Model Railway Animation: Part 4, LCD Displays: Expanded
By David King

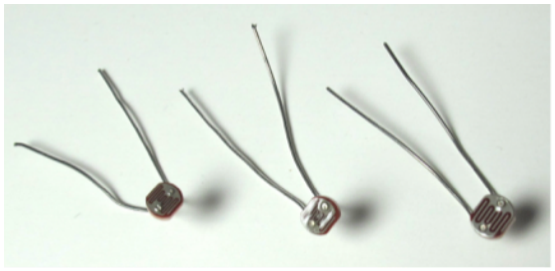
Welcome to the expanded article for using your LCD display and a couple of photo resistors, LDR's, to create a section of rail line that can be used to monitor to speed of your train as it passes through in miles per hour, mph. For this sketch we are going to use the LCD display from article part 4 in The Canadian along with 2 LDR's from our kit. Most of the kits we are using contain 2 or more LDR's but if you only have 1 you can obtain more from one of the many on-line electronic parts suppliers or at a local electronics retail store. Here is a link to a suitable sensor from the Adafruit site, https://www.adafruit.com/product/161.

Creating the Scale Train Speed Project.

For this project we are adding the LDR's to UNO and we are also using the LCD display to keep us informed of what is happening in the microcontroller. The information displayed will include the pre-start procedure where the LDR's will be calibrated for the ambient light conditions as well as letting us know what scale speed the train is travelling at. One additional component will be a reset button that will allow us to re-start the Uno and cause the LDR's to be re-calibrated. This can become necessary if the ambient light conditions change. On a home layout this occur if walkway or room lights were on along with the layout lights when the Uno was first powered up or if one of the LDR's where covered over by an idle train when calibration took place. On a portable layout this might be required if the lighting in the room changes because all of the lighting wasn't on during setup or as the day goes on and the sunsets reducing the natural light entering the room. As you can see by these examples that any number of conditions can occur which will affect the light that the LDR's were exposed to when they were originally calibrated.

