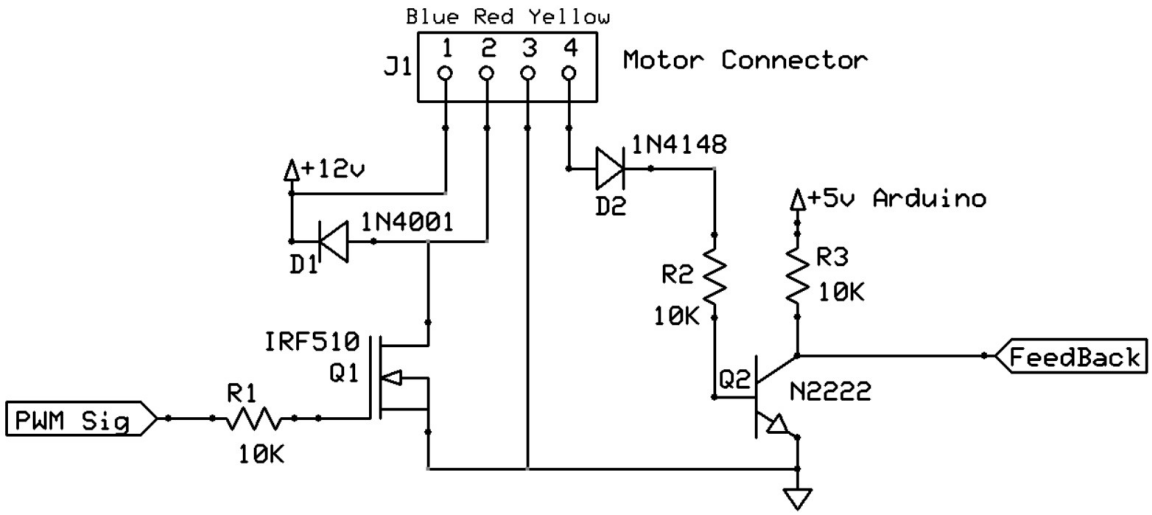
Motor Controller (Cost $15)

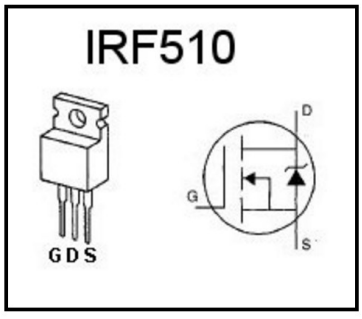
The objective of this project is to control the speed of a motor, by adjusting the duty cycle of a Pulse Width Modulated (PWM) signal. The kit for this project consists of a motor with a built in tachometer, a small circuit to convert the tachometer signal to a digital signal and a motor driver to allow the Arduino to drive the motor. An RPM readout (LCD) and speed control (encoder knob) will be expected.

The motor that is part of this kit is a motor and encoder set. The motor itself will be driven by a power MOSFET (IRF 510) which can be driven by the Arduino through a 10 K resistor. The feedback of the motors speed is accomplished by an encoder that is part of the motor-gearbox-encoder system.

Some of the details of computing RPM from the tachometer signal, requires a couple important points. First the tachometer produces 8 cycles per revolution, so if you measure the cycle time (Δt) in μseconds from rising edge to rising edge you will compute RPM as

$$RPM = \frac{60x10^6 * 1}{8 * \Delta t} = 7.5\, x \frac{10^6}{\Delta t}$$





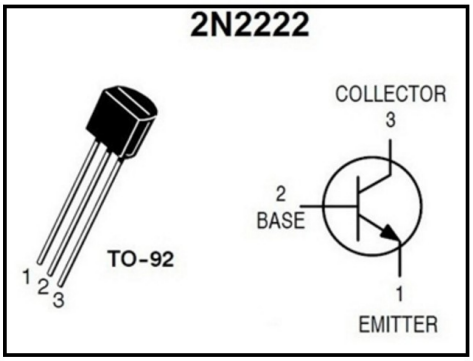


Figure Proj-1A. Schematic for the Motor Controller.

