Python Radio 1: Simple Beginnings

Simon Quellen Field

Simon Quellen Field

Follow

4 min read

·

Aug 17, 2024

Listen

Share

More

The simplest digital radio mode

Press enter or click to view image in full size

Photo by the author

This is the first in a series of articles about using microPython to control radio transmitters, receivers, and transceivers.

We will start with something simple and very cheap. Later articles in the series will present Python programs for many popular digital communication modes used in amateur radio. But for now, we will use the oldest and simplest mode: Morse code. This will get us started with microPython and one of the cheapest radios, the FS1000A transmitter and the MX-RM-5V receiver. Some versions of the receiver have the name XY-MK-5V but they are the same circuit.

When connected to good antennas, I have sent signals over a mile using these. You can get the pair for less than a dollar from AliExpress.com, or about two dollars on Amazon.com if you want faster delivery.

To match the low-price radio, we will use the Wemos D1 Mini as the computer. They are about two dollars at AliExpress.com or three for ten dollars at Amazon.

The latest version of microPython for the D1 Mini is found here: <https://micropython.org/download/ESP8266_GENERIC/>. We want the latest version, which will be the link in boldface type. Download it and rename it to firmware.bin.

We will also need the software to program the D1 Mini, called esptool. It is written in Python, so you type “pip install esptool” to download and install it:

Pip install esptool

When you connect the D1 Mini to your computer via USB, it will appear as a serial port, such as COM3: if you are running Windows. The first step is to erase the flash memory on the D1 Mini:

Esptool –port com3 erase\_flash

Then we load microPython:

Esptool –port com3 write\_flash –flash\_size=16MB -fm dio 0 firmware.bin

The D1 Mini can now accept and run microPython programs. To load them, we use the ampy program:

Pip install adafruit-ampy

Our main.py program for operating the transmitter is short:

From machine import Pin, PWM

From time import sleep

ANY\_CHANGE = Pin.IRQ\_RISING | Pin.IRQ\_FALLING

PIN\_D4 = 2

PIN\_D8 = 15

Xmit\_pin = PWM(Pin(PIN\_D4, Pin.OUT), 1000)

Xmit\_pin.duty(0)

Def key\_changed(pin):

Global xmit\_pin

If pin.value():

Xmit\_pin.duty(512)

Else:

Xmit\_pin.duty(0)

Key = Pin(PIN\_D8, Pin.IN)

Key.irq(trigger=ANY\_CHANGE, handler=key\_changed)

While(True):

Sleep(1)

The program sets the D4 pin to output a square wave at 1,000 Hertz. Pin D8 is an input, and whenever it changes, the key\_changed() method is called. That will make pin D4 emit the square wave or nothing, depending on the state of the D8 pin. The rest of the program just sleeps.

We load the program using ampy:

Ampy -p com3 put main.py

Photo by author

The transmitter has three pins: VCC, GND, and DATA (printed in reverse on the board as ATAD). We connect VCC to the 5 Volt pin on the D1 Mini. We connect GND to the GND pin on the Mini. We connect the ATAD pin to D4 on the Mini. To get more range, connect a short wire to the ANT hole in the transmitter. In a later article, we will construct a long-range antenna.

Press enter or click to view image in full size

Image by WeMos

The Morse code key connects the 3V3 pin (3.3 volts) to pin D8 when it is pressed down. If you don’t have a Morse code key, you can just short these two pins using a wire.

We connect the receiver to a battery and a speaker and we’re done.

Press enter or click to view image in full size

The MX-RM-5V receiver board has 4 pins. There are the VCC and GND pins, on the outside and two DATA pins in between. The DATA pins are connected, so you can use either one. The speaker connects to GND and DATA.

When you key the transmitter (by connecting the 3V3 and D8 pins), we hear a 1,000 Hertz tone from the speaker. With just the simple wire antennas, the range is ten to twenty feet. We will greatly increase that later.

The receiver is not sensitive to an unmodulated carrier (called Continuous Wave, or CW) so we needed to turn the transmitter on and off at 1,000 Hertz to hear anything. You can change that tone higher or lower if you like.

This radio operates at 433.92 megahertz and requires no license to operate.

Python

Radio

Morse Code

Amateur Radio

Microcontrollers