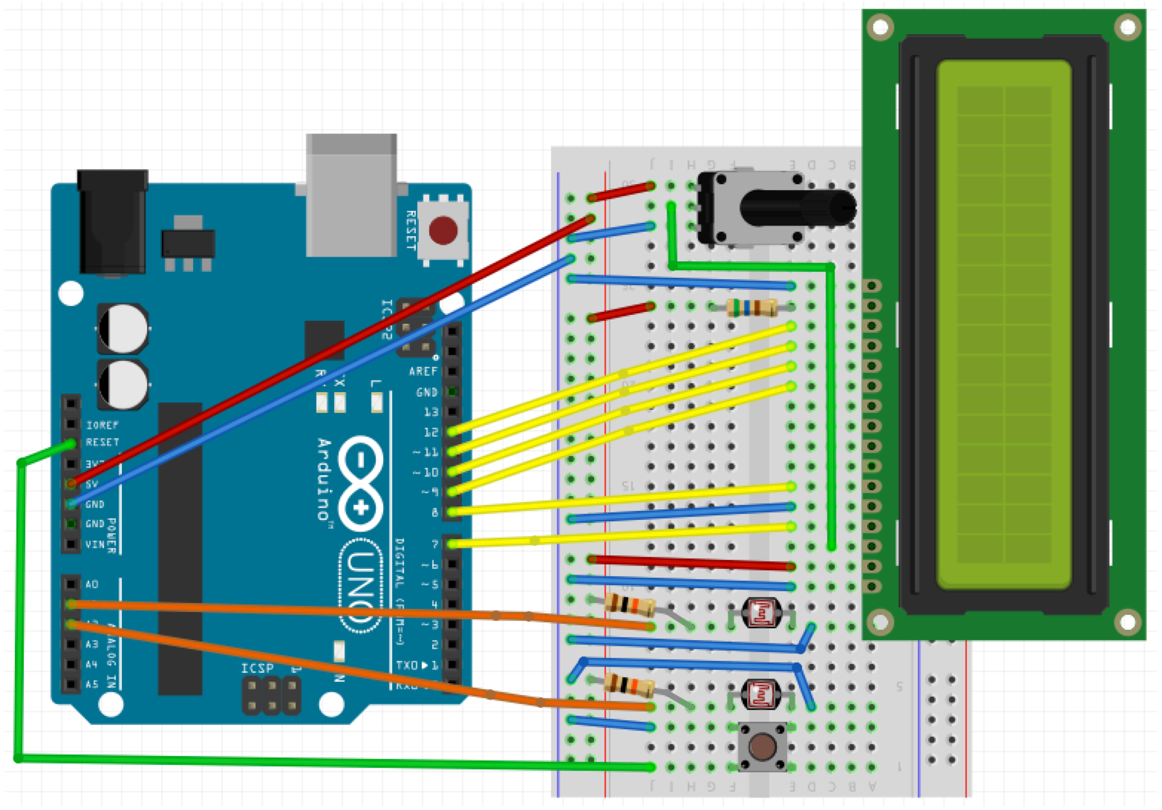
Wiring up the Components

Wiring up the LCD will be identical to the wiring used in the article that was included in The Canadian, issue #64. This time will re-arrange the items on the breadboard as we need to add 2 LDR's, 2 10k ohm resistors and move the location and wiring to the pushbutton. The breadboard is now getting crowded so take your time wiring up these items.



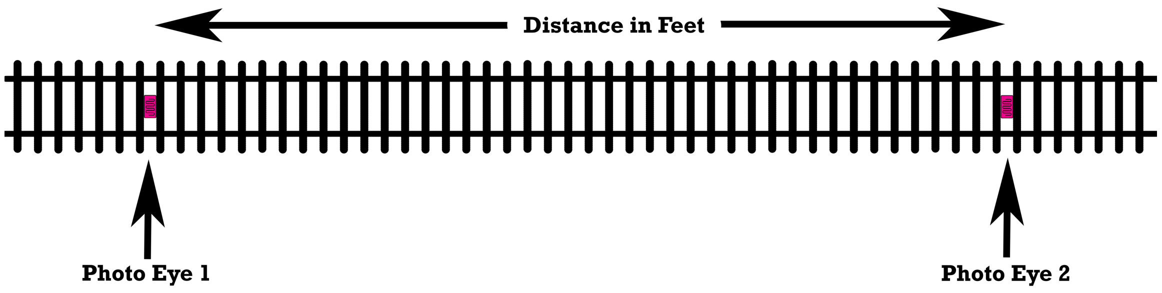
We used the LDR's in a previous project when we created sensors to use with a flashing grade crossing project in Part 2 of this series of articles. If you only have one LDR you can obtain more from any of the electronic supply companies online or in real brick and mortar stores. You will need 2 to complete this project.

Here is an image showing the complete breadboard wiring needed for this project.



Physical Layout of LDR's and Measurement

In the image below, we can see the physical layout that is typical for this project. The LDR's are placed in 2 convenient locations along a section of track. We will need to note the distance from one LDR to the other and measure this distance in feet. The reason for using feet is that this is a measure that most people will be able to manage and also that the display will be showing the speed in miles per hours. Both of these units are based on the Imperial/US measurement systems.



Since the distance measurement uses feet as the base unit you might be required to convert your measurement that may include inches and fractions of an inch to feet. The math to do this is not that hard but let me show you a couple of examples that you may find useful.

For this first example that us say that the distance from LDR to the other is 1 foot 5 inches. To calculate the distance in feet we will need to know how much of a foot is 5 inches. We will convert the 5 inches to a decimal value by dividing 5 by 12. Here is the math.

$$Whole\ feet + \frac{inches}{12} = 1 + \frac{5}{12} = 1 + 0.417 = 1.417\ feet$$

You can even calculate the distance in feet even if your measurement includes fractions of an inch. If we measure the distance as 2 feet, 1 inch and 5/16ths of an inch we can convert this by first calculating what 5/16 is in inches. Then we add the number of inches to this fractional value. Once that is done we use the formula above to calculate the distance in feet.

