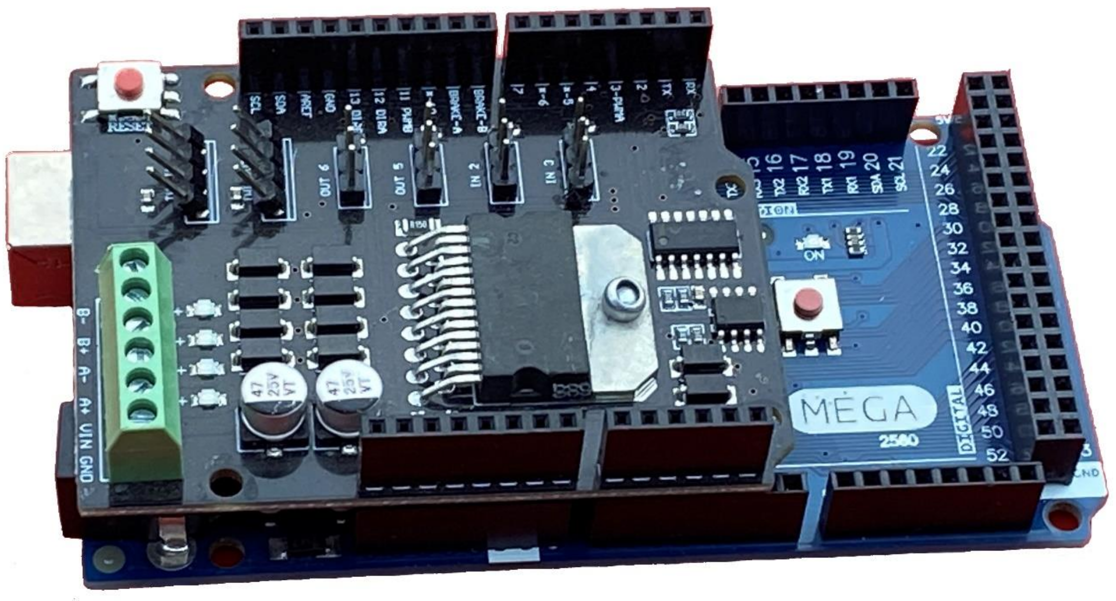
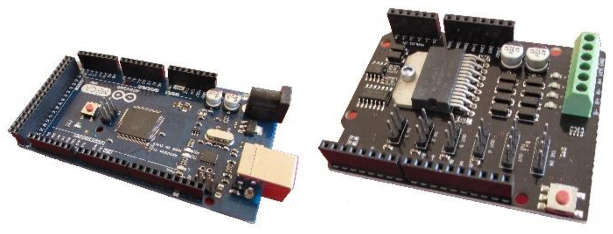
JTElectronics DCC-EX

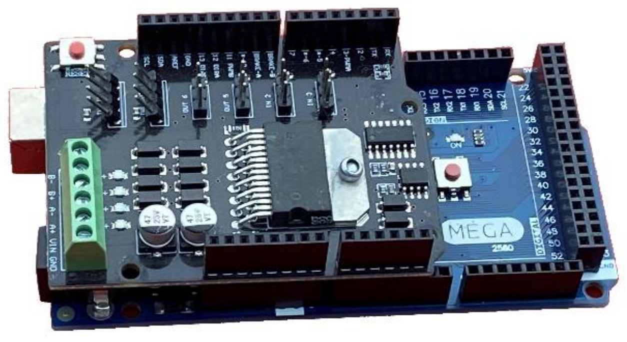
MEGA2560 Controller Module

Model: JTEDCC-MEGA



We have carefully followed various documents and videos to bring you the JTEDCC-MEGA – the "big brother" to our original JTEDCC DCC controller module – installed with the DCC-EX system firmware. This hardware and firmware setup allows full functionality of the impressive DCC-EX system, and you can add stand-alone automation operation.





ARDUINO MEGA-2560 + JTEMOT1 MOTOR DRIVER BOARD = JTEDCC-MEGA

The JTEDCC-MEGA module fully constructed and tested DCC basestation controller for a fraction of the price of commercial units. It comprises of an Arduino based microcontroller shield and a motor controller shield. It connects to your computer via a USB cable and also connects to two tracks – your mainline, and a separate programming track (if required).

Once the JTEDCC-MEGA module is connected to a power supply, your track(s) and your computer you can use the JMRI software program your DCC Decoders (CV's) and to run trains on the track. Also, you can setup the computer as a JMRI network server and control your DCC layout over WiFi from a phone or tablet.

The JTEDCC-MEGA module has been fully rewired, programmed and tested so you don't have to worry about hardware selection, component modifications, Arduino bootloaders and programming – it has all been done for you! All documentation and software for both the DCC-EX controller and the JMRI software is available online for free. You are paying for the hardware boards and their configuration, so the system is ready to use.

The JTEDCC-MEGA module is supplied with a 300mm USB cable for connecting to your computer. You will need to provide a power supply for the upper "Motor Driver" board. Beware that a lot of Arduino boards similar to this one will fail if you apply more than 15 volts due to component substitution. The JTEDCC-MEGA module has been modified to work on up to 20V (absolute maximum) input as most small scale train layouts will work on a DCC supply voltage of 12 or 15 volts DC. A 15VDC 5Amp Toshiba Laptop charger will work well as a power supply for "HO" scale locomotives, wired to the VIN and GND terminals on the 6-Way terminal strip. The DCC-EX firmware in the Arduino board is continually monitoring track current of both tracks and will turn off power to the tracks when a current limit safety threshold level is reached. This is to help prevent a short on the track from damaging the Motor Driver board or your locomotive.

