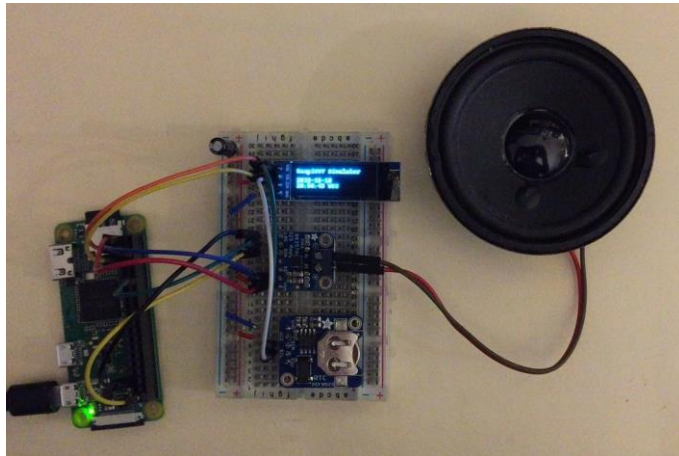


RaspiWWV

2018-10-22 R.Grokkett

Simulated WWV Shortwave Audio Time Broadcast



Hear it on YouTube: <https://youtu.be/FzX1HpYpx6E>

Remember the days when you would sit listening to WWV time signals on your Shortwave radio (*tick, tick, tick... At the tone, the time will be...*)?

Oh! You missed out on that? Now you can (re-)experience those moments and have your own WWV clock, no Shortwave Radio and no Internet connection required. Isn't that what you always wanted?

***Actually**, this is a project to help show you how to connect a tiny OLED screen, a Real Time Clock (RTC) and/or an Audio Amplifier all to a single Raspberry Pi Zero! All with the bonus of being able to listen to WWV time "signals" anytime you wish.*

Unlike the real WWV, the audio time signals as well as the admittedly tiny clock display will be only accurate to about one second of drift per day using the typical low-cost RTC module. You can improve this by using a higher quality (more expensive) RTC, or by just leaving the Raspberry Pi connected to the Internet, but this would never replace the atomic clocks of the real WWV. (See *precision time keeping with Raspberry Pi* articles, such as <https://www.satsignal.eu/ntp/Raspberry-Pi-NTP.html>)

Otherwise, for most uses, the accuracy is probably sufficient. Most uses? Well, besides falling to sleep to the droning sound of WWV, I used the shortwave radio version to timestamp astronomical observations; using a tape recorder (remember those?), I would record meteor observations, lunar occultations* or other events I saw, with WWV time signals in the background. "Mark!" The recording could then be transcribed with the sightings and timestamps in the comfort of home.

For those with such nostalgia, be sure to dig up an old transistor radio (and remember those?) to use as a case for your project. Note that the case will need to be large enough to install a battery, if you want portability!

The “WWV” software, written in Python, displays the time and plays the appropriate audio. The Raspberry Pi clock and RTC will resync to Network Time Protocol (NTP) servers whenever it connects to the Internet. (Anytime it is within range of your WiFi, if you are using a Raspberry Zero W.)

Hardware Parts

- Raspberry Pi Zero W
- MakerFocus 0.91 Inch I2C SSD1306 OLED Display Module
*Amazon <http://a.co/d/ioakKen>
(or other SSD1306 128x32 OLED display)*
- Real Time Clock (RTC) module PCF8523
<https://www.adafruit.com/?q=PCF8523>
- I2S 3W Class D Amplifier Breakout - MAX98357A
<https://www.adafruit.com/product/3006>
- Small Speaker (I salvaged a 2” speaker from old PC speakers)
- Breadboard, Pi Breakout cable/connector to breadboard, wire
- Optional: 5v 1amp battery (used to charge a cellphone)
*Such as Anker PowerCore 5000 Portable Charger <http://a.co/d/1utjWPg>
Alternately, you can use a LiPo battery + 5v converter/charger
<https://www.adafruit.com/product/2011>
<https://www.adafruit.com/product/2465>
Or 5V 2amp power supply*
- CR1220 battery for RTC
- Case – old transistor radio

Raspbian OS Setup

Install **Raspbian Stretch Lite** onto a 4GB or larger SD card. (steps below)

1. If you haven’t already, install Raspbian Lite version onto a 4GB or larger microSD card. You DO NOT need the GUI version, as this project does not use a monitor or keyboard.
<https://www.raspberrypi.org/downloads/raspbian/>
2. You will need to access the Raspberry remotely via SSH. On Windows, you can use PUTTY SSH terminal program. On Mac, just bring up a command terminal window.
<https://www.putty.org/>

Did you know?