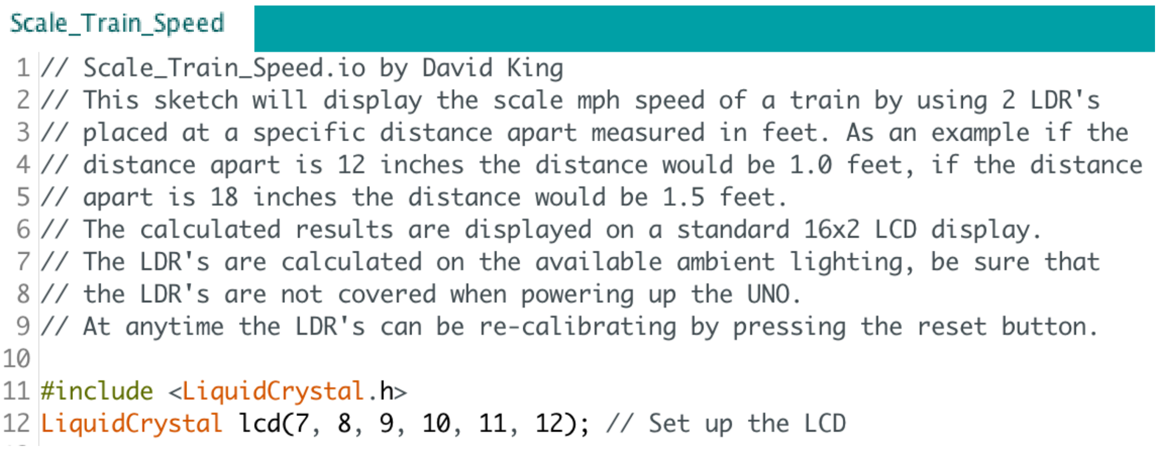
$$Whole\ inches + fraction = 1 + \frac{5}{16} = 1 + 0.3125 = 1.3125\ inches$$

$$Whole\ feet + \frac{inches}{12} = 2 + \frac{1.3125}{12} = 2 + 0.109 = 2.109\ feet$$

Creating the Sketch

This sketch is one of the longest sketches that we have created for this series of articles, but it has many variables and it works well once completed. You will find that is a project that you would like to add to your own layout.

I'll break the code into smaller blocks and I'll to explain what is happening in each section. So here we go.



```
 1 // Scale_Train_Speed.io by David King
 2 // This sketch will display the scale mph speed of a train by using 2 LDR's
 3 // placed at a specific distance apart measured in feet. As an example if the
 4 // distance apart is 12 inches the distance would be 1.0 feet, if the distance
 5 // apart is 18 inches the distance would be 1.5 feet.
 6 // The calculated results are displayed on a standard 16x2 LCD display.
 7 // The LDR's are calculated on the available ambient lighting, be sure that
 8 // the LDR's are not covered when powering up the UNO.
 9 // At anytime the LDR's can be re-calibrating by pressing the reset button.
10
11 #include <LiquidCrystal.h>
12 LiquidCrystal lcd(7, 8, 9, 10, 11, 12); // Set up the LCD
```

In the section of code shown above I start by including a description of what we are trying to achieve. On rung 11 I include the library file for operating the LCD along with the setup of the pins on rung 12.

```
13
14 const int calOn = 13;         // On-board LED for calibration active
15
16 const int photoOne = A1;      // Set up for 1st LDR
17 int photoOneValue;
18 int photoOneHigh = 0;
19 int photoOneSet;
20 boolean photoOneEnable = false;
21 boolean photoOneHome = false;
22
23 const int photoTwo = A2;      // Set up for 2nd LDR
24 int photoTwoValue;
25 int photoTwoHigh = 0;
26 int photoTwoSet;
27 boolean photoTwoEnable = false;
28 boolean photoTwoHome = false;
29
30 long int startTime = 0;       // Set up for timing fuctions
31 long int endTime;
32 boolean timerDone = false;
33
34 long int resetTimerStart = 0; // Set up for reset functions
35 long int resetTimerCurrent;
36 boolean resetTimerStarted = false;
37 boolean resetTimerDone = false;
38
39 float mph;                      // scale speed of the train in mph
40 float distance = 2.109;         // distance between sensors in feet
41 float seconds;                  // time to travel between sensors in seconds
42 float scale = 87;               // scale factor for N is 160, HO is 87, O is 48
43
```

