PATH1 **Rev. 9.12.2020**

Step 1:

Go to: http://learn.parallax.com/propeller-c-tutorials

Click on "Propeller Brains for your inventions". Read the page and follow the links at the bottom of each page. While you are going through this tutorial you should know the meanings of unfamiliar words such as:

microcontroller
multicore microcontroller
multitasking
multiprocessing
functions
library

CAUTION:
There are many words in the English language which have more than 1 meaning. Functions and library are examples of such words. You need to know the meaning of those words in the context of these tutorials.

Step 2:

For what follows you MUST have the BlocklyProp client installed. You did that over the summer. If you did not do this during the summer, then do it now by going to https://www.ionaphysics.org/lobby/robotics/classroom/outline/letter%20to%20students.pdf and following the directions you find there.

Step 3:

A lot of computer programming courses start by programming the computer to say "Hello World". We will do that together in class.

Afterwards, we will add some complexity to that program.

Then we will move on to programming with hardware.

The first job will be to program the microprocessor to blink an LED (light-emitting-diode). The LED is already on the Activity board. Go to the following link and follow the instructions.

http://learn.parallax.com/tutorials/language/blocklyprop/blink-light

Follow the instructions on the **first page** and be sure you understand what is going on. Later experiments will depend upon this understanding.

Step 4:

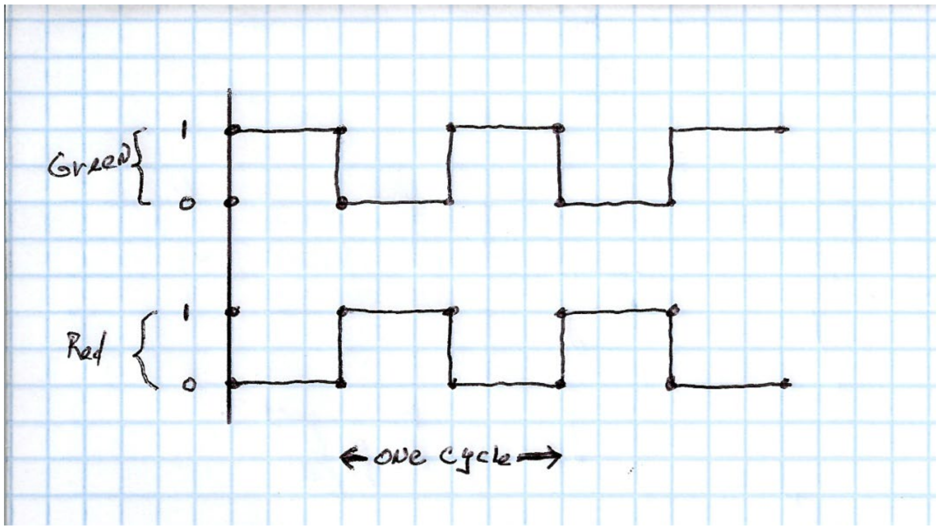
Continue to follow the instructions by clinking the links at the bottom right. STOP when you get to the link which says "Check Pushbuttons".

Step 5:

This is our first engineering project: The traffic light.

Go to : http://en.wikibooks.org/wiki/Fundamentals_of_Transportation/Traffic_Signals and read about the timing of traffic lights. The formulae will probably be very interesting to those of you who are inclined toward mathematics. They do show how a fairly simple object may actually be more complex than it seems. (If you want to pursue this topic see extension 2 below which goes into great detail.) However at this point you only need to get the flavor of the article. It is important to note that there are fixed and adaptive schedules. The fixed schedule is what we will work with during the next few steps.

Assume we have an intersection with a fixed schedule traffic light with the following timing diagram:



Consider time along the X axis. You can see that when the green light is on (1), the red light is off (0), and vice-versa. The length of one cycle is not given. In real life that would be several seconds. However, for the sake of keeping your debugging time reasonable, we will make the time of one cycle 4 seconds. It should be obvious that the cycle time would be easy to adjust when we go from the lab to the real world.

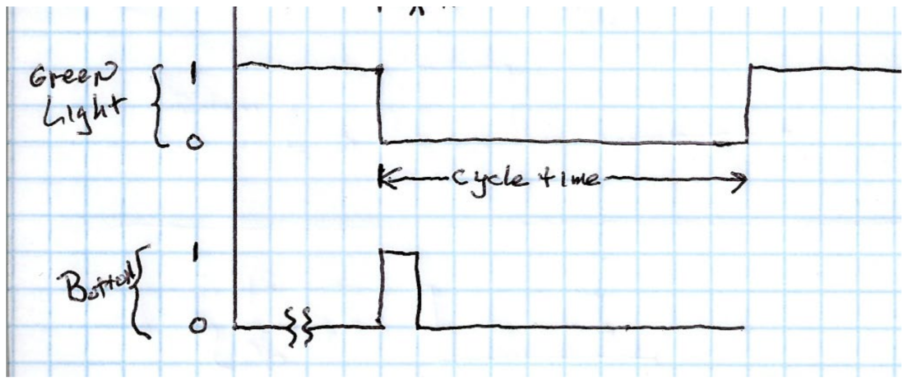
(A) You will be given specifications for the LEDs. You will then calculate the value(s) of the series resistors. You will then draw a schematic diagram of the circuit and have it approved before continuing……. (Continuation) Wire a red LED, and a green LED according to the approved schematic. You may assume they control one direction and the other direction is exactly the opposite. (Red in direction A causes Green in direction B. There is NO yellow (amber) light.

