ECE 4501/6501: Advanced Embedded Systems

Mini Project 0
Getting Familiar with ARM Programming Environment
Due Date: Wednesday, Sep. 12, 2018, 11:59 PM

In this mini project, you will set up and familiarize yourself with the Texas Instruments Tiva™ C Series Launchpad Evaluation Board, Keil µVision IDE, and the TM4C123G ARM Cortex-M4F microcontroller. You will then run a simple HelloWorld program in simulation and on the actual LaunchPad board.

Go to the following link to accept the Mini Project 0 assignment in the **GitHub Classroom**: https://classroom.github.com/a/yzg2RpIy. You will get access to a private repository created for you in the @UVA-embedded-systems organization on GitHub which contains the starter files for the mini project. You can use this repository for developing and testing your code. Your submission will also be posted to this repository.

In Part 1, you will set up your ARM programming environment. In Part 2, you will make sure all the required drivers for programming, debugging, and serially communicating with the LaunchPad board are installed. Part 3 walks you through the process of creating and running a project in Keil in "simulation" and "board" modes. You will be repeating this process every time you need to create a project.

Mini Project Parts

Part 1) In this part, you will install the Keil µVision IDE and software to simulate the LaunchPad board.

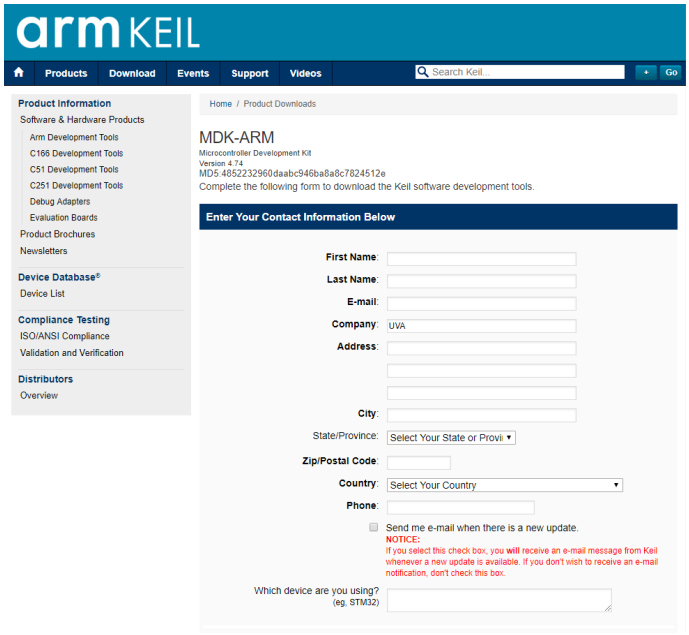
Note: Keil can only run on Windows (XP, Vista, 7, 8, or 10). If you'd like to install the software on a Macintosh or Linux, the easiest way is to use a virtualization software such as:

- Parallels http://www.parallels.com/
- VMware Fusion http://www.vmware.com/products/fusion/
- VirtualBox https://www.virtualbox.org/ (FREE)

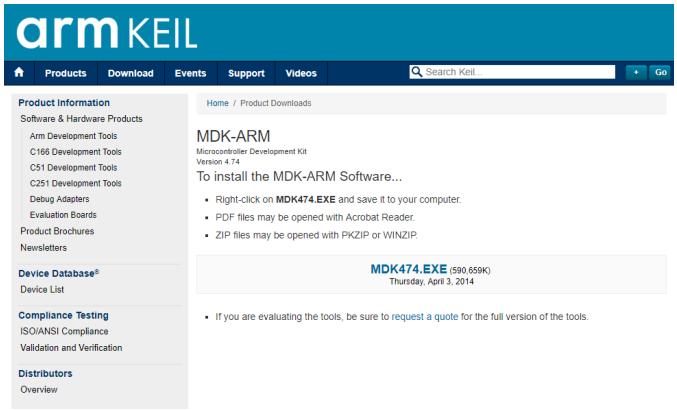
And load a Windows image from the least expensive source you can find. UVA students can download Windows at no cost from UVA Software: http://its.virginia.edu/software/displayPackages.php?tId=32