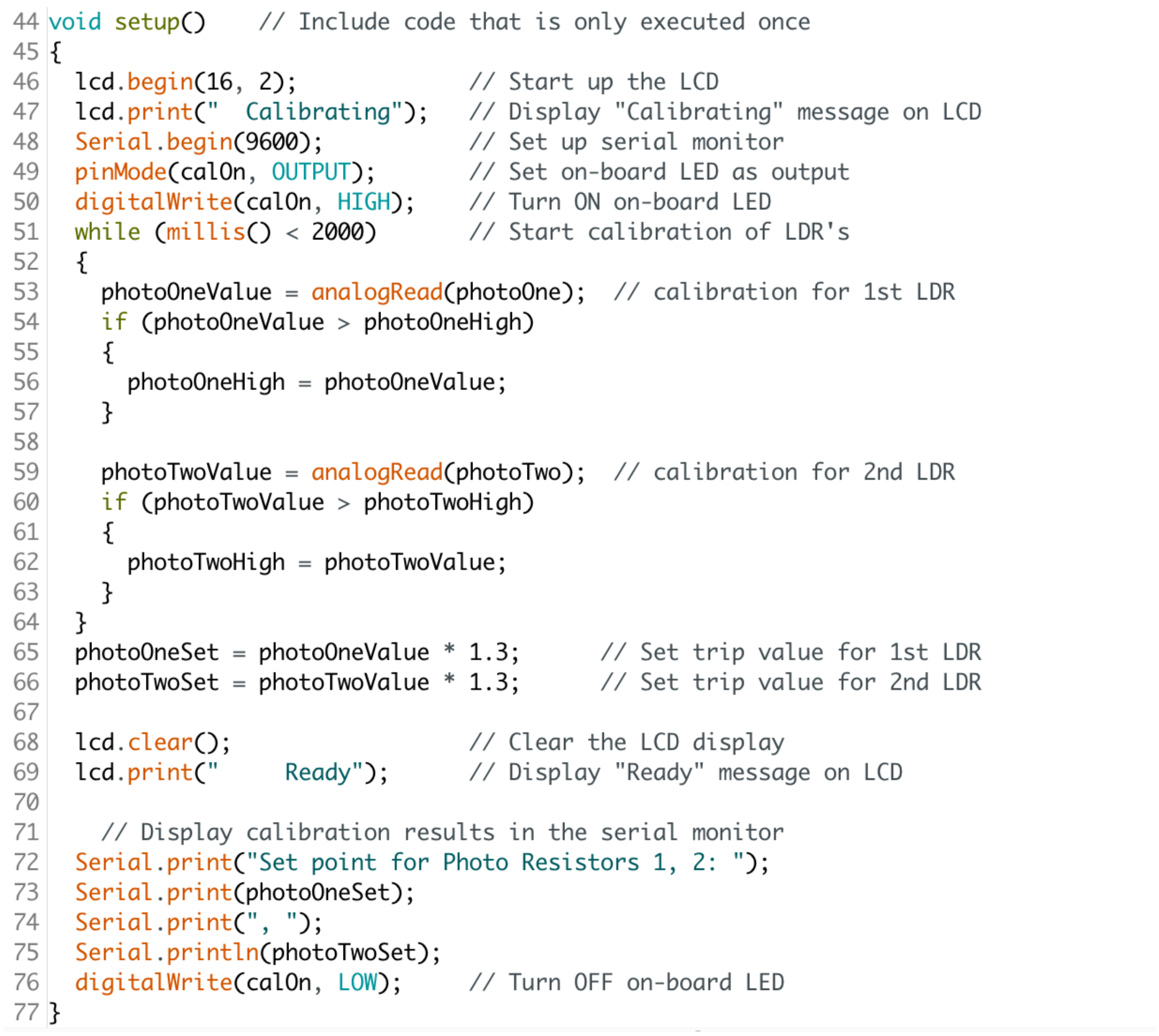
This second block of code is used to declare all of the variables that will be used in the sketch. Rung 14 is used to set pin 13 for use as a calibration in use LED.

Rungs 16 to 21 are used to set up the variables required for one of the LDR's that we are using to detect a train. Rung 16 set the pin to use for connection to the LDR. The remain rungs, 17 to 21 set up the remaining variables. Rungs 23 to 28 follow the same pattern as rungs 16 to 21 but are used with the second LDR.

Rungs 30 to 32 are setting up the variables used with calculating the elapsed time between when the LDR's are blocked.

Rungs 34 to 37 are used for many of the required resets needed so that we can calculate speeds many times without having to power down and then power up the Uno to calculate MPH for multiple trains.

In rungs 39 to 42 you will need to adjust a couple of these values as these are the variables needed to calculate the proper MPH for your layout based on the scale and distance between the LDR's. On rung 40 it is important that you enter the actual distance from one LDR to the other LDR. Use the math equation that I used earlier in this article to calculate the distance in feet. In rung 42 you will need to enter the scale of your train where this set up to be used. Remember that this is the scale, so HO would be 87, O would be 48, S would be 64 and so on for other scales. The gauge of your trains doesn't matter as an example both HO and HOn3 would both be 1/87 scale.