Sure, you might find similar items cheaper from other sources but the items I sell have all relevant configuration and firmware installed, contain the most reliable hardware components and modifications, and are fully tested!

NOTE: If you are going to power the JTEDCC-MEGA module from a DC power source connected to the VIN and GND terminals, AND ALSO POWERING THE JTEDCC-MEGA WITH 5 VOLTS VIA THE USB SOCKET, you must cut the VIN link on the bottom of the motor driver shield (or bend/cut the Vin pin). Failure to do so will result in the 5-volt regulator failing on the Mega2560 shield and therefore probable failure of the whole system. See this link for instructions on cutting the Vin trace link: https://dcc-ex.com/ex-commandstation/get-started/assembly.html

WHAT YOU DON'T GET...

I can only provide very basic help on getting the JTEDCC-MEGA module wired up and working. I am no expert on track layouts or operating the JMRI software so if you are having issues try some google searches and you will find the solution.

This JTEDCC-MEGA module is supplied as preassembled Arduino compatible shields like in the photos. It will be damaged by shorting out the pins, connections, or components, or by either of the shields (boards) coming into unusual contact with external voltages or metal/conductive objects. We recommend using the JTECASE1 case to house your JTEDCC-MEGA module to protect it.

The JTEDCC-MEGA module will also be damaged by incorrectly connecting power to the wrong terminals of the motor driver shield. Make sure you connect your DC power supply to the VIN and GND terminals ONLY. Make sure you connect ONLY your tracks to the A-, A+ and B-, B+ terminals and that no other power supply or train controller is connected to the tracks.

PLEASE REFER TO THE WIRING DIAGRAMS IN THIS DATASHEET FOR CORRECT WIRING INFORMATION

Like most electronic equipment the JTEDCC-MEGA circuit boards contain static sensitive devices and may be damaged by high voltages present during electrostatic discharge. Avoid electrostatic discharge by handling the JTEDCC-MEGA module as little as possible. To prevent damage, we suggest you put the JTEDCC-MEGA module into an enclosure or locate it somewhere protected from contact with random external objects...

SOFTWARE LINKS

JMRI Computer Control Software	http://jmri.org
DCC-EX BaseStation Documentation	https://dcc-ex.com/
Wireless Throttles with JMRI	http://www.nmra.org.au/Clinics/WiThrottle%20Clinic%202012%20handout.pdf
General DCC Information	https://dccwiki.com/

MODULE SPECIFICATIONS

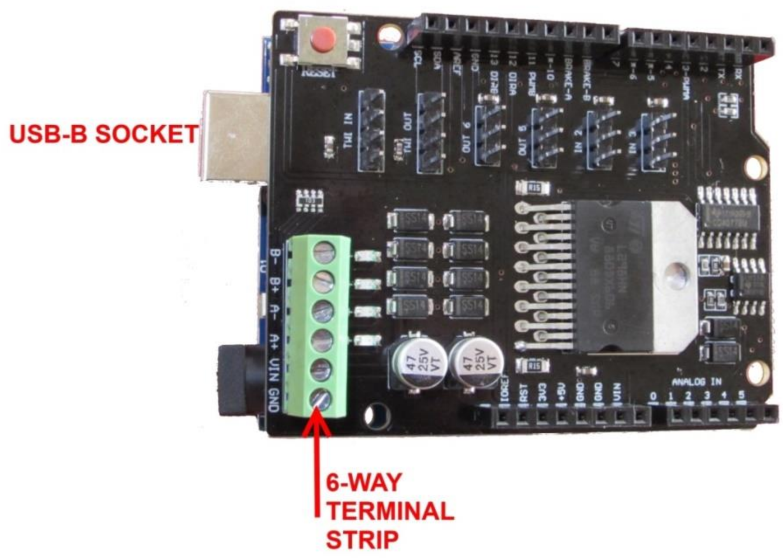
DC PPOWER SUPPLY	9 to 20 Volts DC via "Vin" and "Gnd" connections
TRACK CURRENT	Board can handle 2 Amps per channel/track, but limited to about 1.6 Amps by DCC-EX software

NOTE: If you are going to power the JTEDCC-MEGA module from a DC power source connected to the VIN and GND terminals, AND ALSO POWERING THE JTEDCC-MEGA WITH 5 VOLTS VIA THE USB SOCKET, you must cut the VIN link on the bottom of the motor driver shield (or bend/cut the Vin pin). Failure to do so will result in the 5-volt regulator failing on the Mega2560 shield and therefore probable failure of the whole system. See this link for instructions on cutting the Vin trace link: https://dcc-ex.com/ex-commandstation/get-started/assembly.html

LENGTH	126mm
WIDTH	77mm
HEIGHT	38mm

This document is updated from time to time as new information becomes available – usually due to people asking relevant questions regarding usage or configuration. The "Document Updated" date in the bottom-right corner of each page shows what document date you have. The latest version of this datasheet document can be downloaded from http://www.jtelectronics.co.nz/products/documents/ or Google "JTEDCC-MEGA"…

