Kennesaw State University (KSU)
College of Computing and Software Engineering (CCSE)
CS-3503 Computer Organization & Architecture

Lab-5: Getting Started with your Microcontroller Board

Name: _____ **Date:** _____

Learning Objectives:

- Access an Integrated Development Environment (IDE) for Mbed OS applications
- Build and run an application within an Mbed OS IDE for your microcontroller board.
- Identify a compatible project on the mbed site and input into your Integrated Development Environment (IDE).
- Learn to import a project, build the project and observe the effects.
- Learn to import the class project, build, and observe output messages on the terminal.

Introduction

What is the ARM Mbed OS?
The Mbed OS is an open-source operating system for IoT Cortex-M boards. The Mbed OS provides an abstraction layer for the microcontrollers it runs on, so that developers can write C/C++ applications that run on any Mbed-enabled board.

The *full profile* of Mbed OS is a real-time operating system (RTOS) (includes Keil RTX and all RTOS APIs), so it supports deterministic, multithreaded, real-time software execution. The *bare metal profile* doesn't include Keil RTX and is not an RTOS. It is designed for applications that do not require complex thread management and focuses on minimizing the size of the final application.

Building Applications with ARM Mbed OS
You have 4 options for building applications using the Mbed OS:

1) Build with Arm Keil Studio
    - Free, browser-based IDE for evaluation and development of embedded, IoT, and Machine Learning software for Cortex-M devices. The successor IDE to Mbed Online Compiler (See 2 below)
2) Build with the Mbed Online Compiler
    - Zero-installation web IDE to explore Mbed OS; great for rapid prototyping and education.
3) Build with Mbed Studio
    - Dedicated desktop IDE, including all the necessary tools to work with Mbed OS.
4) Build with Mbed CLI
    - Command-line tool requiring manual setup of tool but providing the greatest degree of flexibility.

What is the ARM Keil Studio?
Arm Keil Studio is a free to use, browser-based IDE for the evaluation and development of embedded, IoT, and Machine Learning software for Cortex-M devices. With a cloud-hosted workspace for your code, comprehensive source control integration, and a powerful C/C++ editor, you can edit your projects from any computer, share them with colleagues and export them for desktop usage in Keil µVision. You can compile projects using Arm Compiler 6, flash the projects directly to supported development boards, and debug from supported browsers without the need to install any software.

Our goal is to make it quicker and easier for you to evaluate reference designs, reducing the time it takes to get your embedded projects to market, while also providing a point of integration for Arm ecosystem partners who provide professional software, tools, and services.

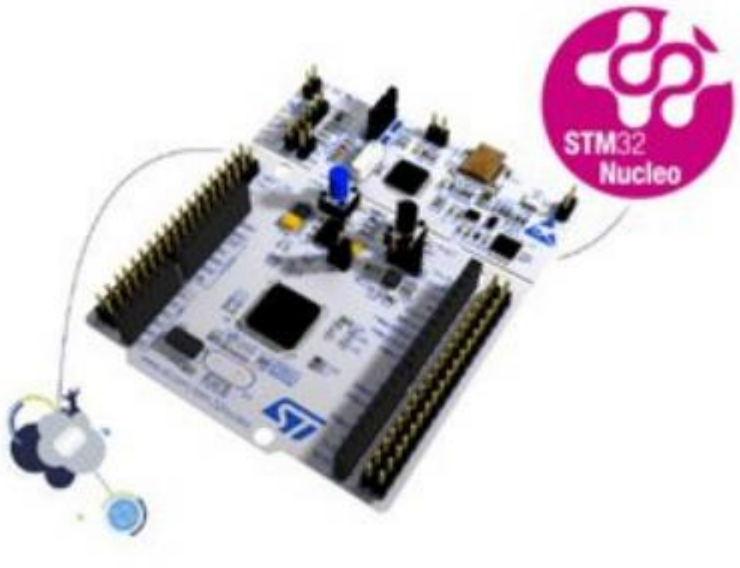
Arm Keil Studio demonstrates next generation IDE technology and new concepts for CMSIS project formats. We support a range of software examples, showcasing Keil RTX, FreeRTOS, and IoT connectors for Amazon AWS IoT, Microsoft Azure IoT Hub, and Google Cloud. Keil Studio is the successor to the Mbed Online Compiler and allows you to develop Mbed OS 5 and 6 projects on supported Mbed Enabled boards. Keil Studio also provides limited support for Mbed 2. To get started, you can import Mbed projects from your Online Compiler workspace or mbed.com.