Features and Their Weighting

1) Encoder controls the RPM setting which the controller is trying to maintain. (4 points)
2) Reasonably good control of the RPM, under varying loads. (3)
3) RPM is calculated using provided formula (3)
4) RPM is displayed on the LCD (2)
5) Presentation skills (3 points)

LED CUBE (Cost $20)

The objective of this project is to create a three-dimensional graphical display. These displays called LED cubes are best demonstrated by a video, the link to which is given below.

https://www.youtube.com/watch?v=yst4eL0-Eco

There is also a LED cube construction video, similar to the way we will be making ours.

http://www.instructables.com/id/4x4x4-LED-Cube-Arduino-Uno/?ALLSTEPS

https://www.youtube.com/watch?v=vf_IpviMiFU

Be aware that we will be programming ours differently, but the construction of the cube will be much the same. Now to that point the following is the circuit diagram for the circuit board that will be in the kit.

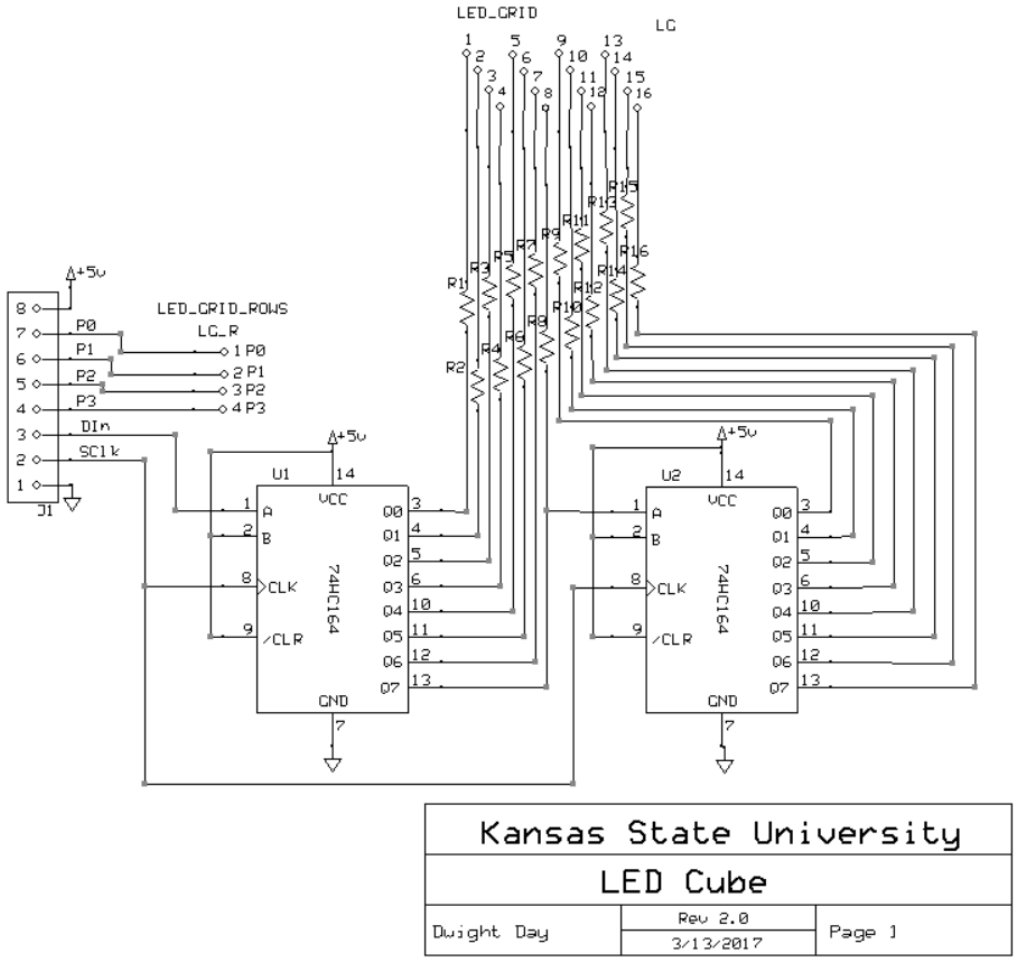


Figure Proj-4A. Driver Circuit for LED Cube PCB.