If you install Raspbian on an SD card using a PC, you can create two files on the card to configure WiFi and SSH access before you boot it on a Raspberry?

For this, assume your SD card is currently mounted as K: on your PC:

- 1) Install the latest Raspbian Lite image to the SD.
<https://www.raspberrypi.org/downloads/raspbian/>
- 2) With notepad, create a file called just “**ssh**” and use Save As “All files” to **K:\ssh**
The file can contain anything. It’s the **filename** that is important. Must **NOT** be “**ssh.txt**”!!!
- 3) With notepad, create a second file called “**wpa_supplicant.conf**” with following:

```
ctrl_interface=DIR=/var/run/wpa_supplicant
GROUP=netdev
update_config=1

network={
    ssid="mySSID"
    psk="mypassword"
    key_mgmt=WPA-PSK
}
```

Use Save As “All files” to **K:\wpa_supplicant.conf**

Again, do not let Notepad change it to “wpa_supplicant.conf.txt”!!

When you boot the Raspberry the first time, Raspbian will look for these and connect to your Wifi. You will have to look on your Router for the IP address, though, since its auto assigned.

3. Insert the microSD card into the Pi and plug in the power now. It will take a few minutes to boot.
4. To remotely log in to your Raspberry Pi, you will need to find its IP address.
You can try:

\$ ssh pi@raspberrypi.local

(Or from Putty, enter hostname **pi@raspberrypi.local**

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address) Port

Connection type:

☐ Raw ☐ Telnet ☐ Rlogin ☒ SSH ☐ Serial

Load, save or delete a stored session

Saved Sessions

Note: If this fails, you will need to see if your Router will show the IP addresses of your local devices.

Example: `ssh pi@192.168.X.X`

Default password is “**raspberry**”

Once booted and logged in, start by updating your Pi’s OS, change the “pi” password, and turn on I2C interface:

```
sudo apt-get update
sudo apt-get upgrade
sudo raspi-config
```

Select:

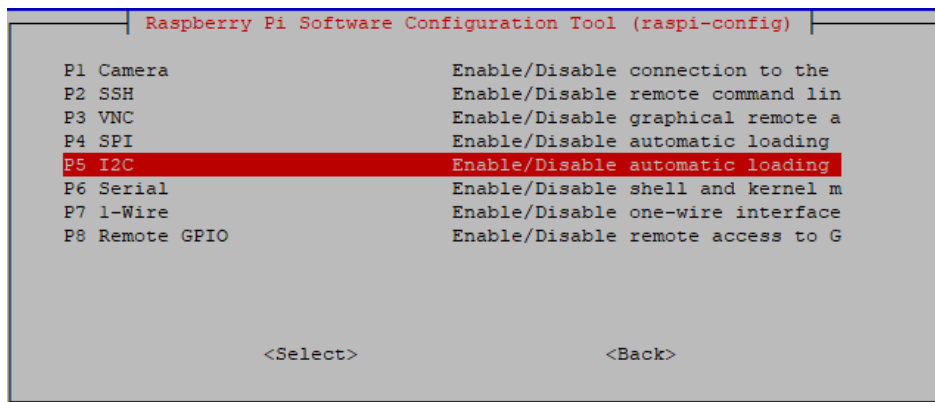
Change user password

Interfacing Options -> I2C Enable -> Yes

Raspberry Pi Software Configuration Tool (raspi-config)

1 Change User Password	Change password for the current u
2 Network Options	Configure network settings
3 Boot Options	Configure options for start-up
4 Localisation Options	Set up language and regional sett
5 Interfacing Options	Configure connections to peripher
6 Overclock	Configure overclocking for your P
7 Advanced Options	Configure advanced settings
8 Update	Update this tool to the latest ve
9 About raspi-config	Information about this configurat

<Select> <Finish>



Be sure to leave the **Locale** and **Timezone** (UTC) unchanged.