A – Install Keil µVision 4.74. A slide show of the process to download and install Keil can be seen here:

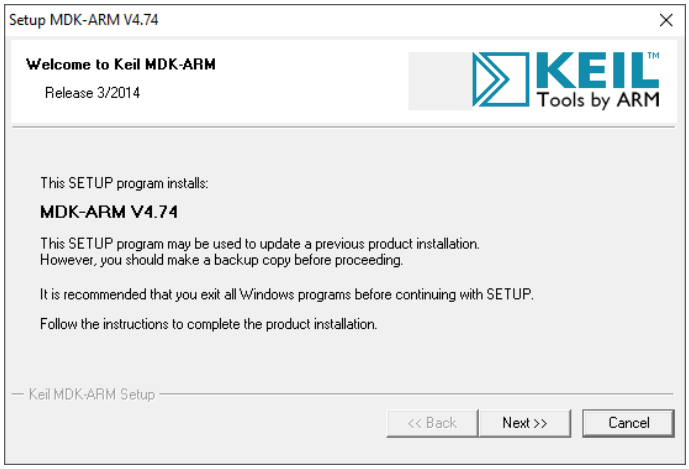
1) Go to https://www.keil.com/demo/eval/armv4.htm. Enter your contact information with your valid address, phone, and email. Set the Company field to your school or your company. Set your device to **TM4C123GH6PM**. Fill in all fields of this form and click **Submit**.



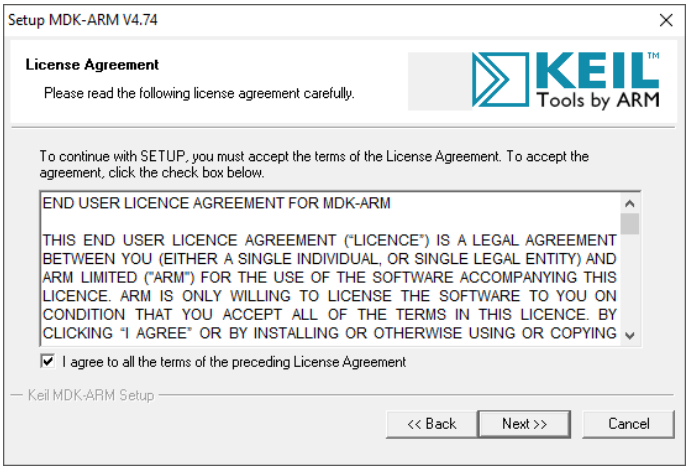
2) Click on the **MDK474.EXE** link, downloading the 590,659K file to your computer.



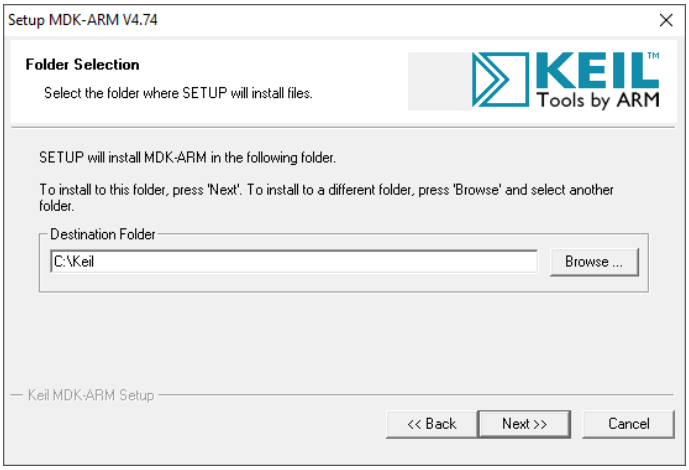
3) Run the **MDK474.EXE** file, and click **Next**.



