NAME(S) _____

CS 341 – Lab 1
Computer Architecture and Organization
Introduction to the Arduino Microcomputer

Equipment: Arduino UNO microcomputer, PC with Arduino IDE installed, and a USB cable. 1K ohm resistor, Light Emitting Diode (LED), and wires.

The Arduino microcomputer is a low end microprocessor on a small printed circuit board. In our lab configuration, it includes a small breadboard for prototyping and connecting peripherals. The TA will help you to form teams appropriate to the number of students in the lab and amount of available equipment. Your team will be provided a plastic box containing the parts kit for use in all of the labs this semester. Please be careful handling the parts as some of them are delicate and keep track of them to return them to the box/envelope when you are done. Return the box to the TA at the end of each lab session.

In our lab, the Arduino UNO will be the target host for our experiments and the PC attached to it via the USB cable will be the development host. You will use the Arduino.exe IDE program to edit, compile, download, and monitor the execution of your "sketches" (experimental programs). The "sketch" software is written in a language that is based on C and C++. There are a few

extensions to the language and a custom library to access the peripheral hardware input and output ports.

Your first assignment is to set up your Arduino system and run the example "Blink" sketch. Then, you will build an LED circuit and make a few modifications to that sketch.

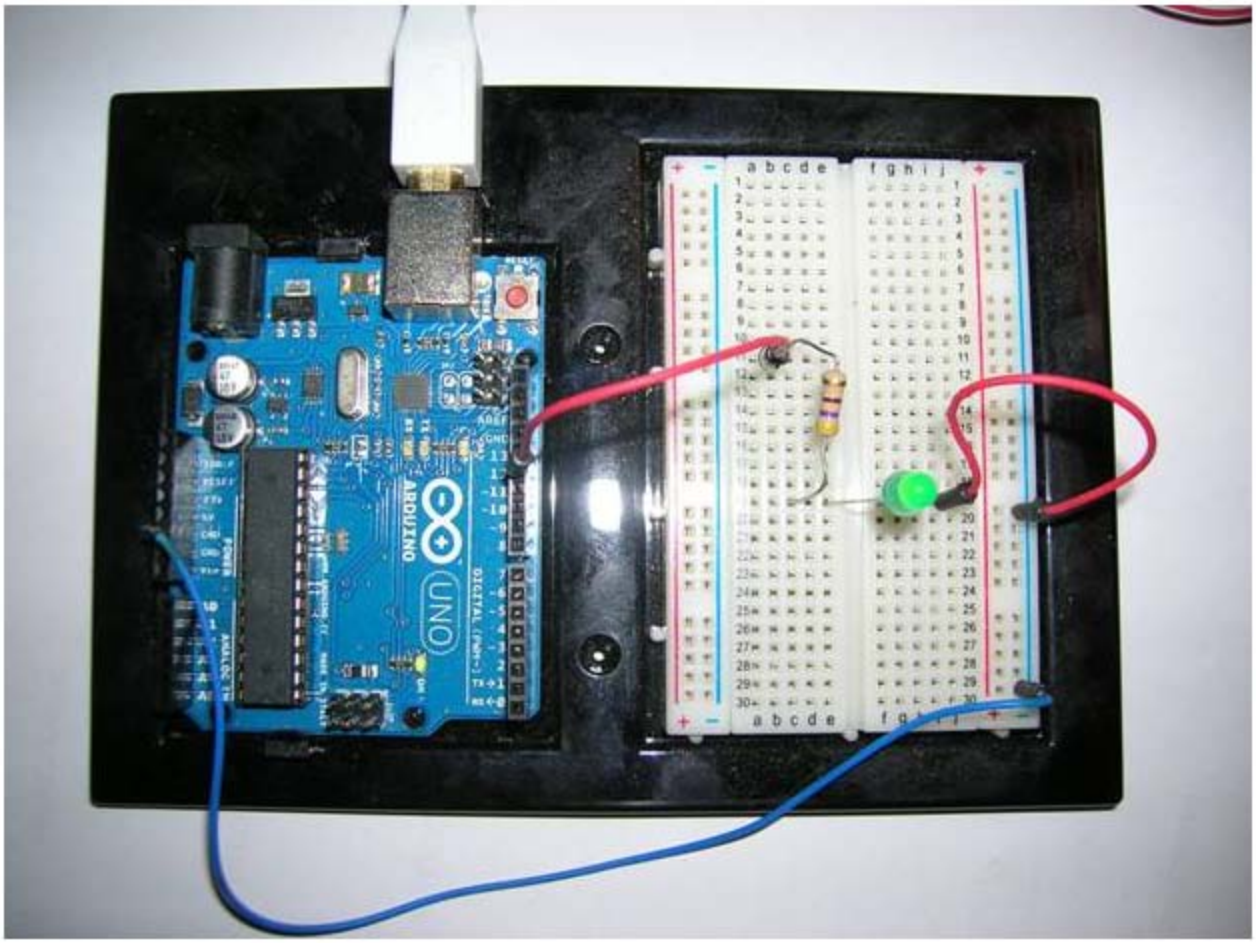
Connect the PC to the Arduino UNO board using the USB cable. Open the Arduino.exe program. Use the menu: File >> Examples >> 01.Basic. Select the Blink program which will open a new window containing the source code for the Blink program. Note the structure of the code in the sketch. There are two functions: setup and loop. These are run by a main function that you don't see or need to write. The implicit main function looks like this: main.c.

