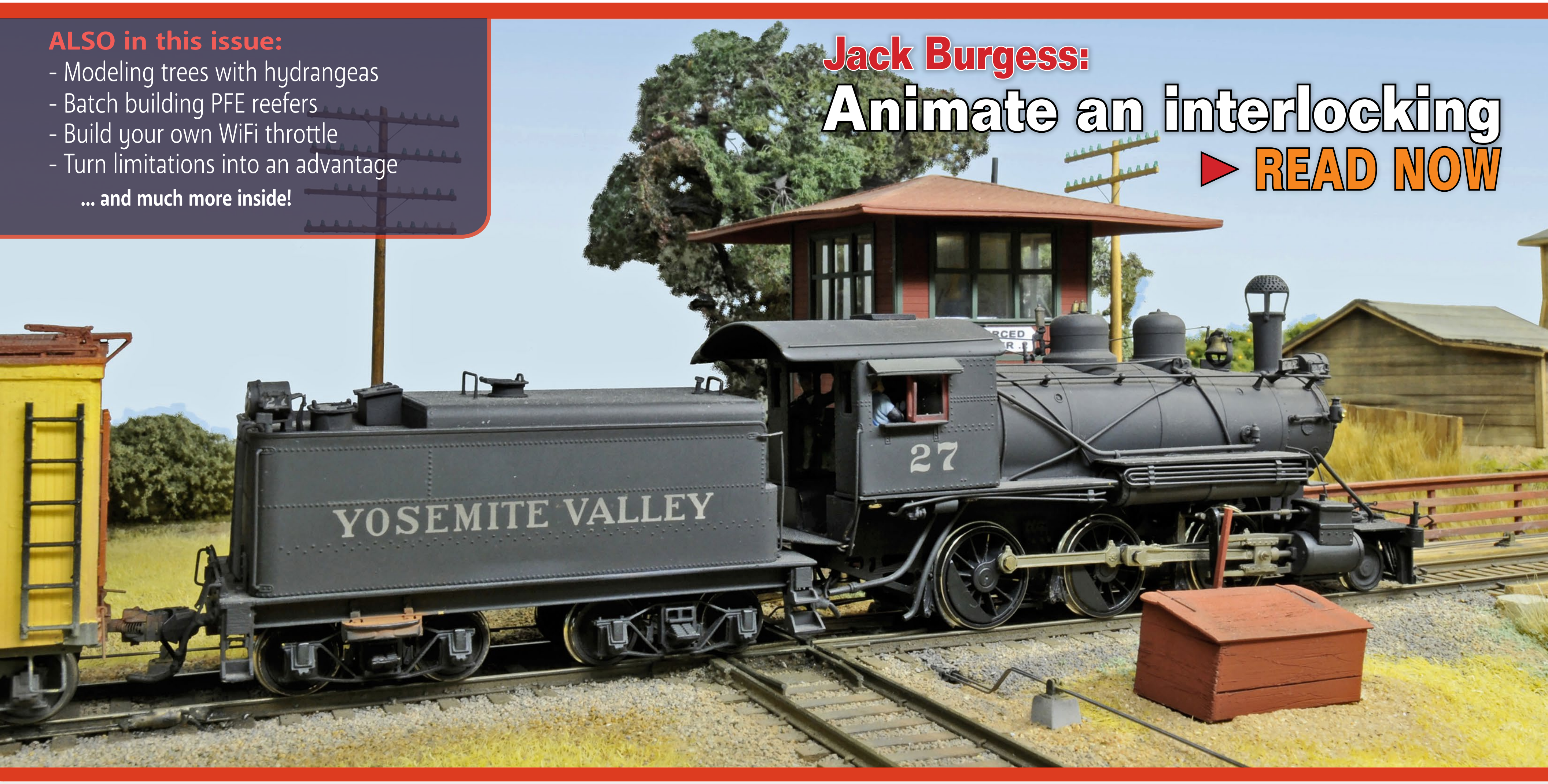
April 2019 - $1.99

MRH RUNNING EXTRA

More modeling every month



ALSO in this issue:

- Modeling trees with hydrangeas
- Batch building PFE reefers
- Build your own WiFi throttle
- Turn limitations into an advantage

... and much more inside!