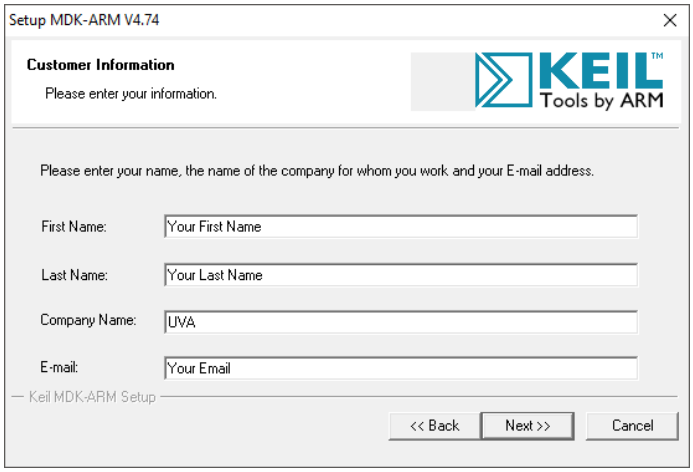
4) Read the license agreement, agree to terms, and click **Next**.



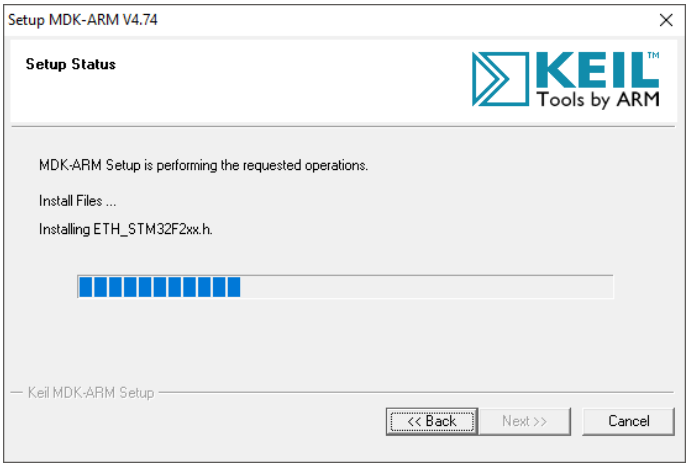
5) Install the application some place easy to find such as C:\Keil or D:\Keil, and click **Next**.



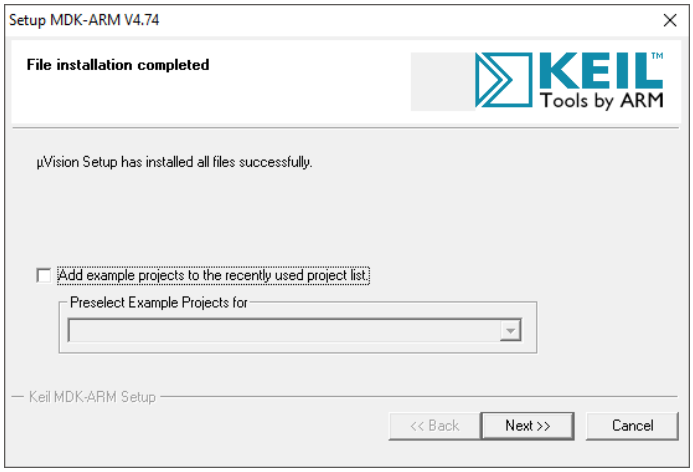
6) Update these fields with your correct information, and click **Next**.



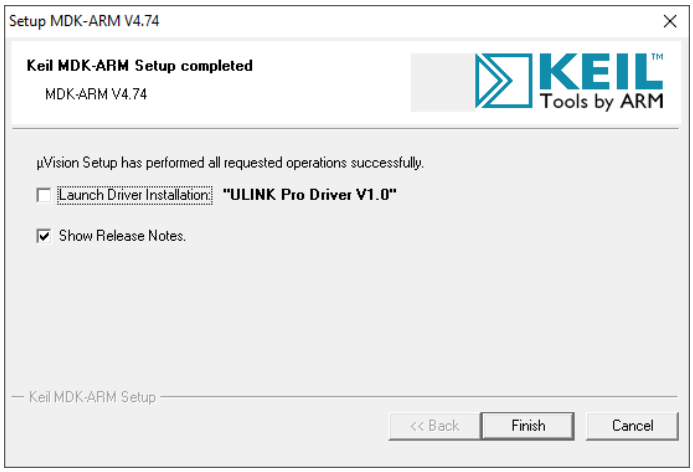
7) Wait while it installs. It will take some time.