A complete description of this circuit and setup can be found under “Files->Course Notes->Hardware Documentation->E_LED_Interface.pdf”

Features and Their Weighting

1) At least three separate modes of operation (flash pattern) for the cube.
   One of the patterns needs to demonstrate and test that each LED is addressable. (7)
2) Mode change should be changed by pressing a button, which is properly debounced. (5)
3) Presentation skills (3 points)

Simon - The Game. *(Cost $10)*

This project is a simple game that was one of the first electronic games. If you are not familiar with it take a look at this video. https://www.youtube.com/watch?v=1Yqj76Q4jJ4 It is hard to explain the game and more verbiage here would not help.

The system should implement the following process.

1) Starting in a Wait state, then on the press of the button, start the game by randomly picking one of four colors (four LED's) and turns on this first LED for 1 second.
2) Then if the user press the matching button within 5 seconds, the game continues.
3) The system randomly picks another color.
4) Then plays back the previous colors, along with the new color, each for 1 second.
5) The user must press the buttons, in the same sequence, with a time out of 5 seconds between each press.
6) If completed correctly the system moves back to step 3.
7) If not correct, do a display, flashing the LED's in some kind of interesting pattern, and then go back to the wait state.

Other features that are needed are 1) setting of the random seed to help making the pattern more random, 2) keep a maximum score (longest sequence remembered), along with current length being attempted, and 3) save the length of longest sequence to EEPROM, and reload it on start up.

This project has a minimal set of hardware, consisting of four LED's, each having a different color, four resistors, to limit the current through the LED's, and four switches. The Kit will come with a circuit board on which you can build the system or if you have one you can wire it up on a protoboard.

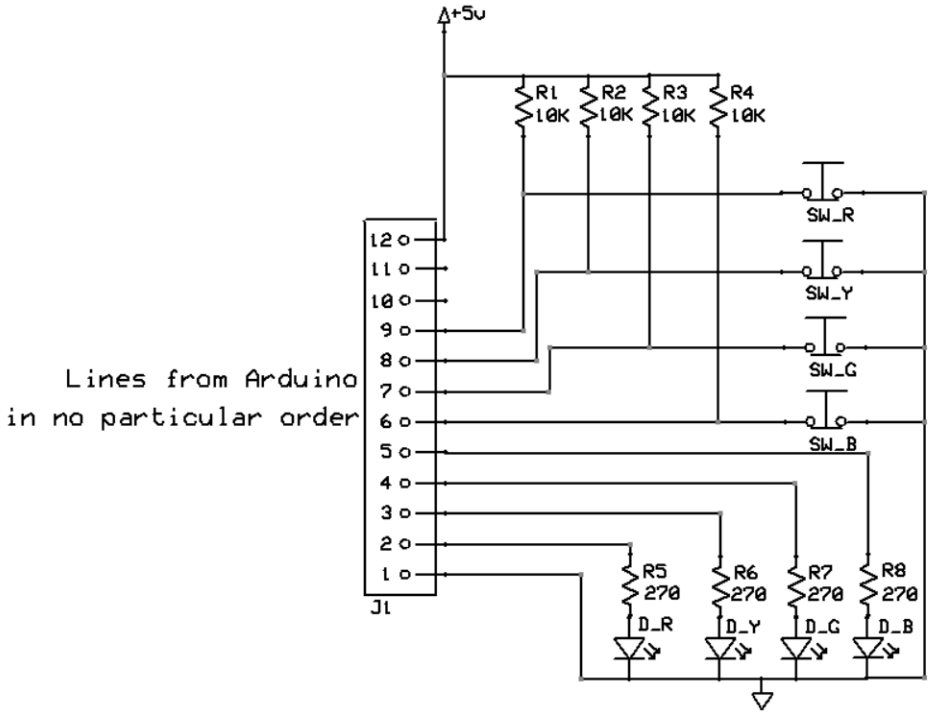


Figure Simon.1 Basic Circuit.

Features and their weighting

1) Show setting of random seed (5)
2) Current length and maximum length achieved displayed on LCD. (5)
3) Maximum length stored to EEPROM and held through power down. (5)

Pulse Rate Monitor (Cost $30)

The objective of this project is to display your pulse rate on the LCD and control a Pulse Width Modulated (PWM) signal sent to an LED using the Timer1 system or the `AnalogWrite()` function. This project is based on the PulseSensor – a simple, light-based device for monitoring pulsatile changes in blood volume or a photoplethysmogram (PPG). An example PPG waveform captured using the PulseSensor is depicted in Figure 1. A PPG waveform is generated at each heartbeat. In Figure 1, two complete PPG cycles are presented.