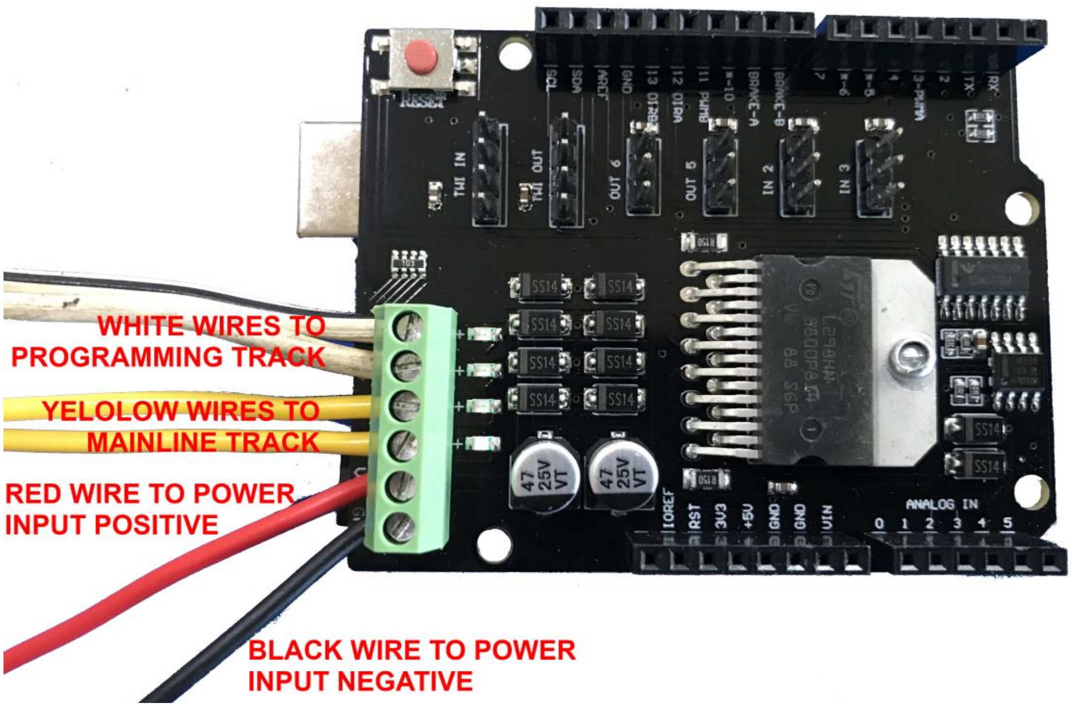
JTEDCC-MEGA MOTOR DRIVER MODULE (JTEMOT1) - TOP VIEW



TRACK 6-WAY TERMINAL STRIP WIRING CONNECTIONS

B-	Programming Track (or second mainline track)	
B+	Programming Track (or second mainline track)	
A-	Mainline Track	
A+	Mainline Track	
VIN	Power Input Positive Connection	9 to 20VDC
GND	Power Input Ground/Negative Connection	

NOTE: 20VDC is the maximum the JTEDCC-MEGA board will handle. Your system will use a lower voltage. Please check the "Power Supplies" section below for power supply voltage recommendations!!


WHITE WIRES TO PROGRAMMING TRACK
YELOLOW WIRES TO MAINLINE TRACK
RED WIRE TO POWER INPUT POSITIVE
BLACK WIRE TO POWER INPUT NEGATIVE

Automation example using built-in EX-RAIL

You can use the JTEDCC-MEGA controller module and the free DCC-EX **EX-CommandStation** firmware to create an automation file and upload it to the JTEDCC-MEGA Arduino board. DCC-EX keep changing the look and feel of their "EX-Installer" automated firmware installer app so these instructions may change… Official instructions are at: https://dcc-ex.com/ex-commandstation/get-started/installer.html

1. Before attempting automation, make sure you can drive locomotives using a throttle in the JMRI software, so you know your setup and wiring is OK. Make sure you "turn track power ON" in your throttle software before trying to drive the locomotive!
2. Close JMRI software before proceeding to release serial port connection to the JTEDCC-MEGA module.
3. Make sure your computer is connected to the internet so it can download the required files
4. Plug the JTEDCC-MEGA module into your computers USB port
5. Download "EX-Installer" app from https://dcc-ex.com/download/ex-commandstation.html#ex-installer
6. Open the downloaded EX-Installer app file to execute it
7. Click on "Manage Arduino CLI"
8. Click on "Install Arduino CLI" if you haven't done this before
9. Click on "Select Your Device"
10. Select the detected "Arduino Mega 2560" device on the detected serial port
11. Click "Select Product To Install"
12. Click "EX-CommandStation" product
13. Click "Configure EX-CommandStation"
14. Select "STANDARD_MOTOR_SHIELD" motor driver board
15. Turn on "Advanced Config" option switch
16. Click "Advanced Config" button
17. Paste in the EXRAIL automation script below into the "myAutomation.h" file area on the right (CTRL-V for Paste)
18. Click "Compile And Upload"
19. Click "Load" and DCC-EX will be compiled and uploaded into the JTEDCC-MEGA module
20. Click "View Device Monitor" button and you should see debug output from the DCC-EX firmware
21. Close the "Device Monitor" window
22. Close the "EX-Installer" app

Now that the myAutomation.h file has been added to the firmware on the JTEDCC-MEGA module, your locomotive will be automatically sent the appropriate DCC commands to shuttle back and forth when it is detected by the sensors at each end of the required track. Just run the EX-Installer app again to edit and upload any required changes to the myAutomation.h EXRAIL script file eg. locomotive address, speeds, or different delays etc. EX-RAIL does a lot more than just locomotive automation so google "**EX-RAIL Detailed Reference**" and **EX-RAIL Command List**" for the required information.