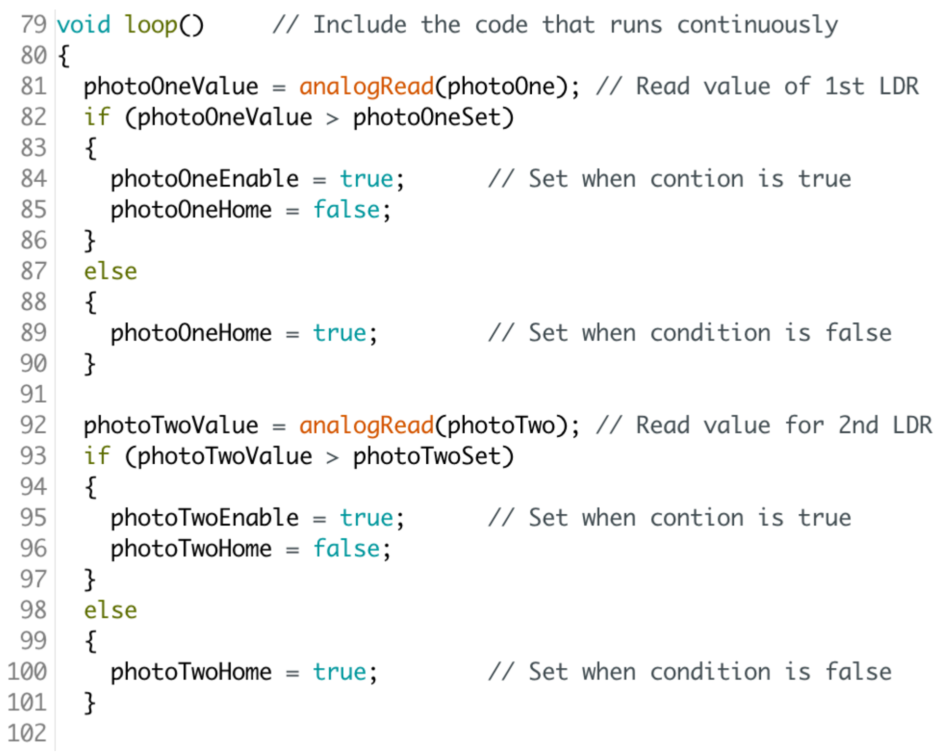
```
44 void setup()    // Include code that is only executed once
45 {
46   lcd.begin(16, 2);               // Start up the LCD
47   lcd.print("  Calibrating");     // Display "Calibrating" message on LCD
48   Serial.begin(9600);             // Set up serial monitor
49   pinMode(calOn, OUTPUT);         // Set on-board LED as output
50   digitalWrite(calOn, HIGH);      // Turn ON on-board LED
51   while (millis() < 2000)         // Start calibration of LDR's
52   {
53     photoOneValue = analogRead(photoOne);  // calibration for 1st LDR
54     if (photoOneValue > photoOneHigh)
55     {
56       photoOneHigh = photoOneValue;
57     }
58 
59     photoTwoValue = analogRead(photoTwo);  // calibration for 2nd LDR
60     if (photoTwoValue > photoTwoHigh)
61     {
62       photoTwoHigh = photoTwoValue;
63     }
64   }
65   photoOneSet = photoOneValue * 1.3;      // Set trip value for 1st LDR
66   photoTwoSet = photoTwoValue * 1.3;      // Set trip value for 2nd LDR
67 
68   lcd.clear();                    // Clear the LCD display
69   lcd.print("     Ready");        // Display "Ready" message on LCD
70 
71     // Display calibration results in the serial monitor
72   Serial.print("Set point for Photo Resistors 1, 2: ");
73   Serial.print(photoOneSet);
74   Serial.print(", ");
75   Serial.println(photoTwoSet);
76   digitalWrite(calOn, LOW);       // Turn OFF on-board LED
77 }
```

The void setup() holds the code that is only required to run once. On rung 46 the LCD instructions are started and followed on rung 47 with the instruction to display a message on the LCD letting us know that calibration of the LDR's is taking place. Next on rung 48 the serial monitor is started so that we can monitor any details using our computer. Rung 49 and 50 are used to set pin 13, for the on board LED, as an output pin and then we turn on the LED.