In order to calculate pulse rate, you will need to measure the amount of time between consecutive PPG cycles (the heartbeat interval or HBI). For this project, you will use six heartbeat intervals to compute an average pulse rate using the following formulas.

$$Average\ HBI\ (in\ seconds) = \frac{HBI1 + HBI2 + HBI3 + HBI4 + HBI5 + HBI6}{6}$$

$$Pulse\ Rate(beats\ per\ minute) = \frac{1}{Average\ HBI\ (seconds)} * \left(60\ \frac{seconds}{1\ minute}\right)$$

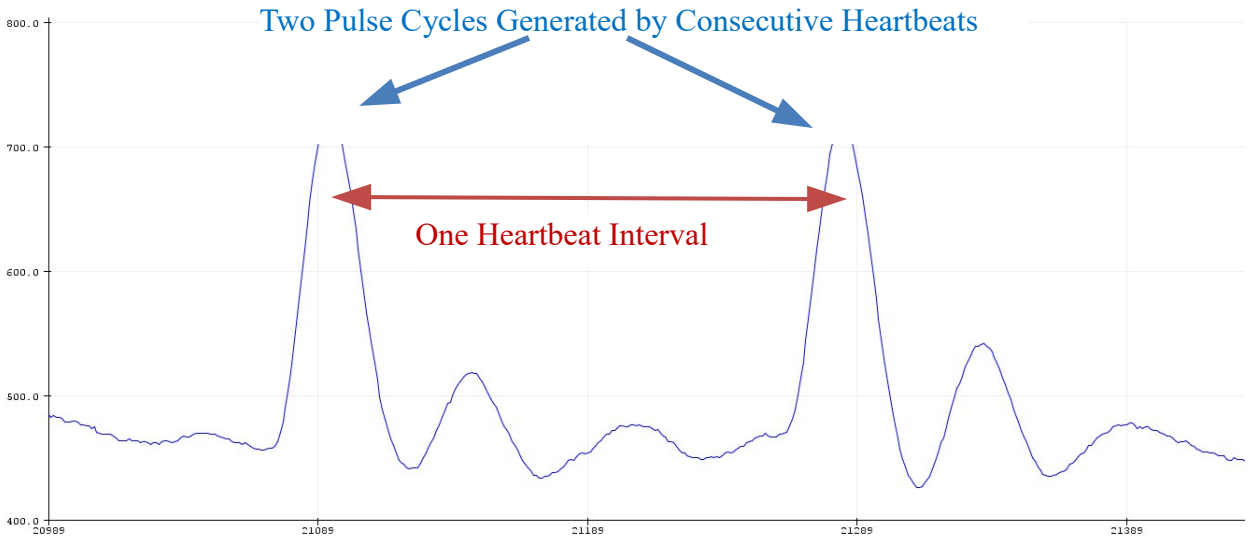


Figure 1. Example PPG captured from the Serial Plotter.

Thus, your code will need to collect six heartbeat intervals before computing a new estimated pulse rate. The transition diagram to accomplish this is illustrated in Figure 3. You will notice there are two threshold values that you will need to determine. Threshold 1 is the value where you decide that a new pulse wave cycle has occured. You can experiment with the PulseSensor (use the Serial Plotter to view the `analogRead()` value) to determine a reasonable value for Threshold 1 (see Figure 2).

You will also not want to count the same pulse cycles multiple times. Therefore, you will need to determine a reasonable wait time before allowing a second cycle to be detected (Threshold 2).

