Dual 3 Position Semaphore Controller

Revision v1.1 11/20/23

Seth Neumann, seth@modelrailroadcontrolsystems.com

Introduction

This document describes the Dual 3 Position Semaphore Controller board and how to assemble and install it.

Formatted: Left, Right: 0.63"

Revision History

V0.1 – first pass – August 1, 2021
V0.2 – Jon Schmidt Comments, Seth Neumann, cleanup
V0.3 – Jon Schmidt Update to sketch 2.4
V1.1 – V 1.1 board, Jon Schmidt sketch 3.2 11/20/23

Table of Contents

Table of Figures

Table of Tables

Formatted: Default Paragraph Font

1 INTRODUCTION

This board is intended to control Train Order Boards and 2 arm Semaphores. It has 2 sets of:

Features:

- Supports one or two SG90 servos to control two semaphore "arms"
- All connections including +5 and ground are brought to an 8 position 0.100 connector
- A 2.1mm barrel jack is provided if you choose to supply power from a 5V wall wart.
- Each aspect (STOP (form 31), APPROACH (form 19), CLEAR has its own adjustment pot
- Each input line as an on-board tactile switch button. The tactile switch is used in testing and adjustment.
- All components are through-hole technology for ease of assembly and repair.
- Each servo has an associated LED which is used in programming

6 lines on the Arduino are not used: D0, D1, (usually TX/RX), A4,A5 (usually SDA/SCL) D8, and D13 (has the onboard LED, could be used for user feedback of some sort).

The servos are positioned so, if equipped with the short horns, they face in and can both be mounted directly below the post of a 2-arm semaphore or train order board. If placed facing out, the horns can control a pair single arm semaphores protecting an ABS block boundary. The servo may also be mounted off-board and all of the usual Radio Control (R/C) adapters and "jewelry" may be used as required for remote mounting or dealing with benchwork obstacles.

This board is based on a design by Seth Neumann, Earl Girbovan and John Plocher for a series of "Introduction to Arduino Programming" clinics that were offered at various NMRA PCR meets from 2017-2019 as well as the NMRA Conventions at Kansas City in 2018 and Salt Lake City in 2019. The code was developed by Jon Schmidt.

Schematic, circuit board layouts, CAD files and code are available on the product page on our website and on my GitHub pagehttps://github.com/SethNeumann/MRCS-Dual-3-Position-Semaphore-Controller

2 IDENTIFICATION AND INFORMATION

2.1. BOARD LAYOUT

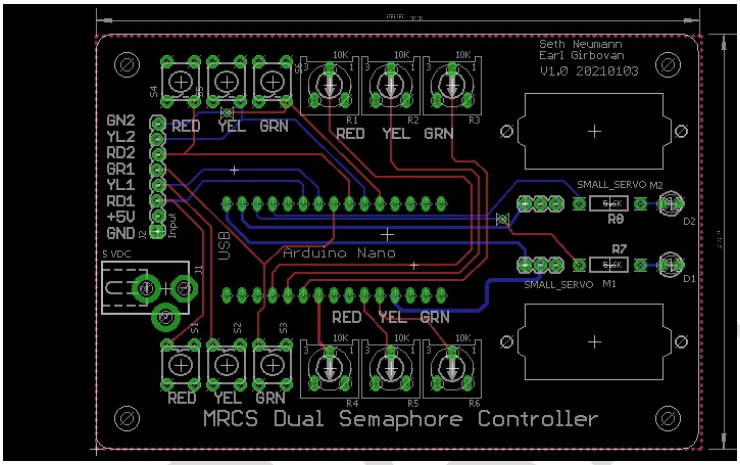


Figure 1 - Version 1.0 Board Layout

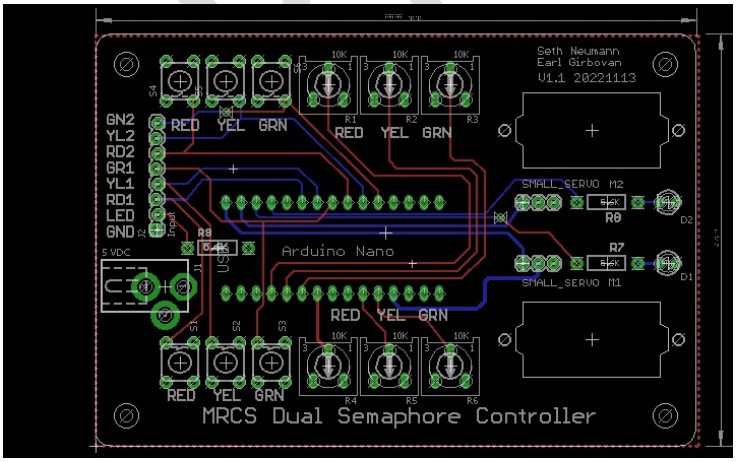


Figure 2 - Version 1.1 Board Layout - note R3

Formatted: Caption, Indent: Left: 0"

