Quad Servo Decoder-Monitor – Part 2

A DCC Accessory Decoder with Feedback

Ready to go

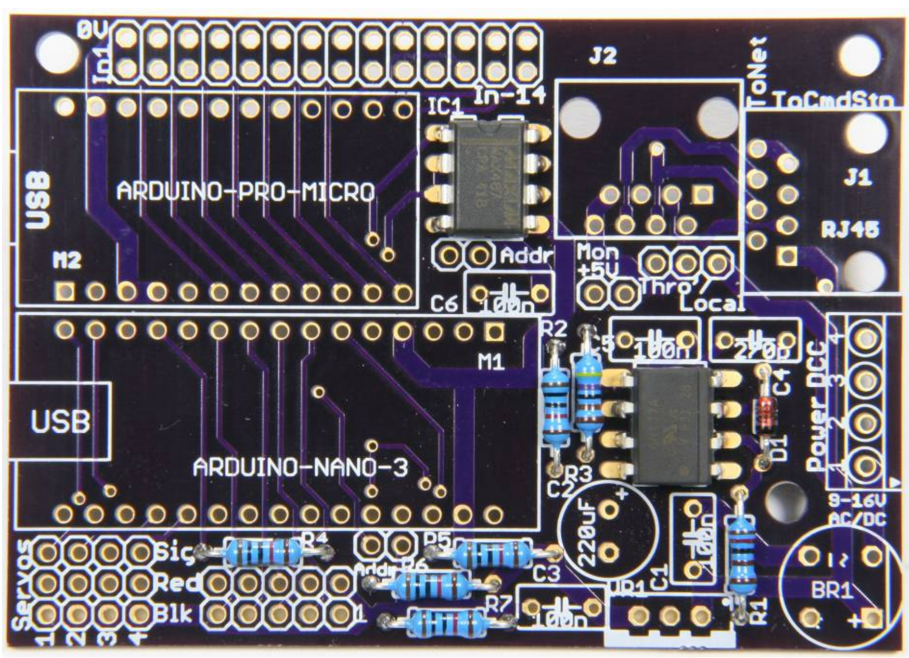
Hopefully, if building a Quad Servo Decoder-Monitor (QSDM) sparked your interest in Part 1, you now have a set of printed circuit boards and a bundle of components ready to be assembled, and have practiced your soldering technique for electronic components – using resin-cored solder in wire form only, of course.

The position and reference number of each component is printed on the top surface of each PCB. Most resistors and capacitors also have their value shown but, where this is not clear, refer back to the component tables in Part 1 to make sure you fit the correct component in each position.

Note that the assembly of the Layout-Input-Monitor PCB, mentioned in Part 1 as an alternative board to use if you only wanted to build the Monitor section of the QSDM, is not covered specifically here. However, just follow exactly the same steps as described below for the main QSDM PCB and its subsidiary units.

Building the Quad Servo Decoder-Monitor

Fit those components with least height to the QSDM PCB first, ie. the diode (D1) and resistors (R1-R7), as shown in [1], so that, when you turn the PCB over and lay it down to solder the component wires on the underside of the board, the components do not fall out of the holes. Ensure that the diode is fitted in the decoder section the right way round, as shown on the PCB markings. It does not matter which way round the resistors are placed on the PCB.



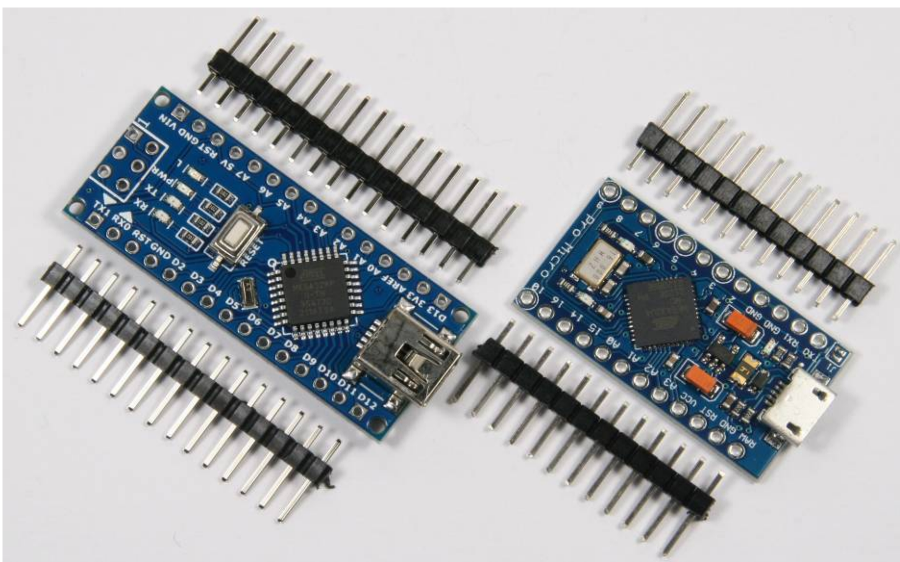
1. Partially-assembled Quad Servo Decoder-Monitor board

A tip here is to solder just one wire from each component, then turn the PCB over and check that all components are still flush with the PCB. If not, make them so (by briefly melting the one soldered joint while holding the component flat – but watch your fingers – soldered wires are hot!) before soldering the remaining wire(s) of the component. Clip the excess component wires flush with the soldered joints on the underside of the PCB using a small pair of sidecutters.

When you fit the 6N137 optoisolator and the MAX487 transceiver (as the next tallest components), ensure that the notch or dot at one end of each package is towards the nearest edge of the PCB, again as indicated by the board markings, and that all pins are through the PCB holes, with none having been bent under the devices. Solder two diagonally-opposite pins on the chips first, and then check that the devices are still flat on the PCB. If not, it is easy to melt the solder on the appropriate corner pin while pushing the device down into the correct position, before soldering the remaining pins.

The next step is to fit the Arduino Nano-3 and Pro-Micro modules to the QSDM PCB but, before you do so, you can carefully connect each module in turn to one of your computer's USB ports, using the cables listed in the QSDM parts table, to check that each module powers up correctly. For the Nano-3, you will normally see two LEDs lit – one constant (labelled POW) and one flashing (labelled L) for a short period, depending on the current internal state of the Nano module. The Pro-Micro module should behave similarly, with the LED next to the USB connector constantly lit, and the LED at the other end of the module flashing for a brief time. You do not need to have any of the Arduino software installed on your computer to perform this check which gives you some confidence that the modules will work – before soldering them to the PCB.

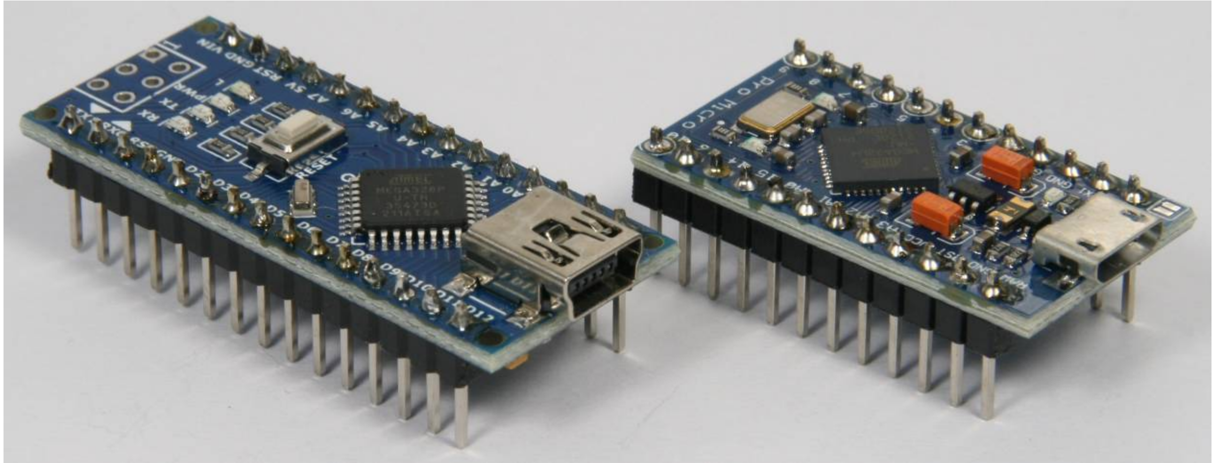
If you purchased the Nano-3 and Pro-Micro modules 'unsoldered' then you will need to attach the supplied pin header strips, as shown in [2], to the modules before fitting them to the PCB –



2. Header pin strips supplied with the Arduino Nano-3 and Pro-Micro

A tip here, to ensure that the header strips are fitted square and flush, is to hold the QSDM PCB horizontally above your bench, either in a vice or using a 'third-hand' jig, and then drop the pin headers through the respective rows of holes in the PCB, long pins down.

