Project 2 - Speedometer

July, 2020
David Ackmann – Gateway Division NMRA

You know you always wanted one. From the time you placed your first locomotive on the loop of your Christmas "Plywood Pacific" and cranked the knob of the throttle all the way to the right, you wanted one. You wanted a train speedometer. I know, because I wanted one too.

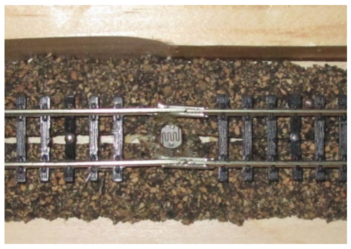
Recently, sixty years later, I still wanted one, but what I want now is a bit different than what I have seen described in the past. I could indeed build a point-to-point speedometer, a "speedometer-on-a-stick", to be taken out and used only when I could remember where I stored it, but that wouldn't fit my needs. I wanted a speedometer permanently embedded in a loop of my layout, always at hand, always operational. Something simple, that would work well with momentum enabled throttles or locomotives. I made it happen, and so can you.

I built my speedometer using an Arduino Nano microprocessor (AMAZON ASIN: B07G99NNXL, 3 for $13.99), a Nano Expansion Board (AMAZON ASIN: B07MGVC18K, 5 for $11.99), a 4 line display (AMAZON ASIN: B01GPUMP9C, $12.99), a single light dependent resistor (aka: LDR, photocell, optical sensor; AMAZON ASIN: B01N7V536K; 30 for $4.15), a power supply (AMAZON ASIN: B07VBR327W, 2 for $7.99) , a 10K resistor, some jumpers, and a 4 position barrier strip, permanently embedded in my layout, all for about $30.

To get started, I selected a spot where I wanted to mount the speedometer display and the Arduino microcomputer. Anywhere on the layout fascia, or on the top of the layout, something about 4 by 8 inches, would work fine. I decided where in the loop of track I wanted to start the timer, and there is where I needed to place the LDR.

I took the light dependent resistor and soldered a wire to each of the leads; 24 gauge or smaller wire is fine, but I made sure the wires were long enough to reach from the LDR to the place I wanted the Arduino and display, plus about 12" more. Because the two leads are uninsulated and close together, I decided some heat-shrink tubing or "liquid electrical tape" should be used to insulate the leads.

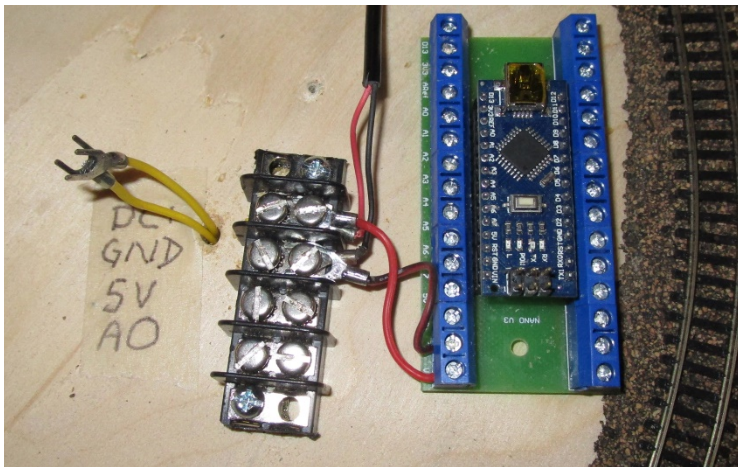
I drilled a 1/8" hole through the layout where I decided to place the LDR (3/16" might work a bit better). Depending on the scale of your layout, you may need to remove a tie. I decided that the top of the LDR should be even with the top of the rail ties. I made sure the LDR was facing straight up; it tended to wiggle out of place, but a bit of glue on the



underside of the LDR helped hold it in place.

