Laboratory 12 on Arduino Uno

The experiment has two tasks.

Task 1: blink the LED with abandon at a rate of 5Hz.

Task 2: Alternate between two LEDs being on with the press of a button.

Task 1: Output Only

This task is very much like the one we did in class on Tuesday, but this time you will do it yourself and wire up the LED on the bread board. This is an output-only task. The process in all micro-controller circuits is three-fold:

1. Wire the circuit.
2. Write the program.
3. Program the chip.

Start by wiring up the circuit

Circuit construction

Power, ground, signal

Lets plan on using Pin 10 for output.

We will run our circuit using the power provided to the Arduino by the computer via the USB connection (voltage at +5V from ground). Connect the 5V power slot on the Arduino to one rail of your bread board and the ground slot to another bread board rail. Start a third wire from Pin 10 to carry your output signal to the board, plan on using HIGH for LED on and LOW for LED off.

Sketch the circuit you will build for your output before you read any further, it will contain, among other things an LED.

Your circuit should connect Pin 10 to the LED through a 330Ohm resistor to ground. If you have not already built this circuit please do so now.

Writing the program

Your program must do 5 things.

Set Pin 10 as output in the software

Turn on the LED

Wait

Turn off the LED

Wait

To accomplish these things we will need three functions: pinMode(pin #, mode code), digitalWrite(pin #, value code), delay(#ms)

Your goal is to have a flashing period of T=1/f=1/5Hz=0.2s. This is the total of the time the LED spends on and the time it spends off. Giving them equal time your sketch (program) should look like this:

```
void setup(){
        pinMode(10,OUTPUT);
}
void loop(){
        digitalWrite(10,HIGH);
        delay(100);
        digitalWrite(10,LOW);
        delay(100);
}
```

Program your controller.

Check your code and if it is sound program your controller with it. Upon completion of programming your circuit should start to blink. You may need to point the Arduino IDE to your board, to do so select the Tools menu, then Port and select Arduino Uno.

Modify to check understanding

1. Change your program so that the LED blinks at the same frequency but spends twice as long on as off.
2. Change the blink frequency to 50Hz.
3. Add another LED that is on when the one you already are using is off and vice versa.

Task 2 Output and Input.

There are two methods for reacting to input: polling and interrupts. We will use interrupts today. We will take input from a push-button that is high when unpressed and is low when pressed. We will use the same three steps as before: build circuit, write program, program chip.

Circuit Construction

Lets plan to use Pins 9 and 10 for output and to use pin 2 for input (only pins 2 and 3 can be used for interrupt on the Arduino Uno).

This time around we will need both input circuits and output circuits. You can use the same output circuit as before. Sketch both the input and output circuits here before reading any further.

Your output circuits should run from Pin 9 ->LED A->330 Ohm ->GND and from Pin 10 -> LED B ->330 Ohm ->GND. The input circuit should connect Pin 2 to the switch, have a pull-up 10k resistor to 5V and connect to ground on the far side of the switch. Build these circuits if you have not done so already.

Writing the program

You will need accomplish x things:

Set pins 9 and 10 as output and pin 2 as input.

Turn pin 2 into an interrupt pin.

Turn on LED A and off LED B

Turn off the LED A and on LED B if the button has been pressed.

In addition to the functions that we used before we will need the following functions: attachInterrupt(details to follow), digitalPinToInterrupt(pin #), a function of your own to run when the interrupt happens. Your function must be type void and take no arguments.

The function attachInterrupt(...) takes three arguments the interruptPin, the interruptFunction to execute, and the interrupt condition.

- interruptPin must be passed through the function: digitalPinToInterrupt().
- interruptFunction is any void, argument free function that you define.
- interruptCondition is one of the following 5 options
    - LOW: trigger when pin is at a low value
    - RISING: trigger on positive edge
    - FALLING: trigger on negative edge
    - CHANGE: trigger on either edge.

We will use LOW.

Try putting all of these things together to write your own code. If you get stuck the following one possible code:

```
void setup(){

        pinMode(9,OUTPUT);

        pinMode(10, OUTPUT);

        pinMode(2, INPUT);

        attachInterrupt(digitalPinToInterrupt(2), flip, LOW);

}

void loop(){

        digitalWrite(10,HIGH);

        digitalWrite(9,LOW);

}

void flip(){

        digitalWrite(10,LOW);

        digitalWrite(9,HIGH);

}
```

The above code repeatedly turns on the LED unless someone has the nerve to press the button.

Program your controller

Enough said.