Use the "right arrow" icon on the top tool bar to compile and upload/run the code on the Arduino board. Watch the LED's on the Arduino board during the upload to observe it occurring. Once the program starts running, the yellow LED will be flashing at a rate of one second on and one second off.

Now study the code in the source of the sketch. Find the lines that control the flash rate for the blinking LED. Change those lines to slow down the flash rate to one half of the current rate. Repeat the steps for compiling, uploading, and running your modified code. Show the TA when you have this working. NOTE: When you try to save your modified sketch, the IDE will prevent you from saving it and overwriting the example file. You will need to save it under a new file name.

Now disconnect the Arduino board from the USB port. *NOTE: When you are adding, changing, or removing wiring on a prototype connected to the Arduino UNO board, always disconnect the power from the USB port and check your wiring carefully before reconnecting it to the USB port. Otherwise, you may damage the Arduino board. If you have any doubts, show your wiring to the TA before reconnecting it to the USB port.*

Insert an LED and a resistor in series on the breadboard to pin 12 on the Arduino board following the design shown in this picture. The longer leg on the LED should be connected through the resistor to the Arduino output pin and the shorter leg of the LED should be connected to ground. Typical colors to use for DC wiring applications are: black for ground, red for plus voltage (+5 V in our case), and other colors for signal connections. (The 12 VDC electrical system wires in cars and boats are color coded this way.) Although the electrons in a circuit actually flow from the negative pole towards the positive pole, the direction of the current is conventionally described as though it were flowing from the positive pole to the negative pole. The negative pole is normally considered to be the "ground" and may be connected to the chassis of a car or an "earth ground" in some applications. Note: The 5 VDC that we are using in these experiments is classified in the national electrical code as Separated Extra Low Voltage (SELV) and is considered to be "safe" if you are exposed to it.



The resistor should be 1K ohms which you can determine from the color of the bands around the body of the resistor (brown = 1st digit 1, black = 2nd digit 0, red = two zeroes). See the following standard resistor color code table for determining the value of resistors by their color code:

Color	First Band (1st Digit)	Second Band (2nd Digit)	Third Band
Black	0	0	no zeros
Brown	1	1	one zero
Red	2	2	two zeros
Orange	3	3	three zeros
Yellow	4	4	four zeros
Green	5	5	five zeros
Blue	6	6	six zeros

Violet	7	7	seven zeros
Gray	8	8	eight zeros
White	9	9	nine zeros

There may be a fourth color band for the tolerance of the resistor: Silver is +/- 10 % and Gold is +/- 5 %. If there is no band, the tolerance is +/- 20 %. We don't need very accurate resistance values for an application like this one, so any tolerance value will be acceptable.

Study the code in the source of the sketch. The Arduino source code is written in C (with a few proprietary extensions that we will discuss later). Find the line(s) that control the pin number being used to flash the LED. Change those line(s) to use pin 12 instead of pin 13 to flash the LED on your breadboard. Repeat the steps for compiling, uploading, and running your modified code.

The Tools > Serial Monitor menu allows you to write code that prints data from your Arduino board to a window that represents a serial monitor. In the sketch code, the Serial object represents the monitor. It has a begin function to set the baud rate (9600 bps) and print and println functions similar to the Java System.out print and println methods. You can use them to write from the Arduino board to the monitor display. Add the following lines of code to the setup method in your sketch.

```
Serial.begin(9600);
Serial.println("Hello world.  Watch me blink an LED.");
```

Compile, upload, and run it. Then, use the Tools > Serial Monitor menu to open the display in a window to see the "Hello world. … ". Experiment with this feature until you understand how it works. You will be using it in later lab assignments.

To get credit for this assignment, show the TA after you have the LED constructed on your breadboard and flashing and the serial monitor display working.

As a team, write your lab report to explain what you did, how you did it, and what you learned about interfacing hardware to a microprocessor and its software (the sketch). Attach the final source code for your sketch. Turn in your report at your next lab session.

____ / 10