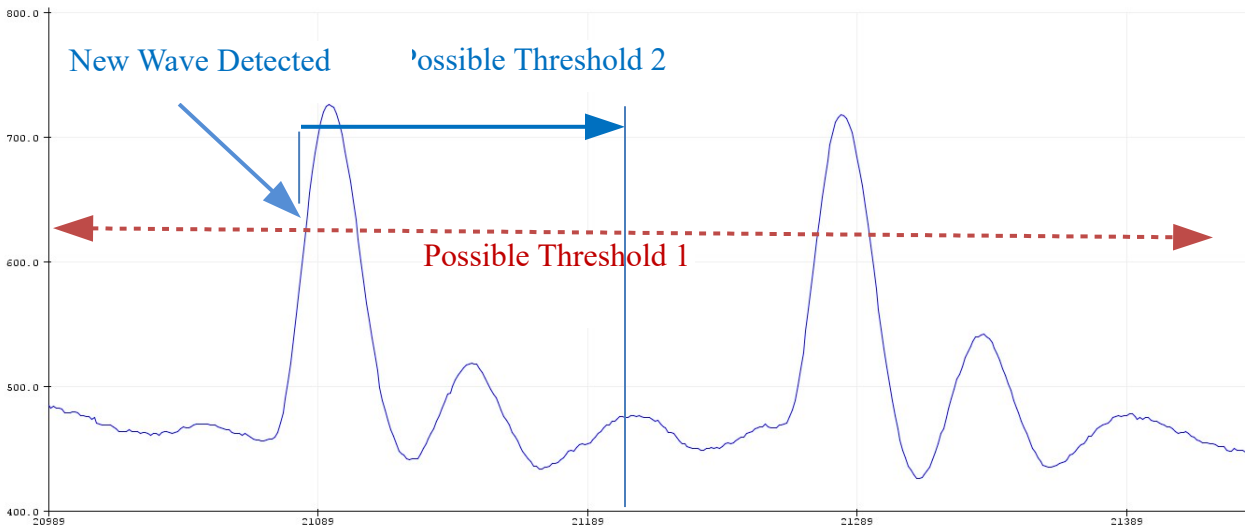


Figure 2. Example threshold values.

With each new detected cycle, (the `analogRead()` value goes above Threshold 1), you will need to record the amount of time that has passed since the last cycle ("Get Delta Time"), and increment a counter. The counter will be used to determine when six heartbeat cycles have been detected so that a new pulse rate value can be estimated and displayed on the LCD.

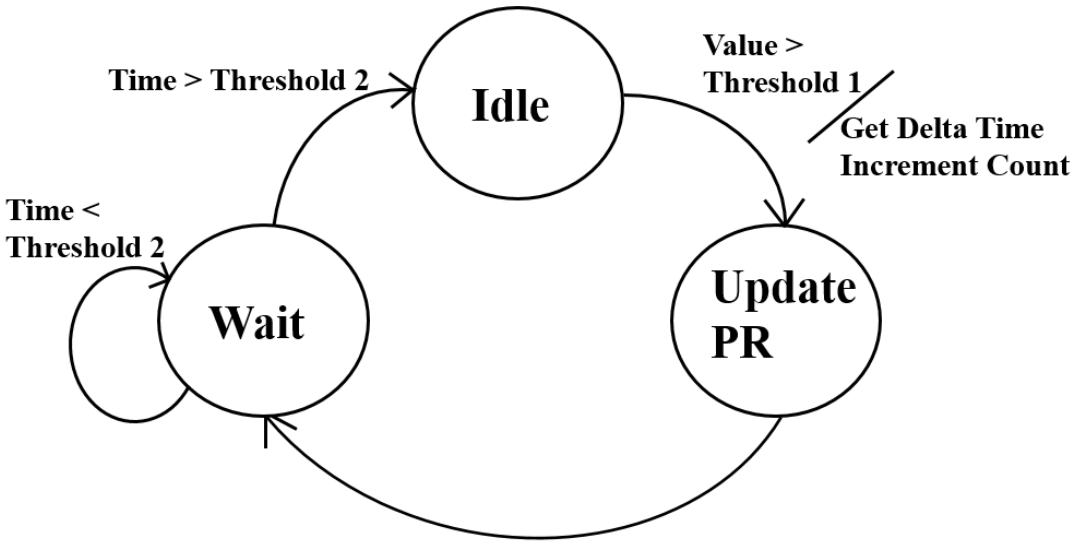


Figure 3. Pulse Rate Monitor State Transition Diagram.

To checkoff this project, you will need to display the estimated pulse rate (in beats per minute) on the LCD. Further, the `analogRead()` value should be used to control a PWM sent to an LED (see Figure 4). The LED's brightness will be controlled by the pulsatile waveform. The LED should dim and brighten with each cycle – providing a visual reference for when heartbeats are occurring.

