**Software Platform Support: DCC++ & DCC++EX**

* [Hardware](https://www.jmri.org/help/en/html/hardware/dccpp/index.shtml#hardware)
* [Limitations](https://www.jmri.org/help/en/html/hardware/dccpp/index.shtml#Limitations)
* [Connecting](https://www.jmri.org/help/en/html/hardware/dccpp/index.shtml#connecting)
* [JMRI Tools](https://www.jmri.org/help/en/html/hardware/dccpp/index.shtml#tools)
* [Documentation](https://www.jmri.org/help/en/html/hardware/dccpp/index.shtml#documentation)

**Overview**

DCC++ is an Open Source Arduino Microcontroller based [Do-it-Yourself](https://www.jmri.org/help/en/html/hardware/dccpp/index.shtml#documentation) Affordable DCC Command Station.  
Both products are written in C++ and utilizing the Arduino IDE and Platform & MS Visual Studio editors.

First, a Special thanks to Gregg E. Berman who had the original vision & idea for a model railroad Command Station using an Arduino Uno or Mega microcontroller with a Arduino Motor Shield, along with Mark Underwood (Twindad) who was instrumental in developing the first JMRI interfaces to DCC++.   
  
DCC++ Base Station was first released in August 2015 and was enhanced through 2017 by various github users.   
In the Spring 2020 Fred Decker (FlightRisk) headed up a group of new and existing DCC++ developers to Enhance the DCC++ project. Our first attempt to build up and out the Vision on the original code left us with a technological software decision of stay with the Original or build a New design with updated code structures. We opted to fix up and keep Gregg’s Original DCC++ 1.2.1+ Base Station code and Rename it to DCC++ Base Station ‘*Classic’* this version can still can be downloaded from the new dcc-ex website and is named DCC++ Base Station Classic.  
Note: For a look back at history and where it began, see Gregg’s Original DCC+++ build videos at the bottom of this page.  
  
In summer 2020 we began New development on a DCC++ EX ‘Extended’ Command Station.  
Our First General release was DCC++EX 3.0 Command Station in November 2020 and is now continuously being enhanced and supported at DCC-EX.com under Command Station Downloads.   
The current Development release can be found at DCC-EX.com. under ‘CommandStation-EX GitHub’.   
Our thanks also go to Steve Todd (MSteveTodd) who was instrumental in developing the Engine Driver App and is our go to for JMRI interfaces and new Engine Driver enhancements with DCC++ EX.

Feature in Both Versions;

* Compliant with NMRA DCC standards
* 2-byte and 4-byte locomotive addressing
* Simultaneous control of multiple DCC locomotives
* 128-step speed control
* Control all cab functions F0-F28
* Activate/de-activate all accessory function addresses 0-2048
* Programming on the Programming Track
  + Write configuration variable bytes
  + Set/clear specific configuration variable bits
  + Read configuration variable bytes
* Programming on the Main Operations Track
  + Write configuration variable bytes
  + Set/clear specific configuration variable bits
* Control DCC turnouts and sensors
* Directly access and control Arduino IO pins for accessory functions and sensors
* Computably with JMRI Decoder Pro, Panel Pro and other offerings

What’s New & Improved with DCC++ EX Command Station (CS)? [New DCC++EX Command Station](https://dcc-ex.com/about/rewrite.html?highlight=whats%20new)    
  
 A few of the many new designs and enhancements include;

* WiFi Ready & Built into the design via either a WiFi Shield or on a Mega2560 + ESP8266 WiFi Onboard chip
* DCC Signal Generation code “Waveform Generator” redesigned & Improved DCC Signal
* Application Program Interface API Enhancements for internal and third party App Interfacing
* Faster Cleaner rewritten DigitalRead() & DigitalWrite() Library routines for programming decoders
* Completely re-wrote current sense and ACK detect routines for better protection & programing accuracy of decoders.
* User defined ACK Min 🡨pulse🡪 Max detection values to improve reading of CV’s on older DCC decoders i.e., QSI.
* Upgraded Current reading to use milliamp for cleaner easier to gage circuit cut off protection
* User Customizable mySetup.h File that’s your Unique Command Station’s Startup configuration
* JMRI integration and enhancements with new Command & Diagnostic Functions
* And no jumper wires on the motor shield, it’s done internally now with EX version