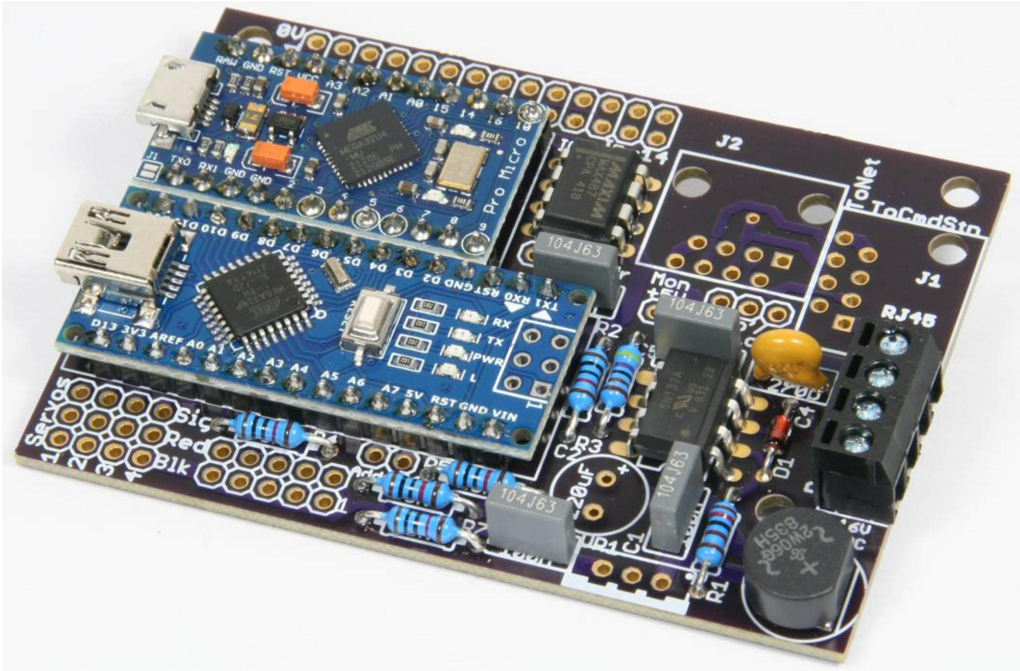
Now fit each Arduino module on to its header strips and proceed to solder it to the header pins – generally starting with the corner pins and checking that the module is sitting square on the header strips before continuing with the rest of the pins. With the Nano-3, make sure that you don't fit any of the corner holes on to the header strips by mistake – each header pin must come through a solder pad, as can be seen in [3] –



3. Header pin strips fitted to the Arduino modules

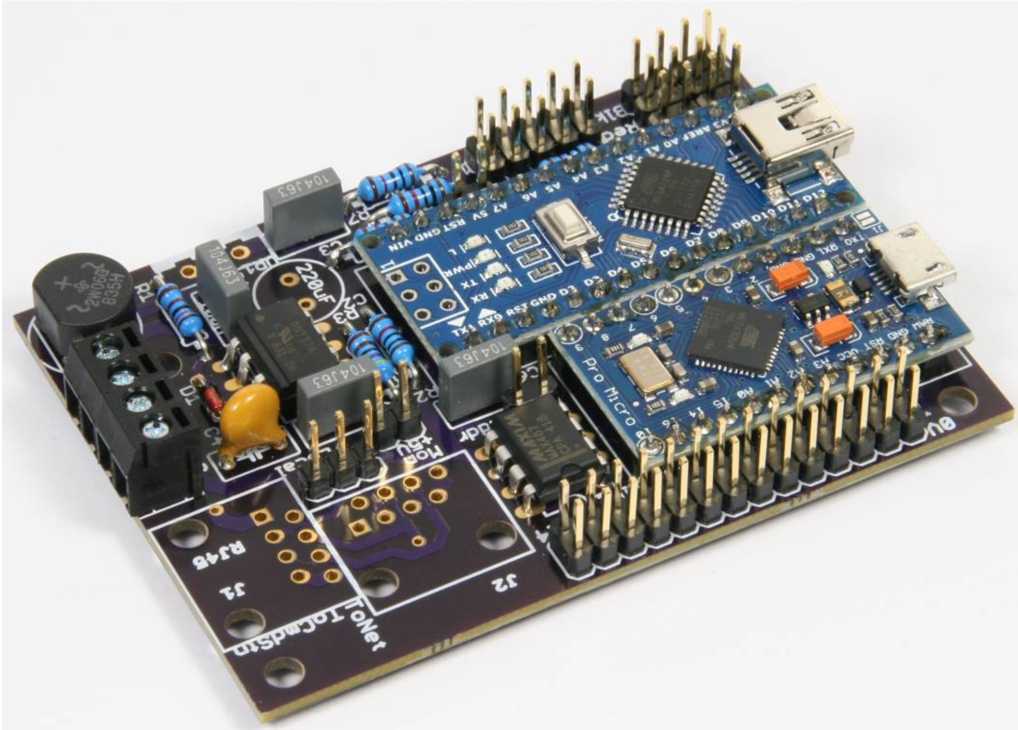
As always, make the solder joints in as short a time as you can, to minimise the heat applied to the module. Note that there is no need in this application to fit the additional 6-pin header, which is usually supplied, to the end of the Nano-3 module furthest from the USB connector.

Fit and solder the Nano-3 and Pro-Micro modules on to the QSDM board, making very sure that both module's USB connectors are towards the edge of the board, followed by the smaller capacitors (C1, C3-C6), bridge rectifier (BR1), and terminal block. Capacitors C1 and C3-C6 can be fitted either way round on the board. The flat edge of the bridge rectifier should be next to the shorter edge of the PCB, as shown in [4] –



4. Continuing assembly of the QSDM board

Now fit the pin headers. If, as recommended, you purchased single long strips of, say, 24 pins, carefully cut them into the required sizes. Place and solder each group of pins in place separately, ensuring that they are flush to the PCB with the pins vertical, but make your soldered joints as quickly as you can – it's very easy to melt the plastic. You should now have a board looking like [5] below –



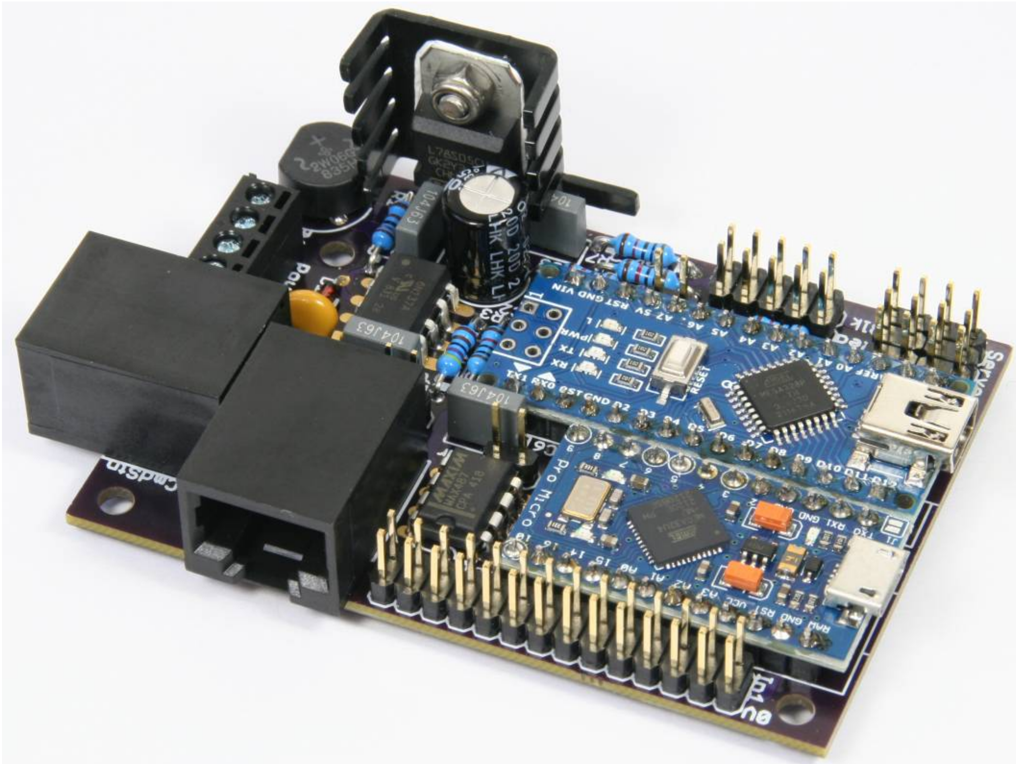
5. All pin headers in place

As the tallest components, the 220uF capacitor (C2) and the two RJ-45 connectors will be fitted last. Ensure that capacitor C2 is mounted the right way round, with the stripe indicating the negative (-) terminal towards the nearest edge of the PCB.

The RJ-45 connectors have two mounting pegs which fit through the holes drilled in the PCB. Take care that the eight pins at the rear of each connector are located in the relevant holes in the PCB **before** you press the connector and its mounting pegs into place. Fitting connector J1 (labelled 'ToCmdStn') **first** lets you tilt each connector to ensure the pins are positioned correctly.