Jack Burgess: Animate an interlocking

► READ NOW

TABLE OF CONTENTS | RUNNING EXTRA

April 2019

RATE THIS ISSUE

More content than ads will pay for ...

TABLE OF CONTENTS | MRH

April 2019

Build a Wi-Fi throttle

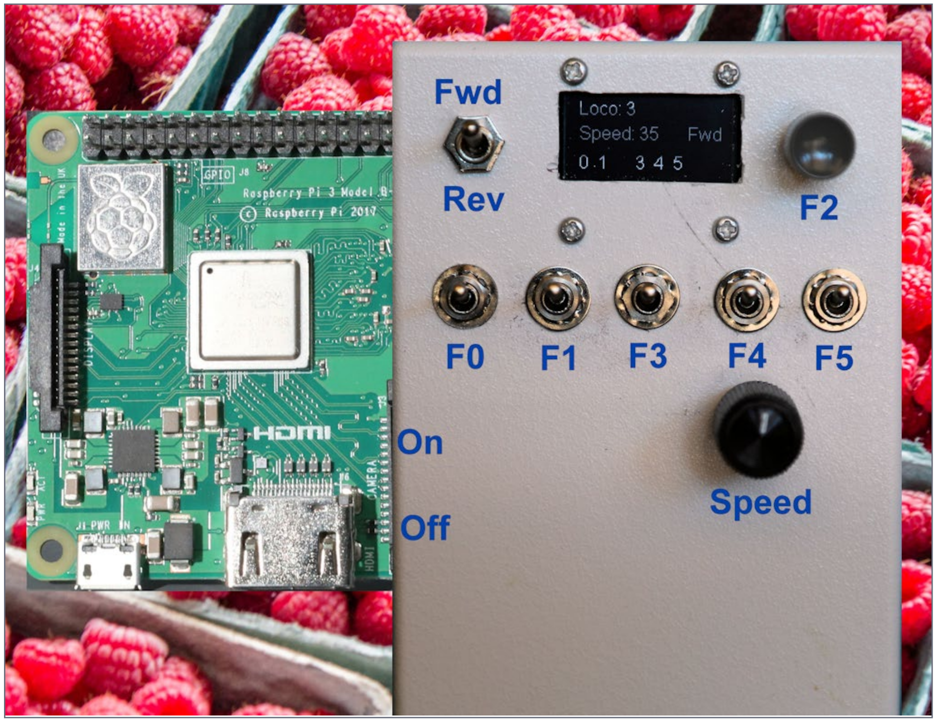
MRH Running Extra | April 2019

Dr Geoff Bunza shows how you can build your own Wi-Fi throttle ...

★★★★★ Rate this article

WiFi Throttle: Bonus extras

Model railroaders running JMRI have the capability of connecting their cell phone or wireless tablets as a remote throttle for DCC control. I have often heard modelers who would like newcomers to operate their layouts with a simplified throttle (reducing the possibility of inadvertently changing things visitors know nothing about). This article describes the construction of a simple WiFi throttle that connects directly to the JMRI WiFi server, just like the one described in the previous Raspberry Pi article. It is not dependent on your DCC command station connected to JMRI. It will support loco address selection, six (or more) function switches, speed and direction control, and runs on a standard nine-volt battery. While providing basic throttle functions, it can be customized



1. The simplified remote throttle.

and expanded by the inventive modeler to perform multiple functions at the touch of a button. You can also limit its functionality as you wish. It uses simple readily available parts for a one- or two-evening project.