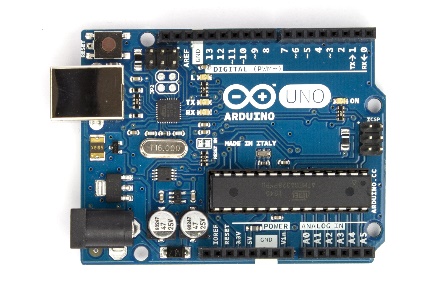
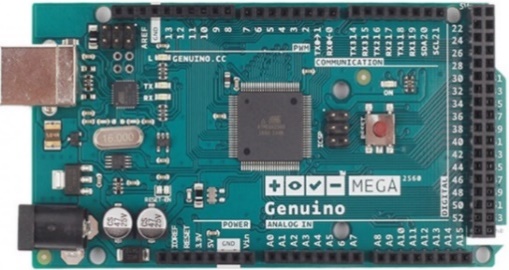
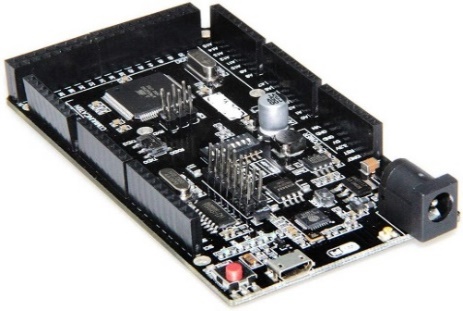
Note: DCC++ Classic & DCC++ EX software are separate projects from JMRI.   
The JMRI discussion groups can only provide limited support for DCC++ itself. If you have questions about building a DCC++ systems, operating it, etc., you should get help via DCC-EX Team members.  
You can seek us out here for **Software & Support:**

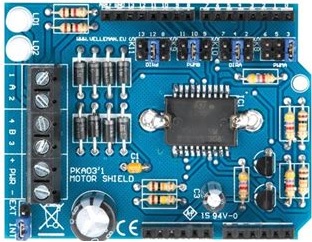
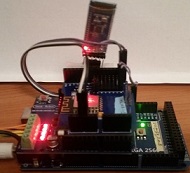
* [DCC-EX.com/](https://dcc-ex.com/)
* [DCC++ EX Discord Server](https://discord.gg/y2sB4Fp)
* [TrainBoard DCC++ Forum](https://www.trainboard.com/highball/index.php?forums/dcc.177/)

**Supported Hardware**

**Command Stations**

Currently DCC++ Classic & DCC++EX Command Station can be built from either the Arduino Uno or Arduino Mega platforms. DCC++EX will operate on a Arduino Uno & Mega and to some cases a Nano..  
  
The features of WiFi ESP8266 Command Station implementation are currently supported on the Mega2560 with a ESP8266 Wifi Shield and the new Mega2560 + ESP8266 WiFi Onboard combo board.

      
 Arduino Uno328 R3 Arduino ATMega2560 R3 MakerFab 8266Wifi Shield ATMega2560 +   
 ESP8266 WiFi R3

**Command Station Motor Shields (Boosters)**  
The Motor Shields are placed on top of a Arduino Processor Board and they Boost the 5vdc digital signals up to 12vdc - 18vdc then send the DCC signal out to the Main track Cab A and the Programming track Cab B connections.  
  
      
 Arduino L298P MS Pololu MC33926 MS IBT2 a 5amp-43amp Booster Arduino Mega + L298P MS   
 + WiFi Shield + Bluetooth  
 Standalone Command Station

**Control Stations and Throttles** [**WiFi Setup**](https://dcc-ex.com/get-started/wifi-setup.html)  
JMRI could support either Arduino platform via Serial USB cable Or Bluetooth add-on module added to a Command Station, or a Ethernet Network connection.   
Although a WiFi network interface to JMRI is available it isn’t practical as WiFi Protocol has a higher degree of overhead processing cycles and operating memory.  
 The WiFi interface is best used for WiThrottle Clients to a JMRI Server loaded on a Windows PC or Linux Pi 3 or 4.  
  
