mbed Lab 1: Hello World and Pushing Buttons

CS 220

Introduction

`Hello, World!` is traditionally a programmer's first program in a new language or in a new development environment. In this lab, you'll start with the mbed analogue to `Hello, World!` to orient yourself both to the mbed development environment and the Freescale FRDM-KL25Z (KL25Z) ARM processor-based microcontroller board that you'll be working with in labs for the rest of the semester.

Lab Objectives

1. Create an mbed development account and configure it for the KL25Z.
2. Download and run KL25Z programs.
3. Make simple modifications to a KL25Z program, controlling the board's built-in tri-color LED using the mbed's `DigitalOut` interface.
4. Make further modifications to the program to accept control from a pushbutton switch, using the mbed's `DigitalIn` interface.

Lab Parts List

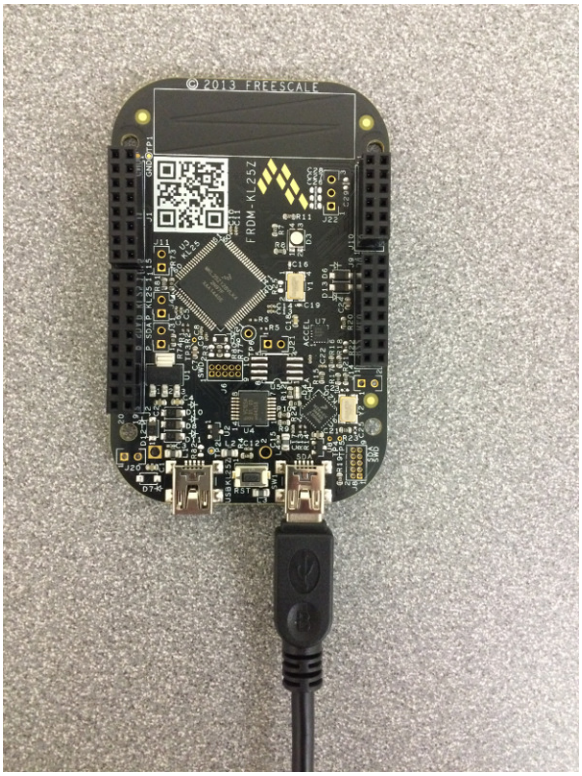
1. KL25Z board and Quick Reference Card
2. USB cable
3. Breadboard, with the normally-open momentary pushbutton switch already mounted on it.
4. Two wires

Lab Procedure

Read each of the following steps **completely** before acting on them.

1. Visit the `developer.mbed.org` website. Create an account and sign-in.
2. Go to the *Boards* page by opening the *Hardware* drop-down and selecting *Boards*. From there, go to the FRDM-KL25Z page. Find the *Add to your mbed Compiler* button and click it. (This step is necessary so that the compiler knows what ARM architecture to generate code for. Code generated for one ARM chip's architecture won't necessarily run on another ARM chip. I wonder why...)

3. Connect the small end of your USB cable to the KL25Z's USB port labeled `SDA`. Refer to this figure:



4. Connect the other end of your USB cable to your PC/Mac. After a moment, you'll notice that an `MBED` drive has appeared in `This PC` or `My Computer` (Windows) or on your desktop (Mac).

5. If you're using your personal Windows laptop/PC for the lab, and your PC is running a version of Windows older than Windows 10, see the course web site for instructions explaining how to install a needed Windows driver.

6. Back in your web browser, on the KL25Z board page, jump/scroll down to the *Downloading A Program* section and follow the steps there to download and run `Hello, World!`. I've had some minor glitches downloading the program directly into the MBED drive, so I recommend downloading the program onto your desktop and then dragging the program into your MBED drive. Downloading the program onto the mbed momentarily unmounts the MBED drive; you can safely ignore any operating system warnings about this. The mbed's LED is pretty bright. You might want to put on your sunglasses, or welding goggles, while you're running the program.

7. Scroll down the web page a bit further and import the `mbed_blinky` program. The import process should open the development IDE in a new browser tab. When the *Import Program* dialog box opens, check the update box so that you pick up the most recent versions of any library files. You'll need your original KL25Z board browser tab in order to refer to mbed documentation, so keep it open.

8. Open the `main.cpp` source file (Yes, you're about to become a C++ programmer.). Here's a copy, with comments drawing analogies to Java constructs:

```
#include "mbed.h"          // Similar to Java's import.

DigitalOut myled(LED1);    // DigitalOut is a class name.  LED1, a KL25Z
                           // pin name, is the argument to DigitalOut's
                           // constructor.  myled is the name of the
                           // created object.  Most KL25Z pins can be
                           // programmed as inputs or outputs.

int main() {               // Similar to "public static void main()" in Java.
    while(1) {
        myled = 1;         // Set the pin to 1.
        wait(0.2);         // Wait for 0.2 sec.
        myled = 0;
        wait(0.2);
    }
}
```

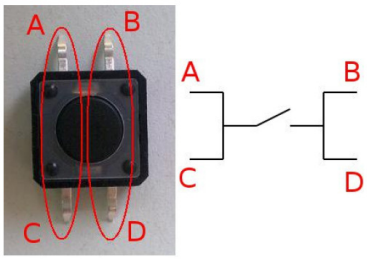
   The pin names `LED1`, `LED2`, and `LED3` have aliases — `LED_RED`, `LED_GREEN`, and `LED_BLUE`, which are more descriptive; you may prefer to use these more descriptive names in your programs.