The heart of the throttle is a relatively new module, the ESP32, or more accurately, the ESP32 Development board from EzSBC.com. It needs to be loaded in a manner similar to an Arduino controller, with nearly the very same tools. The ESP32 has a small voltage regulator on board so you can power it with a 6V-12V DC power supply (I use a standard 9V battery). The higher voltage is lowered to 3.3 volts for the ESP32. As with most of my projects, you don't need to know any programming to use this. Everything is pre-built for you. Additional instructions will guide you if you want to make changes. If you can operate text editor software, you can build your own throttle – and more than one can be used simultaneously with JMRI. Let's get started.

STEP 1: Gather your parts

Here are the materials you will need:

QTY	DESCRIPTION	
1	ESP32-01 breakout and development board	www.ezsbc.com/index.php/wifi01-35.html
6	SPDT subminiature toggle switches	www.allelectronics.com/item/smts-4/spdt-on-on-sub-min-toggle-switch/1.html
1	SPDT mini toggle switch center-off	www.allelectronics.com/item/mts-5pc/spdt-on-off-on-mini-toggle-switch-pc-terminals/1.html
1	10K linear miniature potentiometer	www.allelectronics.com/item/ptw-10k/10k-bourns-pot-pre-wired/1.html
1	Knob for potentiometer	
1	0.96 Inch OLED display white (or other color) with SSD1306 controller & I2C interface	www.ebay.com/itm/382557870211
1	Plastic case	www.ebay.com/itm/131978994077
2	Micro JST 1.25mm 6-Pin connector sets wired	www.ebay.com/itm/302746030692
1	9 volt battery holder snap	www.ebay.com/itm/163376406979

Use whatever knob fits your 10K pot.

The JST connectors are optional. Anytime I want disassembly ease, I use these.

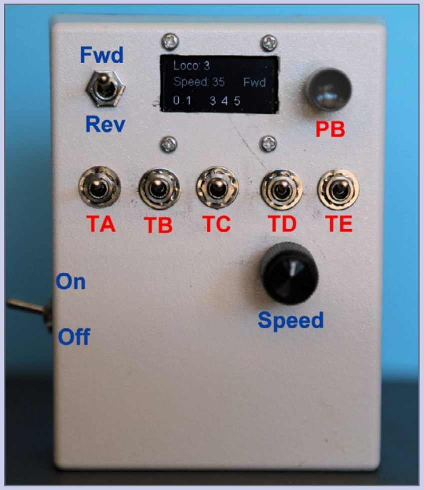
STEP 2: Lay out your case and prepare it

Here is the general layout of my throttle.

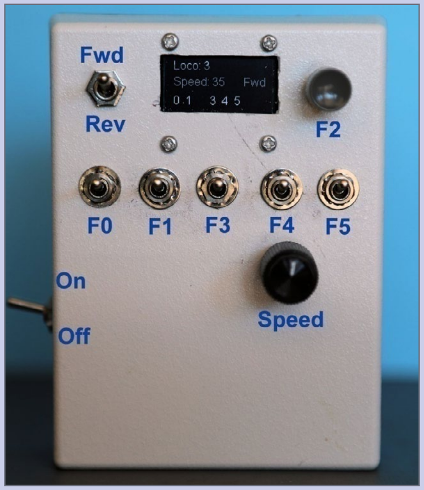
I installed only five subminiature toggle switches and one push-button for the six functions (F0-F5) I wanted. You can add more or use fewer.

The code has one value to set:

```
    int num_functions = 6;     // Number of functions
    supported by this throttle UP TO 8 MAX
```



2. Simplified throttle internal names.



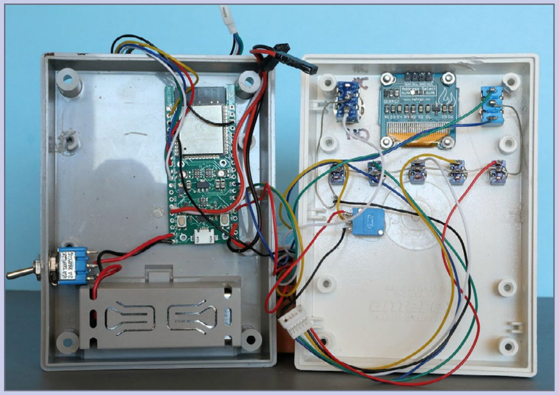
3. Simplified throttle functions assigned.