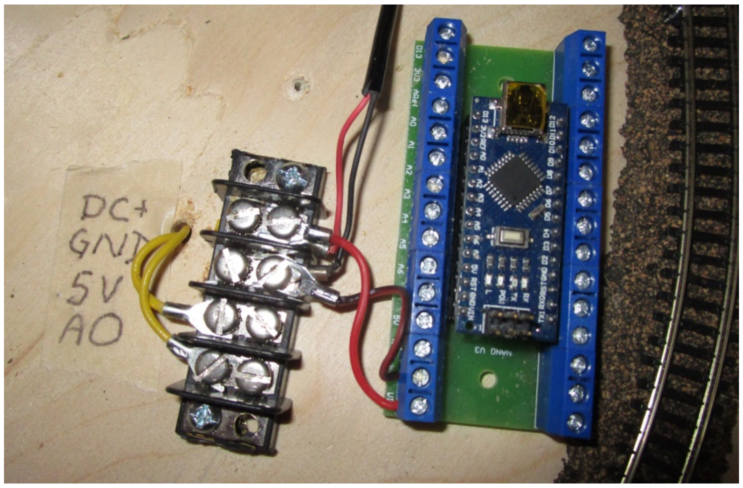
I then ran the LDR wires to where I wanted the display to be, drilled a 1/4" hole through either the fascia (or layout top if that works better for you), and ran the wires through the hole. I like to solder "spade connectors to the end of the wires, but if that is not your "thing", make sure you at least tin the wire ends.

Near the hole, I mounted a 4 position "barrier strip" (see Chapter 5 of the "Amazing Arduino Animations" website at http://daackm.github.io). I took a piece of masking tape and labeled the 4 terminals as "DC+", "GND", "A0" and "5V".

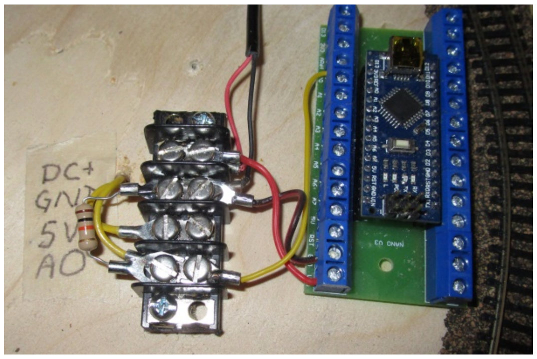


I needed to get power to the Arduino, so I clipped off the connector from a 12VDC "wall wart". My cord had outside insulation, so I removed about 3" to expose the ground and powered wires. I stripped off about 3/8" of insulation from the end of each wire and soldered a spade connector to the ground wire of the power supply, finally attaching it to the GND position of the barrier strip. Similarly, I soldered a spade connector to the positive wire of the barrier strip and attach it to the DC+ position. I ran a black wire from the GND position of the barrier strip to the GND socket of the expansion board, and then ran a red wire from the DC+ position of the barrier strip to the VIN position of the expansion board.

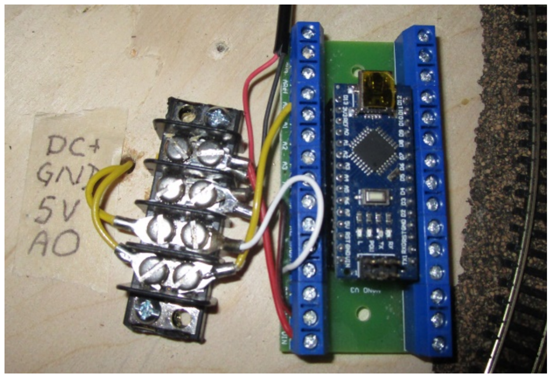
I then attached the wires of the LDR to the A0 and 5V terminals of the barrier strip.



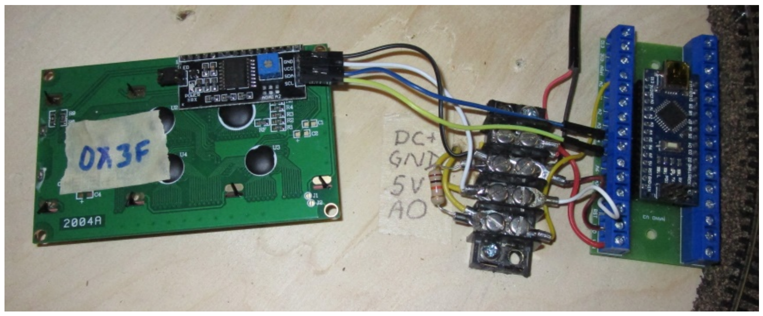
Next I wired in the 10K ohm resistor. I soldered spade connectors to each end, and attached one connector to the A0 terminal of the barrier strip and the other connector to the GND position of the barrier strip.