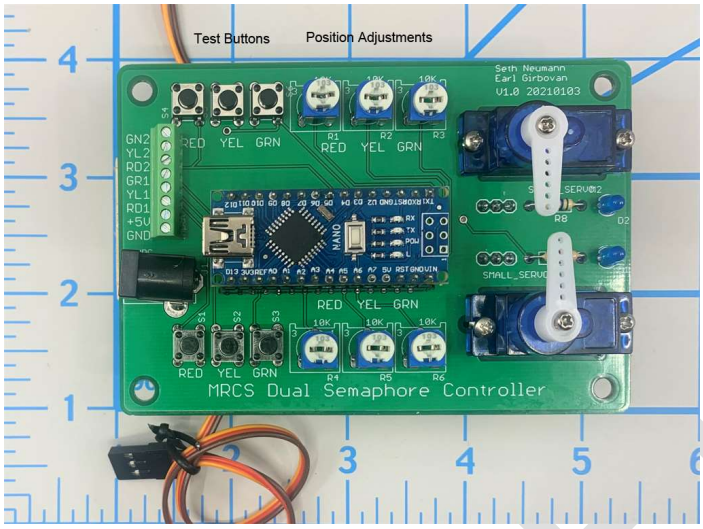


Figure 3 – Dual 3 Position Semaphore Board Top View

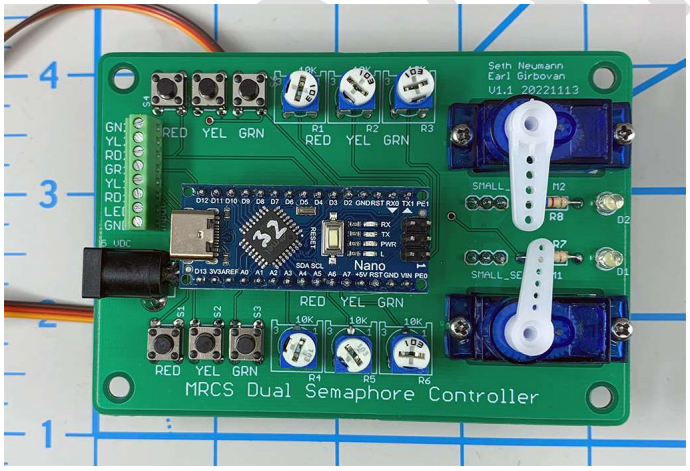


Figure 4 - Dual 3 Position Semaphore board V 1.1

You can't see R3 in this photo (it's under the USB connector) but the +5 screw terminal input has been changed to a current limited output to support a LED for the spectacle. This was in response to customer requests.

Formatted: Indent: Left: 0"

Formatted: Normal, Indent: Left: 0", Don't keep with next

This board is sold as a single unit, assembled, and tested or as bare board. If you are interested in alternate connectors, large quantities, or custom modifications please contact us.

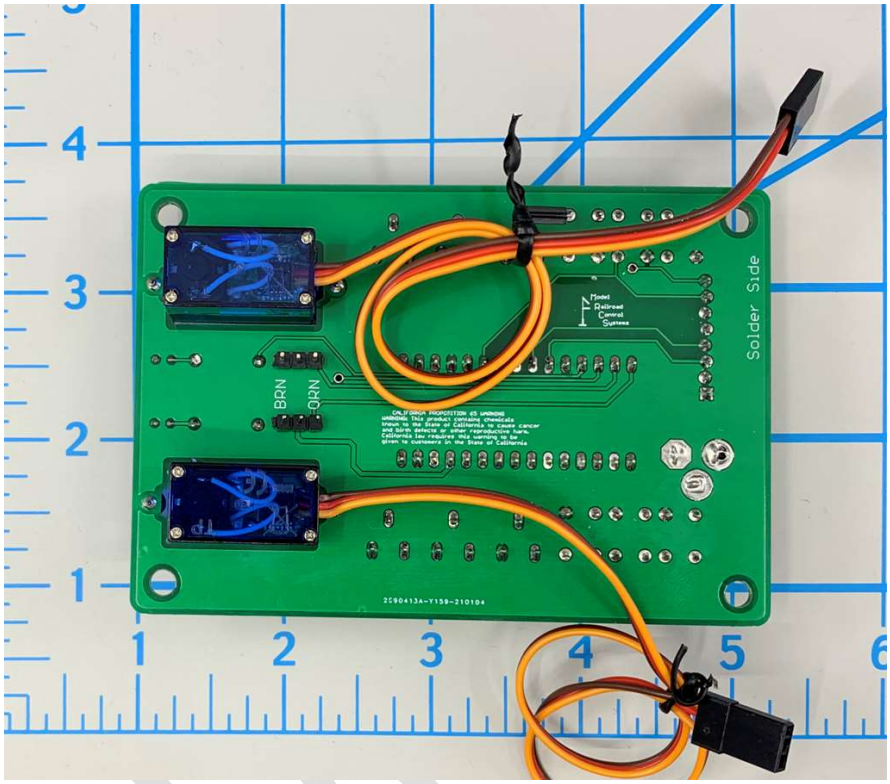


Figure 5 - Dual 3 Position Semaphore Board Bottom View

Note servo connectors mounted on back (bottom) of board

2.2. BILL OF MATERIALS

Qty	Value	Device	Package	Parts	Description
6	10K	TRIM_US-RS3	RS3	R1, R2, R3, R4, R5, R6	POTENTIOMETER
1	5 VDC	CONNECTOR-DC-POWER-RA	DCJ0202	J1	DC POWER JACK
2	5.6K	RESISTORPTH-1/4W	AXIAL-0.4	R7, R8	Resistor
1	Input	CONNECTOR-M08LOCK	1X08_LOCK	J2	Header 8
1	NANO	NANO	NANO	NANO	Arduino Nano
2	female header 15 pos	for Nano	female header 15 pos		
2	SMALL_SERVO header	SMALL_SERVO	1X03	M1, M2	Small servo motor
2	SG90 Servo				
6	SWITCH-MOMENTARY-2PTH	SWITCH-MOMENTARY-2PTH	TACTILE-PTH	S1, S2, S3, S4, S5, S6	NO pushbutton switch
2	Blue	LED3MM	LED3MM	D1, D2	
1	dual 3 position semaphore	circuit board	69x100		circuit board with Nano, 6 pots, 6 buttons, 2 servo
4	screw #2 x 1/4 sheet metal				