8) Deselect example projects (they will be installed later), and click **Next**.



9) Deselect ULINK Pro Driver V1.0 (the drivers will be installed later), and click **Finish**.



Note: The **Release Notes** contain links to information about the Keil development tools.

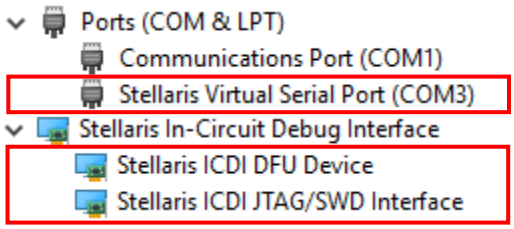
B – Download and install TExaS in your Keil folder. TExaS is a collection of DLL plugins developed by Prof. Valvano at the University of Texas at Austin that provides simulation of the LaunchPad: http://faculty.virginia.edu/alemzadeh/teaching/ece_6501/software/TExaS_Install.exe

Deliverables for Part 1: Successful installation of the software on your machine.

◆

Part 2) In this part, you will ensure that the Stellaris® In-Circuit Debug Interface (ICDI) and Virtual COM Port drivers are installed. These drivers are needed for debugging and downloading code to the microcontroller and for serial communication.

1) You may not need to install the drivers manually on Windows 10. Connect your Tiva LaunchPad board through its "Debug" port to your computer and select the "Debug" mode on the board. Check the Windows Device Manager for the following:



2) Note that the device names may be slightly different from the ones shown in the figure. Follow Steps 3 and 4 if Windows indicates missing drivers.
3) Download the Launchpad drivers from the TI website: http://www.ti.com/tool/stellaris_icdi_drivers
4) Then Install the LaunchPad drivers by following the instructions provided here: http://www.ti.com/lit/ml/spmu287c/spmu287c.pdf

Deliverables for Part 2: Ensure the drivers are installed.

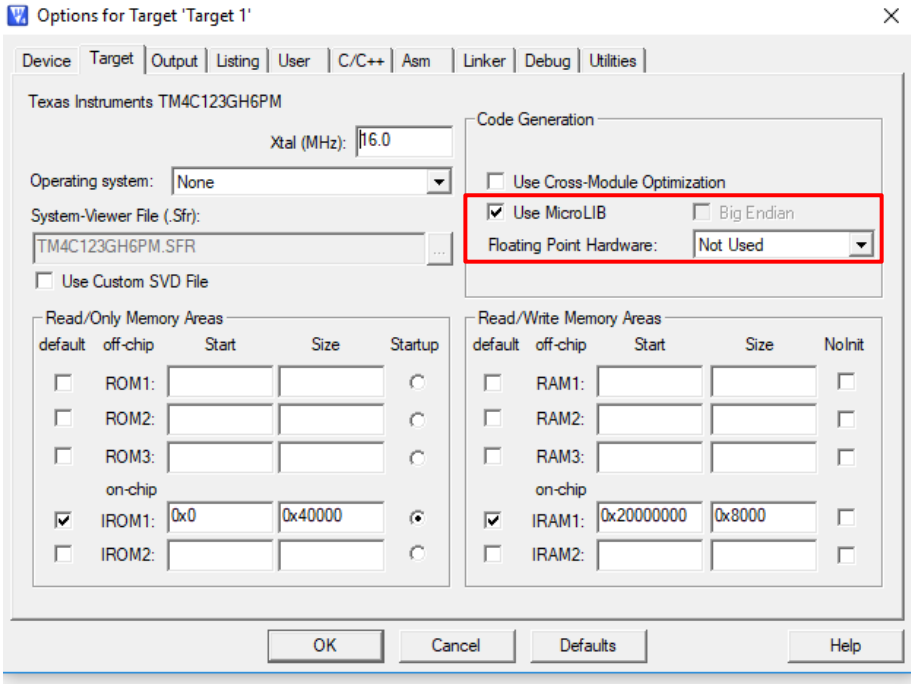
◆

Part 3) In this part, you will create your first project in Keil and run a HelloWorld program both in simulation and in board modes.