DCC++ & JMRI Supported WiThrottle Apps; [JMRI WiThrottles](https://www.jmri.org/help/en/package/jmri/jmrit/withrottle/UserInterface.shtml)   
(Android) Engine Driver  
(Apple iOs) WiThrottle  
(Android) DCCpp CAB  
 among others   
  
In Addition the New DCC++EX allows these Throttle Apps to Directly connect via WIFI to a Standalone DCC++EX + WiFi enabled Command Station to run multiple Locomotives in Operations mode. Or via Bluetooth to a DCC++EX Command Station enabled with an add-on Serial Bluetooth module.

**Computer Interfaces**

Currently, four interfaces are supported:

* Serial/USB: Direct connection to Arduino via a USB cable, Or via a Serial Bluetooth module
* Simulator: Simulated Command Station for off-line operation and testing.
* Network: TCP/IP connection from JMRI to a Command Station with an Ethernet
* DCC++ Over TCP: Remote network connection to a local JMRI instance connected to a Command Station

**Simulator**

To use the Simulator, simply choose that option in the JMRI Preferences. Note that some features do not work in the Simulator. For example, the Simulator does not (yet) support assigning, storing, and remembering Sensor and Turnout assignments. The Simulator interface is functional enough to keep JMRI "happy" while working on offline projects such as Operations or Panel and Logix design, but is not intended to be a full-fledged Command Station Emulator.

**DCC++ Over TCP**

For the DCC++ Over TCP, a host computer must be connected to the Command Station over a Serial or Network connection (or a simulator). This host computer then runs the DCC++ Over TCP Server. A remote computer (or several remote computers) can then use the DCC++ Over TCP (Server) interface to remotely access and control the Command Station.

**Limitations**

JMRI

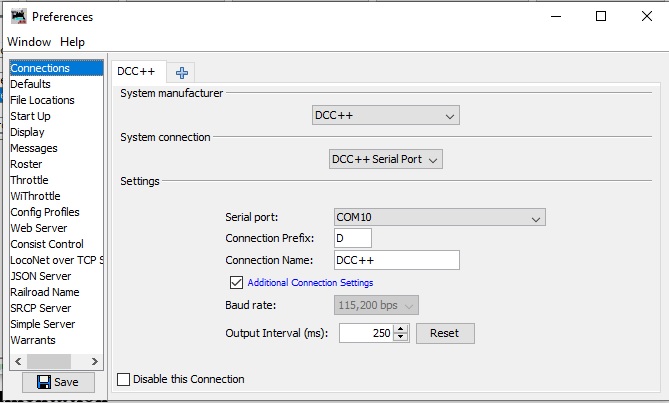
JMRI currently supports the DCC++ Classic 1.2.1+ version and the new DCC++ EX 3.0 version and above.  
  
Some limitation may arise if WiFi CS connection is used to interface to a JMRI Application server on a PC or Pi instead of using a Serial USB or serial Bluetooth add-on module interfaced to the Command Station.

**Connecting**

**Configuring an Arduino DCC++ Classic & DCC++ EX Command Stations**

In order to use an Arduino as a Command Station, you must first assemble the device and download 0ne of the Command Station firmware onto it. Please follow the [Assembling a DCC++ Command Station](https://dcc-ex.com/get-started/assembly.html)

**Connecting to a DCC++ & DCC++EX via Serial USB Or Serial Bluetooth  
Note: both the USB Cable and the Bluetooth Serial interfaces use the Rx0 Tx0 pins and can Not both be physically connected to Tx0 Rx0 at the same time.**