The EX-CommandStation code files were extracted and stored in folder C:\Users\<You>\ex-installer\CommandStation-EX if you want to manually edit the myAutomation.h file using Notepad to change speed settings etc. Just run the EX-Installer app to upload the full DCC-EX EX-CommandStation code again, including your new changes.

Don't forget to wire the locomotive sensors to the JTEDCC-MEGA module – to digital pins "2" and "4" in the example "myAutomation.h" file below but you can use any available digital input pins and update the automation file accordingly. Adding the "JTESEN1" board makes it easy to connect a few sensors to the JTEDCC-MEGA module.

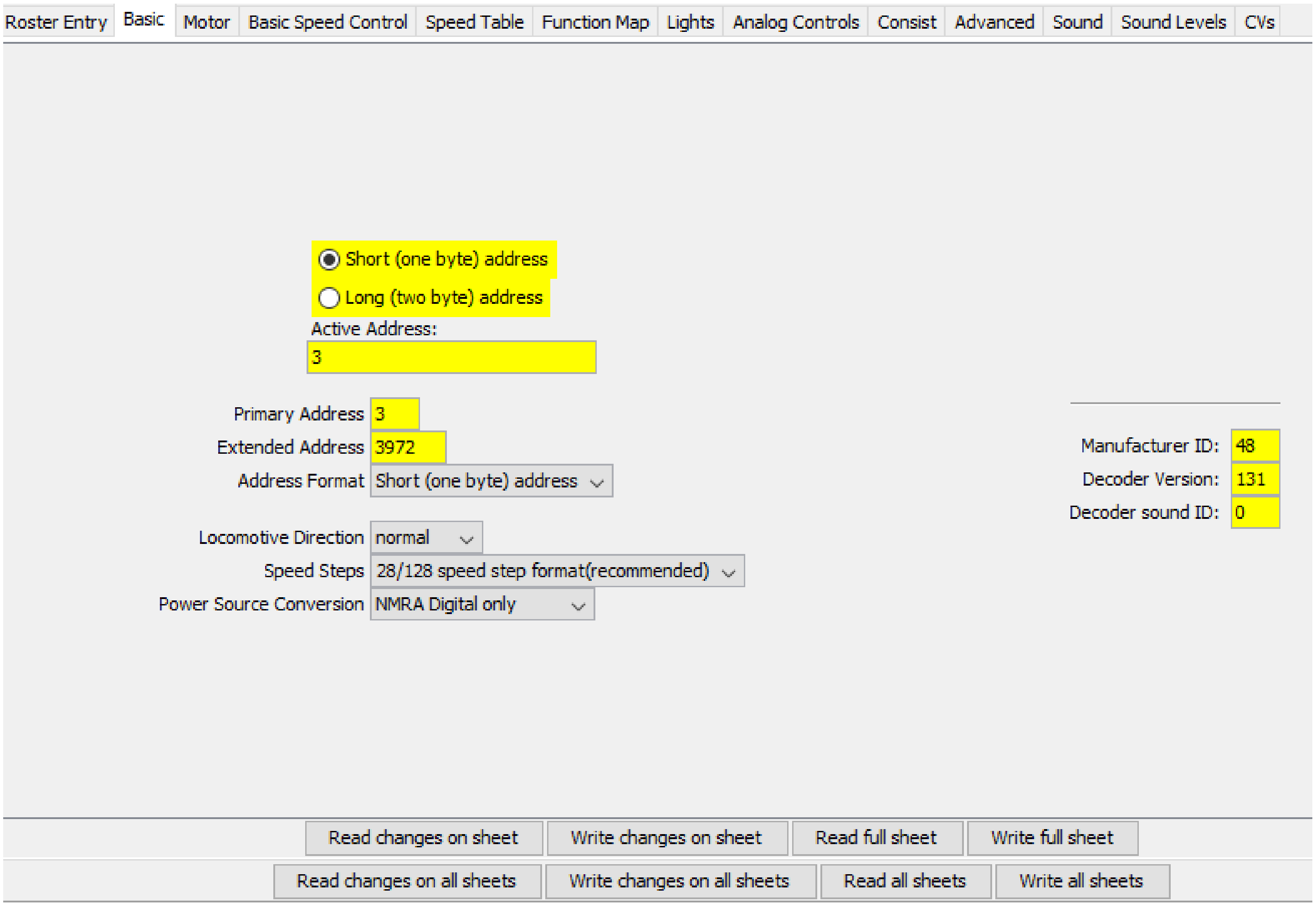
Below is an example “myAutomation.h” EXRAIL automation script file. It assumes the locomotive DCC address is “3” and the sensors are connected to Arduino input pins “2” and “4” as detailed in the comments. The EXRAIL script file is written similar to the “C++” programming language where all text on a line following a double-backslash “\\” will be ignored so the \\ is used to add comments to make it easier for humans to understand, and also used to “comment out” a particular line so it won’t be used.

```
EXRAIL                      //myAutomation.h file must start with the EXRAIL instruction.

//Configuration for locomotive #1
ALIAS(LOCO_1_ADDRESS, 3)        //DCC Address “3” of locomotive #1
ALIAS(LOCO_1_FORWARD_SPEED, 40) //Forward speed “40” 0->127
ALIAS(LOCO_1_REVERSE_SPEED, 35) //Reverse speed “35” 0->127
ALIAS(LOCO_1_SENSOR_1, 2)       //Sensor “2” at end of "forward" direction
ALIAS(LOCO_1_SENSOR_2, 4)       //Sensor “4” at end of "reverse" direction

//Send locomotive on it’s way
ALIAS(SHUTTLE_1_SEQUENCE, 1)
SETLOCO(9999)                   //select locomotive 9999 – a dummy locomotive address
SPEED(0)                        //set the speed to 0.  This will turn the track power ON
SENDLOCO(LOCO_1_ADDRESS, SHUTTLE_1_SEQUENCE)     //send LOCO_1 off along SHUTTLE_1 route
DONE                          //This just ends the startup thread 0,leaving 2 others running

//Define the SHUTTLE_1_SEQUENCE of commands
SEQUENCE(SHUTTLE_1_SEQUENCE)
FON(3)                          //Set Loco Function 3, Horn on
DELAY(1000)                     //Wait 1 second
FOFF(3)                         //Horn off
FWD(LOCO_1_FORWARD_SPEED)       //Move forward
AT(LOCO_1_SENSOR_1)             //Until we hit LOCO_1 SENSOR_1
STOP                            //Then stop
DELAYRANDOM(5000,10000)         //Wait for a random delay between 5 and 10 seconds
FON(2)                          //Ring bell
REV(LOCO_1_REVERSE_SPEED)       //Reverse
AT(LOCO_1_SENSOR_2)             //Until we get to LOCO_1 SENSOR_2
STOP                            //Then stop
FOFF(2)                         //Bell off
DELAYRANDOM(5000,10000)         //Wait for a random delay between 5 and 10 seconds
FOLLOW(SHUTTLE_1_SEQUENCE)      //And follow SHUTTLE_1 command sequence again

ENDEXRAIL                       // marks the end of the EXRAIL automation program
```