1) In Keil, create a new project and save it somewhere on your machine: Project -> New uVision Project
2) Choose the Device: "TM4C123GH6PM"
3) When asked to "Copy 'startup_TM4C123.s' to project folder and add file to project?", choose "No". An alternative startup code **`startup.s`** is provided to you.
4) Add the source files from your GitHub repository to the project (Make sure **`tm4c123gh6pm.h`** and **`startup.s`** are included).
5) Change the project configurations: Project -> Options for Target 'your project name'

In the Target Tab, inside the “Code Generation” panel:

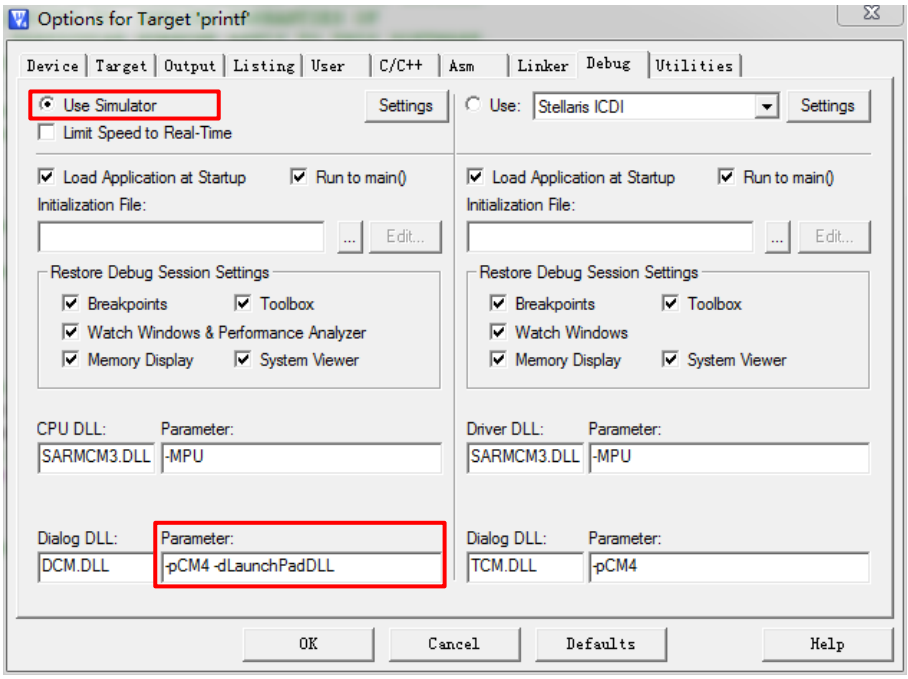
- Check “Use MicroLIB”
- Select “Not Used” for “Floating Point Hardware”



Note: You will follow these steps every time before creating a project in Keil. You will follow the instructions in **A** for running your code in simulation or **B** for running it on the actual LaunchPad board.

A – Run in Simulation: We use the simulation mode to run and debug code in a simulated environment rather than on the physical LaunchPad. The simulator configures the µVision Debugger as a software-only product that simulates the instruction set of an ARM Cortex-M based microcontroller. Developers can test and debug embedded applications before the hardware is ready.