Starting on rung 51 we start a 2 second loop that reads the base value of each LDR (rungs 53 and 59) and saves the highest value in the variables located on rungs 56 and 62. Once the 2 second loop has completed the highest values saved are then multiplied by 1.3, a 30% increase in the value, and the results are stored in the variables located on rungs 65 and 66. This multiplying factor of 1.3 can be alter if you find the sensing is either too sensitive or not sensitive enough. The values photoOneValue and photoTwoValue can range from a low of 0 to a high of 1023. These numbers represent the possible range that can be read on pins A1 and A2. If these numbers are low, then that means that there was lots of ambient light and if these numbers are high there was very dim ambient lighting.

Rungs 68 and 69 are used to clear the LCD and then display a new message of Ready to let you know the calibration has finished its calibration. Rungs 72 to 75 are used to display the results on the serial monitor.

A final step on rung 76 is to turn off the on board LED to let you know that the calibration process and the void setup() have been completed.

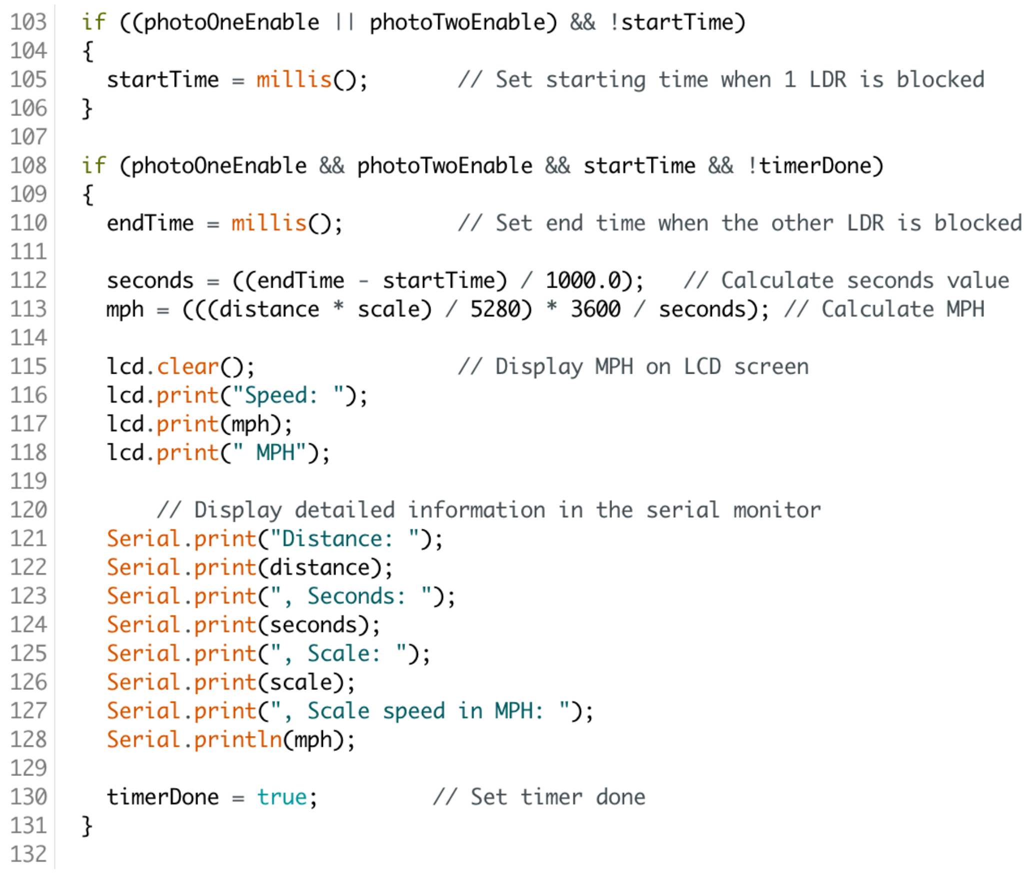


```
 79 void loop()     // Include the code that runs continuously
 80 {
 81   photoOneValue = analogRead(photoOne); // Read value of 1st LDR
 82   if (photoOneValue > photoOneSet)
 83   {
 84     photoOneEnable = true;      // Set when contion is true
 85     photoOneHome = false;
 86   }
 87   else
 88   {
 89     photoOneHome = true;        // Set when condition is false
 90   }
 91 
 92   photoTwoValue = analogRead(photoTwo); // Read value for 2nd LDR
 93   if (photoTwoValue > photoTwoSet)
 94   {
 95     photoTwoEnable = true;      // Set when contion is true
 96     photoTwoHome = false;
 97   }
 98   else
 99   {
100     photoTwoHome = true;        // Set when condition is false
101   }
102 
```

Starting on rung 79 and continuing to rung 158 is the void loop() which is the main portion of the sketch and this section runs continuously until either the reset button is pressed or the power is removed from the Uno.