Write a program which controls the traffic light and verify that it operates according to the previous timing diagram with cycle time = 4 seconds.

Extension 1: Now suppose you want to connect another red and another green for the other direction. There are two possibilities.

> (i) You could connect the new LEDs in parallel with the existing LEDs by simply connecting the new red in parallel with the old green and the new green in parallel with the old red. Consider that, and decide how that would influence the voltages and currents. Would a single microcontroller pin be able to handle two LEDs? Would the value of the series resistor need to change? Discuss this with your partner. Verify your answer by speaking with your teacher.
>
> (ii) You could also connect the new LEDs to two different microprocessor pins. That would mean adding programming to control the other two pins.

(B) You are now in a different location. There is a single street, with no cross street, but there is a pedestrian crosswalk. Here the light is always green until a pedestrian pushes a button. When the button is pushed the light changes according to the following timing diagram:



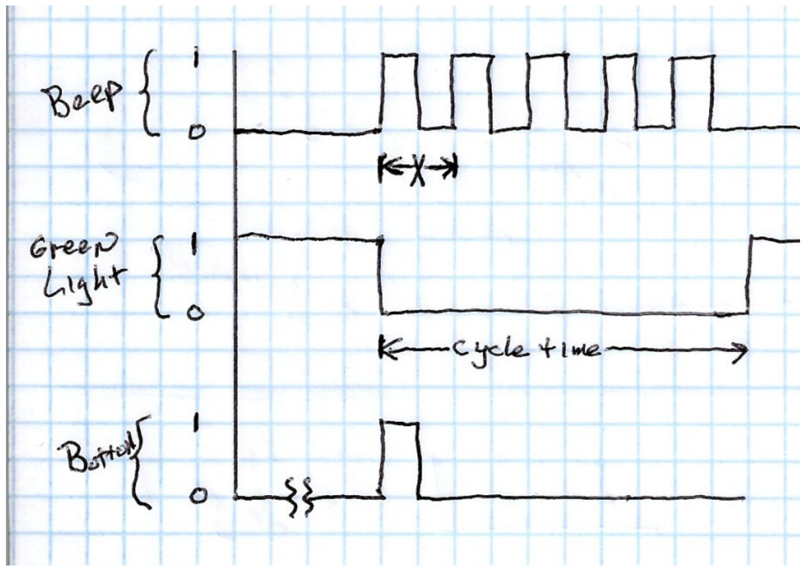
Explanation: The light remains green until the button is pushed. The length of time you hold the button does not matter. Once the button is pushed the green light goes off and stays off for the cycle time. We will make the cycle time 20 seconds. After 20 seconds has passed the green light comes back on.

In order to learn how to use the pushbutton you will need to go to the following link:

http://learn.parallax.com/tutorials/language/blocklyprop/circuit-practice-blocklyprop/check-pushbuttons

Modify your circuit by adding an active high button. Now write a new program which will respond according to the timing diagram above.

(C) Now we consider the fact that the above crosswalk happens to be in a location where there are blind people. We need to add an audible indication that it is safe to cross the street. Add a piezo speaker to your circuit and reprogram the light according to the following timing diagram:



Step 6:

Go to: http://learn.parallax.com/propeller-c-simple-circuits/seven-segment-display and learn about the 7-segment display.

Step 7: Return to the circuit you constructed for step 5C above and add a 7-segment display counting down. Since there is a 20 second walk interval, the display should be dark for the first 10 seconds and then it should count down 9-8-7… during the last seconds until it reaches 0 when the beeping stops and the light again turns green for the traffic.

Extensions:

1. It may come as a surprise that some traffic signals are not timed properly. Look at this laboratory exercise concerning the dilemma zone which can be created by an improperly timed signal.
http://www.ionaphysics.org/lab/resources/YellowLight.pdf
You might want to take some intersection in your town which has a traffic signal and determine whether or not there is a dilemma zone. (One of my students did this several years ago and used the information to get a traffic ticket dismissed.)

2. Here is a link to a good tutorial on the actual considerations involved in timing a traffic signal:
http://www.webpages.uidaho.edu/niatt_labmanual/Chapters/signaltimingdesign/Introduction/index.htm

You might want to investigate some intersection in your town and determine how well or how poorly the signal is timed.

3. Look at the Signal Timing Manual of the Federal Highway Administration to see all the gory details of timing signals in a real city environment.
http://www.ionaphysics.org/lobby/robotics/classroom/outline/SignalTimingManual.pdf

Humorous postscript:

Because the light cycle introduces delays, it frequently can be more efficient to leave an intersection uncontrolled (without a traffic light). In that case the motorists simply cooperate. View the following video which is an actual intersection in India. There is great cooperation there!

https://www.youtube.com/watch?v=RjrEQaG5jPM

This one is even better. It appears that a traffic light can be replaced by beeping horns!

https://www.youtube.com/watch?v=pLUm3Q-7iZA

Here endeth the traffic signals projects.