Table 1- Bill of Materials Ver 1.0

Qty	Value	Device	Package	Parts	Description

6	10K	TRIM_US-RS3	RS3	R1, R2, R3, R4, R5, R6	POTENTIOMETER
1	5 VDC	CONNECTOR-DC-POWER-RA	DCJ0202	J1	DC POWER JACK
3	5.6K	RESISTORPTH-1/4W	AXIAL-0.4	R7, R8, R9	Resistor
1	Input	CONNECTOR-M08LOCK	1X08_LOCK	J2	Header 8
1	NANO	NANO	NANO	NANO	Arduino Nano
2	SMALL_SERVO	SMALL_SERVO	1X03	M1, M2	Small servo motor
6	SWITCH-MOMENTARY-2PTH	SWITCH-MOMENTARY-2PTH	TACTILE-PTH	S1, S2, S3, S4, S5, S6	Various NO switches- pushbuttons, reed, etc
2	White	LEDLED3MM	LED3MM	D1, D2	

Formatted Table

Formatted Table

Formatted Table

Formatted Table

Table 2 – Bill of Materials Version 1.1

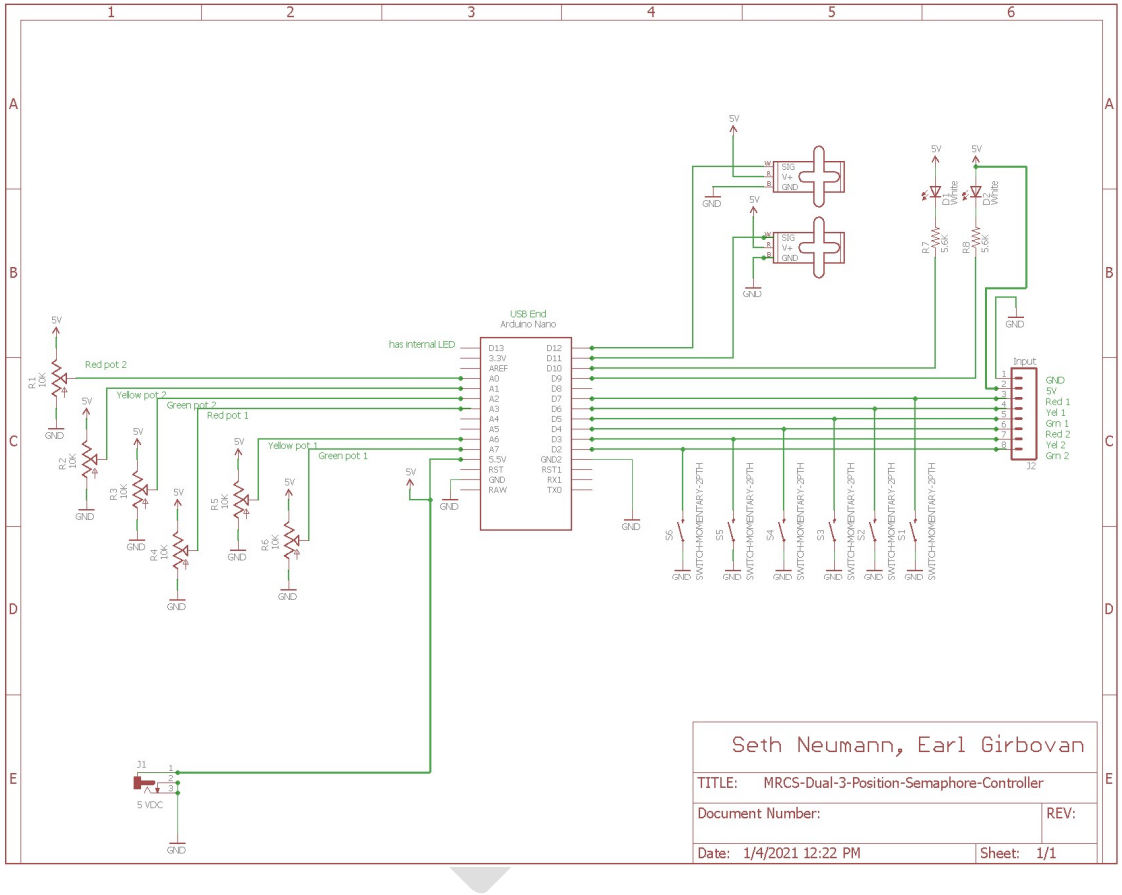


Figure 6 -Version 1.0 Schematic

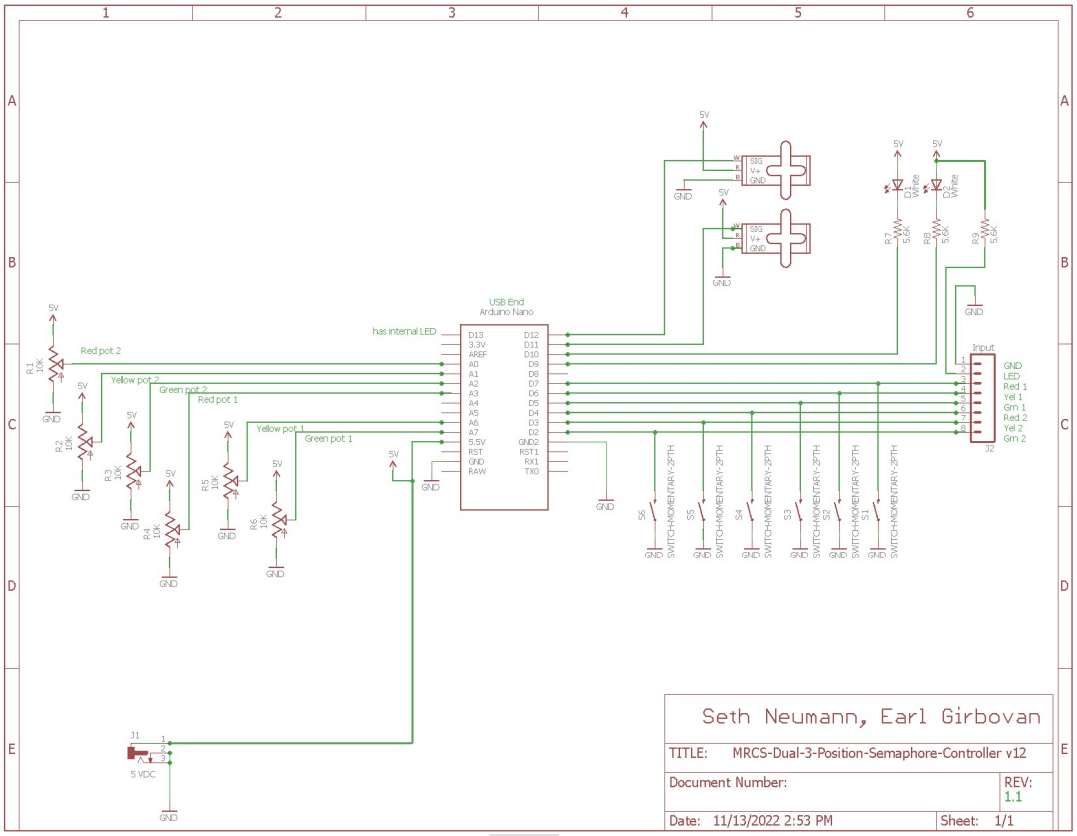


Figure 7 - Version 1.1 Schematic

Formatted: Caption

3 OPTIONS

3.1. CONNECTORS

- The 8-position connector is on 0.100 centers (staggered slightly to hold the connectors in place during assembly). While our standard connector is the 0.100 screw terminal, you may substitute any 0.100 connector you prefer. If you are ordering an assembled and tested unit from MRCS and you would prefer a different connector, please contact us at sales@modelrailroadcontrolsystems.com and indicate your preference and we'll provide a quotation.

3.2. LEDS

- I use blue LEDs with 5.6K limiting resistors for the on board LEDs (used as indicators of programming status), there is no magic to this: I usually have extra blue LEDs because of the packaging I buy my LEDs in, but any color will work fine. The supply is 5V. In case you are new to building up electronics, here's a handy table of typical LED colors and values, the more experienced reader will have his/her own preferences and whatever you have on the bench is probably fine!

Color	Forward drop	Typ Current	Suggested Resistor value*
Red	1.5V	10mA	330 ohms
Yellow	1.8V	10mA	330 ohms
Green	1.8V	10mA	330 ohms
White	3.0V	0.5 mA	5.6K ohms
Blue	3.0V	0.5 mA	5.6K ohms

The formula is supply voltage (5) – Forward Drop (1.5v for Red) = 3.5/current in A, = .01 = 350 ohms. * Using nearest standard value.

Table 3 - Suggested LED limiting resistor values

3.3. SERVOS

The board is set up for SG90 servos which are adequate for our purpose and inexpensive and have a range of a little less than the specified 180 degrees. You can use any other analog servo you like but it probably won't fit in the holes in the board. There is no reason not to mount the servos off board if you need to.

3.4. OPTIONAL LED POWER

The change in version 3.1 is power in series R3 is provided for a LED to illuminate the spectacle of the semaphore. The standard value of R3 is 5.6K, but you change it to use a grain of wheat bulb or some

Formatted: Body

