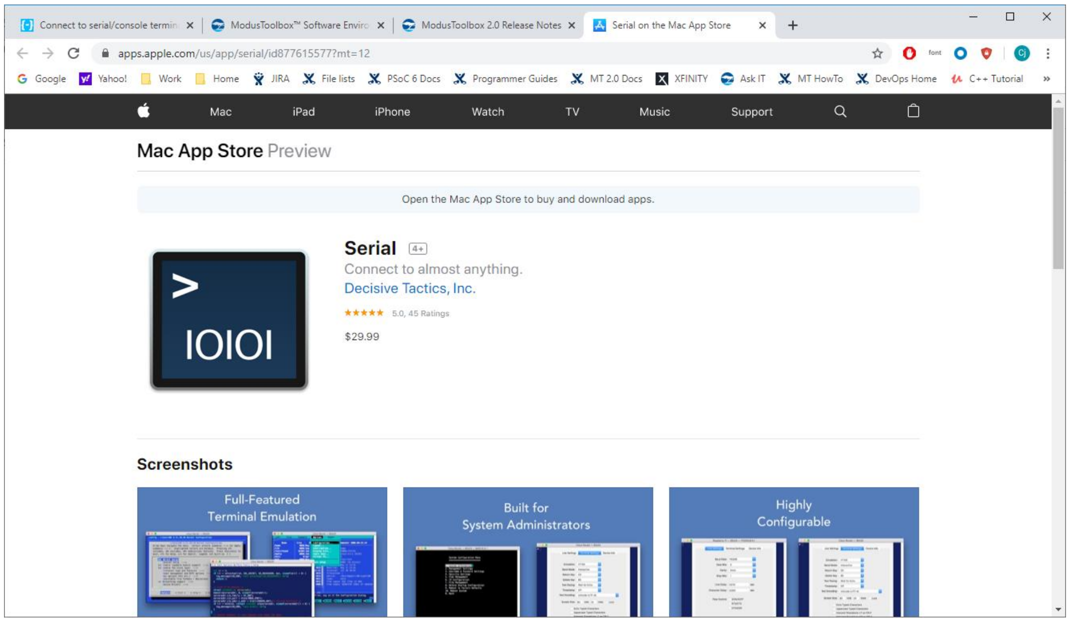
Serial App

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:

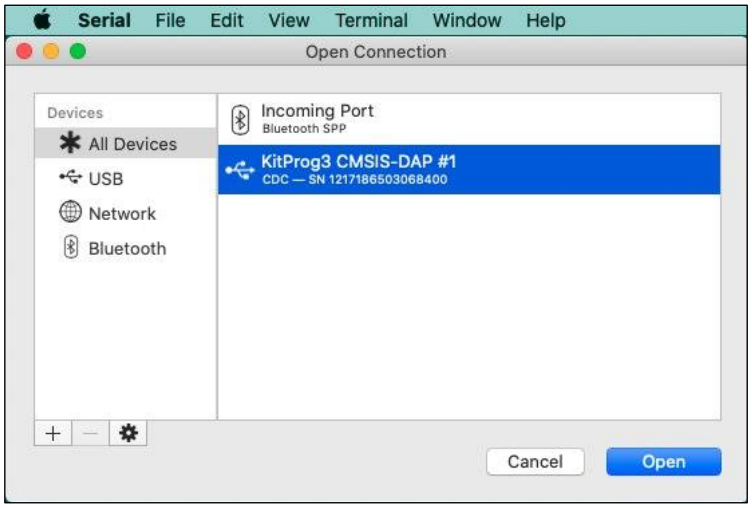
Creator's website: https://www.decisivetactics.com/products/serial/


decisivetactics.com/products/serial/?source=mas
DECISIVE TACTICS
Products
Support
About
Contact
Serial
TRY FREE
BUY $29.99
Full-featured serial terminal for the Mac
Connect your Mac to routers, switches, firewalls, IoT devices and much more with ease.

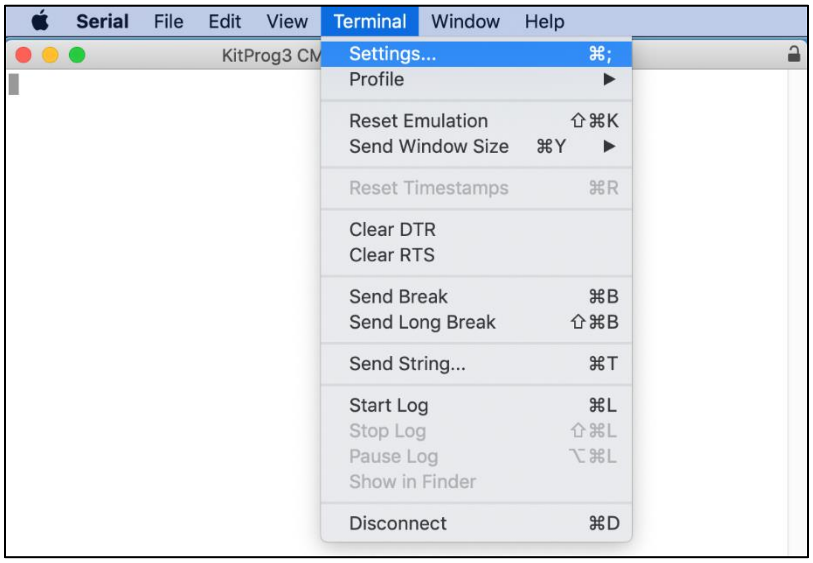
Mac App Store: https://apps.apple.com/us/app/serial/id877615577?mt=12