Select FINISH

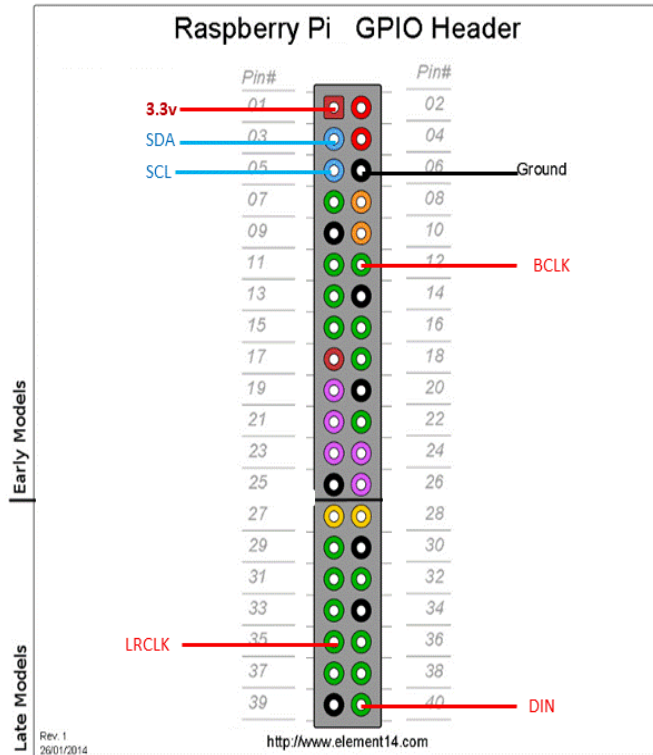
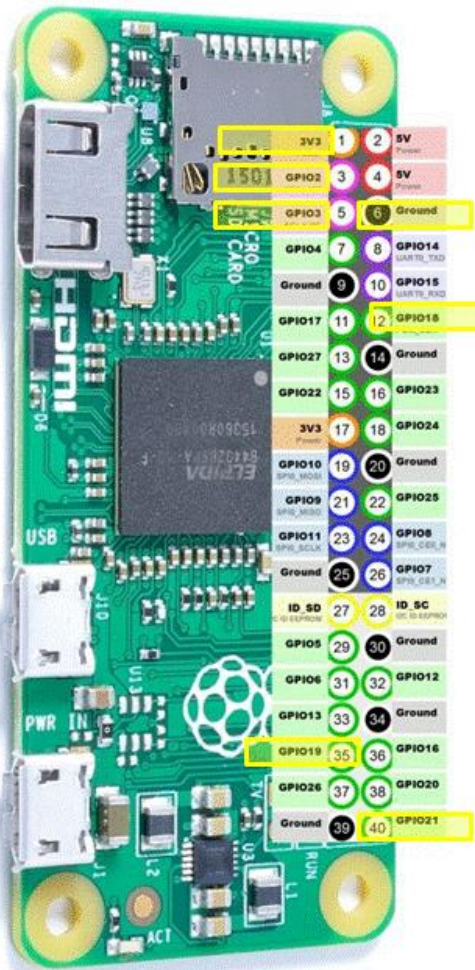
Note that WWV runs in UTC time, not local.

You can wait to reboot after the next software install:

RaspiWWW Software Install

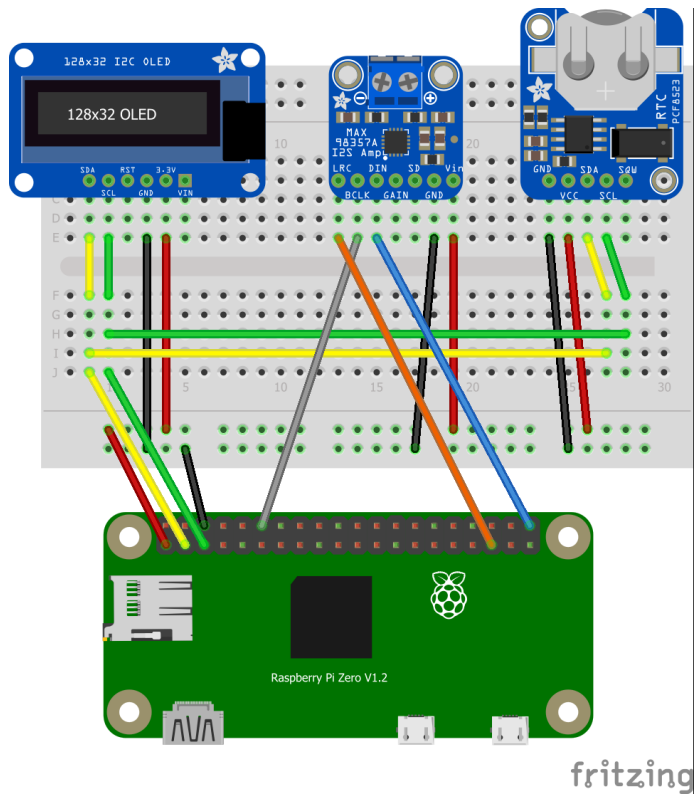
```
cd /home/pi
sudo apt install -y git
git clone https://github.com/rgrokkett/RaspiWWV.git
sudo shutdown now
```