Prelab

For your prelab activity, you will use Arm Keil Studio for your IDE and then follow the instructions under "Get started with an Mbed OS Blinky example" to build and run an example program (Blinky program) within that IDE. This program causes your LED to blink on and off.

Instructions for NUCLEO-F401RE Users

1. Note: NUCLEO-F410RB users should ignore these instructions and refer to special instructions after step 3.



2. To build with Arm Keil Studio, following the directions found at the link below. https://developer.arm.com/documentation/102497/1-5/Tutorials/Get-started-with-an-Mbed-OS-Blinky-example?lang=en

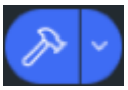
3. Verify that LED1 blinks on and off at the specified rate.

Instructions for NUCLEO-F410RB Users

The NUCLEO-F410RB boards are currently not compatible with the latest version of the Mbed OS (Mbed OS 6). Program development with these boards require an older version of the Mbed OS (Mbed OS 5). Please follow the instructions below to build and run the Blinky program for your board.

1. Create an Mbed account at the url below.
   studio.keil.arm.com

2. Log into Keil Studio after creating your account.
3. Create a new project, select File > New Project….
   The New Project dialog box opens.

4. Click the Example project drop-down list and go to the Mbed list. Select mbed-os-example-blinky under **MBED OS 5**.
5. Confirm the project name.
6. Check the default options:
   - Keil Studio sets the newly created project as the active project. Build and run commands only apply to the active project. Clear the checkbox if you do not want to make the new project active.
   - Keil Studio initializes the project as a Git repository. Clear the checkbox if you do not want to turn your project into a Git repository. See Configure a project for source control and collaboration for more details.
7. Click Add project.
   The project loads to your workspace and is the active project. The README.md file of the project displays. Review the file to learn more about project settings and board requirements.
8. The Target hardware drop-down list shows the build target set for the project. A build target tells Keil Studio how to build Mbed OS so that it matches your hardware. To select a build target, you can either:
   - Use the Target hardware drop-down list. Your target hardware name most likely matches the board name.
   - Connect your board to your computer. The first time you connect your board, you have to click the Connect to target hardware button to the right of the Target hardware drop-down list. After the first successful connection, Keil Studio detects the board and suggests a matching target.
9. To build the project, click the Build project button,  . Build project builds Blinky and stops.

10. If you have a board connected, click Run project  . Run project builds Blinky and flashes it to your board.
    You might have to restart your board for Blinky to run.

    For Mbed projects only: Select the Use Daplink option in File > Settings > Open Preferences > Run category to build and flash projects to your board. When this option is not selected, the Run project button is not available.

Laboratory Procedure:

1. Consider the code from main.cpp of the mbed-os-example-blinky program.

```
1) /* mbed Microcontroller Library
2) Copyright (c) 2019 ARM Limited
3) SPDX-License-Identifier: Apache-2.0
4) */
5)
6) #include "mbed.h"
7)
8) // Blinking rate in milliseconds
9) #define BLINKING_RATE     500ms
10)

11)int main()
12){
13)    // Initialise the digital pin LED1 as an output
14)    DigitalOut led(LED1);
15)
16)    while (true) {
17)        led = !led;
18)        ThisThread::sleep_for(BLINKING_RATE);
19)    }
20)}
```

   Line 9 defines the on and off time of the led (BLINKING_RATE).
   Line 14 initializes variable led to map to LED1 of your board.
   Lines 16 to 19 defines a while loop that inverts the state of led (line 17) and then waits BLINKING_RATE before the next iteration of the loop (line 18).

2. Modify the while loop in main.cpp to print a “Hello ARM world” statement repetitively to a terminal screen whenever led is on.