apps.apple.com/us/app/serial/id877615577?mt=12
Mac
iPad
iPhone
Watch
TV
Music
Support
Mac App Store Preview
Open the Mac App Store to buy and download apps.
Serial 4+
Connect to almost anything.
Decisive Tactics, Inc.
★★★★★ 5.0, 45 Ratings
$29.99
Screenshots
Full-Featured Terminal Emulation
Built for System Administrators
Highly Configurable

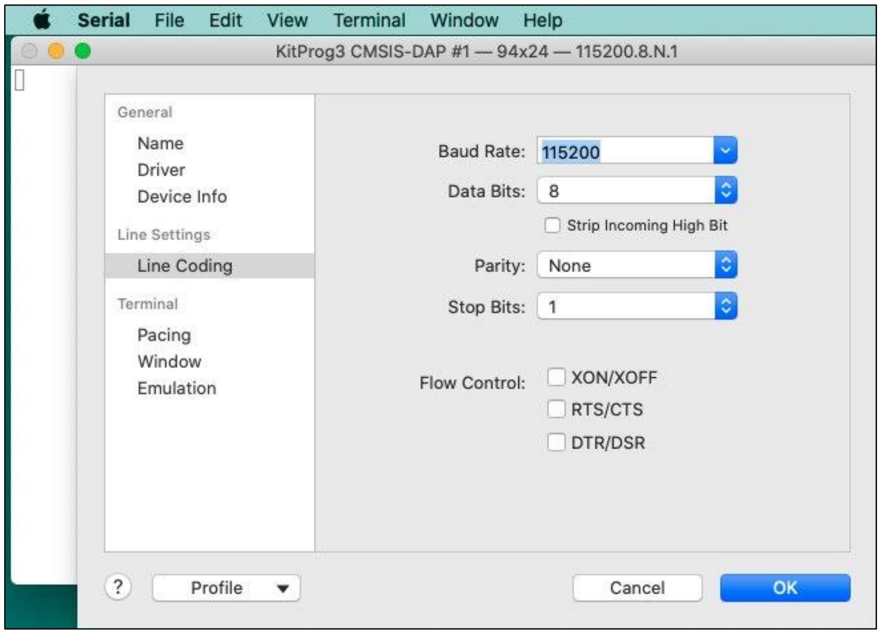
1. After installing it, run the Serial app to open the port selector. This shows the attached board (CY8CKIT-062S2-43012) is set to CMSIS-DAP mode:



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):



Baud Rate: 115200
Data Bits: 8
Strip Incoming High Bit
Parity: None
Stop Bits: 1
Flow Control: XON/XOFF, RTS/CTS, DTR/DSR

5. Click **OK** to apply.

1.11.3 Linux

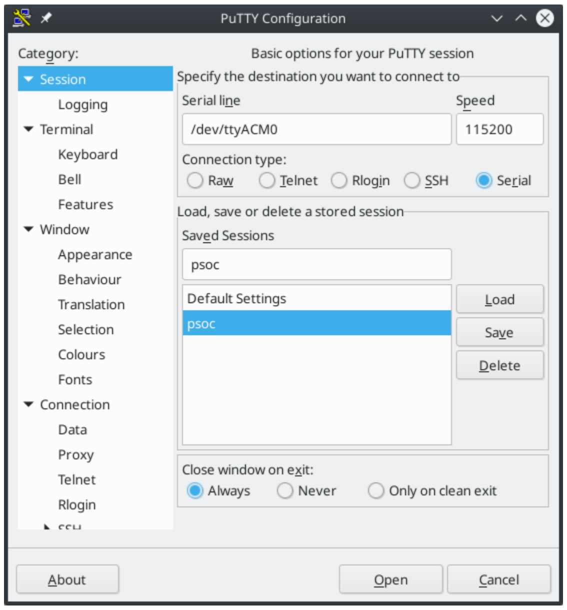
You can use PuTTY on Linux similar to Windows. The only difference is that the device name is in the form of /dev/ttyACMn, where n is usually a small number. To get a list of such devices, type the following on the command line:

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

This should return something like this:

```
/dev/ttyACM0
```

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



Basic options for your PuTTY session

Specify the destination you want to connect to

Serial line: /dev/ttyACM0
Speed: 115200

Connection type: Raw Telnet Rlogin SSH [✓ Serial]

Load, save or delete a stored session

Saved Sessions: psoc

Default Settings
psoc

Close window on exit: [✓ Always] Never Only on clean exit

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 (MTB2-x-SW-Install-Instructions). Make sure the software is installed and configured correctly.