1. To connect your computer to a DCC++ Command Station, first install the appropriate drivers.
   * For the DCC++ Command Station with JMRI, you may need to install a device driver on your computer. Please refer to the [Arduino Getting Started pages](https://www.arduino.cc/en/Guide/HomePage) for instructions.
     + [Windows](https://www.arduino.cc/en/Guide/Windows) (See **Step 4**)
     + Mac: No drivers should be required
     + [Linux](http://playground.arduino.cc/Learning/Linux)
2. **Note:** If you have already installed the Arduino IDE software, the device drivers should already be installed.
3. When the device drivers are installed, connect the Arduino Command Station to your computer using a standard USB cable.
4. Once the drivers are installed, you are ready to configure JMRI.  
    Start one of the JMRI-based programs, Then go to the preferences panel. This opens automatically the first time a program is run, or you can select it from the "Edit" menu.
5. Select "DCC++" from the top selection box ("System Manufacturer").
6. Select "DCC++ Serial Port" from the second selection box ("System Connection")
7. Select the appropriate USB/Serial port from the third selection box ("Serial Port")  
    [^} Check the Additional Connection box just to verify that you set up the PC Device Manager baud rate to 115200 at Arduino setup time.
8. Click "Save". You maybe be asked if it's OK for the program to quit, click "Yes".
9. Restart the program. You should be up and running and See a On/Off button on the right side of the Decoder Pro and Panel Pro Panels  
     
   

**Connecting to a DCC++ & DCC++ EX Systems using Network**

1. First, connect the DCC++ Command Station to wired or WiFi network and record the IP Address and Port number. You may need to connect the Arduino to a host computer temporarily and use the Arduino Serial Monitor to get this information.
2. Now you are ready to configure JMRI. Start one of the JMRI-based programs, then go to the Preferences panel. This opens automatically the first time a JMRI program is run, or you can select it from the "Edit" menu (from the Application menu on OS X).
3. Select "DCC++" from the top selection box ("System Manufacturer").
4. Select "DCC++ Ethernet" from the second selection box ("System Connection").
5. Input the IP Address and Port Number of the DCC++ Command Station.
6. Click "Save". You'll be asked if it's OK for the program to quit, click "Yes".
7. Restart the program. You should be up and running.

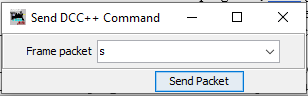
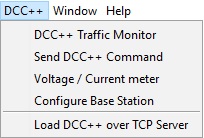
**Connecting to a DCC++ Classic & DCC++ EX Systems using DCC++ Over TCP**

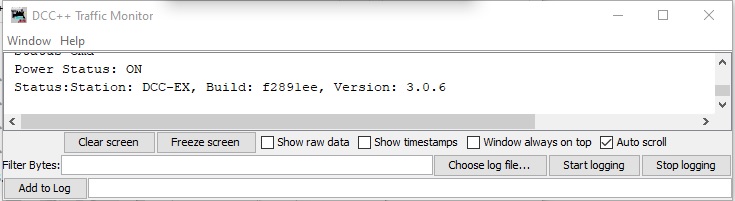
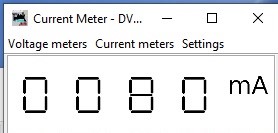
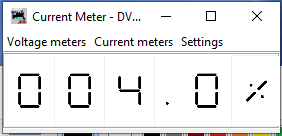
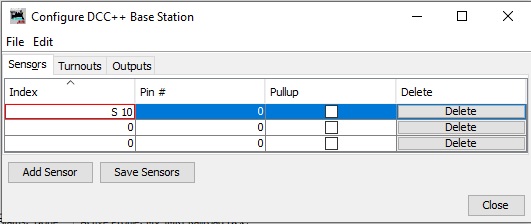
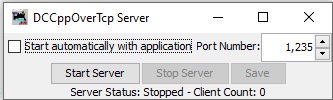
1. First, connect the DCC++ Command Station to the host computer via the Serial or Network interface as described above.
2. On the host computer select "Load DCC++ Over TCP Server" from the DCC++ Menu. Check the settings, then press the "Start Server" button. If you want the server to automatically start when JMRI is launched, click the checkbox provided.
3. The Host computer must be running with the DCC++ Server active before launching the Client computer.
4. On the client computer, launch JMRI on the Client computer.
5. In the Connection Preferences, select DCC++ as the System Manufacturer.
6. Select "DCC++ Server" as the System Connection Type.
7. Enter the network name or IP address of the host computer and verify that the port number is the same as on the host computer.
8. Save your Preferences and restart.