Finally, fit the voltage regulator with the metal tab facing outwards. It is important that a heatsink, as listed in the QSDM parts table, is attached to the voltage regulator to keep its operating temperature at a comfortable level.

Carefully straighten one of the bottom fins (beware of the soft aluminium cracking and breaking the fin away from the main heatsink body), to allow the heatsink to be mounted with the fins facing inwards towards the centre of the PCB in the final assembly [6] –

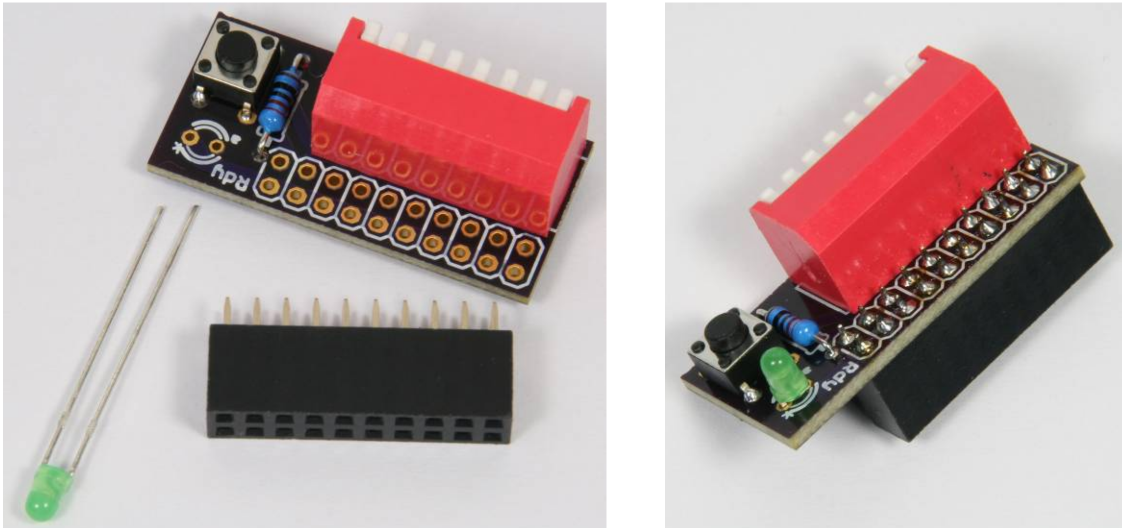


6. Assembled Quad Servo Decoder-Monitor

Building the subsidiary units

A Keypad, together with its extension cable, is required to set up the servos which will eventually be attached to the decoder section of the QSDM. If you do not already have a Keypad (after building the original Quad Servo DCC Decoder) then please refer to the original February 2020 MRH article (online.fliphtml5.com/buups/wpwb/index.html#p=49) for full assembly details.

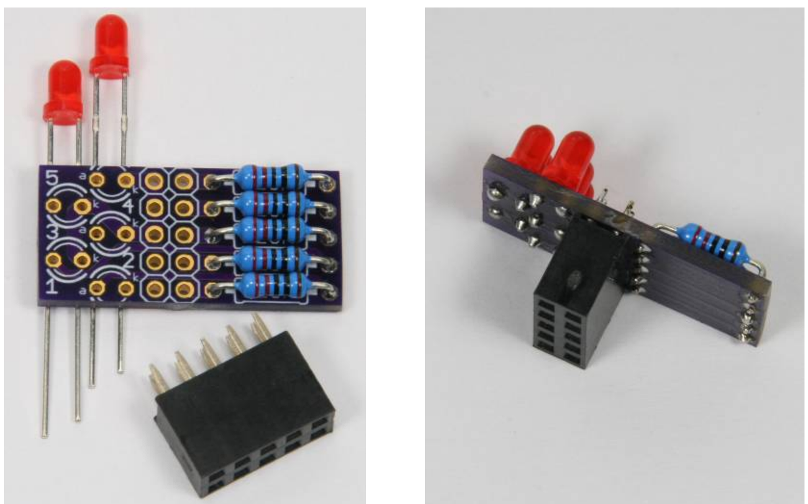
The other unit that you need to assemble is the Set Address module. Fit and solder resistor R1, tactile switch S1, and the 8-way DIP switch S2 to the PCB. The LED is fitted as shown in [7] below, with the longest lead going through the PCB hole marked ‘a’ (furthest from the short edge of the PCB). Note that LEDs are particularly sensitive to heat, especially when fitted flush to the PCB, so do not linger with your soldering iron.



7. Set Address module assembly

Lastly, fit the 2 x 10 socket header on the **underside** of the board, ie. on the opposite side to the other components.

As an option you can also build a Status View unit with five resistors and five LEDs fitted as shown in [8] below –



8. Assembled Status View module

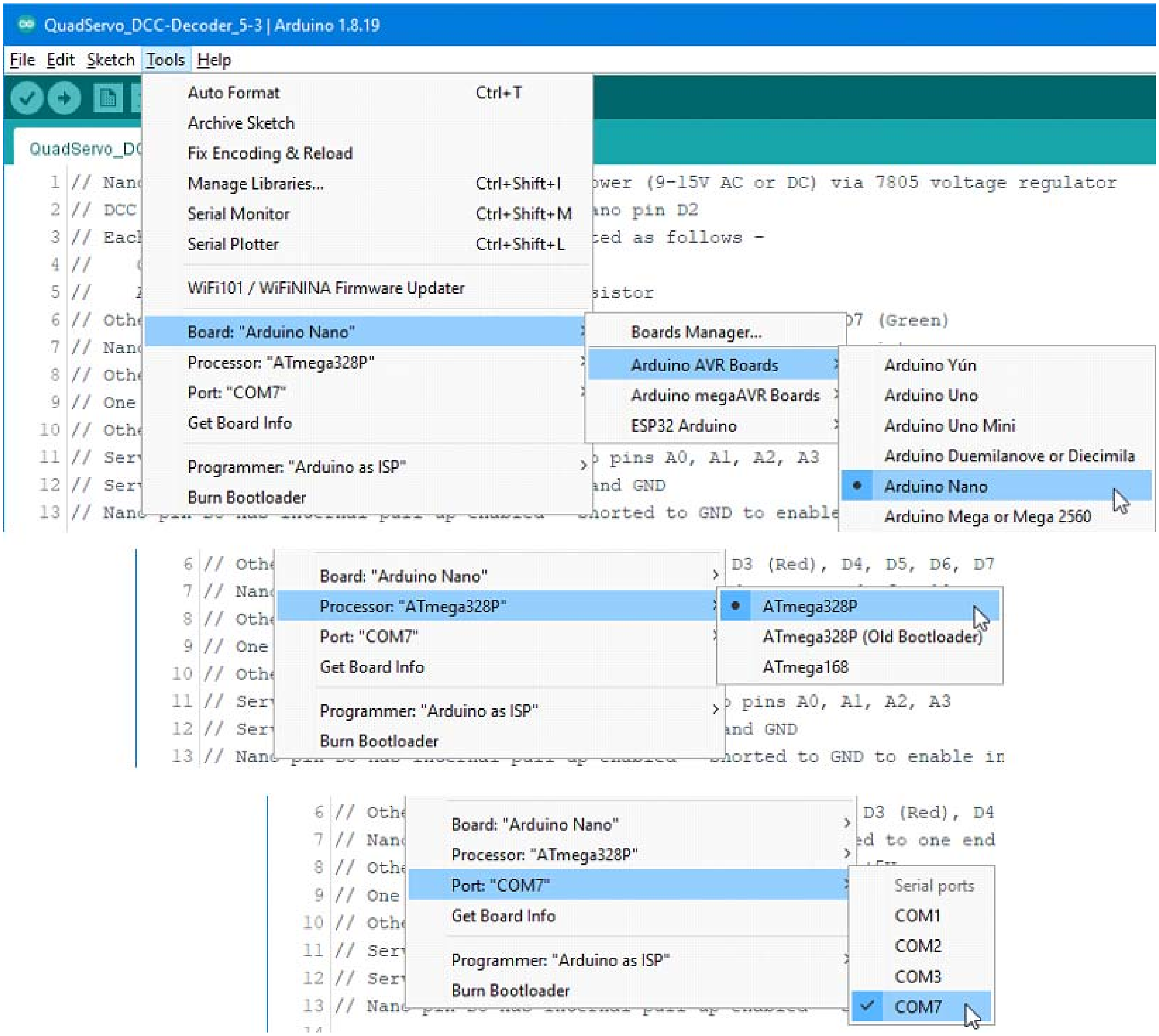
Again, the 2 x 5 socket header is fitted on the **underside** of the board.

Carefully inspect each of the completed boards to check that all of your soldered joints are bright and shiny, and that the solder has wicked through the PCB holes to the component side of the board. Check also that there are no solder bridges between copper pads or component pins anywhere on either side of each PCB. Use of a x5 or x10 hand lens or jeweller's loupe is highly recommended for this inspection.