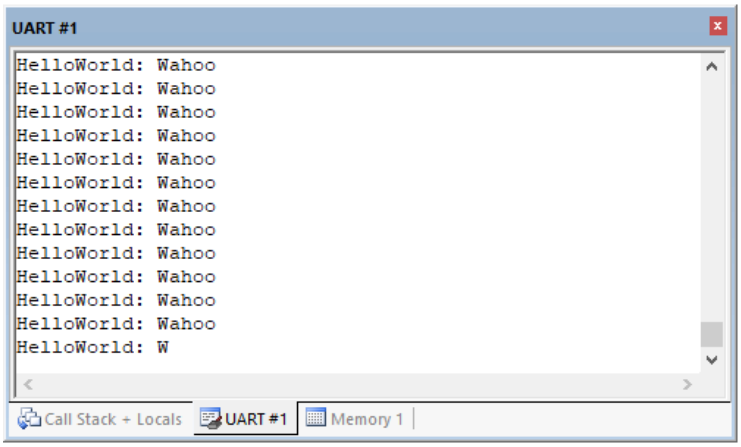
1) In the Debug Tab, select “Use Simulator” in the left panel and add “-dLaunchPadDLL” to the “Parameter” box:



2) In order to see the results of HelloWorld program in simulation:

- Build the project in Keil: Project -> Build target
- Start a debug session: Debug -> Start/Stop Debug Session
- Open up a serial window: View -> Serial Windows -> UART #1
- Run the program: Debug -> Run

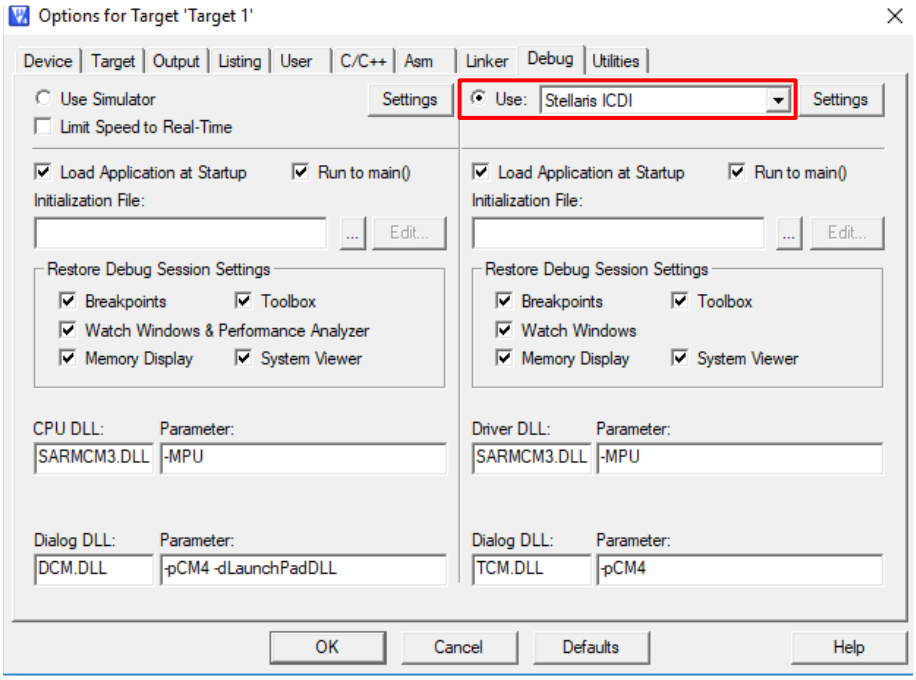
Now you should be able to see the **`printf`** results on the serial window.



The Serial window is one way of communicating with the board, through the UART interface in simulation. It accepts serial input and output data streams. The window displays serial output data received from a simulated CPU, while characters typed into a serial window are input to the simulated CPU. This allows testing a UART interface prior to having the target hardware.