Programming Decoder CV's

You can use the JTEDCC-MEGA controller and the free JMRI software to easily program the CV values in your decoder. It is very simple to use – just open your locomotive in the roster, select the tab containing the required CV settings, adjust the setting and click "Write changes on sheet" button

Below is a screenshot of the "Basic" tab where you can set the decoder's DCC address (either Long or Short address)



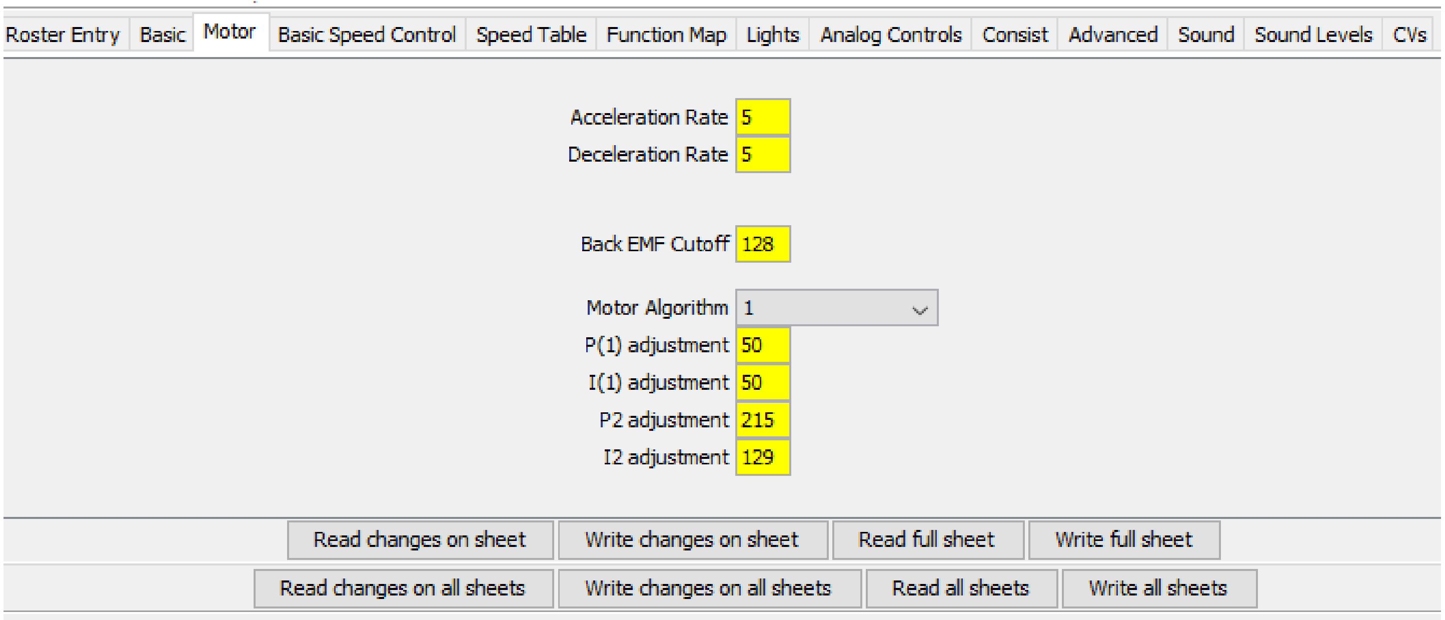
[✓ Short (one byte) address] Long (two byte) address

Active Address: 3

Primary Address	3
Extended Address	3972
Address Format	Short (one byte) address
Locomotive Direction	normal
Speed Steps	28/128 speed step format(recommended)
Power Source Conversion	NMRA Digital only