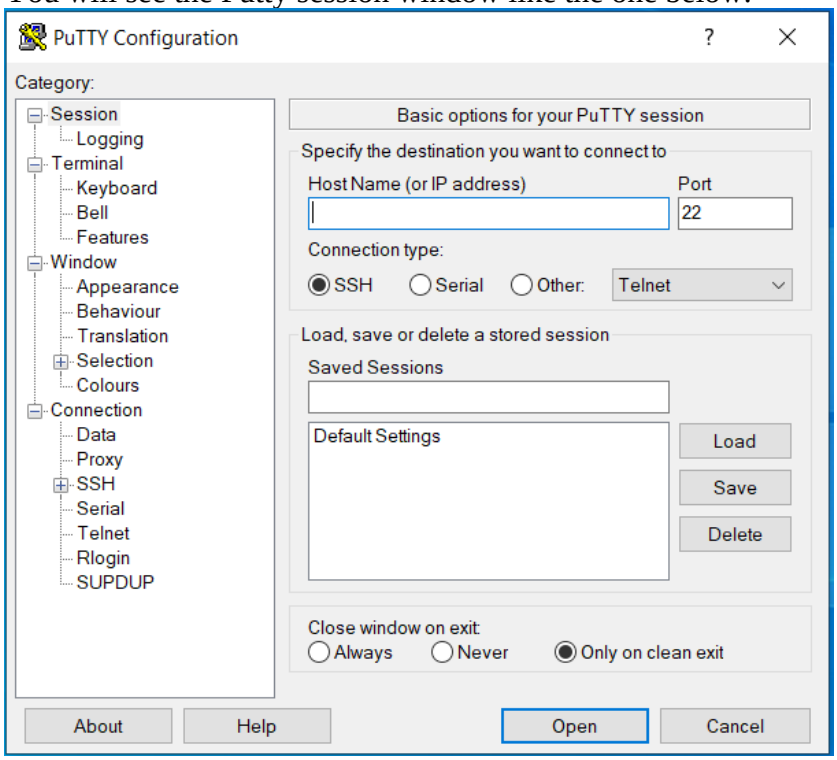
```
while (true)
{
    led = !led;
    if(led == 0){
        printf("Hello ARM world.\n");
    }
    ThisThread::sleep_for(BLINKING_RATE);
}
```

3. To view the “Hello ARM world” statement you will need a terminal application if using an online IDE. Download a suitable terminal application (Putty for Windows or CoolTerm for Mac).

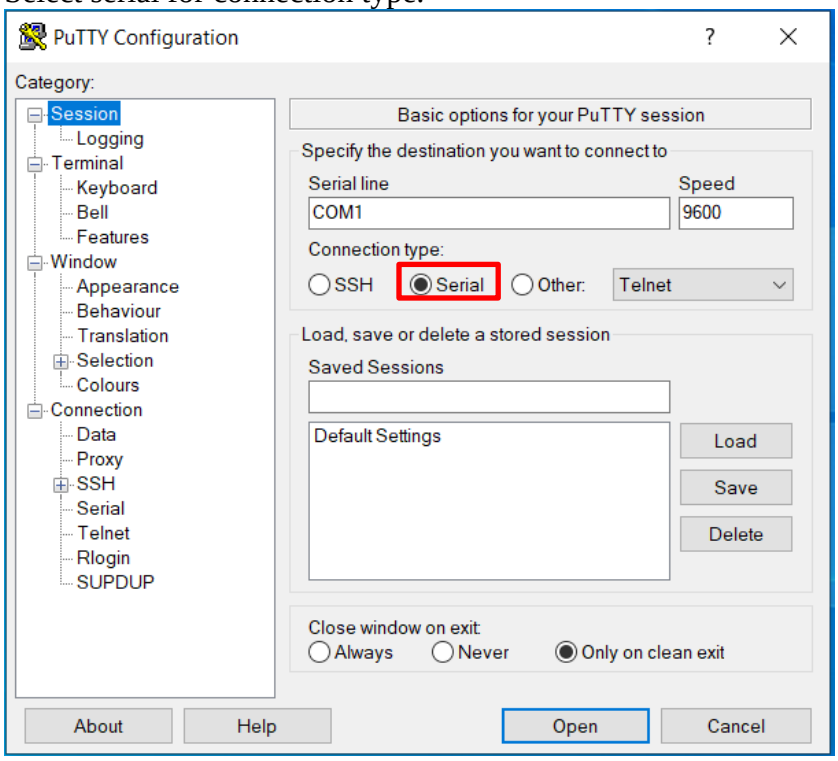
   For Windows Users:

   i. Navigate to the website below and follow the instructions to download the installer for Putty.
      https://www.putty.org/
   ii. Install Putty by following the installation instructions.
   iii. Open the Putty application.

iv. You will see the Putty session window like the one below.



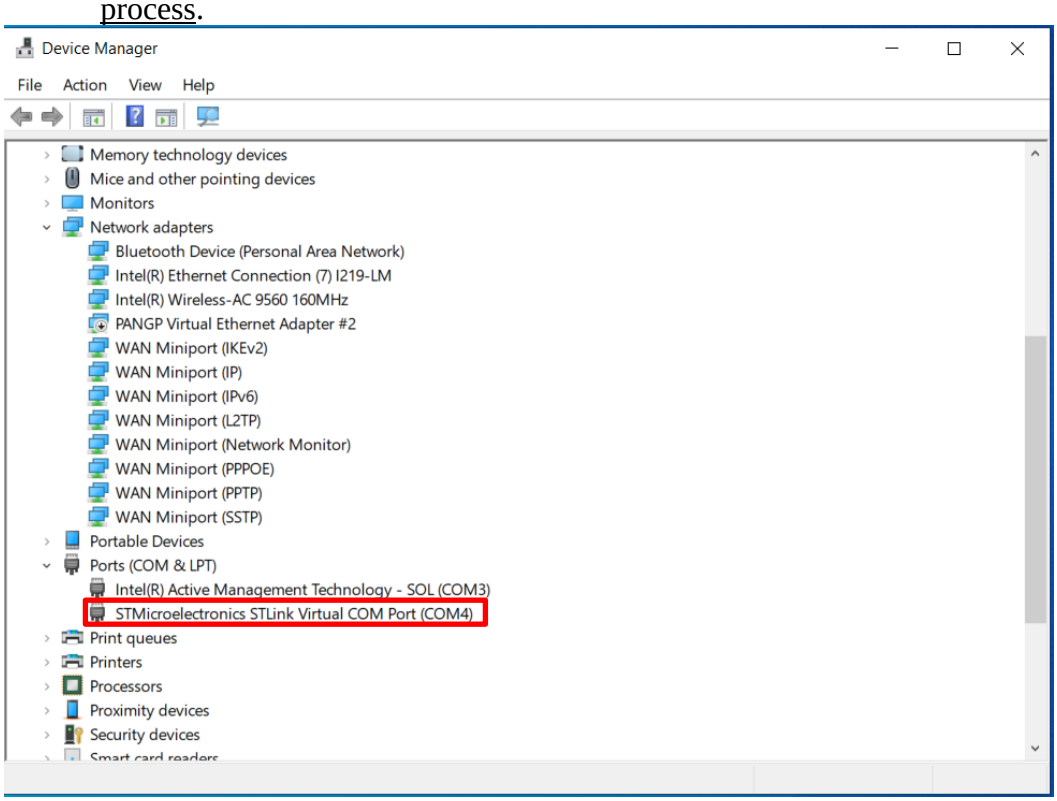
v. Select serial for connection type.