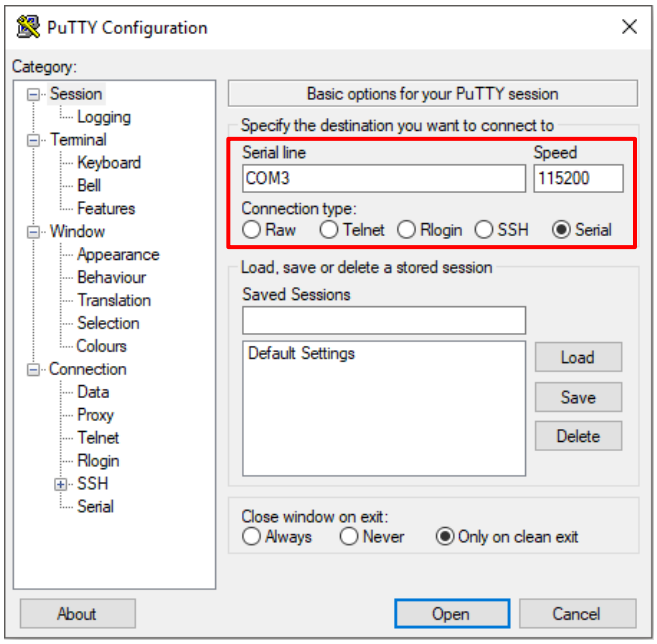
B – Run on Board: We choose the "board" mode to run software on the actual TM4C123G microcontroller. Your LaunchPad board needs to be connected to your PC.

1) In the Debug Tab, select "Use" on the right panel and select “Stellaris ICDI” from the dropdown list.



2) You will need to install PuTTY to see the results of the **`printf`** statement from the actual board. PuTTY is an open-source software which allows serial communication.

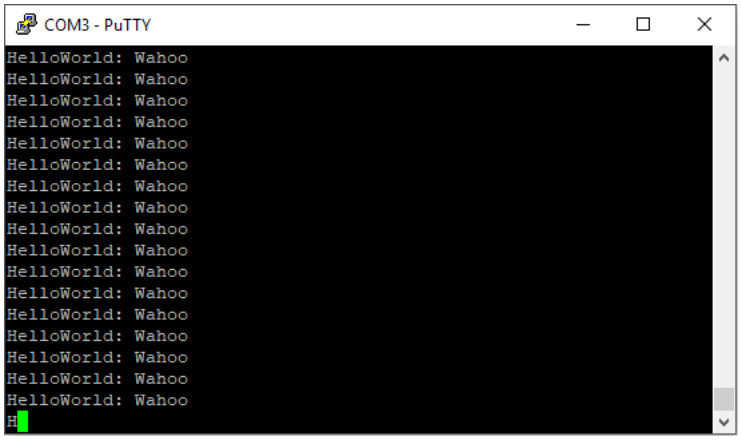
- Install and run PutTTY (you could also download a binary): http://www.putty.org
- Select “Serial”.
- Check in the port number of “Stellaris Virtual Serial Port” device from the Windows Device Manager.
- Set the “Serial Line” to the port number (e.g., COM3) and set the “Speed” as 115200.



- Click **Open**. The PutTTY terminal will show up.

- Build the project in Keil: Project -> Build target
- Start a debug session: Debug -> Start/Stop Debug Session
- Run the program: Debug -> Run

Now you should be able to see the **`printf`** results on the PutTTY terminal.



Deliverables for Part 3: After you make sure the HelloWorld program runs nicely both in simulation and in board modes, change the **`main`** function to print "HelloWorld: <Your Name>". Take snapshots of the results in both simulation (in the UART window) and board (in the PutTTY terminal) modes.

◆

Mini Project 0 Deliverables and Grading

The following table lists the deliverables and their corresponding points. Commit and push your code to your private repository on **GitHub** before the submission deadline. Include a PDF report containing all the deliverables (any calculations, snapshots, answers to questions, and links to videos). The first page of your report should include your name and computing ID. **Note** that the latest commit before the deadline will be considered your submission. Any major changes to your submitted code and report after the deadline and before posting the grades will be considered a late submission.

Survey: After completion of this mini project, please go to the following link and complete the survey: https://virginia.az1.qualtrics.com/jfe/form/SV_d1h4cxVAUYWQdNz.
This will be anonymous feedback but will be counted towards your class participation. Attach a **snapshot** of the completion page to your project report.

Deliverables	**Points**
• A **snapshot** of the UART window in simulation showing the results of the **`printf`** statement.	50
• A **snapshot** of the PuTTY serial terminal showing the results of the **`printf`** statement.	50
Total	100