different value that looks good to you with the LED you are using. If you don't want to illuminate the spectacle, you can use a jumper and the "LED" terminal becomes +5 which can be used to power the board or to provide +5 to something else if you're using the barrel jack for power.

4 ASSEMBLY

4.1. DETAILED ASSEMBLY INSTRUCTIONS

[ ] All of the components are through-hole technology with wire leads. A lead bender is a useful tool for forming the leads at 90 degrees for easy insertion into the pad holes. See "suggested tools" below if you are new to assembly.

The general rule is to install the lowest components first, working towards components that are higher off the board. This enables you to support the low components as you solder them. Most of the components will stay in place as you flip the board over to solder but if not use a small piece of cardboard to hold them in place as you flip the board over. Use 0.015 solder unless otherwise indicated to help control the flow of solder onto the work.

[ ] Resistors

[ ] Install R7, R8, R9. Use your lead bender to make 0.400 bends. There is no polarity but for consistency I put the gold "tolerance" band towards the Arduino. R9 is for the external LED to light the spectacle. If not using it you can insert a wire jumper.

[ ] LEDs

[ ] install 3mm LEDs D1, D2 align the flats on the bottom (Arduino) side. See section 3.2 above for typical values if you choose not to use Blue LEDs.

[ ] Tactile Switches

[ ] install switches S1 – S6. They should snap into the holes and stay put as you turn the board over to solder them

[ ] Pots

[ ] install variable resistors (Potentiometers) R1 -R6. . These should also snap into the holes and stay put as you turn the board over to solder them

[ ] 15 position female headers for Arduino Nano

[ ] install the female headers, use a Nano (they usually come with the male headers installed) to hold the female headers in alignment. A properly grounded electronic soldering iron (see tools below) will not harm the Nano, but experienced assemblers will have a dead Nano around to use as a fixture.

[ ] Screw Terminal

[ ] Install the 8 position screw terminal with wire openings pointing off the board.

[ ] Install the DC Power Jack (if used)

[ ] Use 0.031 solder for this. Fill the holes with solder so there are no gaps. You can mount the servos in either orientation.

[ ] Install the Servo Headers

[ ] install the 3 position male servo headers, on the bottom of the board facing down

[ ] Install the Servos (if mounting on board)

[ ] slip the servo cable through the mounting hole, and gently wiggle the servo into the hole (foil label may come off in the process) and push it firmly against the board. The secure with 2 #2 x ¼ sheet metal screws. You may want to hit the tip of the screw with a file to make it less sharp when you're done

THIS WOULD BE A GOOD TIME TO MAKE SURE YOU DON'T HAVE ANY SHORTS!