Next, I ran a short, yellow wire from the A0 pin of the barrier strip to the A0 pin of the expansion board. I also ran a white wire from the 5V terminal of the barrier strip to the 5V pin of the expansion board.



Next I wired the display. Because the display cannot mount flat on the fascia board (there is a "I2C piggyback" board attached to the back which gets in the way), I used plastic "standoffs" and 1" #4-40 sheet metal screws to fasten the display to the layout; at this point you may want to only attach the display with two screws, because you will probably want to remove it once all components are in place to adjust its brightness; that's what the blue potentiometer on the back of the piggyback board is for.



From the GND position on the barrier strip, I ran a black jumper with a spade lug on one end and a female socket to the GND pin of the display. Similarly, I ran a white female jumper wire from 5V on the barrier strip to the VCC pin on the display. I ran a male-female jumper from the Nano's A4 socket to the display's SDA pin, and then another male-female jumper from the Nano's A5 socket to the display's SCL pin.

I often make a spreadsheet of all connections, along with the colored wired I used.

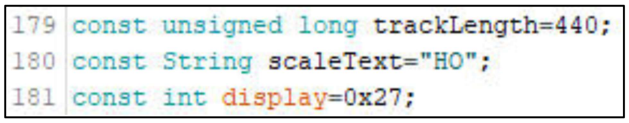
The overall electrical schematic is shown below.

```
                      Barrier               Arduino
                        Strip                 Nano                   Display

                      +-------+             +-------+               +-------+
                      |       |             |       |               |       |
Power                 |       |             |       |               |       |
Supply ---------------0  VIN  O-------------O  VIN  |         +-----O  VCC  |
  +                   |       |             |       |         |     |       |
                      |       |  /----------|       |----+    |     |       |
                      |       | /           |       |    |    /     |       |
Power                 |       |/            |       |    |   (      |       |
Supply----------------0  GND  O-------------O  GND  O    +--(-------O  GND  |
  -                   |       |\---+        |       |        (      |       |
                      |       |    10K      |       |    |+---\     |       |
                      |       |/---+        |       |    |          |       |
     /----------------O  A0   O-------------O  A0   O    |  +-------O  SDA  |
    /                 |       |             |       |    |  |       |       |
 LDR                  |       |             |       |    |  |       |       |
    \                 |       |             |       |    |  |       |       |
     \----------------O  5V   O-------------O  5V   O----+  |   +---O  SCL  |
                      |       |             |       |       |   |   |       |
                      |       |             |       |       |   |   |       |
                      +-------+             |  A4   O-------+   |   +-------+
                                            |       |           |
                                            |       |           |
                                            |  A5   O-----------+
```

Now, before I installed the Arduino into its expansion board, I needed to download a program, or “sketch” as they are called in Arduino-speak, into the Arduino to make the speedometer work. Using the free Arduino Interactive Development Environment that I downloaded to my Personal Computer (see “Part 4” of this tutorial series), I wrote a sketch that did three main things: 1) when the locomotive passed over the light dependent resistor, it started a timer; 2) when the locomotive again passed over the LDR, it would check the timer, display the time it took to complete the loop, and then calculate the average speed and display it as well; 3) finally, and immediately, the sketch would restart the timer and do it all again.

But, to make this work for your layout, you will need to modify the sketch by entering the <u>length of your track loop</u> and the <u>scale of your layout</u>, which will let the Arduino do the proper calculations for your loop; these values can be found around line 178 of the sketch.



```
179 const unsigned long trackLength=440;
180 const String scaleText="HO";
181 const int display=0x27;
```