Uploading Arduino software

Assuming you have installed the Arduino IDE on your computer, as described in Part 1, and downloaded the two sketches for the Nano-3 and Pro-Micro, respectively, the next job is to upload the code to the Arduino modules.

Start the Arduino IDE by double-clicking on the sketch for the Nano-3 (QuadServo_DCC-Decoder_5-3.ino), and check that the Board, Processor, and Port settings on the Tools menu are set appropriately, as shown in the example screenshots in [9] below –

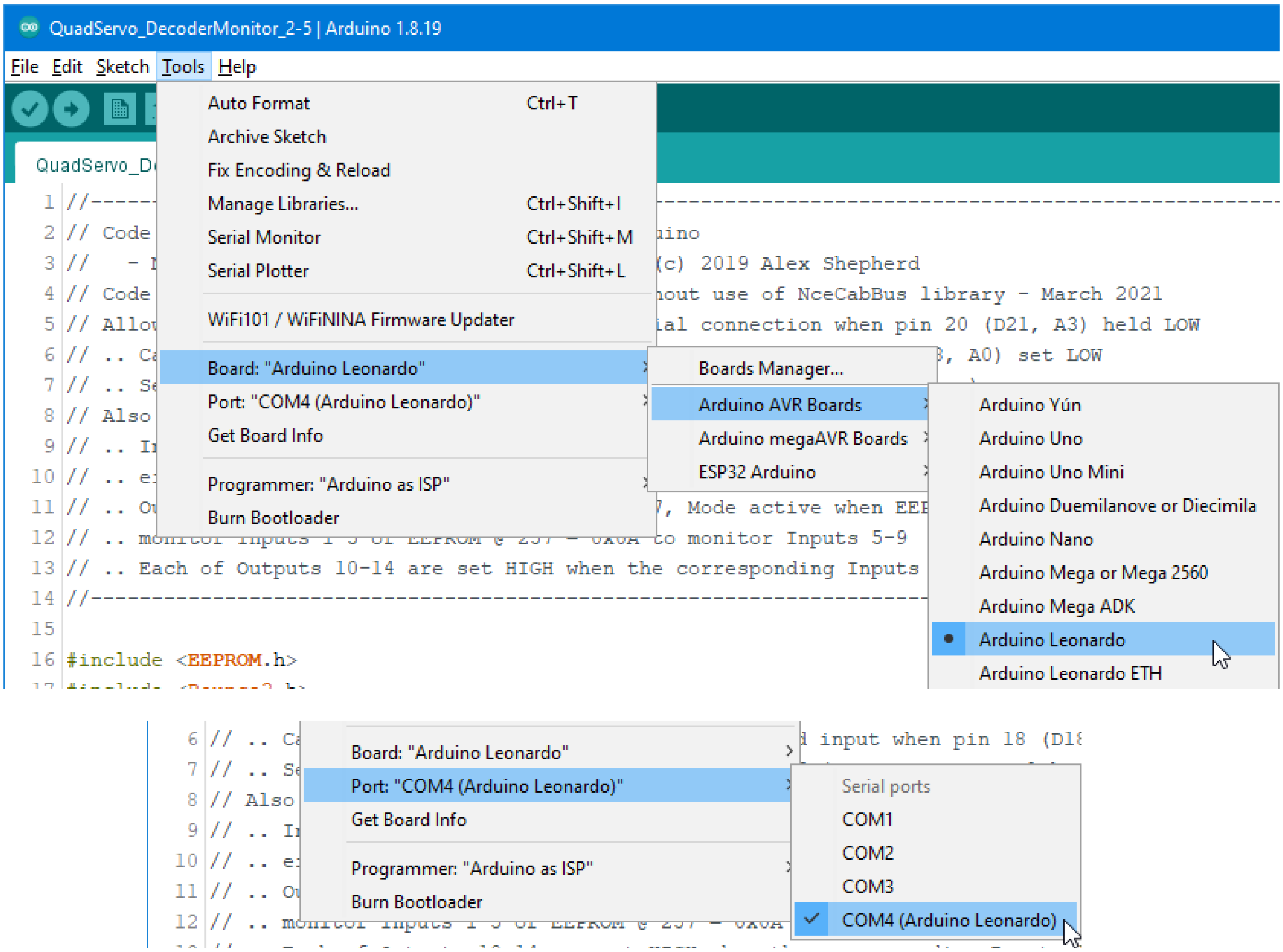


9. Arduino IDE setup for Nano-3 module

Connect a Mini-B USB cable from your computer to the Nano-3, then click the Upload (➔) button to transfer the compiled sketch to the module. You should see the Tx and Rx LEDs on the Nano-3 light for varying lengths of time as the transfer proceeds. If you have any difficulty finding the right Port, a useful tip is to disconnect the USB cable and look at the list of available ports (if any) displayed when you click Port on the Tools menu. Now close the Tools menu, plug the USB cable back in, then look at the Port list again – there should now be one extra port shown which is, therefore, the one to use.

Once the sketch has been uploaded, the Nano-3 will retain all of the code even when it is switched off so you can simply remove the Mini-B USB cable and close the Arduino IDE window on your computer.

Restart the Arduino IDE by, this time, double-clicking on the Pro-Micro sketch (QuadServo_DecoderMonitor_2-5.ino) then, from the Tools menu, reset the Board and Port settings to those for an Arduino Leonardo. The Port will almost certainly be different from that used by the Nano-3, but is usually clearly identified, as can be seen in the screenshots in [10] below –

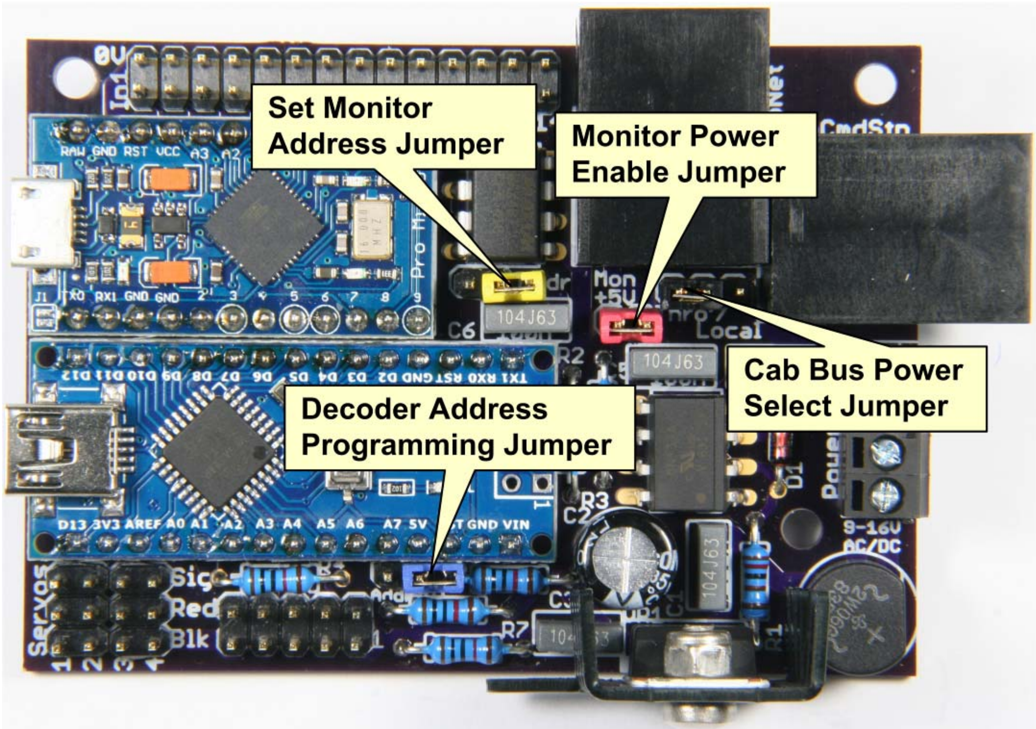


10. Arduino IDE setup for Pro-Micro module

Connect a Micro-B USB cable from your computer to the Pro-Micro, then click the Upload ( ) button to transfer the compiled sketch to the module. Again, you should see the Tx and Rx LEDs on the module light for varying lengths of time as the transfer proceeds. Once the sketch has been uploaded, remove the Micro-B USB cable and close the Arduino IDE window on your computer.

Ready for setup

Before proceeding to the next stage of QSDM setup, you need to add four jumper links to various pin headers as shown in [11] below. Note that you can use any colour of jumper that you like – I use different colours simply to help me identify which jumper is which –



11. QSDM jumper link positions

The Monitor Power Enable jumper (red) connects the Monitor section of the QSDM to the onboard 5-volt supply from the voltage regulator, and should be fitted to the header at all times except when accessing the Decoder section configuration variables using JMRI Decoder Pro or my own A-Track application.