9. Modify the program so that it blinks a white light at a rate of 1 sec. on and 1 sec. off. (Hint: Add additional DigitalOut constructors to your code.) Compile the program, which will have the side-effect of downloading it. (Put your sunglasses back on!) Confirm that the running program behaves as it should.

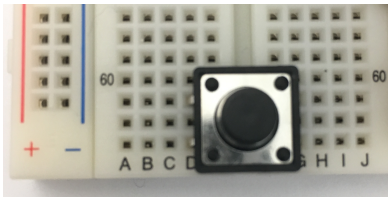
   Does setting an LED's pin to 1 turn the LED on or off?

10. If you've never used a breadboard, or it's been a while since you've used one, navigate to the Wikipedia page on the topic and read the section on Solderless Breadboards. You can stop after you've read the subsection covering Jumper Wires.

11. For this next part, you'll work with a pushbutton switch and the `DigitalIn` mbed interface. The pushbutton switch is what's known as a normally-open momentary switch. Internally, its electrical connections are like this:



The pushbutton should alredy be mounted on the breadboard and oriented so that pins C and D line-up with column D on your breadboard with pin C in row 61 and pin D in row 63. Pins A and D should line-up with column G, like so:



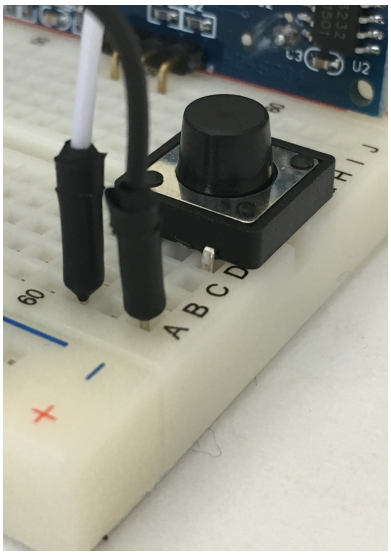
12. You're about to wire the pushbutton to the KL25Z. It's generally a very bad idea to modify the wiring of an electronic device while it's under power, so disconnect the USB cable at the KL25Z end. Otherwise, you may damage the KL25Z in which case your wallet will also be damaged! Be careful while disconnecting the cable, because mini USB connectors aren't exactly known for their durability.

13. For the following wiring steps, use the KL25Z Quick Reference Card to locate KL25Z pins.

14. Grab two jumper wires.

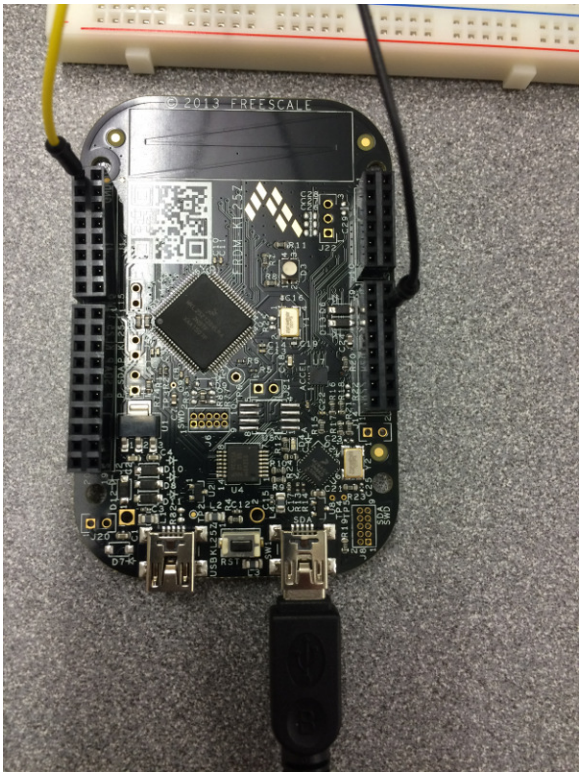
15. On the KL25Z, locate a GND pin (I used the GND pin across from pin PTE4.) Using one wire, wire one electrical side of your pushbutton to GND. Now, locate pin PTA1 and, using a second wire, wire it to the pushbutton's other electrical side.

    Your connections should look pretty much like this:



(Turn the page.)

and this:



Have your partner double-check that you have the KL25Z oriented the same way as in the diagram above and that you've connected the correct pins. If you want to triple-check, have me take a look at your wiring job.

16. When the switch is open, the input pin is logical 1 ($V_{in} = 3.3V$). Closing the switch (pushing the pushbutton), directly connects the KL25Z's input pin to logical 0 (Ground, usually abbreviated `GND`).

17. Reconnect the KL25Z to power. The white light should resume blinking.

18. The documentation for the DigitalIn interface may be found at
https://os.mbed.com/docs/v5.6/reference/digitalout.html
You'll notice that there's sample code.

19. Using the sample code as an example (What pin will you pass to your DigitalIn object's constructor? It's not the pin shown in the sample code.), modify your program so that the pushbutton controls the LED. With the pushbutton released, the LED should be off, and with the pushbutton pushed, the LED should be on.

20. To finish-up, disconnect both ends of the USB cable and then disconnect the two wires. Put the wires back into the wires bag and then put all the parts back into the storage box.