On rung 81 the 1st LDR value is read. On rung 82 this value is compared to the high calculated value that for this 1st LDR that was determined during the calibration. If the value is greater, enough light has blocked, the LDR then rungs 84 and 85 are executed. If the value is not greater than the code on rung 89 is executed. The code on rung 84 set photoOneEnable to true, meaning that the LDR has been blocked from the ambient light, to say that a train has been detected. On rung 85 the value photoOneHome is set to false if a train is blocking the LDR. Rung 89 sets the photoOneHome to true if no train is blocking the LDR.

Rungs 92 to 101 repeat all of the steps that were included in rungs 81 to 90 but this time these steps are in reference to the 2nd LDR.



```
103  if ((photoOneEnable || photoTwoEnable) && !startTime)
104  {
105    startTime = millis();       // Set starting time when 1 LDR is blocked
106  }
107
108  if (photoOneEnable && photoTwoEnable && startTime && !timerDone)
109  {
110    endTime = millis();         // Set end time when the other LDR is blocked
111
112    seconds = ((endTime - startTime) / 1000.0);   // Calculate seconds value
113    mph = (((distance * scale) / 5280) * 3600 / seconds); // Calculate MPH
114
115    lcd.clear();                // Display MPH on LCD screen
116    lcd.print("Speed: ");
117    lcd.print(mph);
118    lcd.print(" MPH");
119
120        // Display detailed information in the serial monitor
121    Serial.print("Distance: ");
122    Serial.print(distance);
123    Serial.print(", Seconds: ");
124    Serial.print(seconds);
125    Serial.print(", Scale: ");
126    Serial.print(scale);
127    Serial.print(", Scale speed in MPH: ");
128    Serial.println(mph);
129
130    timerDone = true;          // Set timer done
131  }
132
```

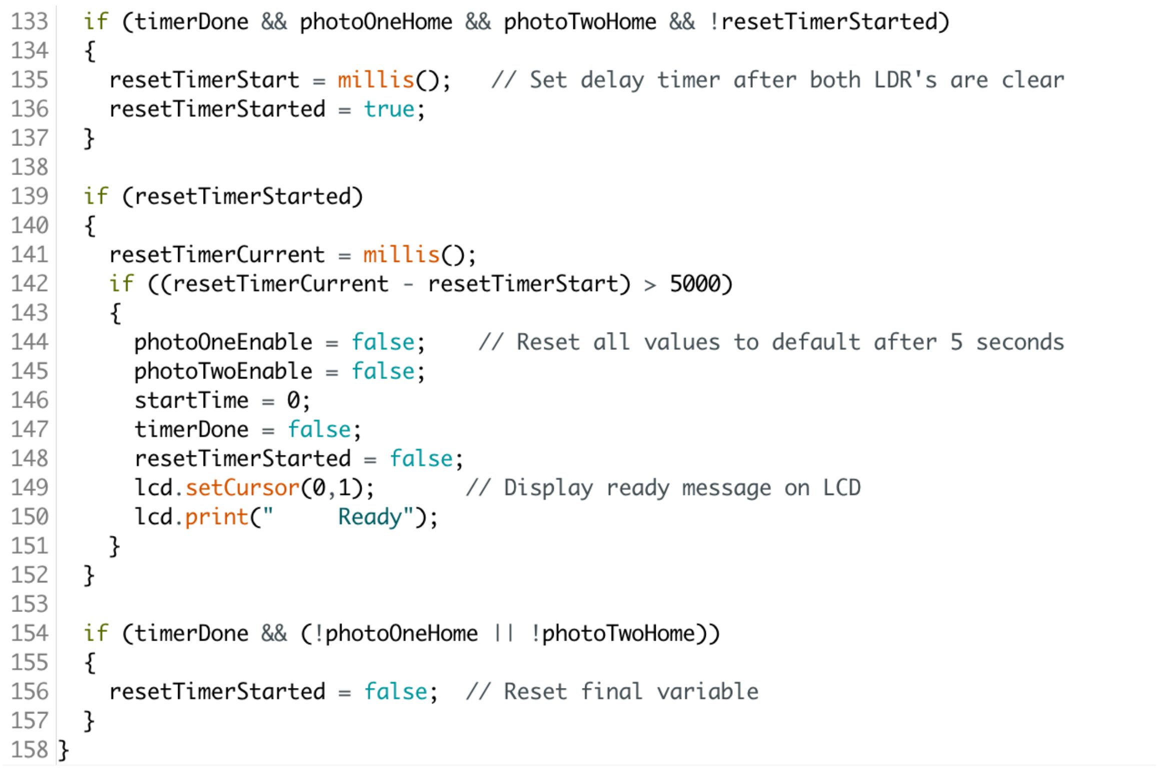
On rung 103 if either LDR has been triggered and the timer has been started we record the starting time. On rung 108 we check for the other LDR to be trigger and that this hasn't happened earlier in this cycle. On rung 110 record the time that this most recent has occurred. Rung 112 is used to calculate the elapsed time between when the two LDR were triggered. This time is measured in milliseconds so divide this result by 1000 to change the units to seconds.