The Cab Bus Power Select jumper (black) is normally fitted to the two leftmost pins of the 3-pin header to allow the 12-volt supply carried by the NCE Cab Bus to pass through the RJ-45 sockets. Fitting this jumper on the rightmost pair of pins allows the QSDM to supply power to the downstream section of the Cab Bus, but this would only be necessary for a very large, extended layout and would also require that QSDM to be powered from a 13.5-volt supply.

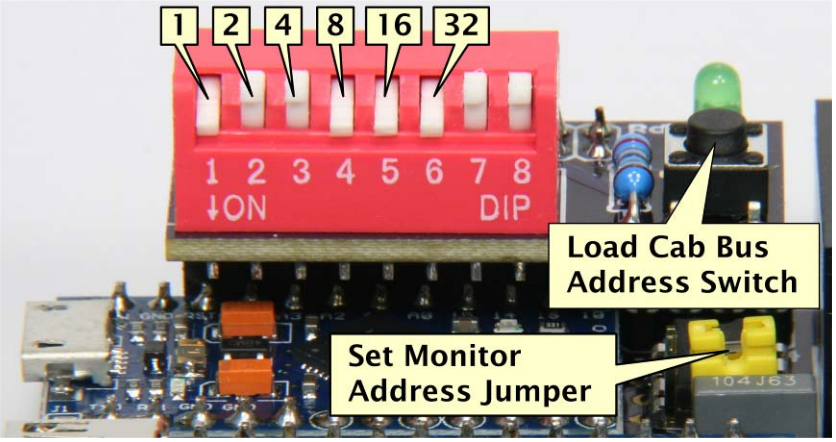
The other two jumpers, Set Monitor Address (yellow) and Decoder Address Programming (blue), are only fitted during setup of the QSDM Monitor and Decoder sections, respectively, and will normally simply be ‘parked’ on one of the relevant pair of header pins.

QSDM decoder section setup

After connecting the QSDM to the track DCC and to its own power source (9-volt recommended), and connecting up to four servos to the appropriate 3-pin headers, the setup of the QSDM Decoder section is identical to that described for the Quad Servo DCC Decoder in Part 2 of the original article in *MRH March 2020 (online.fliphtml5.com/buups/hfkw/index.html#p=51)*, or as found on the A-Train Systems website (www.a-train-systems.co.uk/download.htm#Projects) – so I won’t repeat all the steps here.

QSDM monitor section setup

Setup of the Monitor section only requires you to set a Cab Bus address for the particular QSDM, by using the switch block fitted to the Set Address unit. Block switches 1 to 6 set the required Cab Bus address, by being set ON or OFF with a small flat-blade screwdriver or similar tool, and are assigned the numerical values shown in [12] below –



12. Setting QSDM Cab Bus address

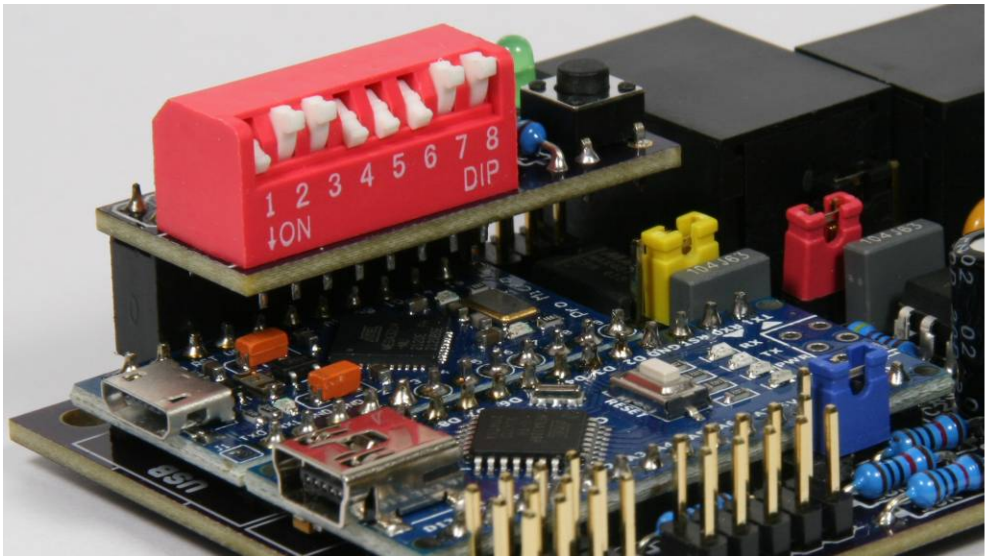
The Cab Bus address is selected by adding the values of switches 1 to 6 which are ON, so that the address set in [12] will be 57 (1+8+16+32). Switches 7 and 8 are used to control the way in which the optional Status View module operates, as described later in the article.

Using the NCE Power Pro, the Cab Bus address can have any value between 2 and 63, although NCE recommend that you use addresses from 41 and above. With the NCE Power Cab, only addresses 2 to 10 are available, with some restrictions (see the NCE website at ncedcc.zendesk.com/hc/en-us/articles/201802345-Cab-Ids-101 for more details).

Apply power to the QSDM Monitor section by connecting the Pro-Micro to a USB port on your PC. It is not necessary (nor desirable) at this point to make any other connections to the QSDM.

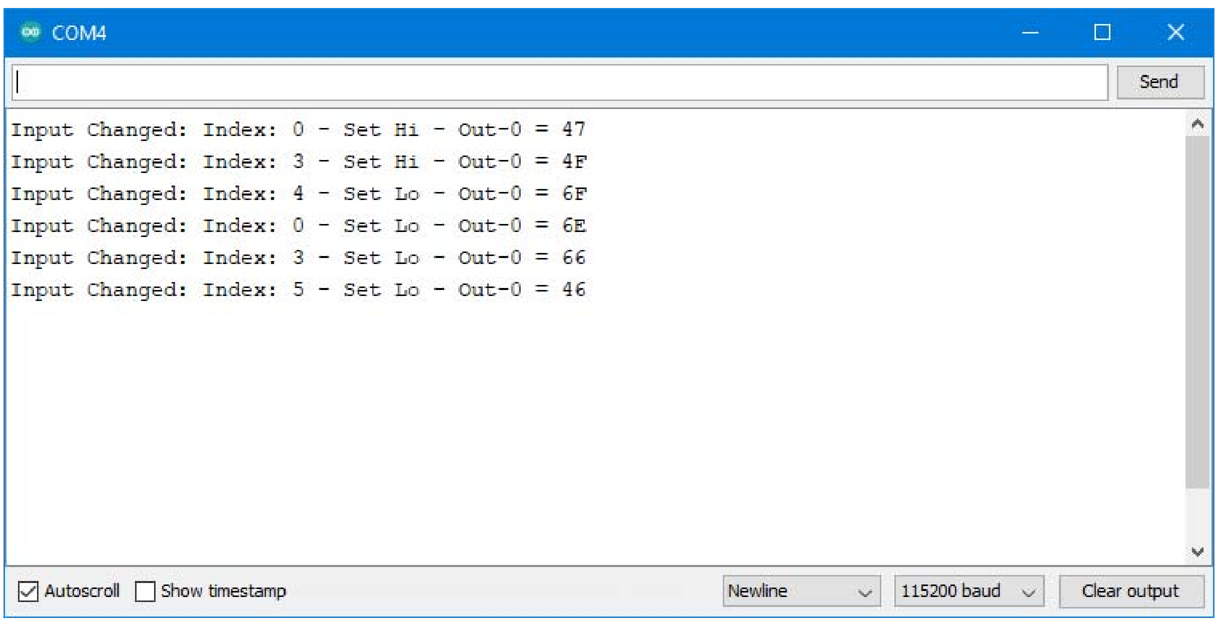
Open the Arduino sketch for the QSDM Decoder section (QuadServo_DecoderMonitor_2-5.ino) and then start the Arduino Serial Monitor by clicking on its icon ( ) in the top-right corner of the Arduino window. The speed of the serial link should be set at 115200 baud, selected from the drop-down list located at the bottom-right corner of the Serial Monitor window.