Once both the Host and Client instances of JMRI are configured, you can use the Client JMRI just the same as if it were directly connected to the Command Station. The client JMRI instance can be on the same computer, in the same house, or across the country from the host.

**Connecting to the DCC++ & DCC++ EX Simulator**

1. Start one of the JMRI-based programs, Then go to the preferences panel. This opens automatically the first time a program is run, or you can select it from the "Edit" menu.
2. Select "DCC++ Simulator" from the top selection box. There are no additional configuration steps required to use the DCC++ Simulator.
3. Click "Save". You'll be asked if it's OK for the program to quit, click "Yes".
4. Restart the program. You should be up and running.

**JMRI DCC++ Tools**

**Documentation**

**JMRI Help**

Additional documentation will be provided on other pages linked here:

* [Configuring Sensors](https://www.jmri.org/help/en/html/hardware/dccpp/Sensors.shtml)
* [Configuring Turnouts and Outputs](https://www.jmri.org/help/en/html/hardware/dccpp/Turnouts.shtml)

**Additional DCC++EX & Third Party info**

For more information on how to construct and program a DCC++ EX Command Station v3.0 and up  
 and additional DCC++EX blogs:

* [DCC-EX Website](https://dcc-ex.com/)
* [DCC-EX Reference Doc](https://dcc-ex.com/reference/index.html)
* [DCC++EX Firmware Download](https://dcc-ex.com/download/commandstation.html)
* Support & Development on [DCC++ EX Discord Server](https://discord.gg/y2sB4Fp)
* [DCC++EX YouTube](https://www.youtube.com/channel/UCJc-2ByHFUie-cKmhJ1zetQ)
* [DCC++EX YouTube Blog](https://www.youtube.com/results?search_query=DCC%2B%2B+EX)
* TrainBoard Blogs: [DCC++ Update Project 2020](https://www.trainboard.com/highball/index.php?threads/dcc-update-project-2020.130071/)
* [Facebook: DCC++ and Arduino Model Railroads](https://www.facebook.com/groups/1406785379394934/)
* [Facebook: Arduinos for Model Railroads](https://www.facebook.com/groups/926390087391995)
* [Arduino & JMRI](https://www.jmri.org/help/en/html/hardware/arduino/index.shtml)
* [Arduino Project Site](http://www.arduino.cc)
* [Sparkfun](http://www.sparkfun.com) - one of many places to acquire hardware

For more information on how to construct and program a DCC++ CLASSIC Command Station v1.2.1+:

* DCC++Original 2015 Into Videos; [DCC++ Project Starter by Author Gregg Berman](https://www.youtube.com/channel/UCJmvQx-fe0OMAIH-_g-_rZw)   
   *Note: The Original DCC++Base Station required jumper wires the New DCC++EX does Not*.
* DCC++ Original github repository [GitHub Repository](https://github.com/DccPlusPlus)
* [DCC++ Base Station Classic Download](https://dcc-ex.com/download/commandstation.html)
* [Trainboard.com Discussion](http://www.trainboard.com/highball/index.php?threads/introducing-dcc-a-complete-open-source-dcc-station-and-interface.84800/)
* [Facebook: DCC++ and Arduino Model Railroads](https://www.facebook.com/groups/1406785379394934/)
* [Facebook: Arduinos for Model Railroads](https://www.facebook.com/groups/926390087391995)
* [Arduino Project Site](http://www.arduino.cc)
* [Sparkfun](http://www.sparkfun.com) - one of many places to acquire hardware