☐ **ModusToolbox Installer**

- Can you start the IDE?
- Can you create a workspace?
- Can you run ModusToolbox Command Line?
  - **For Windows**: Start modus-shell by running cygwin.bat from the ModusToolbox/tools_2.2/modus-shell folder.
  - **For macOS/Linux**: Open a terminal window and type `make`.

☐ **Python**

- From modus-shell (Windows) or a terminal window (macOS/Linus), does the following command return a version of 3.x.x (e.g. Python 3.7.7)?

```
python --version
```

☐ **Mbed Studio**

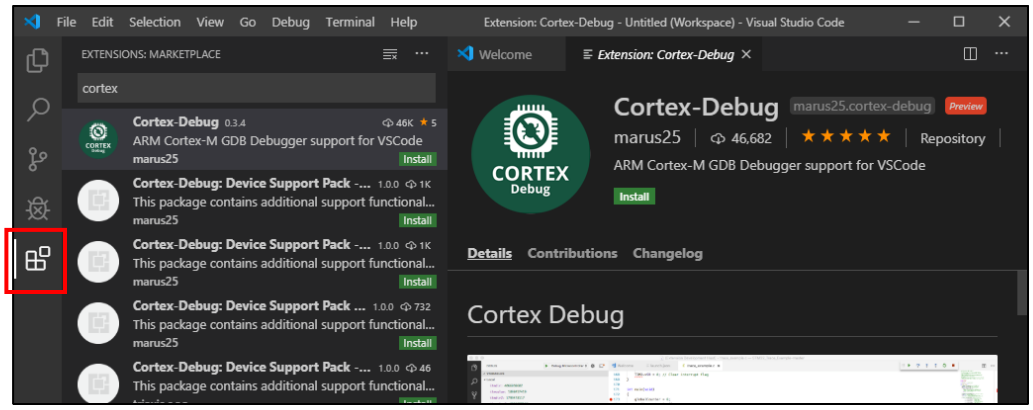
- Can you start the IDE?

☐ **Mbed CLI**

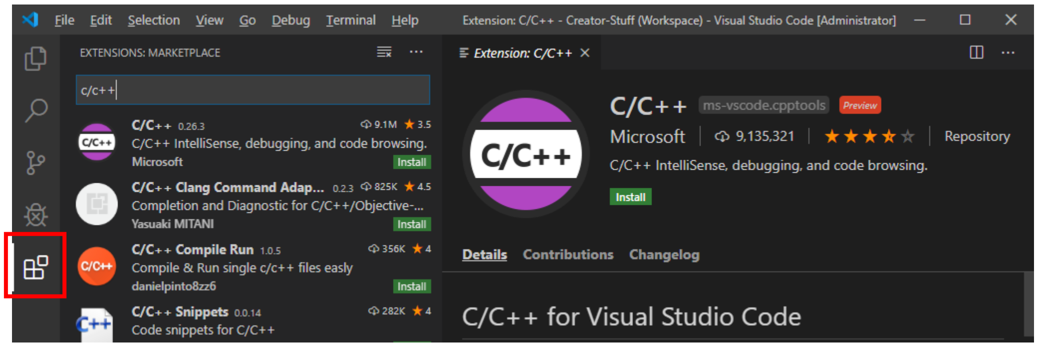
- Were you able to setup and configure the virtual environment for Mbed in the installation instructions?

☐ **Visual Studio Code**

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


File
Edit
Selection
View
Go
Debug
Terminal
Help
Extension: Cortex-Debug - Untitled (Workspace) - Visual Studio Code
EXTENSIONS: MARKETPLACE
cortex
Cortex-Debug 0.3.4
ARM Cortex-M GDB Debugger support for VSCode
marus25
Install
Cortex-Debug: Device Support Pack -... 1.0.0
This package contains additional support functional...
Welcome
Extension: Cortex-Debug
Cortex-Debug
marus25.cortex-debug
Preview
marus25
46,682
Repository
ARM Cortex-M GDB Debugger support for VSCode
Install
CORTEX Debug
Details
Contributions
Changelog
Cortex Debug