Power down and unplug the Pi to do wiring next, before trying the software.



Early Models

Late Models



Figs: Final pins and wiring (see below for individual components)

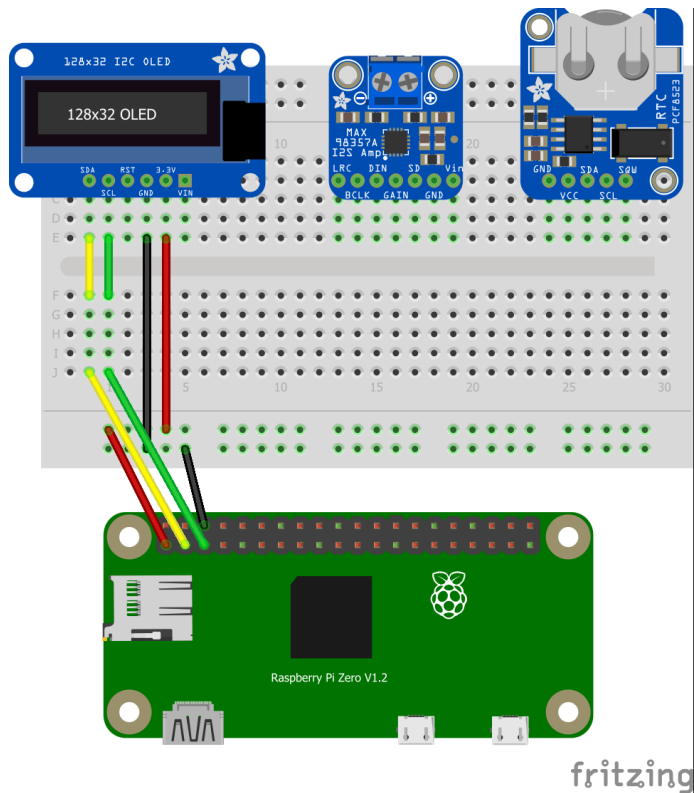
OLED Display Setup

(from <https://learn.adafruit.com/ssd1306-oled-displays-with-raspberry-pi-and-beaglebone-black/overview>)

Wire in SSD1306 OLED display as follows:



OLED	Pi
SCL	SCL pin 5 (GPIO 3)
SDA	SDA pin 3 (GPIO 2)
VCC	3.3v pin 1
GND	GND pin 6



Power up the Raspberry and log back in and run:

```
ssh pi@raspberrypi.local
```

Log in with your new password

```
sudo apt-get install -y python-smbus
```

```
sudo i2cdetect -y 1
```

```
pi@raspberrypi:~$ sudo i2cdetect -y 1
    0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
10:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
20:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
30:  --  --  --  --  --  --  --  --  --  --  --  3c  --  --  --
40:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
50:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
60:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
70:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
pi@raspberrypi:~$
```

You should see a “3C” address show up.

If not, check your wiring again and be sure I2C is turned on in **raspi-config**.

Next, run the OLED install script to install the Adafruit Python OLED SSD1306 library:


```
cd /home/pi/RaspiWWV
sh installOLED.sh
```

Once Finished, run the OLED test program:

```
python testOLED.py
```

If you get a display, then continue to installing the Real Time Clock (RTC)

If not, check your wiring and the Adafruit tutorial (url above) for more info.