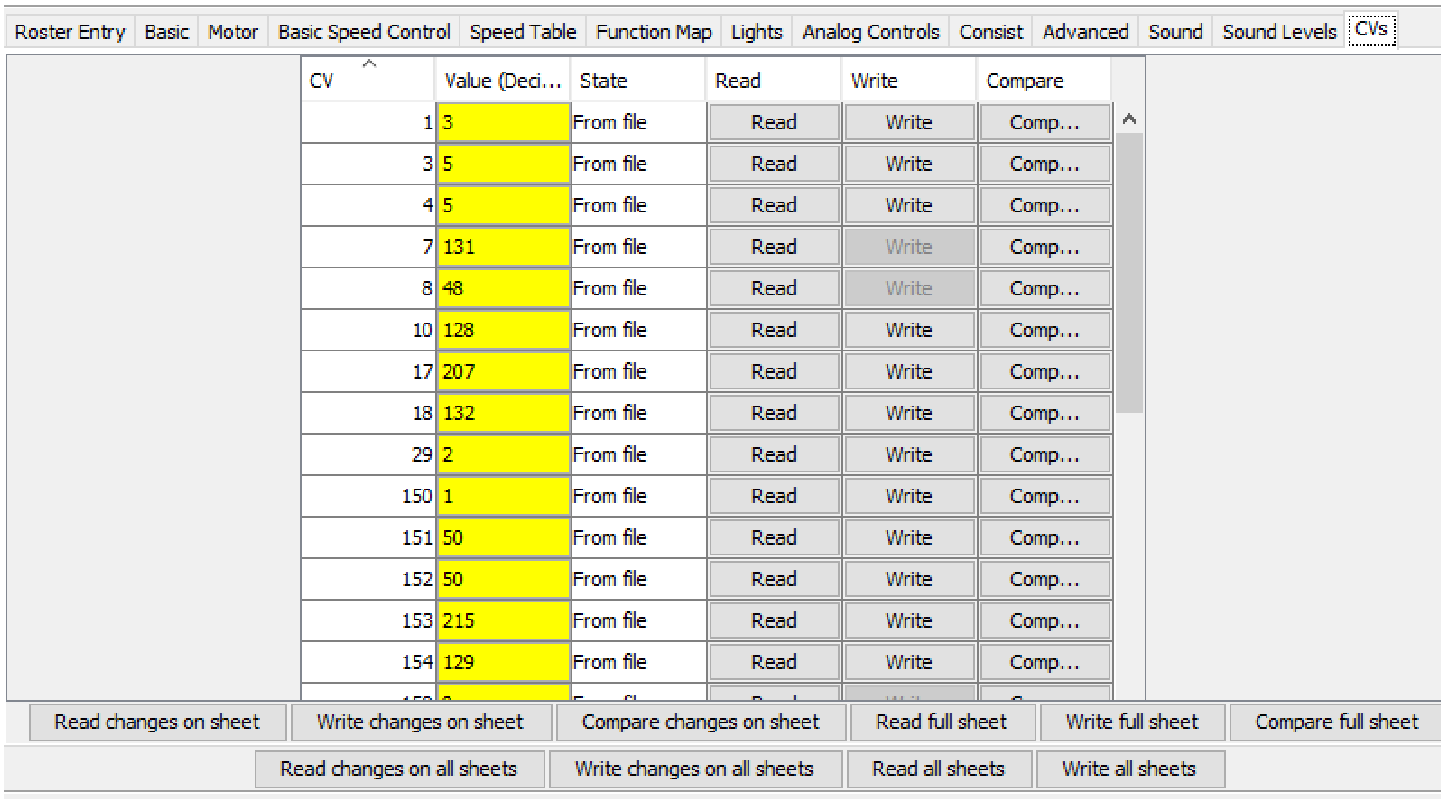
Manufacturer ID:	48
Decoder Version:	131
Decoder sound ID:	0

Below is a screenshot of the "Motor" tab settings for this particular decoder



Acceleration Rate	5
Deceleration Rate	5
Back EMF Cutoff	128
Motor Algorithm	1
P(1) adjustment	50
I(1) adjustment	50
P2 adjustment	215
I2 adjustment	129

The CV's tab will show you the actual CV numbers read from the decoder:



CV	Value (Deci…	State	Read	Write	Compare
1	3	From file	Read	Write	Comp…
3	5	From file	Read	Write	Comp…
4	5	From file	Read	Write	Comp…
7	131	From file	Read	Write	Comp…
8	48	From file	Read	Write	Comp…
10	128	From file	Read	Write	Comp…
17	207	From file	Read	Write	Comp…
18	132	From file	Read	Write	Comp…
29	2	From file	Read	Write	Comp…
150	1	From file	Read	Write	Comp…
151	50	From file	Read	Write	Comp…
152	50	From file	Read	Write	Comp…
153	215	From file	Read	Write	Comp…
154	129	From file	Read	Write	Comp…

Power Supplies

Being a DIY controller solution, there are many possibilities to use an old DC power supply you have.
Some information on DCC power requirements is here: https://dccwiki.com/Power_supply which recommends a maximum power supply voltage of 15V for HO scale and 12V for N scale.

NOTE: You must power the JTEDCC-MEGA module using the VIN and GND terminals on the "Track 6-way Terminal Strip". You should not use the DC power socket (beside the USB socket) to supply power. You will need to cut the plug off the power supplies shown below, use a multimeter to determine exactly what wire is Positive and what wire is Negative, and connect the wires to the appropriate VIN and GND terminals on the "Track 6-way Terminal Strip".

