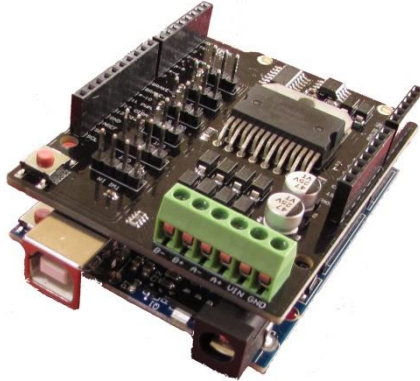


JTElectronics DCC++ JMRI

Controller Module

Model: JTEDCC



We have carefully followed various documents and videos to bring you the JTEDCC - a fully constructed and tested DCC basestation controller for a fraction of the price of commercial units. The JTEDCC module 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.

Once the JTEDCC module is connected to your track(s) and your computer you can use the JMRI software program your DCC Decoders 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 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 DccPlusPlus 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 module is supplied with a 300mm USB cable for connecting to your computer. This powers the control circuitry on the bottom "Arduino" board. 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 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 15 or 16 volts DC. A 15VDC 5Amp Toshiba Laptop charger will work well as a power supply, wired to the VIN and GND terminals on the 6-Way terminal strip.

The DccPlusPlus in the Arduino board firmware is continually monitoring track current of both tracks and will turn off power to the tracks when a threshold level is reached. This is to prevent a short on the track from destroying the Motor Driver board. We have increased the current limit threshold to approximately 1600mA which should be plenty to run a couple of N-Scale locos with sound... The JMRI software will also give an indication of the mainline track current and the current limit trip threshold will be around a reading of "59%" which corresponds to approximately 1600mA.

What you don't get... I can only provide very basic help on getting the JTEDCC 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 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.

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

SOFTWARE LINKS

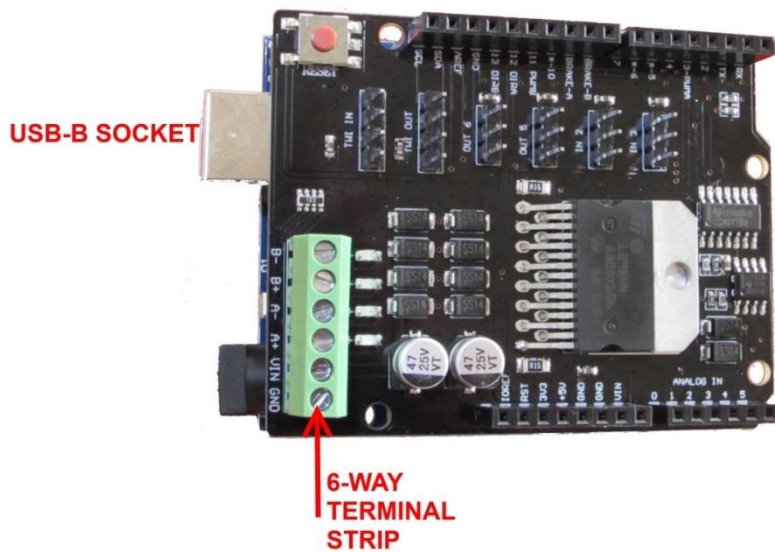
JMRI Computer Control Software	http://jmri.org
DccPlusPlus BaseStation	https://github.com/DccPlusPlus/BaseStation/wiki/What-is-DCC--Plus-Plus
Wireless Throttles with JMRI	http://www.nmra.org.au/Clinics/WiThrottle%20Clinic%202012%20handout.pdf
General DCC Information	https://dccwiki.com/

MODULE DIMENSIONS (APPROX)

LENGTH	77mm
WIDTH	53mm
HEIGHT	24mm

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” ...