Rung 113 is the heart of the calculations as this is where we will calculate the scale miles per hour, mph. We start by taking the distance, from rung 40, and multiplying it by the scale, from rung 42. Next, we divide this number by 5280 as this is the number of feet in a mile. You may have noticed that the distance from rung 40 was also in feet. This result is multiplied by 3600 which is the number of seconds in 1 minute. Finally, we can calculate the scale mph by dividing by the time in seconds that we calculated on rung 112.

Starting on rung 115 we clear the text on the LCD and place the curser in the upper left position on the display. Rungs 116 to 118 displays the mph on the upper line of the display with something similar to **Speed: 64.09 MPH**.

Rungs 121 to 128 are used to send information to the Serial Monitor on your computer. This information includes the distance, time, scale and mph. This will only be displayed if you are connected to the Uno with the USB cable and the Serial Monitor is open.

Once the above is done we use the code on rung 130 to set the timerDone as true. This prevents the calculation from being done again until we reset all of the steps required.



```
133    if (timerDone && photoOneHome && photoTwoHome && !resetTimerStarted)
134    {
135      resetTimerStart = millis();   // Set delay timer after both LDR's are clear
136      resetTimerStarted = true;
137    }
138
139    if (resetTimerStarted)
140    {
141      resetTimerCurrent = millis();
142      if ((resetTimerCurrent - resetTimerStart) > 5000)
143      {
144        photoOneEnable = false;    // Reset all values to default after 5 seconds
145        photoTwoEnable = false;
146        startTime = 0;
147        timerDone = false;
148        resetTimerStarted = false;
149        lcd.setCursor(0,1);       // Display ready message on LCD
150        lcd.print("     Ready");
151      }
152    }
153
154    if (timerDone && (!photoOneHome || !photoTwoHome))
155    {
156      resetTimerStarted = false;  // Reset final variable
157    }
158  }
```

On rung 133 we check that the timerDone is true and that both LDR's are not being covered by a train. Also, we check that the reset timer has not been started. Rung 135 gives us a start time for the reset timer if all of the above conditions have been met. Rung 136 set the reset timer in the true state.

On rung 139 we check for the true state of the reset timer. If the condition is true, then we can set the starting time for the reset timer using rung 141. Rung 142 checks to see if 5 or more seconds have elapsed since we set the starting time of the reset time. If the required amount of time has passed, we can use the code on rungs 144 to 148 are used to reset most of the variables and states that have been set in capturing and calculating the scale mph. Rungs 149 to 150 displays some text on the lower portion of the LCD with the message **Ready** to inform the operator that the scale mph cycle has been completed.

Finally, on rung 154 we check that the 5 second timer has been completed and that both LDR's are clear of any trains. This allows the code on rung 156 to reset the last variable which also the speed check to start over again for the next passing train.

Conclusion

Believe it or not if you have entered all of the code properly in your sketch and you wired up everything correctly, you should now have working scale mph speed trap for your layout or module. Enjoy and take your time entering the code that many items in the code are very particular that they be entered using all of the proper syntax that is called for in the Arduino programming language.

I'm not sure what topic I will cover in the next instalment of this series and you could help me out with your ideas. Drop me a note, email, addressed to directordavid@caorm.org or membership@caorm.org as I would like to hear from you.

This is a project that you can start at any time even if you have not viewed the previous articles from this series. The first part of this article is located in issue #63 of *The* Canadian. The previous articles are available in the preceding issues (#59, #60, #61 and #62) of *The Canadian*, so enjoy!