You can try hooking up a 12V DC 1Amp power supply from an old internet modem (like in the picture to the right) to get you started with running a single locomotive. This type of power supply should be fine to run a few HO or N scale locomotives.

For HO scale use I have seen good results with using an old Toshiba laptop charger like in the picture to the right, and these are usually rated for 15V DC at 5Amps. You will need to carefully check the label to make sure it's rated at 15V DC as some laptop chargers are 18V or 19V DC and this will be too high! ie. Will damage the DCC Decoder and/or motor in the locomotive!

Jaycar sell a 15V DC 2Amp power supply (Cat. No MP-3492) which will also be suitable for a couple of HO scale locomotives with sound but at $36.90 it's a bit pricey…

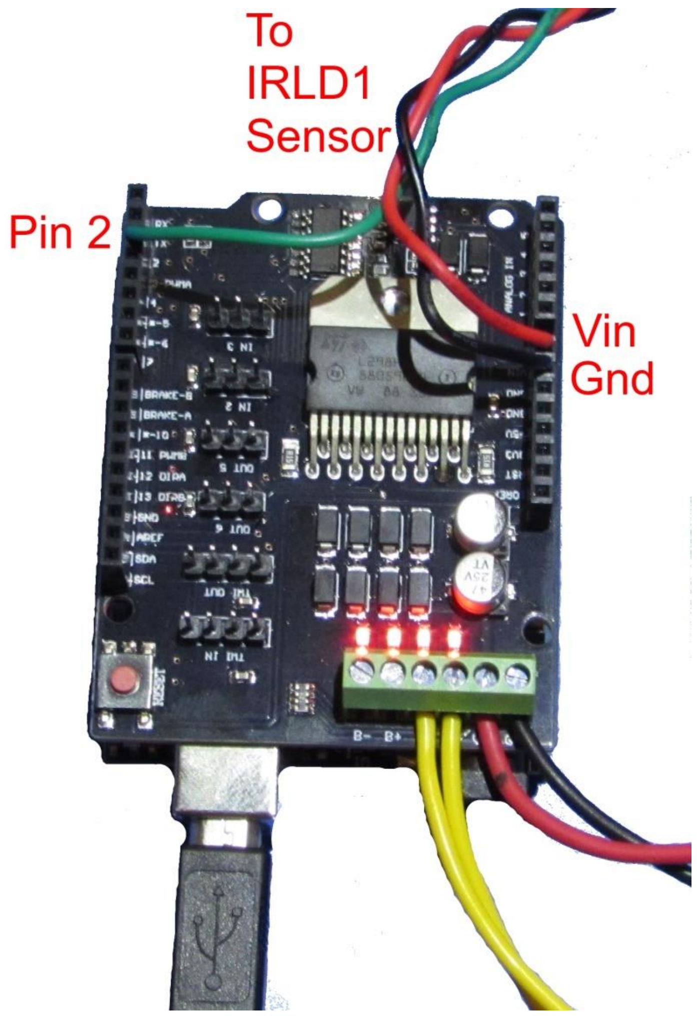
Additional Hardware

The DCC-EX firmware also supports additional hardware connections to allow some track automation. This is configurable in the JMRI software by choosing menu **DCC++/Configure BaseStation** and configuring:

Sensors	as inputs from the unused Arduino Uno pins
Turnouts	as DCC enabled turnouts
Outputs	as digital outputs from the unused Arduino Uno pins

Be careful to not apply more than 5volts DC to the Arduino board input pins! It is advisable to only apply "Ground" signals to the Arduino pins to avoid the possibility of a high voltage being applied.

The **JTElectronics IRLD1 Infra-Red Locomotive Detector** from our Products (http://www.jtelectronics.co.nz/products) can easily be connected to the JTEDCC board as a digital input sensor. Shown here powered by **Vin** and **Gnd** pins, and also connected with the green wire to the Arduino "**Pin 2**" input.