Set the Set Address unit switches to the required Cab Bus address, then carefully plug the unit on to the first 10 pairs of the 14x2 pin header of the QSDM Monitor section, with the switch block overlapping the Arduino Pro-Micro, as indicated in [13] –



13. Set Address module fitted to QSDM Monitor header

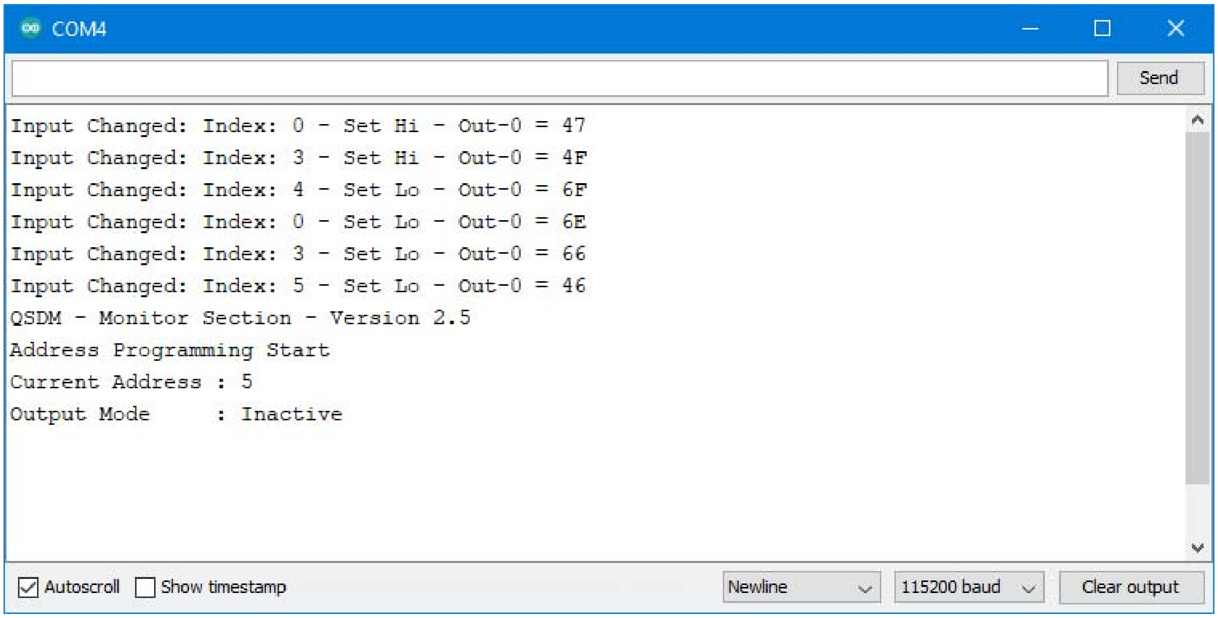
The Serial Monitor window will display a few messages showing that the state of some of the QSDM Input pins has changed (depending on how the Set Address switches have been set) as shown in [14] –



14. Serial Monitor window – initial state

Now, using a small pair of pliers (or your fingers if you are sufficiently dextrous) fit the jumper link to the 2-pin header next to the Pro-Micro (the yellow jumper labelled 'Set Monitor Address Jumper' in [10]) – the jumper is normally 'parked' on one of the header pins.

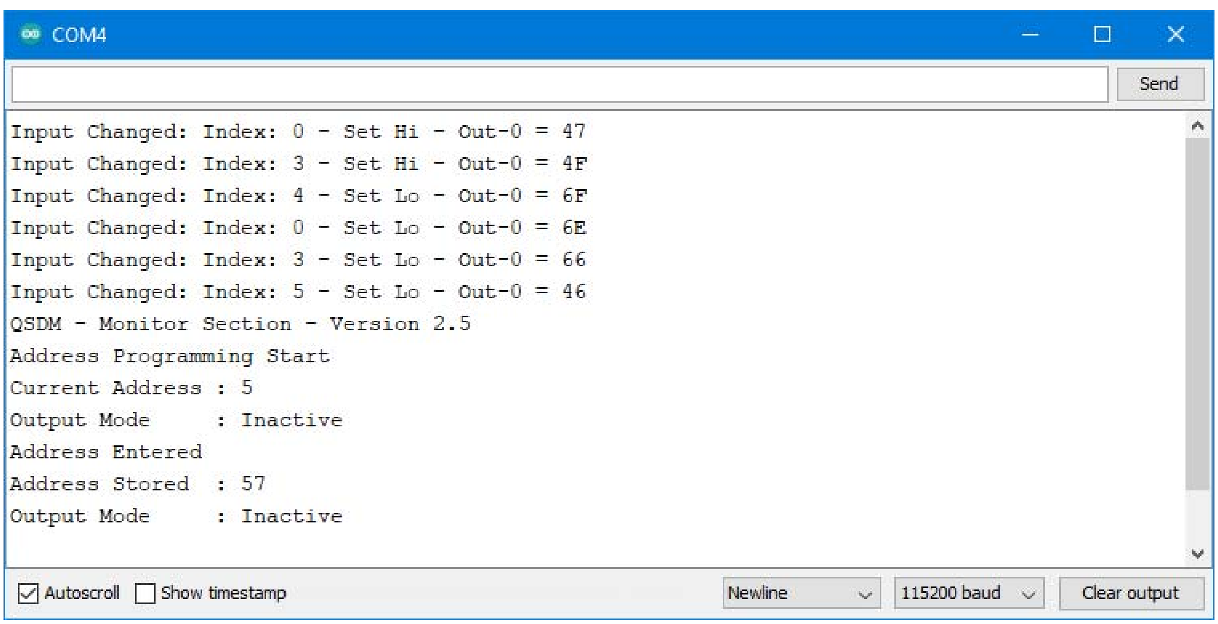
This action should light the green LED on the Set Address unit and display a further message in the Serial Monitor window, as in [15] below, which shows you the currently-stored Address and Output Mode –



15. Serial Monitor window – programming started

Note that, if you do not manage to fit the jumper at the first attempt, so that it makes and breaks contact a couple of times, then you may see some further messages in the Serial Monitor window, indicating that Address Programming has been stopped and restarted.

This is not a problem – once the jumper is firmly in place, simply press the pushbutton on the Set Address unit. This will extinguish the LED and produce a set of messages in the Serial Monitor window, shown in [16], confirming that the Set Address (57 in this case) plus the Output Mode status (as set by switch 7 or 8) has been saved in the Pro-Micro memory (EEPROM). This means that these values will be retained even when the QSDM is powered off –



16. Serial Monitor window – address stored

You can now remove the Set Monitor Address jumper and ‘park’ it on one of the 2-pin header pins for safekeeping. A final confirmation message ‘Address Programming Completed’ will be displayed in the Serial Monitor window, and you can unplug the Set Address unit from the QSDM. If the

Serial Monitor window is still open you will see a few more Input Changed messages reporting the resultant changes in the Pro-Micro inputs – these can be ignored.

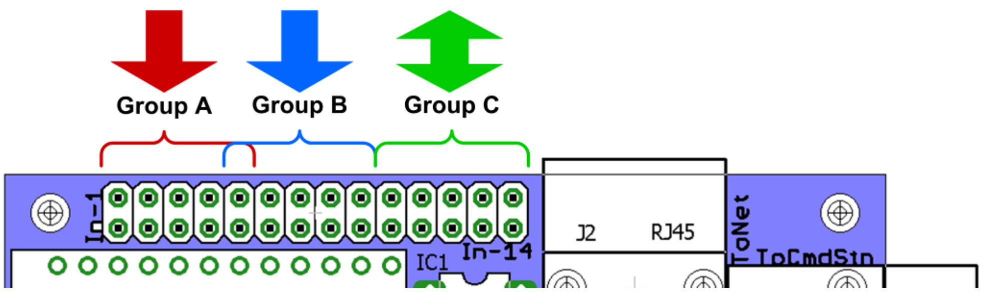
The USB connection to the Pro-Micro from your PC should now be removed, and the QSDM connected into its intended position in the layout control network, using standard Cat5/6 Ethernet cables (as shown in the example network [4] in Part 1 of this article).

Note : It is **essential** that the QSDM is **powered off after programming** (and removal of the Set Monitor Address unit) – the newly entered values will not become effective until the QSDM is restarted.

Note also that, when setting up the address of the QSDM Monitor section, it is not actually essential to have the Arduino Serial Monitor window open – its only purpose is to provide confirmation that the switch settings have been accepted and the intended values have been saved to EEPROM – the Arduino IDE takes no part in the data entry operation.

Status View module setup

The 14 inputs (header pin pairs) to the QSDM Monitor have been divided into three groups, A, B and C, where Group A consists of inputs 1-5, Group B has inputs 5-9 (overlapping with Group A), and Group C has the remaining inputs 10-14, as shown in [17] below –