Use your bench supply with a banana jack to 2.1 mm power adapter and apply 5V. (a 5V wall wart with your volt-ammeter in series will do also, a few clip leads will be handy.) You should be drawing 3-5 mA current and the output of the supply should hold at 5V. If it's drawing more current, look for solder bridges or pins that haven't been soldered. If less, make sure the solder connection to the barrel jack is good.

[ ] install the Nano

Insert the Nano into the two female headers. (I'm assuming you've already programmed it) with the USB connector facing towards the DC power Jack (the board says "USB"). Make sure they headers are lined up (not spilling over one end). Try powering up again, you should be drawing about 50 mA.

[ ] **Plug the servos in**

Note the colors marked on the board: Orange to Orange and Brown to Brown.

4.2. LIST OF HANDY TOOLS FOR THE ASSEMBLER

- Lead Bender
- Side Cutters
- Small Needle nose Pliers
- Temperature controlled solder iron. Don't skimp here, this is a very useful tool for everything you do in model railroad electronics, get one with replaceable tips, the finer the better Weller (such as WE1010 NA) and Hako (FX888D) make very nice irons that balance well in your hand for <$150 both available from Digikey and Amazon.
- 0.015 solder for fine pitch items like 0.100 connectors (also handy on decoders)
- 0.031 solder for larger items like the power connector
- Isopropyl alcohol 91% or 99% for cleaning left over flux off the board
- Bench power supply – this gives you precise control of the voltage and allows you to measure and limit current as you test. You can see if you are drawing the correct amount of current, if not that's an indication that something is wrong. I like these: https://www.banggood.com/Topshak-NPS3010W-110V-or-220V-Digital-Adjustable-DC-Power-Supply-0-30V-0-10A-300W-Regulated-Laboratory-Switching-Power-Supply-p-1474957.html?cur_warehouse=CN&rmmds=search Equivalent units are available from many suppliers.
- Digital Volt-Ohm-Milliamp meter. This really comes down to features like quality of the probes and how the stand works, even Harbor Freight has very functional meters for as little as $10.

5 SOFTWARE AND PROGRAMMING THE ARDUINO NANO

The Arduino Nano is programmed using Jon Schmidt's "semaphore3.2" sketch using the free Arduino Integrated Development Environment (IDE). You can get it at https://www.arduino.cc/en/software. The IDE is available for Windows, Mac and Linux. There are many tutorials and guides on how to download and install the Arduino IDE if it doesn't just work for you. Note that most commodity Nanos use the CH340 serial chip and you may need to find and download the driver. They usually are supplied by the Asian vendors who make the chips and look pretty bare bones, but they work!

When you start the IDE, go to the "tools" menu:

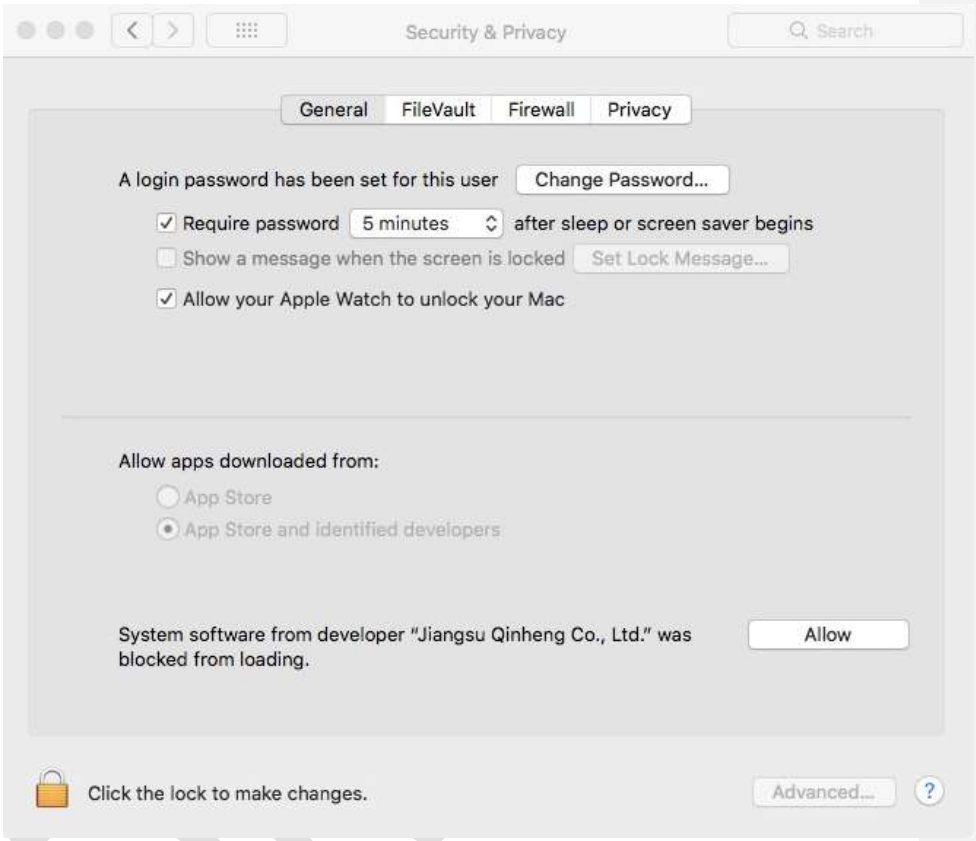
I've found the following settings to work on my Windows 10 machines.

Go to Board, mouse right and select "Arduino AVR Boards" and select "Arduino Nano"
Go to Processor, mouse over to the right and select "ATmega328P (old bootloader)"
Go to Programmer, mouse over to the right and select "AVRISP mkii"
Go to Port and be sure it's not COM1 (if that's the only choice you probably have to install the CH340 driver, or at least reboot to make your machine see it).