Your component choice may be somewhat different from mine. Lay out the switch holes and small window for the OLED display. I started by mimicking the Digitrax throttles which power on when the battery is inserted. But I later added a small switch to turn the battery on and off – quite simple.

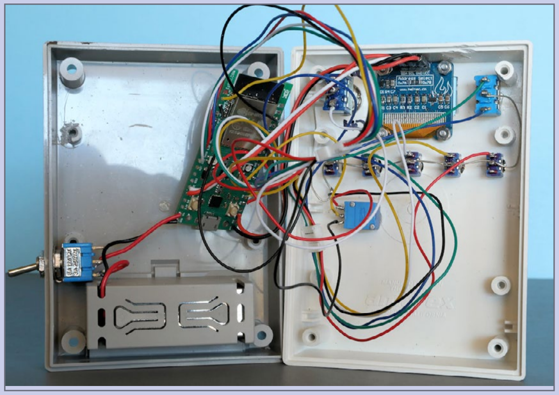
STEP 3: Install your components

Once you decide where everything goes, drill all your holes, insert, and lock down your components. You can see my internal layout in [4] and [5].

Many people like a large knob for their throttle. It is not a big deal to me, but in the box I chose there is plenty of room for a bigger knob, and even a bigger potentiometer. If you want to mount a six-inch knob, simply use a larger enclosure (try www.digikey.com, mouser.com, or allelectronics.com).



4. Component placement inside.



5. Wiring harnesses connected.

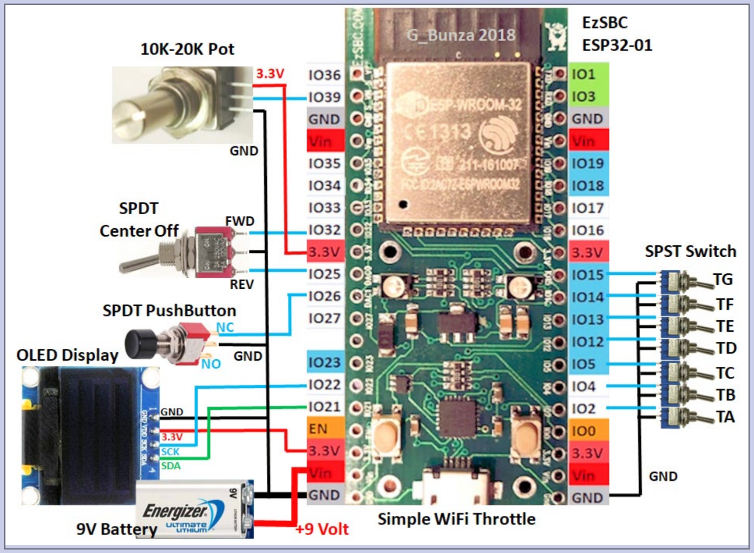
STEP 4: Wire everything

I found it easier to wire after installing all the switches. Since I knew I would take this apart again and again (this was the prototype, remember), I used small connectors with pre-attached wires to connect everything. Diagram [6] shows the basic interconnections, color coded.

The on-off switch is optional, but I recommend it to save battery life in the long run. The battery connector is hidden behind the cover of the battery compartment.

Note that the connections to the SPDT pushbutton need to connect to the common-middle and the normally closed connection. You should take note that the ESP32 is a 3.3V part. All digital connections are therefore 3.3 volts maximum.

Also realize that you can light most "normal" LEDs with the 3.3 volts and a small dropping resistor (like 220 ohms ¼ watt).