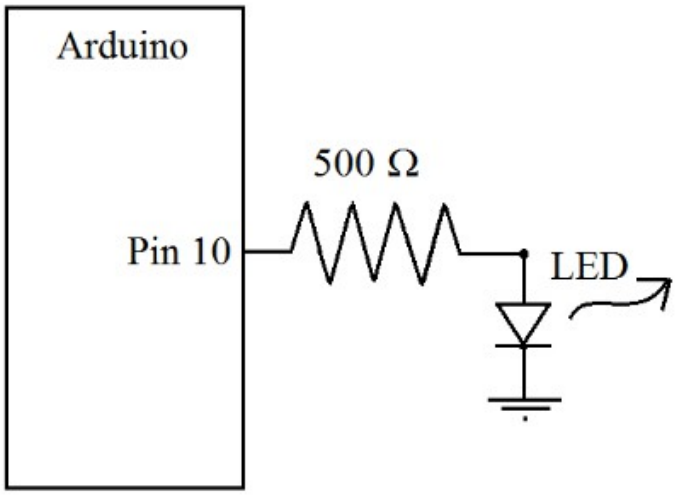


Figure 4. Circuit for LED controlled by a PWM signal.

1) Led pulsing reflecting Heart Rate (HR) (3 points)
2) HR displayed (4 points)
3) HR consistent (averaging) (5 points)
4) Presentation skill (3 points)

Solar Tracker (Cost $10)

The objective of this project is use a servo motor to turn a set of light sensors towards a light source. This system could be used to keep a solar cell pointed directly at the sun. A time of day and angle readout (LCD) are expected.

Consider the voltage on pin 3 (coming from between R1 and R2) as the measure of the total light present. Then if this voltage is too high, indicating that there is not enough light, you should set the position of the servo based on the time of day. For simplicity, simply set the position to 0 for 6 AM, 180 for 6 PM and linearly in between. Note that for after 6 PM, it should simply move to 0 in anticipation.

If the total light is sufficient, you should move the servo, such that voltage on pin 4 (from between R2 and R3) is approximately half of the voltage on pin 3.

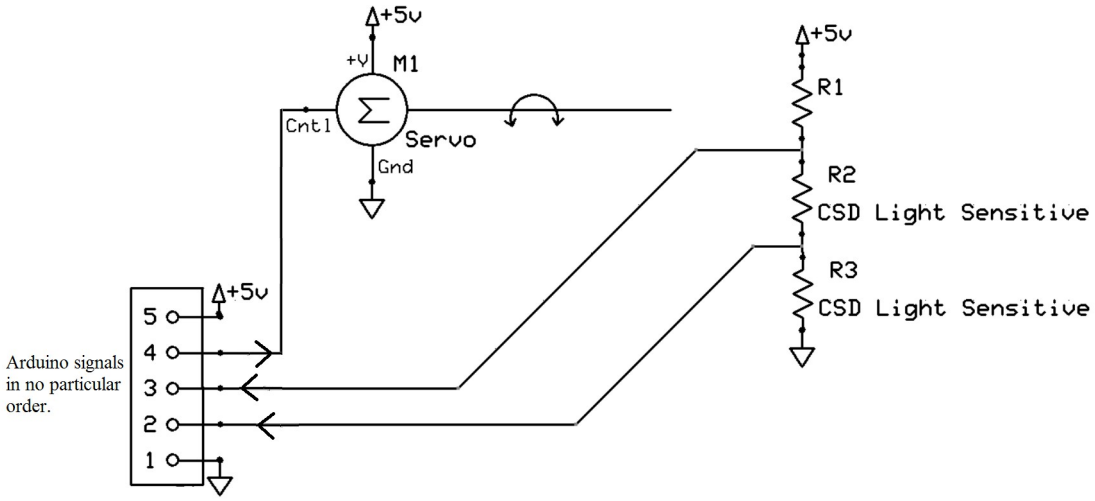


Figure Proj-2A. Schematic for the Solar Tracker.

Features and Their Weighting

1) Ability to track light. (3 points)
2) If light source is insufficient (below a threshold) servo position is set based on time. (3 points)
3) Display clock on LCD. (1 points)
4) Display angle on LCD. (1 points)
5) Ability to set clock using encoder or button (4 points)
5) Presentation skills (3 points)

Sensors are soldered such that they can bend over the edge, pointing the light sensitive portion out.

Figure Proj-2B. Mounting of Light Sensors and Board to Motor