A Mac friend suggests these for the Mac:

Download the Arduino IDE from https://www.arduino.cc/en/Main/Donate
double click the arduino-1.8.5-macosx.zip file and drag Arduino.app into your Applications folder
Download serial driver from https://github.com/adrianmihalko/ch340g-ch34g-ch34x-mac-os-x-driver
double click the ch340g-ch34g-ch34x-mac-os-x-driver-master.zip file
double click the CH34x_Install_V1.4.pkg file to begin installation

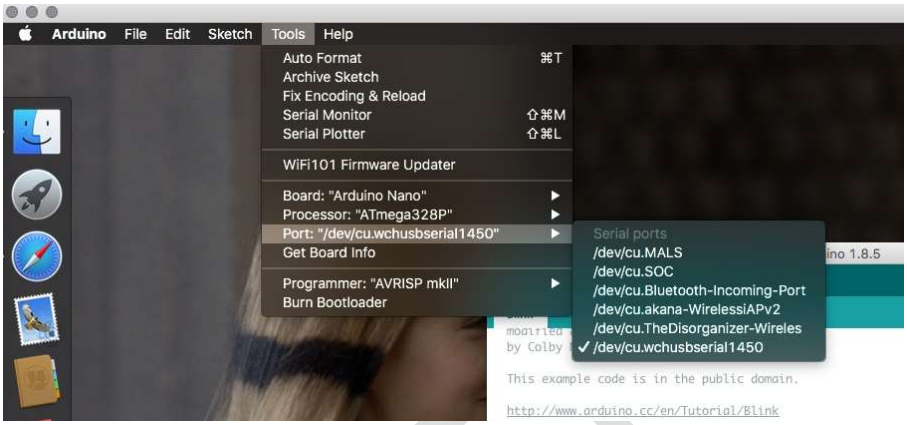
click Open Security Preferences



Click Allow
Restart your Mac - it takes a little time to rebuild the driver cache - don't panic!

Run Arduino.app from your Applications folder. There will be a prompt whether you "really want to do this". Say ok - because hey - we've got a grade crossing to program!

in the Tools Menu set the port to /dev/cu.wchuserial1450 (see below)



To prove to myself this worked I followed the instructions at https://www.arduino.cc/en/Guide/ArduinoNano to try loading the "Blink" sketch under the heading "Open your first sketch". Once you upload it the board red light will blink regularly.

If you're using Linux, you can probably figure this out for yourself.

Since the D0, D1 serial data lines on the Nano are not connected to anything, it's OK to leave the Nano on the Dual 3 Position Semaphore Controller while reprogramming.

6 USING AND MODIFYING THE SEMAPHORE SKETCH

(Check Jon's GitHub Site for the current version: https://github.com/joneschmidt/SemaphoreControl)

Formatted: Body

6.1. DESIGN:

Control two servos for driving semaphores. The board contains 3 variable resistors (pots) for each servo. These pots are labelled Red/Yel/Grn for the 3 possible positions for a blade. There are also 3 pushbuttons, labelled Yel & Grn, for a blade. The buttons and pots are aligned along the top and bottom of the board. The left set controls the left servo, the right, the right servo.
The connector block on the bottom of the board connects to the external buttons or switches or logic inputs for driving the semaphore indication.

6.2. SETUP

The servo will follow the relative positions of the pots for each requested indication. If Red is requested the current setting of the Red pot will be used to position the servo and blade. If Yellow, the Yellow pot is used. Green, Green. Therefore, some care must be used to set the pots appropriately so that the blades fall into the correct position.
This tuning of the pots/blades can happen during normal operation. However, there is a special period during a power-up or reset which can be used.

6.2.1. Setup at Power-up or Reset

At power-up or reset, the board goes into a mode where the pots may be changed to set blade positions. During this time the LEDs on the board will flash in a moderate alternating pattern. If no pot positions are changed for 15 seconds, the board transitions into normal mode and the LEDs will flash in a slower pattern.
However, a change in the position of any pot will cause the corresponding servo & blade to move to that position. This allows a period in which each blade position may be set. A change in any pot will also reset the 15 second timer, so the user may take as much time as needed to set blade positions. If no pots are changed in 15 seconds, the board progresses to normal operating mode.
We recommend that for initial installation, all the pots be set to mid-point and the blades connected. From that point, the red/yellow/green positions can be set as desired.
Rapid flashing of the LEDs is an indication that more than one input signal was active at the same time, possibly as a result of an accidental cross-connection.

6.2.2. Software controls:

```
// signals/servos – don't change
#define NumSigs 2

// FullArc - the maximum degrees of arc to travel
#define FullArc 60
```

The servos can be driven to 180 degrees of motion. This is unnecessary as smaller changes are useful for driving a semaphore. The potentiometer values are mapped to this `FullArc` range.

```
// MinPotChg - the minimum pot change we will act on
```

```
#define MinPotChg 10
```
The software doesn't act on a change to the potentiometers unless that change exceeds this value. The normal pot range is 0 - 1023.

```
// HoldMS - time before looking for a new signal for that semaphore
#define HoldMS 10000
```
The software doesn't act on a change to a semaphore until this time limit from the last change.

```
// latch = false – if true, hold last active button status
#define Latch false
```
Set `false` the software will scan repeatedly the input switches. This assumes that the switches will constantly present their state. Some panels use an enabling momentary signal to enact the signal change. By setting this `Latch to true` the software will maintain the last presented state of the inputs until a new state is detected.

```
// mirror = true - upper moves counter-clockwise, lower moves
clockwise
#define Mirror true
```
The servos are in pairs. Depending on how the card is mounted in relation to the semaphore, you may wish the servos to move in the same direction, or in opposing directions. `Mirror` = false will cause the servos to move in the same direction. `Mirror` = true will cause the servos to move in opposite directions.

```
// debugging
#define Trace 0
// main loop delay in milliseconds – make large for debugging
#define LoopDly 1000
```