6. WIFi throttle diagram.

STEP 5: Download the Arduino development editor and software

I have previously written about installing and loading Arduino software:

- *MRH* December 2016, "A modeler's introduction to the Arduino" mrhpub.com/2016-12-dec/online.
- *MRH* March 2017, "DCC projects using the Arduino" mrhpub.com/2017-03-mar/online/html5.

The ESP32 is a different controller that will make use of the Arduino editor and loading environment, but with a different configuration and library. The same free tools that can be downloaded here: www.arduino.cc/en/Main/Software.

Download and install this if you have not already done so.

View reader comments

STEP 6: Load the ESP32 library

Start up the Arduino application you installed. On the top left menu select **File -> Preferences.**

In the field after "Additional Boards Manager URLs:" paste: dl.espressif.com/dl/package_esp32_index.json.

And click OK.

Go back to the top menu bar and select: **Tools-> Board -> Boards Manager...**

Look toward the bottom in the new Boards Manager list that opens, and find the entry labeled "esp32 by Expessif Systems" and click on that entry. A small label "Install" should appear on the lower right of the entry. Click on "Install." After it finishes installing, click "Close" in the lower right corner of the window.

Now click on **Tools-> Board-> Esp32 Dev Module** from the board list (likely near the bottom).

Close the Arduino IDE application.

View reader comments

WiFi Throttle: Bonus extras

STEP 7: Download Simple Throttle sketch from Additional Materials with this article

Download the Additional Materials that accompany this article, and unzip.

Move **folders** labeled **ESP32_Throttle** and **Blink7f_ESP32** and their **contents** into ...**\Documents\Arduino\** and **esp-oled-ssd1306** and **ESP_EEPROM** into ...**\Documents\Arduino\libraries\.**

STEP 8: Load the ESP32 controller with the Simple Throttle sketch

Connect the micro USB connector to your ESP32, and the other end to your Windows computer. As of this writing, all the Arduino/ESP32 software should work on a Windows PC, and an Apple PC. I know it will not work on a Raspberry Pi.

Start up the Arduino editor, making sure you have selected the ESP32 Dev Board from the Tools-> Board menu.

Next, select from Tools->Board->Port menu the Port your ESP32 is connected to.

I have tested all this with Windows10 and Windows7. If a "Port" number does not show up after you plug in your ESP32, the Win10 and Win7 current Silicon Labs USB drivers are included so you don't have to hunt for them.

I included a special "Blink7f_ESP32" sketch you can use for testing your setup to the ESP32.

STEP 8: Load the ESP32 controller with the Simple Throttle sketch *Continued ...*

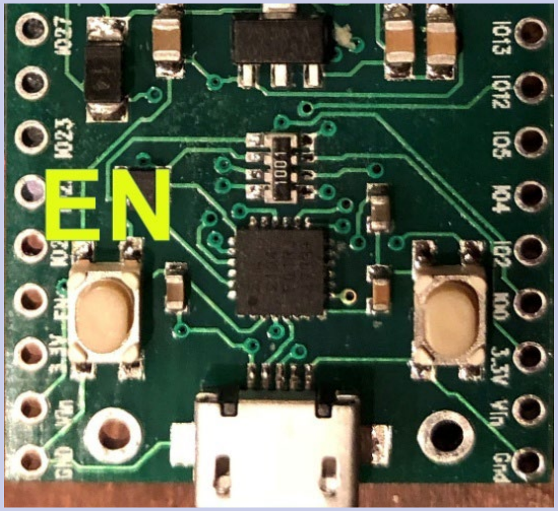
Open the sketch in the folder **Blink7f_ESP32** and download it onto your ESP32. After a short delay, two LEDs should blink in succession red, green, blue, and blue.