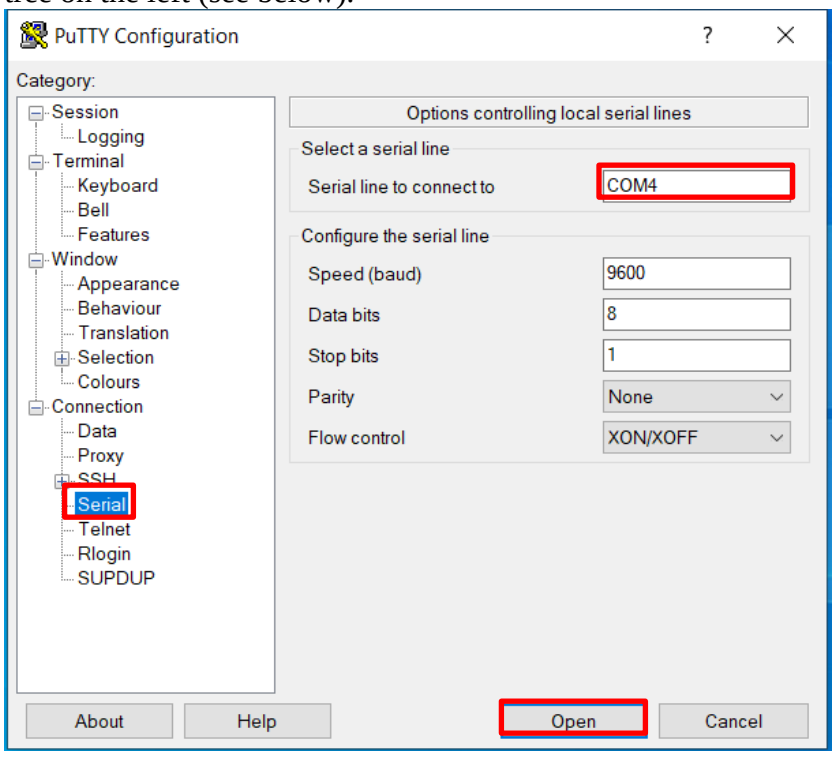
vi. You will need to identify the virtual serial port number of your microcontroller board. Follow the steps below.

a. Open the Device Manager application of your Windows OS.

b. Under Ports (COM & LPT), identify the COM port number for your STMicroelectronics device. Ensure your board is connected during this process.



vii. Return to Putty and select Serial leaf on the Connection branch of the Category tree on the left (see below).



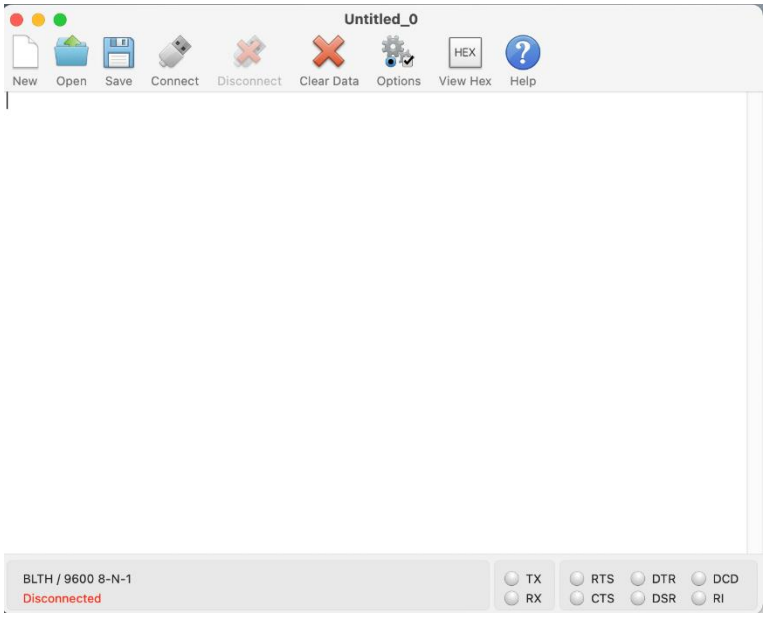
viii. Update the “Serial line to connect to” field with the port number of your board.
ix. Select Open button to start a session.