6.2.3. Version 2.2 Update 5/2/2021:

We have added code to produce a "bounce" in the semaphore action when the blade returns to the Red position. The `BounceArc` controls how much of a bounce. A factor of 3 says that the bounce will be one third of the Green-Red arc, and decrease by 1/3 until the blade comes to rest. A factor of 0 means no bounce. The `BounceDly` controls the speed of the bounce.

```
// Chg 2.2 vvvvvvvvvvvvv
// BounceArc - factor for arc of bounce
//    - (green-red)/Bounce
//    - 0 means no bounce
#define BounceArc 3
//    BounceDly - delay between moves
#define BounceDly 300
// Chg 2.2 ^^^^^^^^^^^^
```

7 TESTING

Testing your Dual 3 Position Servo controller is straight forward

1. I recommend setting BounceArc to 0 while setting the position as it's a little easier to see what's happening, you can reset it later without affecting your blade position settings. You may also want to set HoldMS to 5000 so it is a little more responsive to changes.
2. Power up and starting with one blade set Red, then Green for the ends of travel and then adjust Yellow as desired. Use the pots and buttons to set red, yellow, and green. Remember the board won't look for a change for the time HoldMS after the Setup period
3. Repeat for the second semaphore.

8 INSTALLATION AND CONNECTIONS

8.1 POWER CONNECTIONS

Power is 5 Volts DC, a servo can draw up to 0.25A when moving, the Dual 3 Position Semaphore Controller will only move one servo at a time, so the maximum current is about 0.3A, to be safe use a supply rated for a minimum of 0.5A. The board has a 2.1mm barrel jack for DC power (typically from a wall wart, but you could tap off a local 5V bus or use our "Buckeroo" power distribution board to derive 5V from a 12 V bus. You can also supply 5V to the 5V and ground terminals of the 8 position screw terminal block (V1.0 or make the mod described above) on the Dual 3 Position Semaphore Controller. If you have multiple Dual 3 Position Semaphore Controllers in the same area, you can daisy chain from one to another assuming the power supply can source enough current.

8.2. SERVO CONTROL CONNECTIONS

The standard sketch assumes the semaphore is set to the RED value unless explicitly set to something else.

8.2.1. Setting the position manually with a switch

If a train order operator is controlling an order board, single pole, double throw, center off switch provides a simple interface. The center position is RED, "STOP" or "Stop and sign for a form 31 train order," one of the "ON" positions is YELLOW, "Slow down to pick up a form 19 train order on the fly" and the other ON position is GREEN, PROCEED. The center pole of the switch is wired back to the same ground as the Dual 3 Position Semaphore Controller.

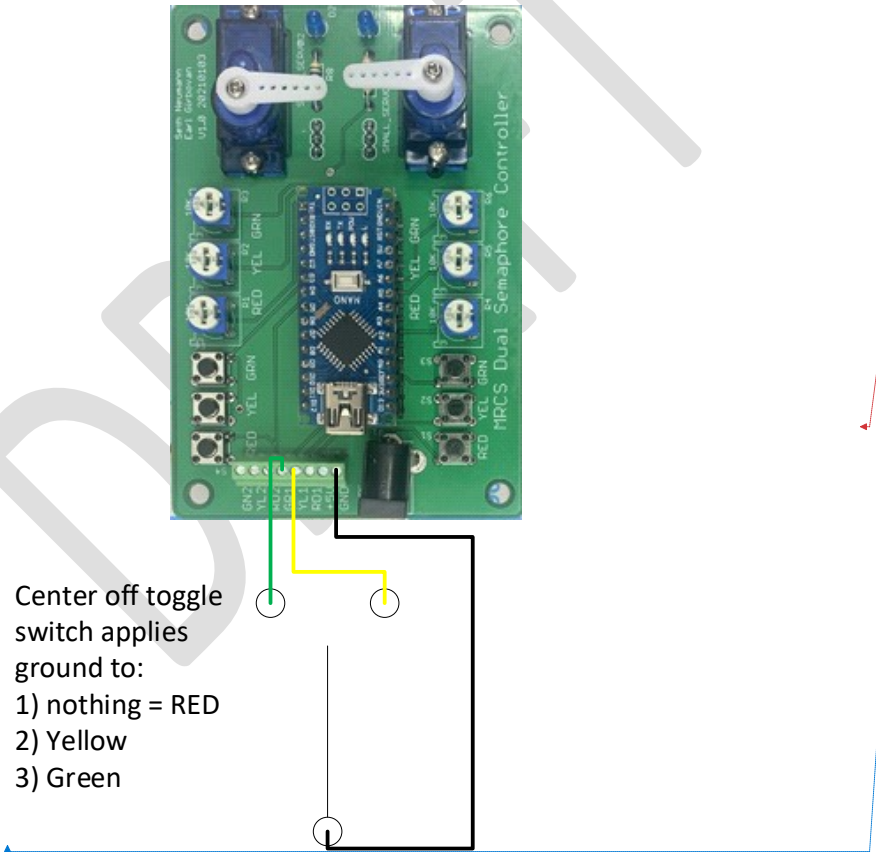


Figure 8 - Controlling D3PCS with an SPDT Center Off Toggle switch

Formatted: Indent: Left: 0.81"

Field Code Changed

8.2.2. Using the Dual 3 Position Semaphore Controller in an Automatic Block Signal (ABS) System

A typical example is Dennis Drury's Dual ABS Signal Controller (DABSC) which combines block occupancy signals and drives the controller. The DABSC has red, yellow and green outputs and they are connected Yellow out from the DABSC to Yellow in on the Dual 3 Position Semaphore Controller and Green out on the DABSC to Green in on the Dual 3 Position Semaphore Controller.

JRMI or Dr Chubb's BASIC Code can do the same for an ABS or APB system, the Dual 3 Position Semaphore Controller looks just like a 3-color signal to the logic except you don't need to connect the RED output as RED is the default position.