When this succeeds, open the folder labeled **ESP32_Throttle** and download the throttle sketch onto your ESP32. If you are using a Windows 7 PC, you will have an additional step:

After the download starts, at the bottom of the Arduino editor window you will see a message like: **Connecting....._____...**

As soon as you see this on screen hold down the “Enable” (EN) button to the left of the USB micro connector (it is immediately next to the pin marked “EN” [7]) and release the button as soon as the text changes in the window.

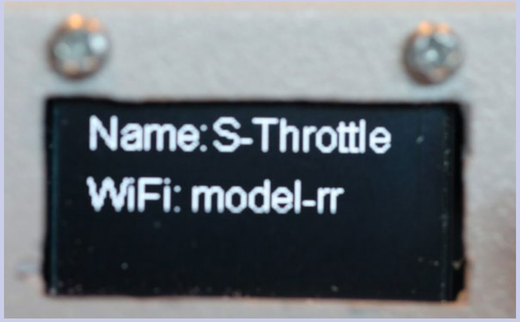
I have not been able to get around this method on Windows 7. I have not tried this on a Mac nor on any other Linux machine other than an RPi.



7. Enable button.

STEP 9: Connecting to your RPi

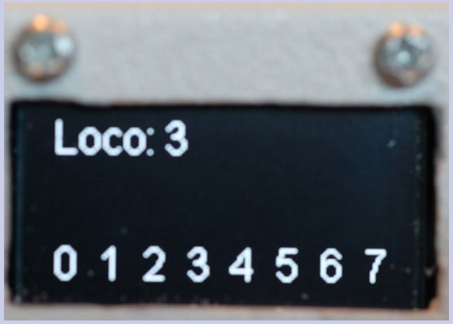
This assumes you set up a Raspberry Pi per the article in the March 2019 *MRH*. If you do not have your RPi powered up, you should get the message on the small OLED screen:



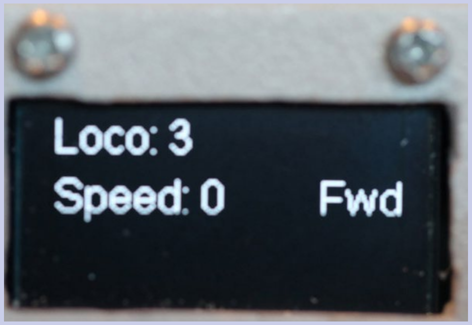
8. Start-Up screen.

This means your Simple Throttle has started up and is trying to connect to your WiFi named (SSID) **model-rr**.

When you turn on your RPi and it finds the **model-rr** WiFi server, it will then display in sequence:



9. Loco assigned and functions test.

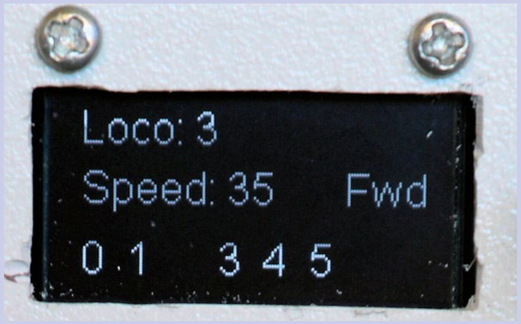


10. Loco 3 selected, speed is 0, and Fwd toggle is set.

★★★★★ Rate this article

STEP 9: Connecting to your RPi *Continued ...*

When you flip a function switch “ON,” the function number will be displayed like [11]:



```
Loco: 3
Speed: 35     Fwd
0 1   3 4 5
```

11. Operating Loco Functions 0, 1, 3, 4, and 5 ON.

View reader comments

STEP 10: Special Operations

Direction Neutral (Center-off): When the direction switch is set to center-off during normal operation, then the speed control will be ignored, and the last known speed sent to JMRI will be displayed. If you change the speed setting while in Neutral, then set the Direction to either FWD or REV. The current knob setting will be sent to JMRI as the new speed setting.