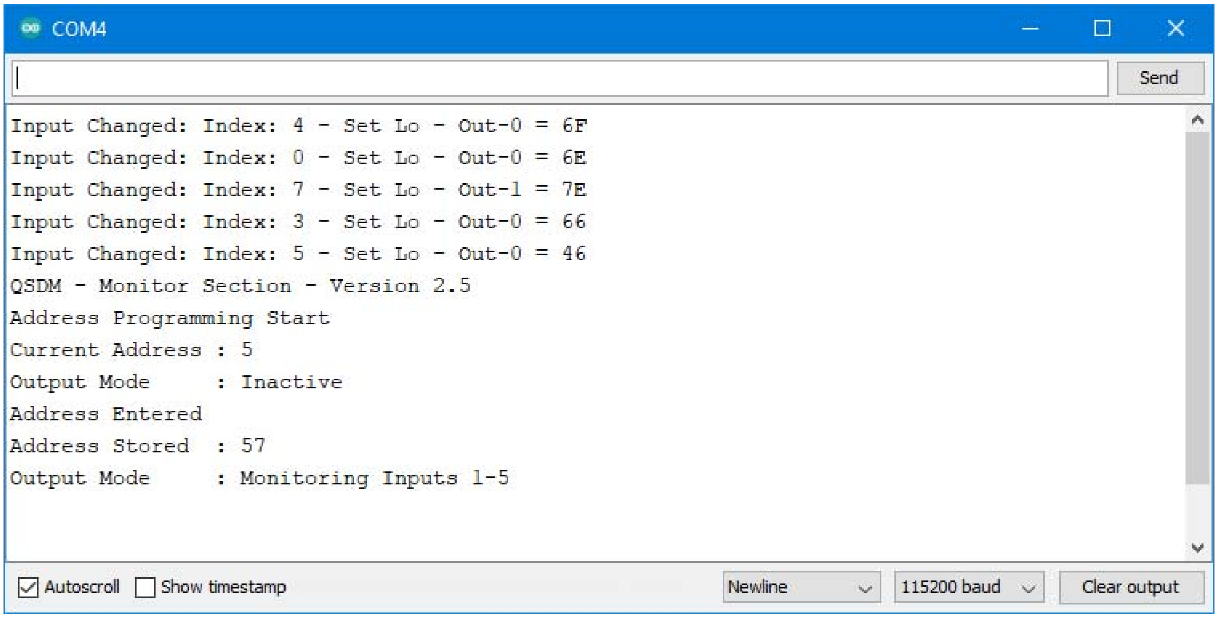


17. Grouping of QSDM header pins

During programming of the Cab Bus address, with the Set Address module plugged on to the QSDM 14x2 pin header, you can set the pins of Group C as outputs (Output Mode) to drive the LEDs of the Status View module.

To enable the LEDs to display the states of the Group A inputs, set switch 8 ON or, to enable the display of the Group B input states, set switch 7 ON. Note that setting both Switches 7 and 8 ON will only enable display of the Group A input states. To disable Output Mode, and allow the Group C pins to be used as inputs, set both switches 7 and 8 to OFF.

If, for example, switch 8 was set to ON during the programming of the Cab Bus address to 57, as shown previously, the Arduino Serial Monitor will display the additional message shown in [18] below, to indicate that the Status View module, when plugged on to the Group C header pins, will display the state of the Group A pins –



18. Serial Monitor window – input monitoring enabled

Alternatively, if switch 7 is set On prior to programming (with switch 8 set Off, otherwise the setting of switch 7 will be ignored) the states of the Group B inner pins will be shown on the Status View module when it is plugged on to the Group C pins, ie. Output Mode will be shown as Monitoring Inputs 5-9.

Hence, assuming we have set switch 8 as part of Cab Bus address programming, when a sensor, such as a microswitch on a turnout servo, or a block occupancy detector, is connected to one of the QSDM Monitor Group A inputs, the state of the sensor will be indicated immediately by whether the LED attached to the corresponding Group C output lights up or not (when power is applied to the QSDM, of course).

Once you are sure that all of your sensor connections are sound, you can simply unplug the Status View module, ready for use during your next QSDM setup.

However, if you do then want to use the pins of Group C as inputs from other sensors, you will need to remove all of the current connections from the QSDM 14x2 pin header – making a careful note of what they are and which way round they are connected. You can then reprogram the Cab Bus address into the QSDM, with switches 7 and 8 set Off on the Set Address module. Finally, carefully reconnect the sensor inputs you removed, before adding additional sensors to the pins of Group C (now reset to be inputs).

Using the QSDM

After all parts of setup have been completed, the QSDM can be powered either from the DCC track or, preferably, from a separate 9 to 15-volt AC or DC supply capable of providing up to 1 amp (or better, 2 amps, if a number of servos are to be operated simultaneously). Note that, unlike the NCE AIU, the Monitor is not powered from the Cab Bus 12-volt line, so should be switched on **before** the NCE Command Station is powered up. When operating the QSDM it is best to use the minimum external supply voltage (9V AC/DC), reducing the power dissipation in the on-board voltage regulator, and keeping its heatsink as cool as possible.

Supply power to the QSDM, and hence to the attached servos, through terminals 1 and 2 of the 4-way terminal block. Take a connection from the track DCC supply to terminals 3 and 4 of the terminal block. This connection supplies commands to the Servo Decoder section of the QSDM but, in normal operation, does not supply any substantial power to the module.

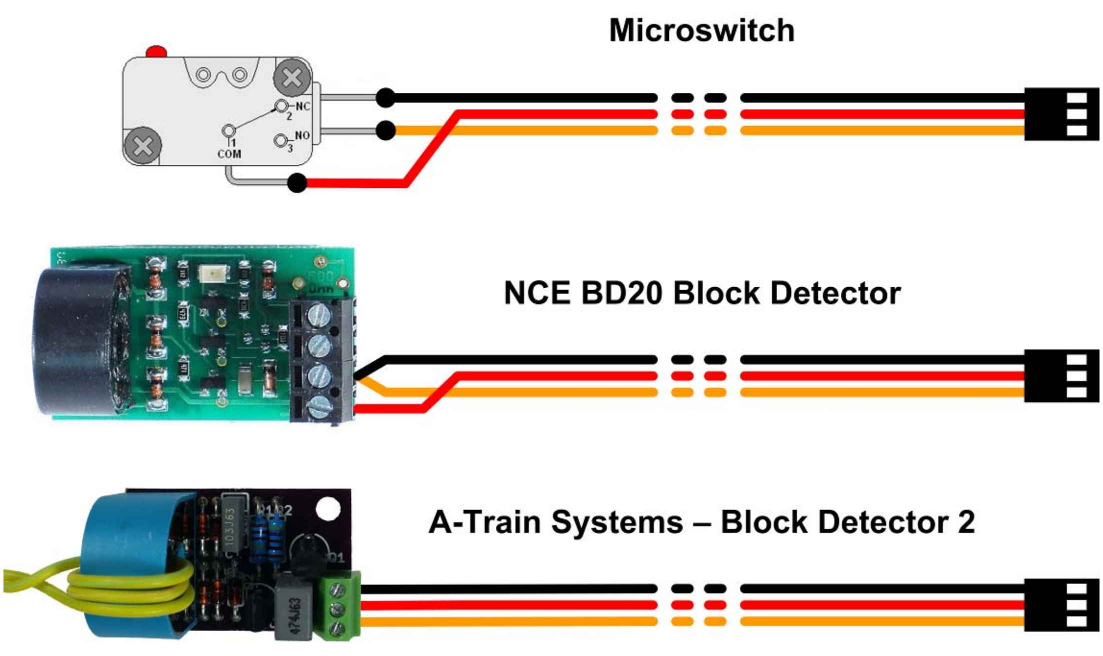
If you wish to operate everything from the DCC bus then you can connect terminals 1 and 3, and 2 and 4, together. However, the higher DCC input voltage will substantially increase the power dissipated in the voltage regulator, so that a larger heatsink would be strongly recommended.

The Monitor section of the QSDM is connected to the Command Station Cab Bus via the 8-way RJ45 sockets J1 (labelled 'ToCmdStn'), using a standard Ethernet cable, and the Cab Bus is continued in daisy-chain fashion to the next QSDM (or AIU or other NCE device) from the second RJ45 socket, J2 (labelled 'ToNet'). The Cab Bus daisy-chain can be extended to incorporate as many attachments as required, bearing in mind that the ultimate limit is 62 devices, since each attached device must have a unique address, and can stretch as far as 1000 feet.

However, very long cables will result in excessive voltage drops which may prevent Handheld Controllers plugged into the end of the cable run from working. This does not affect the QSDMs since they are powered from their own independent supply but, for cable runs in excess of 40 to 50 feet (12 to 15 metres), it is recommended that 12V DC auxiliary power units are connected to the Cab Bus (as shown in diagram [4] in Part 1) to keep any Handheld Controller at this distance operational.

You can attach any type of sensor to one of the 14 Monitor inputs via the 14x2 header pins. The outer row of pins (next to the edge of the PCB) are all connected to ground (0 volts), and an input is regarded as active when the sensor connects the corresponding inner pin to ground (0V). Typical sensors are microswitches attached to turnout motors, and block occupancy detectors such as the NCE BD20, or my own DIY detector design (see my website at www.a-train-systems.co.uk/projects.htm#BlockDet for details).