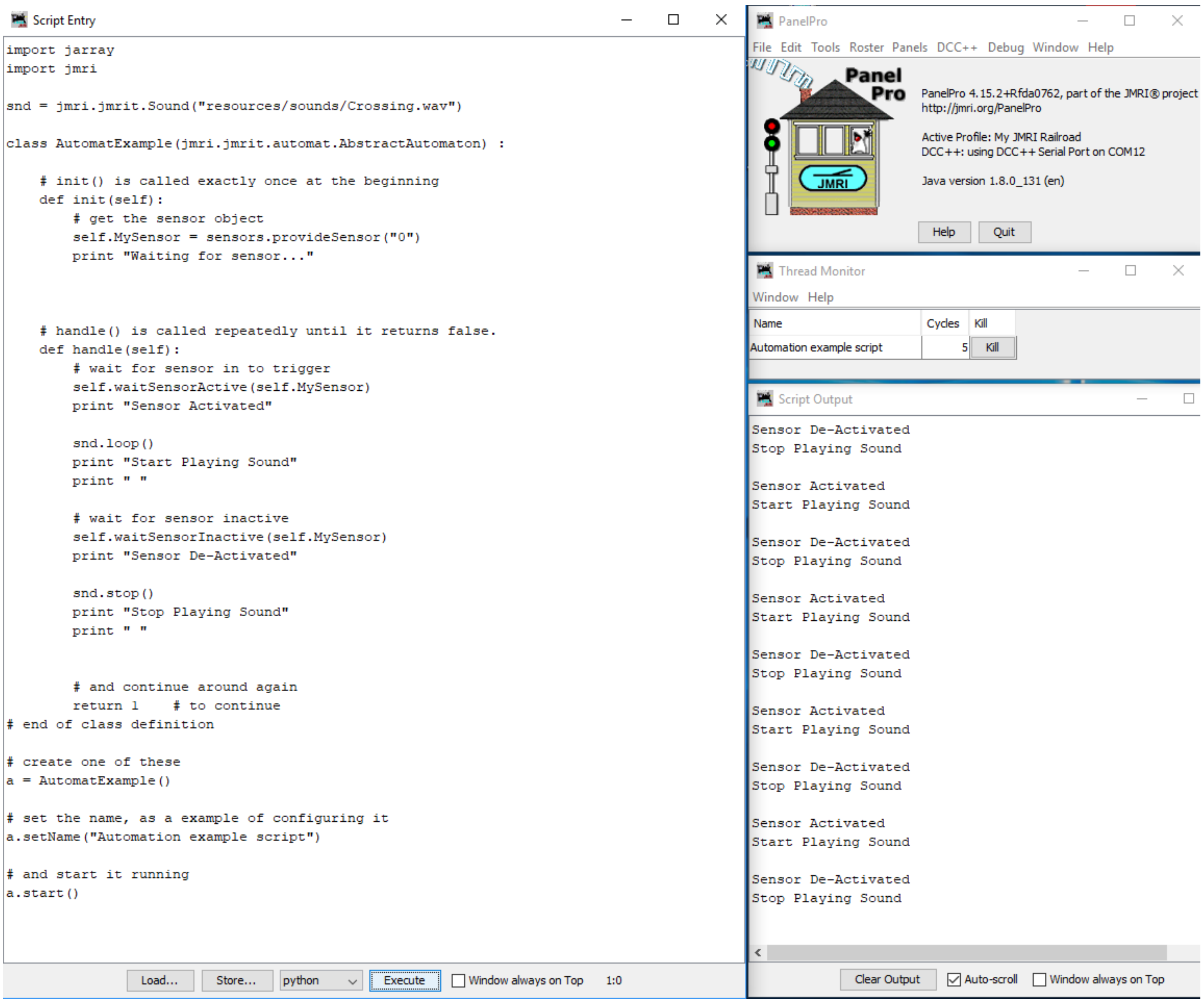


JMRI Scripting for Automation

Below is a simple JMRI/python script to play a sound while a sensor is activated, and stop the sound when the sensor is deactivated. In this example you could put two IRLD1 sensors on either side of a crossing and the "crossing bells" sound would play when a train approaches from either side and continues to play until the train is free from both sensors… The sensors would be connected to the same Arduino input pin (like in the "Additional Hardware" section above) and defined in JRMI as "Sensor 0"

```
import jarray
import jmri

snd = jmri.jmrit.Sound("resources/sounds/Crossing.wav")

class AutomatExample(jmri.jmrit.automat.AbstractAutomaton) :

    # init() is called exactly once at the beginning
    def init(self):
        # get the sensor object
        self.MySensor = sensors.provideSensor("0")
        print "Waiting for sensor..."


    # handle() is called repeatedly until it returns false.
    def handle(self):
        # wait for sensor in to trigger
        self.waitSensorActive(self.MySensor)
        print "Sensor Activated"

        snd.loop()
        print "Start Playing Sound"
        print " "

        # wait for sensor inactive
        self.waitSensorInactive(self.MySensor)
        print "Sensor De-Activated"

        snd.stop()
        print "Stop Playing Sound"
        print " "


        # and continue around again
        return 1    # to continue
# end of class definition

# create one of these
a = AutomatExample()

# set the name, as a example of configuring it
a.setName("Automation example script")

# and start it running
a.start()
```

Below are screenshots of JMRI PanelPro windows when executing the script with the "Script Output" window showing what I am seeing and hearing as the locomotive is detected by the IRLD1 sensor.



```
import jarray
import jmri

snd = jmri.jmrit.Sound("resources/sounds/Crossing.wav")

class AutomatExample(jmri.jmrit.automat.AbstractAutomaton) :

    # init() is called exactly once at the beginning
    def init(self):
        # get the sensor object
        self.MySensor = sensors.provideSensor("0")
        print "Waiting for sensor..."



    # handle() is called repeatedly until it returns false.
    def handle(self):
        # wait for sensor in to trigger
        self.waitSensorActive(self.MySensor)
        print "Sensor Activated"

        snd.loop()
        print "Start Playing Sound"
        print " "

        # wait for sensor inactive
        self.waitSensorInactive(self.MySensor)
        print "Sensor De-Activated"

        snd.stop()
        print "Stop Playing Sound"
        print " "


        # and continue around again
        return 1    # to continue
# end of class definition

# create one of these
a = AutomatExample()

# set the name, as a example of configuring it
a.setName("Automation example script")

# and start it running
a.start()
```

PanelPro 4.15.2+Rfda0762, part of the JMRI® project
http://jmri.org/PanelPro

Active Profile: My JMRI Railroad
DCC++: using DCC++ Serial Port on COM12

Java version 1.8.0_131 (en)

Name	Cycles	Kill
Automation example script	5	Kill

```
Sensor De-Activated
Stop Playing Sound

Sensor Activated
Start Playing Sound

Sensor De-Activated
Stop Playing Sound

Sensor Activated
Start Playing Sound

Sensor De-Activated
Stop Playing Sound

Sensor Activated
Start Playing Sound

Sensor De-Activated
Stop Playing Sound

Sensor Activated
Start Playing Sound

Sensor De-Activated
Stop Playing Sound
```