Changing addresses: Even if you are using this throttle for a young visitor, you may want to change the address of the throttle occasionally, without having to reprogram the ESP32. This can easily be done with the following sequence:

1. Start with the throttle **powered off** and the **Dir**ection switch set to **center-off**.

2. Push and hold down the pushbutton normally used for F2, then power-on the throttle.

3. Release the pushbutton and the screen should blink: **“SET ADDR?”** with the last address set displayed.

4. Set the **Dir**ection switch to increment up (FWD) or down (REV) and push the button to change the address.

5. When done setting the address, release the button, place the **Dir**ection switch to center-off, and hold the button down for one second. The new address is saved and you can start running the loco at this address.

Turning DCC Power ON: Before you release the button in the previous step, and after placing the **Dir**ection switch in the center-off position, if you flip all function switch up (ON), then when

STEP 10: Special Operations *Continued …*

you release the button the command will be sent to JMRI to turn on DCC power. This is not something one would normally expect a novice user to do on his/her throttle but it is provided here for your convenience.

If you are not changing the loco address, then:

1. Before powering on
2. Set the **Dir**ection switch to center-off
3. Turn all function switches up (ON)
4. Turn Power On or connect your battery
5. The command will be sent to JMRI to turn on DCC power.

View reader comments

STEP 11: Changing your throttle configuration

You can change your throttle to work with any JMRI WiFi server.

Near the beginning of the sketch you loaded into your ESP32 you will find the statements:

```
// WiFi Server Definitions
const char* ssid        = “model-rr”;
const char* password    = “model-rr”;
const char* host = “192.168.6.1”;
const int tpPort = 12090;
```

With the Arduino editor you can change the name (SSID) of the WiFi server, the password to access the server, the server's network (IP) address, and the Port number for the server access. When you edit them be sure to change **only** what is between the quote marks (“ ”).

You can also limit the number of function switches that can be used by the throttle from 1-8 by changing the “**6**” in the statement:

```
int num_functions = 6;     // Number of
functions supported by this throttle UP
TO 8 MAX
```

You can also change the switch/button assignments to their respective functions by changing:

```
#define F0_pin    TA   // Function 0 pin assignment
#define F1_pin    TB   // Function 1 pin assignment
#define F2_pin    PB   // Function 2 pin assignment
```

STEP 11: Changing your throttle configuration *Continued ...*

```
    #define F3_pin    TC   // Function 3 pin assignment
    #define F4_pin    TD   // Function 4 pin assignment
    #define F5_pin    TE   // Function 5 pin assignment
    #define F6_pin    TF   // Function 5 pin assignment
    #define F7_pin    TG   // Function 5 pin assignment
```

You can see the pins assigned to each label here:

```
    #define  PB       26   // Pushbotton Active LOW
                              usually assigned to F2
    #define  TA        2   // SPDT toggle active HIGH
    #define  TB        4   // SPDT toggle active HIGH
    #define  TC        5   // SPDT toggle active HIGH
    #define  TD       12   // SPDT toggle active HIGH
    #define  TE       13   // SPDT toggle active HIGH
    #define  TF       14   // SPDT toggle active HIGH
    #define  TG       15   // SPDT toggle active HIGH
```

This gives you the ability to assign the pushbutton to a different function, and to rearrange the order of all functions any way you want.

There are many more variations you can build into your throttle. This article should get you started on a whole series of new adventures in model railroading, quickly, and for relatively low cost! Have fun! ☑

★★★★★ Rate this article

Dr. Geoff Bunza

Geoff Bunza started as a model railroader when he received a Mantua train set for Christmas, at age 6. He fed his interests through college, becoming a member of the Tech Model Railroad Club (TMRC) at MIT while getting his doctorate and three other degrees in electrical engineering. He has collected Lionel HO trains for many years, which spawned his interest in realistic model animation and lighting. Primarily, he models the New York Central Railroad.

Geoff is a member of the New York Central System Historical Society, a life member of the NMRA, and holds an Extra Class amateur radio license. ■

WiFi Throttle: Bonus extras

View reader comments