To link sensors to the QSDM, I use a 3-wire connection conveniently provided by servo extender cables. For microswitches and block occupancy detectors the connections are as shown in [19] below –



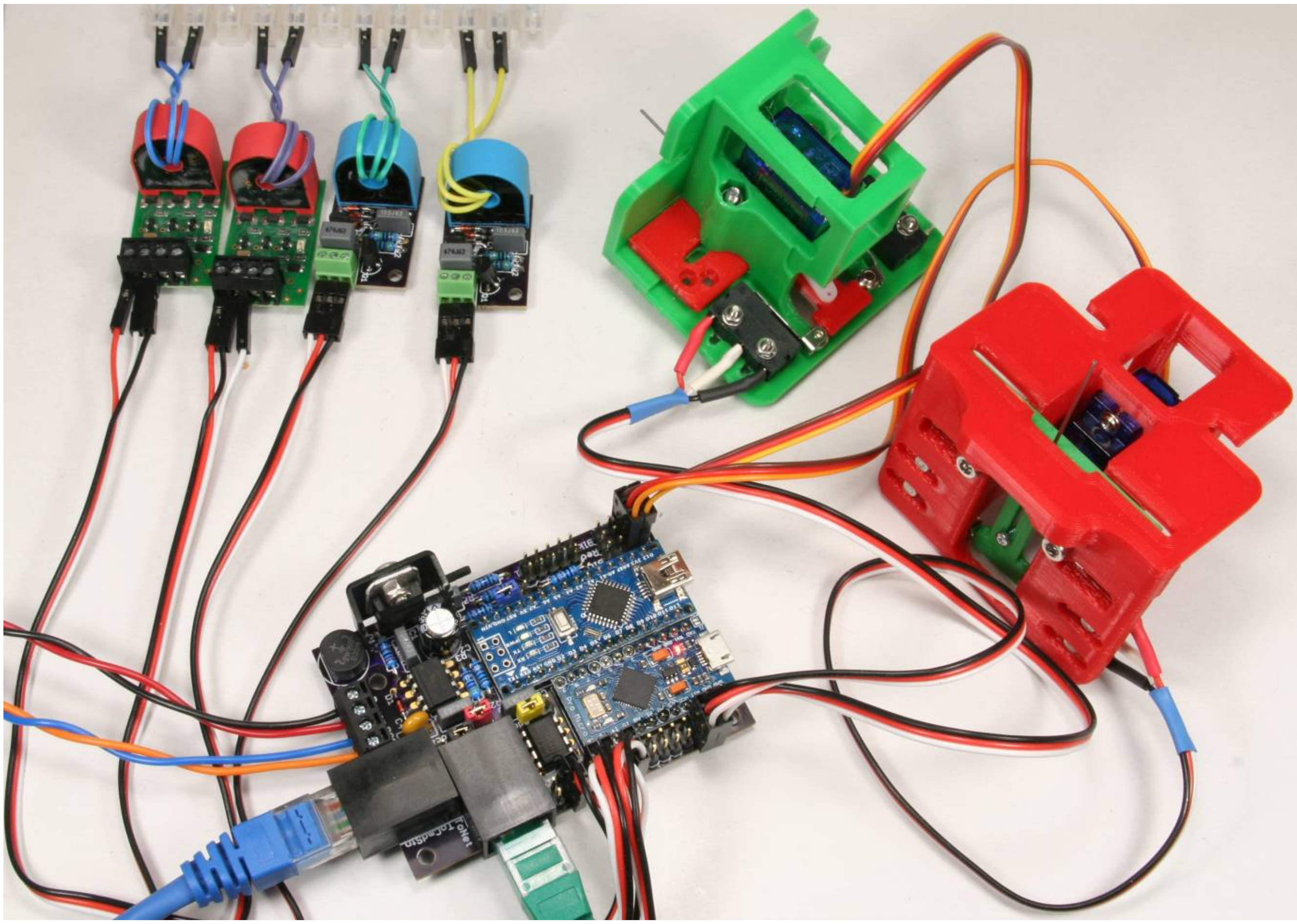
19. Sensor connections to QSDM inputs

The key point to note here is that the ground connection from the sensor is always made by the centre (red) wire. This ensures that, whichever way the 3-way cable connector is fitted to the two pins of the selected Monitor input, the ground connection is always correct, and you are prevented from creating any short-circuits.

For the NCE BD20, the outer two wires of the cable are both connected to the device's Logic output, as shown above in [19], whereas my own AT Block Detector 2 already has the required dual output connections at its terminal block.

Additionally, since microswitches are generally fitted to the servo (or other) turnout motor before the assembly is mounted to the turnout, it is common to find that the sensed state of the turnout is the opposite of what is required. With the cable connection shown in [19] you can fix the problem instantly, simply by reversing the fitment of the cable connector to the relevant pair of pins of the 14x2 header – no resoldering of wires is required.

An example of how the QSDM would be connected in a typical system is shown in [20] below –

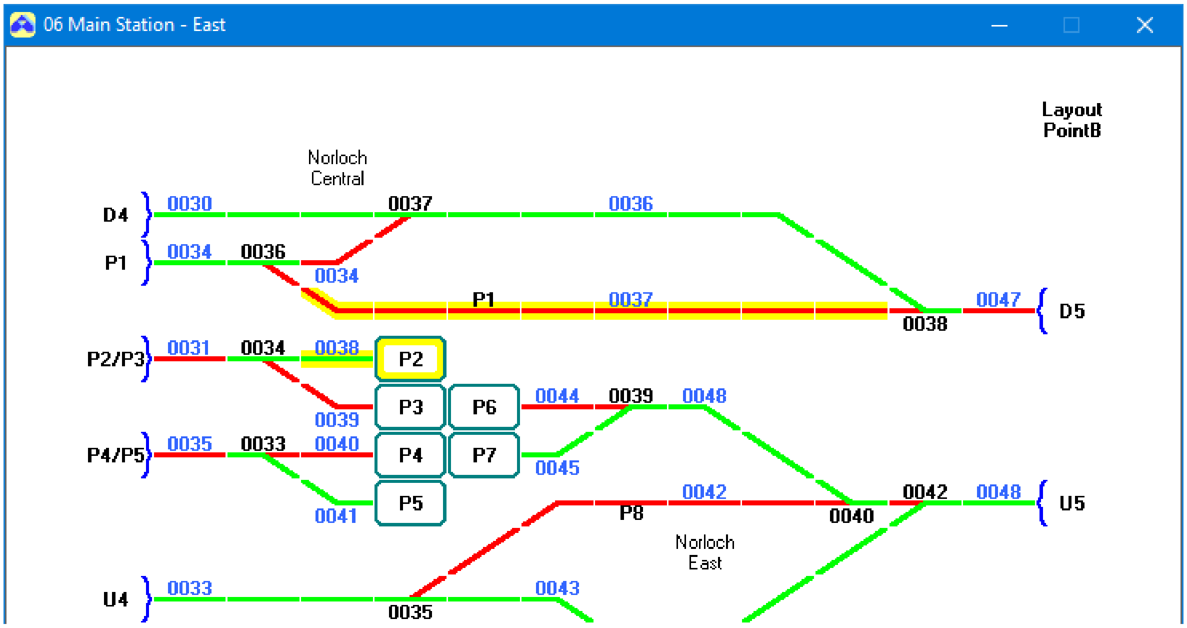


20. Typical connections to the Quad Servo Decoder-Monitor

The two servos attached to the Decoder section are in the (red and green) mounts designed and 3-D printed by my brother Derek. Each servo drives a horizontal slider, with fixings for either a vertical (as shown) or horizontal turnout-actuating rod, and which also operates one or two attached microswitches. One of each pair of microswitches will be connected to a Monitor section input, as can be seen in [20], while the other microswitch will be used to switch the turnout frog polarity.

You can also see that four block-occupancy detectors are connected to other inputs of the Monitor section 14x2 header – two NCE BD20 detectors on the left and two of my own AT Block Detectors.

The state of all QSDM Monitor inputs will then be read at regular intervals from the NCE system by the attached computer application, such as JMRI Panel Pro or my own A-Track software, with the results normally displayed on screen in the form of a layout panel or mimic diagram. An example of one panel of the mimic display produced by A-Track is shown in [20] below –



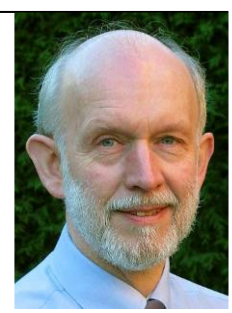
21. An example mimic diagram

Turnout addresses are shown in black text, with the green track indicating the current turnout direction. Track block addresses are shown in blue text, with currently-occupied blocks highlighted in yellow, such as block 0037 and block 0038 leading to terminal platform P2.

Technical Details

For anyone who is interested, the circuit schematics of the QSDM and the subsidiary units can be found on my A-Train Systems website (www.a-train-systems.co.uk/qsdm-download).

Dr Terry Chamberlain



Terry Chamberlain got into model railroading almost by accident in the 1990s when he responded to a request from some modellers in California to build a DCC system based around an Atari personal computer – and he had to build a simple layout to prove that it all worked. Eventually the project evolved into A-Track, a Windows application to provide full computer support for the complete range of NCE DCC systems, with facilities similar to JMRI's Decoder Pro and Panel Pro.

Terry is a professional electronics engineer and spent most of his career in the UK defence industry designing, and managing the development of, large real-time computer systems for the Royal Navy. Now that he is fully retired he is beginning to make progress building the small logging and mining layout he has been planning for many years (after several visits to Colorado) – but keeps getting distracted by new computer and electronics projects for model railroading.