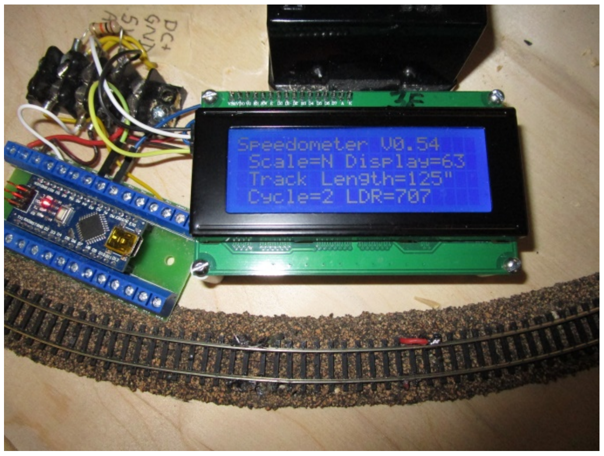
Each Arduino display has an address, and you ***MAY*** need to alter the address of the display around line 181, as described in Chapter 11 of the https://daackm.github.io website. Most displays have an address of “0x27”, but some are “0x3F”, so if yours does not display properly, either change the address and try again, or use the SCANNER program as described in Chapter 11 to verify the proper address setting .

Yes, the Speedometer sketch is quite lengthy, but you don’t need to worry about the details. All you have to do is:

1) Visit https://daackm.github.io
2) Scroll down to Project 2 - Speedometer
3) Click on the Speedometer.pdf file and print the documentation
4) Return to the main menu and download the Speedometer Arduino Code
5) Highlight and copy the code to the clipboard (Control/C),
6) Open the Interactive Development Environment (IDE)
7) Click on the FILE tab
8) Click on the “New” entry
9) Highlight all the default contents
10) Hit the “delete: key
11) Paste the code into a sketch of your own (Control/V)
12) Replace my values for scale and loop length with your own (about line 178).
13) Change the display address, if necessary
14) Save the sketch to your hard drive
15) Download the sketch to the Arduino

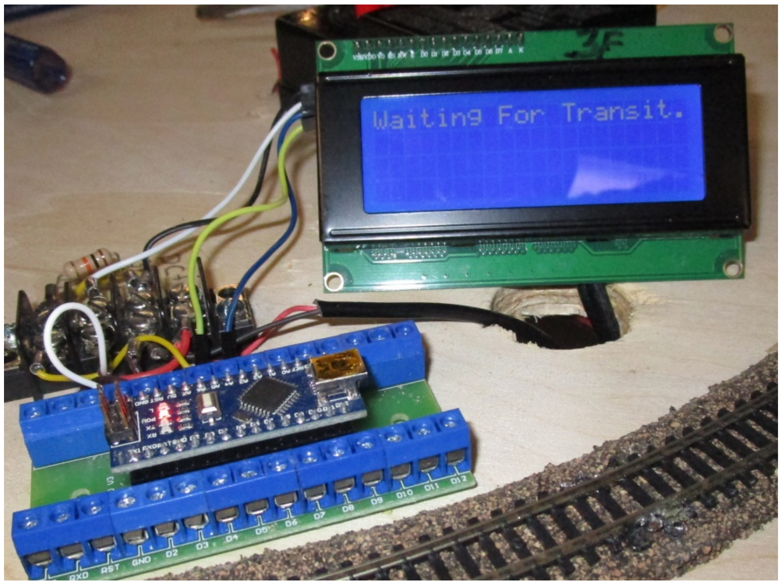
After downloading the sketch and disconnecting the Arduino from my PC, I inserted the Nano into its expansion board, making sure the proper end of the Nano went in to the proper socket holes (hint: the USB connector goes near the D12 socket).

When power is turned on, the display should show the software version number, and the scale and the length of the loop that is provided in the IDE; if it does not, you may need to turn the small blue potentiometer on the back of the display clockwise a bit. If the display shows a bunch of bright squares, you need to turn it counter-clockwise (if nothing lights, I hope you did check the display’s address with the SCANNER program, didn’t you?).