For Mac users:

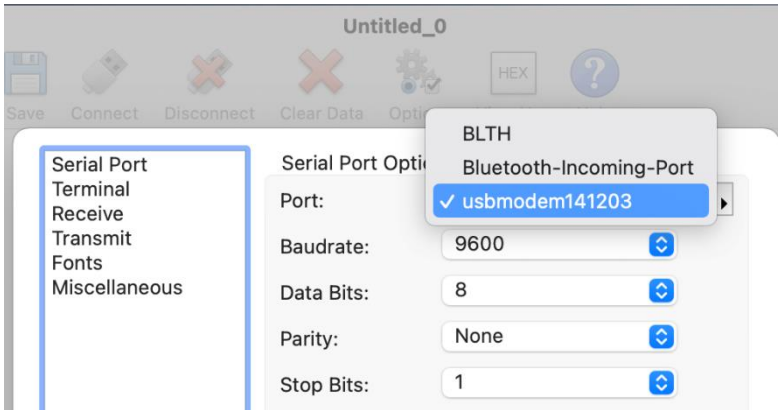
i. Navigate to the website below and download the zip file for CoolTerm.
https://freeware.the-meiers.org/

ii. When you first try to run CoolTerm, it will run into an error as the developer is not registered with Apple. You will have to go to the Security & Privacy settings and allow the application to be opened anyway.

iii. After allowing the application to be opened anyway, the CoolTerm window will open as shown below.

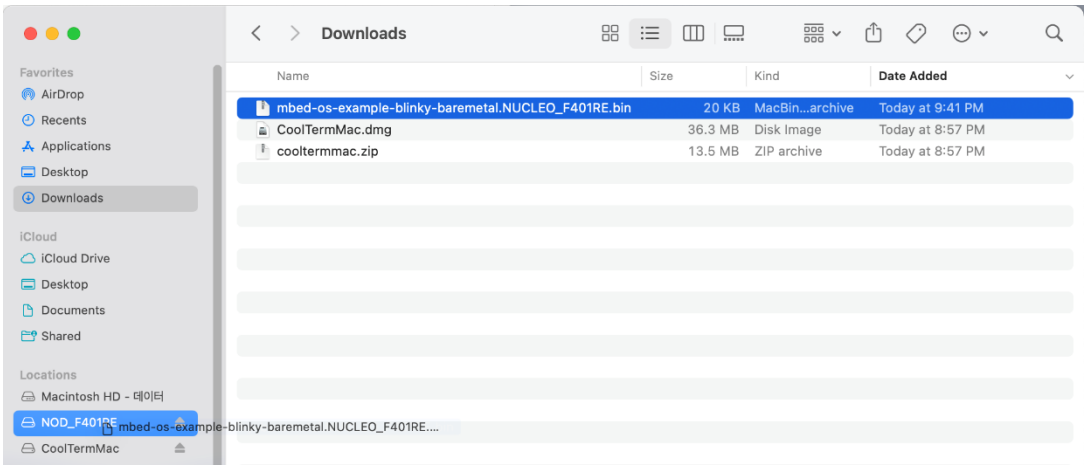


iv. To setup CoolTerm to communicate with the board, the serial port in options must be set as the usbmodem.



v. Once set up is finished, by clicking the connect button, CoolTerm will start to communicate with the microcontroller board.

vi. When using the online compiler, the compiled project will be downloaded to your personal computer. This file must be copied to the microcontroller such that the board can execute the project.



vii. The OS will say that the board was not disconnected properly. Ignore that prompt.

4. Verify that the “Hello ARM world” statement prints every time the LED turns on.
5. **Take a screen shot of the Mbed IDE window and the terminal window for your lab report.**

Discussion Questions

1. Explain how you would change the code in main.cpp so that on/off times of the LED are 1 sec in duration.
2. Write a sample code section showing how you would modify the while loop in main.cpp to print “Goodbye ARM World.” to the terminal when the LED turns off?

Report Requirements

1. mbed IDE window screenshot (25 points)
2. terminal window screenshot (25 points)

Answers to Discussion Questions

1. Explain how you would change the code in main.cpp so that on/off times of the LED are 1 sec in duration. (25 points)
2. Write a sample code section showing how you would modify the while loop in main.cpp to print “Goodbye ARM World.” to the terminal when the LED turns off? (25 points)

No formal lab report is needed for this lab.