LAB #3 – Conditional Compilation & Serial Debugging

The files you will be using for this lab can be downloaded [here](#).

**Prelab:**

1. What is the purpose of the C preprocessor?
2. Using whatever resources you have at your disposal, give some examples of preprocessor macros you use regularly.

Debugging on embedded hardware presents unique challenges. As many embedded devices lack any sort of screen, something as simple as printf debugging is not possible, directly.

As such, we make use of a Universal Synchronous/Asynchronous Receiver Transmitter, or USART. In this lab, you will be performing serial I/O making use of the FTDI chip on your Wunderboard and the screen application in your virtual machine.

You will be making use of the following functions:

1. void USART_Transmit(unsigned char data);
2. void USART_SendString(const char* str);

The first sends a single byte, while the latter sends a string.

---

**Task 1**

For the first part of this lab, you will need to use your `myitoa()` function, but if you do not have a functioning `myitoa()`, then you can use `gcc`’s `itoa()` to do this lab.

In order to test this portion of the lab, you will be using the `screen` utility. In order to run it, open a new terminal window, and execute

```
sudo screen /dev/ttyUSB0 9600
```

This will open a serial console to your Wunderboard. Any messages you print using the USART functions will appear in this console. In order to exit screen, hit Control-A, k, then hit y.
Task 2

After you have it working, display an X pattern on your LED array. If no buttons are pushed, it should be green. If button 1 is pushed, it should be red. If button 5 is pushed, it should be blue. Remember, you wrote code that queried which button was pushed in the previous lab. For each LED light turned on, print out it's location over the USART, with the origin in the upper left corner of the Wunderboard.

You can use either the `set_array_*(row)` functions, which work as you learned last lab, or a new set or "pixel" based functions, `led_*(x,y)`. These functions turn on an individual LED, with the origin in the upper left corner, as is customary in graphics work. See `leds.h` for further details.

Task 3

Assuming you answered the pre-lab questions, you saw some examples of using the preprocessor. One of the useful features of it is the ability to conditionally include code, based on macro definitions. For instance, this is often referred to as a DEBUG macro:

```c
#ifdef DEBUG
    /* your debug code here */
#else
    /* your non-debug code here, if differences exist */
#endif
```

In some cases, you have different function calls for debugged versions, in other cases, debugging code is simple output statements. For this task, ensure that all of your USART function calls are wrapped in DEBUG macros.

**Be sure to get checked off by a lab TA before you leave lab to receive credit for the lab!!**