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


File Edit Selection View Go Debug Terminal Help
Extension: C/C++ - Creator-Stuff (Workspace) - Visual Studio Code [Administrator]
EXTENSIONS: MARKETPLACE
c/c++
C/C++ 0.26.3
C/C++ IntelliSense, debugging, and code browsing.
Microsoft
Install
C/C++ Clang Command Adap... 0.2.3
Completion and Diagnostic for C/C++/Objective-...
Yasuaki MITANI
C/C++ Compile Run 1.0.5
Compile & Run single c/c++ files easly
danielpinto8zz6
C/C++ Snippets 0.0.14
Code snippets for C/C++
Extension: C/C++
C/C++ ms-vscode.cpptools Preview
Microsoft 9,135,321 Repository
C/C++ IntelliSense, debugging, and code browsing.
Install
Details Contributions Changelog
C/C++ for Visual Studio Code

1.12.2 Run the fw-loader Tool

See section 1.10 PSoC Firmware Loader for details.

☐ ☐

- Connect your kit to your computer using the provide USB cable. Make sure to connect to the KitProg3 USB connector on the kit.
- Check the KitProg firmware version on the kit.

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

☐

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

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

1.12.3 Switch KitProg Modes and Verify the Mode

See section 1.10 PSoC Firmware Loader for details.

☐ ☐

- Switch between 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

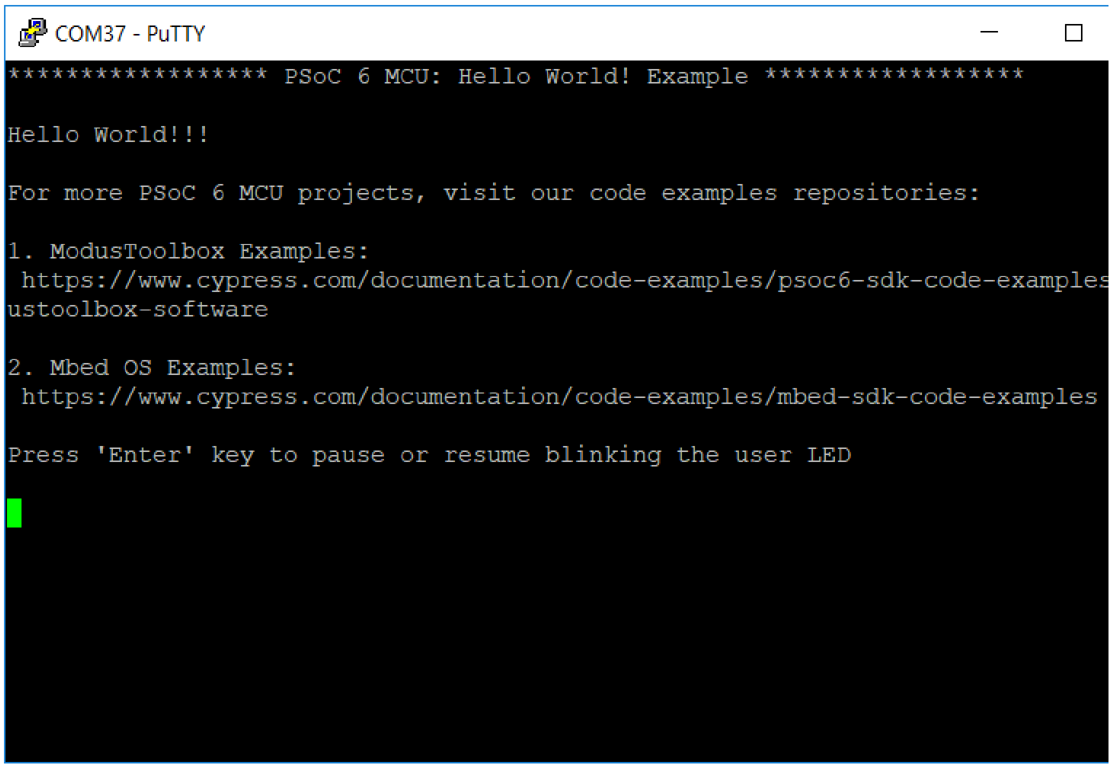
```
./fw-loader --mode kp3-bulk
```

The LED should be solid ON.

1.12.4 Use a terminal program to attach to the UART on the CY8CKIT-062S2-43012

☐

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


COM37 - PuTTY

```
****************** PSoC 6 MCU: Hello World! Example ******************

Hello World!!!

For more PSoC 6 MCU projects, visit our code examples repositories:

1. ModusToolbox Examples:
 https://www.cypress.com/documentation/code-examples/psoc6-sdk-code-examples
ustoolbox-software

2. Mbed OS Examples:
 https://www.cypress.com/documentation/code-examples/mbed-sdk-code-examples

Press 'Enter' key to pause or resume blinking the user LED
```