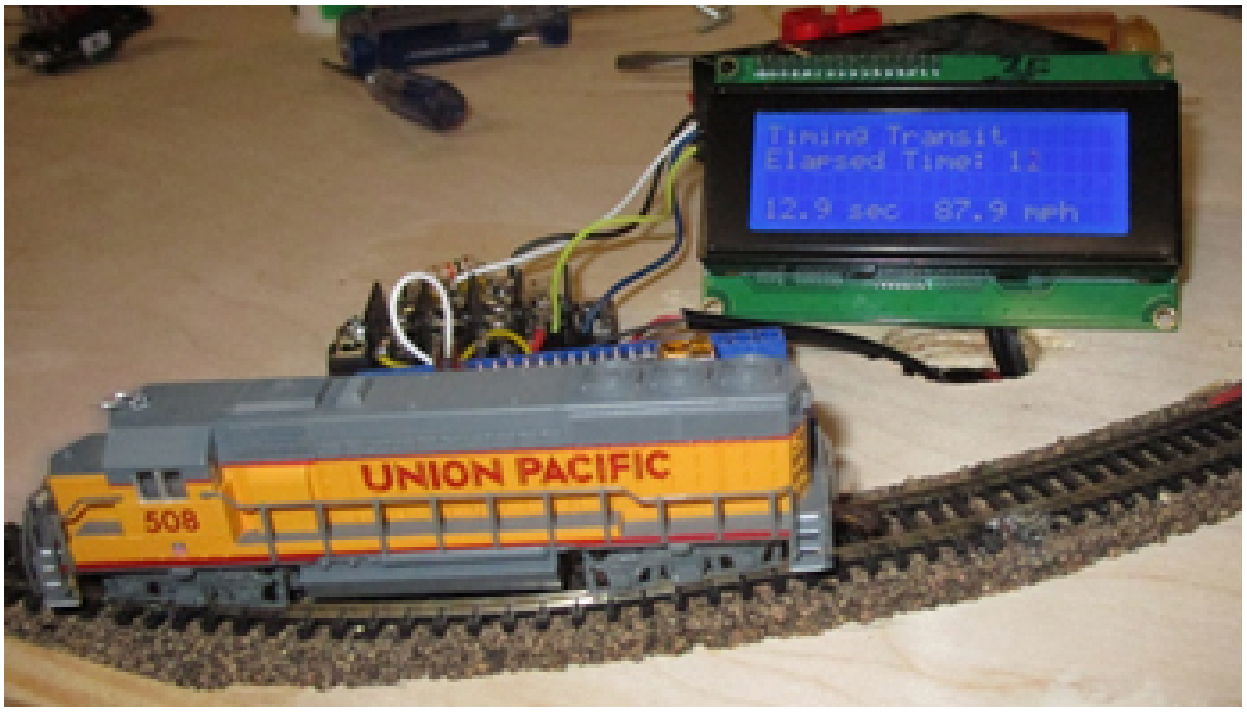


```
Speedometer V0.54
 Scale=N Display=63
 Track Length=125"
 Cycle=2 LDR=707
```

After the initial startup screen, the display should be showing a status of “Waiting For Transit.”. When the locomotive passes over the light dependent resistor, the display status changes to “Now Timing Transit” and also shows the elapsed time since the transit started.

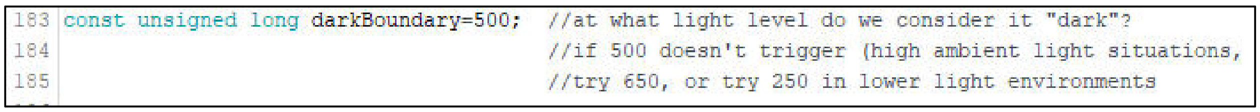


When the engine again passes over the LDR, the display shows the total transit time and the average scale speed, and then restarts the timer. To see it in action, visit http://daackm.github.io/Trailers.html.



In order not to trigger the timer when light hits the sensor between the locomotive and any cars that it may be pulling, a “minimum interval” was written into the sketch, so that there must be at least 2 seconds of continuous light before the transit would be declared to be complete, for smaller scales and shorter loops this value may need to be decreased. There is also a variable which can be set to accommodate layouts with more or less ambient light than I have on my layout.

If your ambient light is significantly different from mine, you may need to change the “darkBoundary” value near line183.



```
183 const unsigned long darkBoundary=500;   //at what light level do we consider it "dark"?
184                                         //if 500 doesn't trigger (high ambient light situations,
185                                         //try 650, or try 250 in lower light environments
```

I hope you enjoy this loop speedometer as much as I do. Thanks for giving it a try.