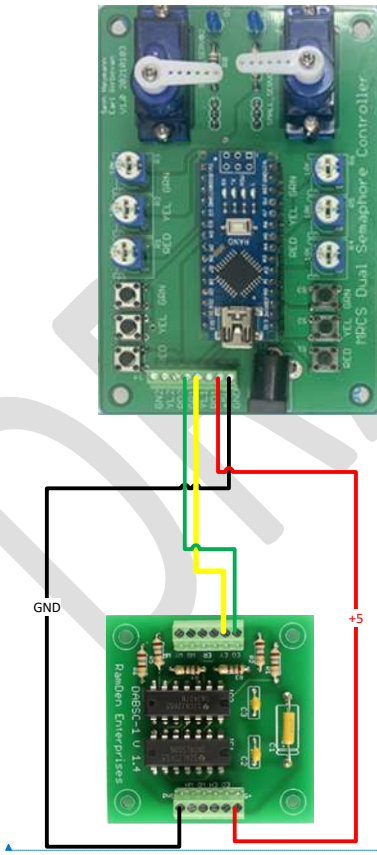


Figure 9 - D3PSC with Dual ABS Controller (DABSC)

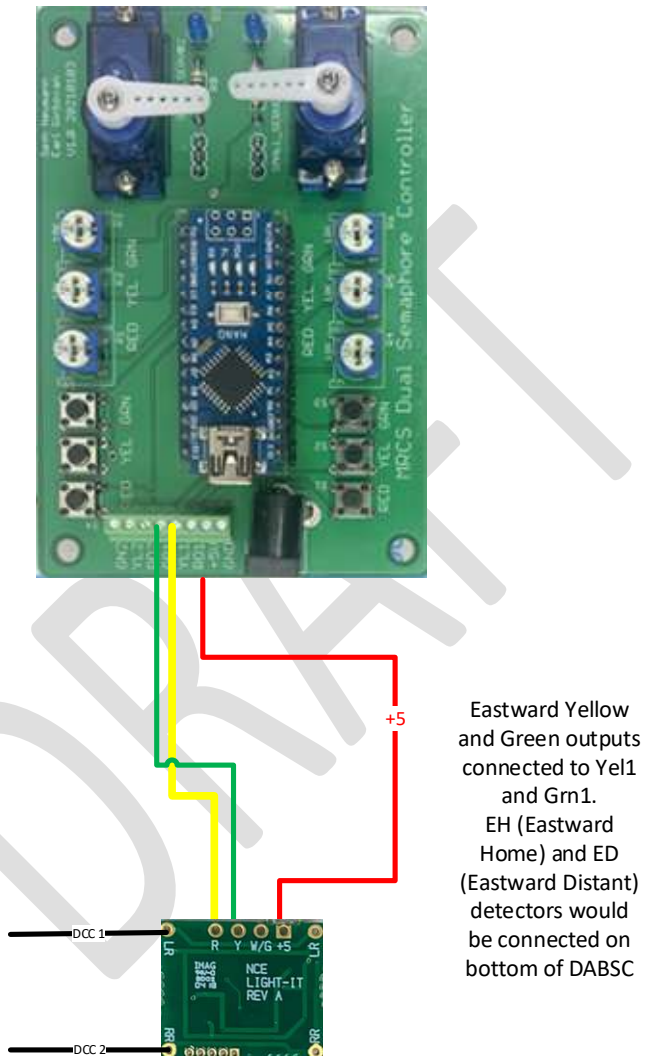
Field Code Changed

Formatted: Caption

8.2.3. DCC Control

George Sinos suggested using an NCE Light-It to control the Dual 3 Position Semaphore Controller (D3PSC).

- the Light-It has 3 outputs, (Wht)Grn/Yel/Red and a 5 volt source. The Light-It is designed to drive LEDs directly and includes 330 ohm resistors in series with each output. Since there is no (official) ground reference, I tied the 5V to the 5V screw terminal on the D3PSC to provide a reference voltage (we're not drawing any power from the Light-It).
- Then I tied the Yel output of the Light-It to the Green1 terminal of the D3PSC (I'll explain later) and the Red output of the Light-It to the Yellow1 terminal of the D3PSC.
- I had previously programmed the Light-It as mobile decoder, address 1000 and kept the configurations for its 3 outputs as effect 1: "steady on when the corresponding function button is pressed" so Green is function 1 on the cab and Yellow is function 2. The D3PSC defaults to RED, so there is no need to connect to the Red1 terminal.
- So why didn't I use the GRN output from the Light-It? It turns out NCE has installed a white LED on the (Wht)/Grn output as an indicator to help with programming, and because of the resistor value selected, the GRN output can only get down to about 2.5 volts when connected to the Nano, which does ensure reliable detection of a "logic low" input. However, the Yel and Grn outputs will pull the voltage down below 0.4V which is well within spec. The Light-It outputs can be re-mapped. You can also cut a trace and remove the LED from the circuit but then you are programming blind
- NCE also supports Accessory Decoder address and Signal address, which would be handy for an ABS-with-semaphores layout. I didn't try these options, but I expect they work fine.
- my test setup powers the D3PSC from a bench power supply and the DCC system is an NCE Power Cab, powered by the included wall wart. It is possible that some DCC systems may be referenced to ground and have incompatible voltages resulting in possible damage to the D3PSC or Light-It, so check with a voltmeter to see if there is any significant voltage between the 5 volt terminals of the D3PSC and Light-It before connecting.



Eastward Yellow and Green outputs connected to Yel1 and Grn1. EH (Eastward Home) and ED (Eastward Distant) detectors would be connected on bottom of DABSC

Figure 10 - D3PCS with NCE Light-It

Field Code Changed

Formatted: Caption, Indent: Left: 1.25"