JTEDCC MODULE - TOP VIEW



6-WAY TERMINAL STRIP WIRING CONNECTIONS

B- Programming Track

B+ Programming Track

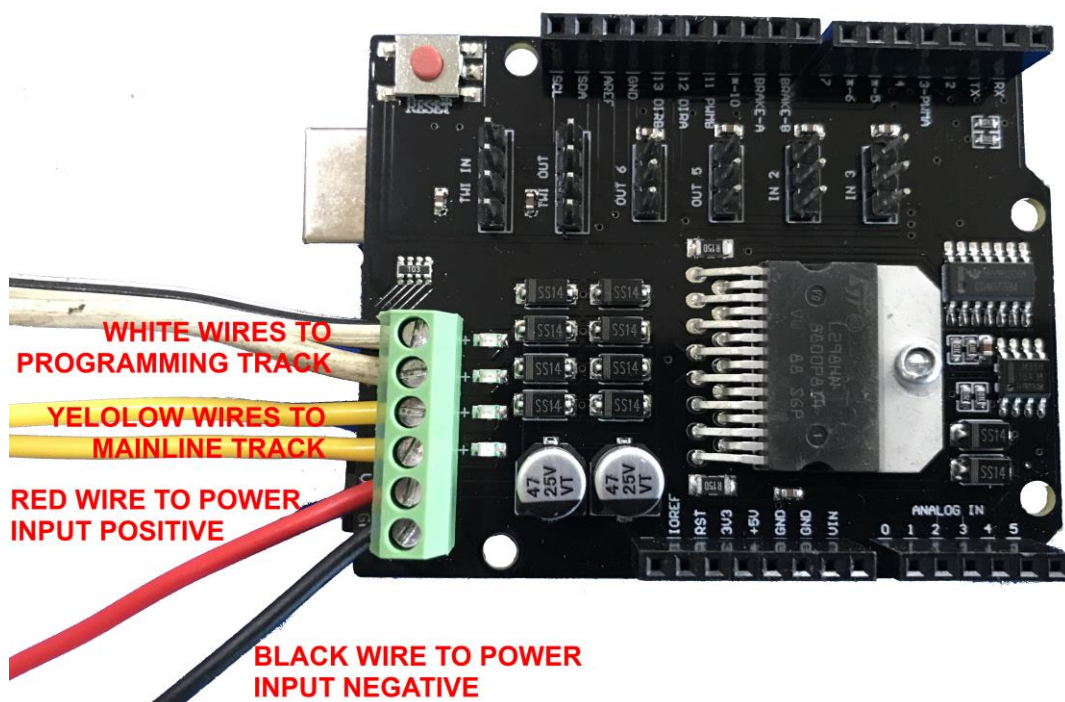
A- Mainline Track

A+ Mainline Track

VIN Power Input Positive Connection 12 to 20VDC

GND Power Input Ground/Negative Connection

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



Power Supplies

Being a DIY controller solution, there's 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.

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 HO or N scale locomotive.



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...



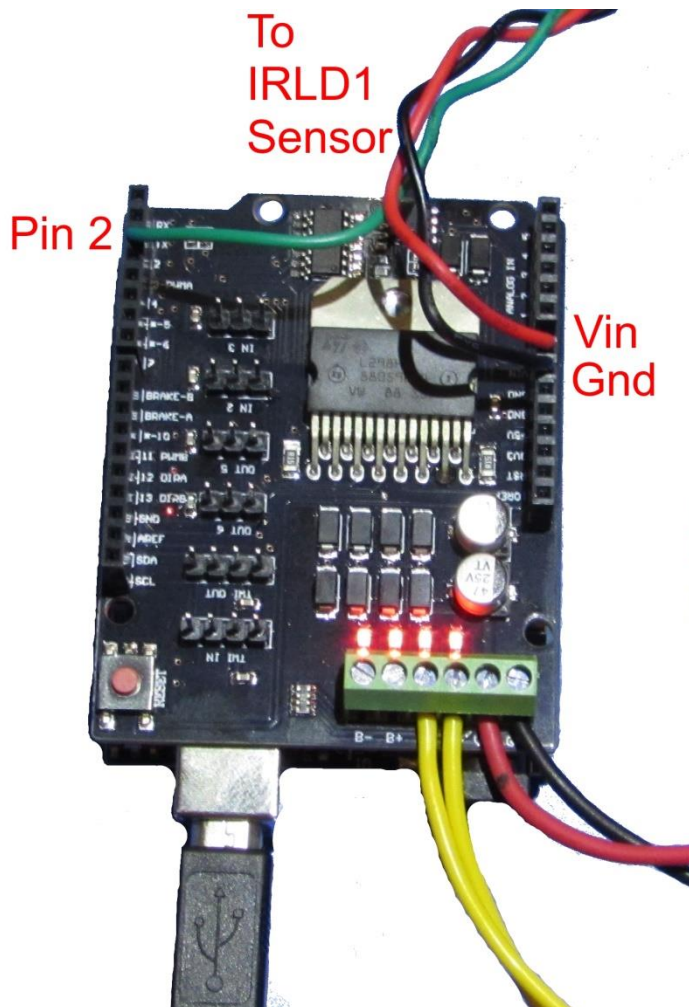
Additional Hardware

The DccPlusPlus 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 JMRI 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.waitForSensorActive(self.MySensor)
        print "Sensor Activated"

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

        # wait for sensor inactive
        self.waitForSensorInactive(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.