Real Time Clock (RTC) Setup

(from : <https://learn.adafruit.com/adding-a-real-time-clock-to-raspberry-pi/wiring-the-rtc>)

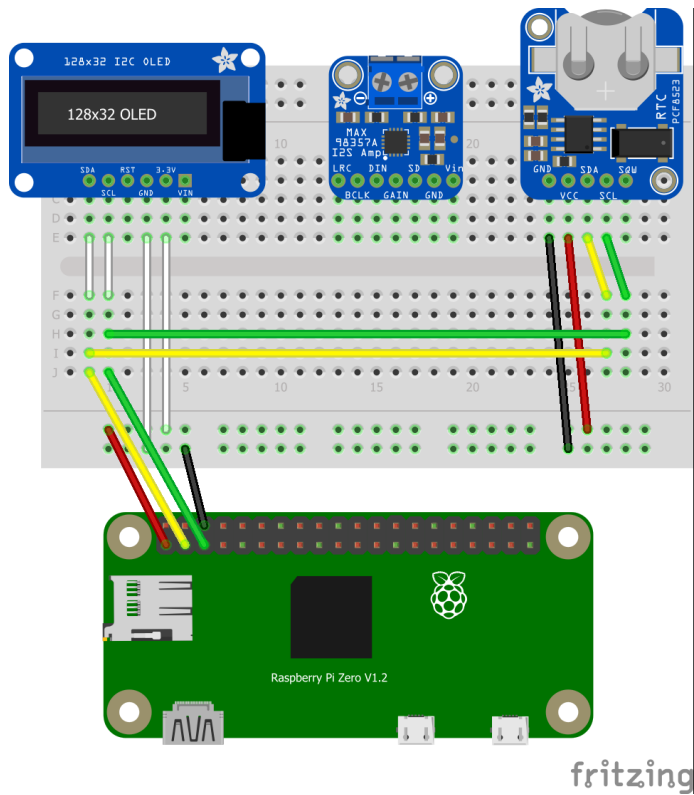
Again, shutdown the Pi and unplug it.

```
sudo shutdown now
```

Wire in the RTC in parallel with the OLED, so now both the OLED and the RTC modules are connected to the same GPIO pins on the Pi. (The I2C bus allows for multiple devices, as long as they have different addresses)



RTC	OLED	Pi
SCL	SCL	SCL pin 5 (GPIO 3)
SDA	SDA	SDA pin 3 (GPIO 2)
VCC	VCC	3.3v pin 1
GND	GND	GND pin 6



Put a CR1220 battery into the RTC

Plug in the power to the Pi and log back in again and run:

```
sudo i2cdetect -y 1
```

```
pi@raspberrypi:~$ sudo i2cdetect -y 1
   0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
10:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
20:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
30:  --  --  --  --  --  --  --  --  --  --  --  3c  --  --  --
40:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
50:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
60:  --  --  --  --  --  --  --  68  --  --  --  --  --  --  --
70:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
pi@raspberrypi:~$
```

You should see a 68 address as well as the 3C address.

Run the RTC install script:

```
cd /home/pi/RaspiWWV
sh installRTC.sh
```

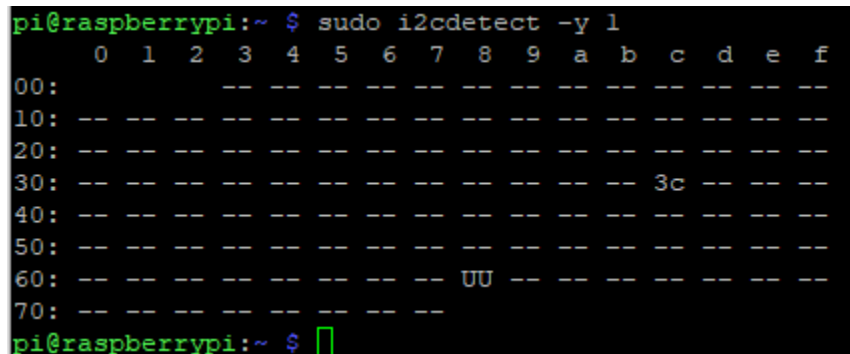
Answer the question concerning which RTC chip type you have.
The default [1] is the pcf8523 chip for the Adafruit RTC listed in the hardware list above.
Once complete, reboot & log back in again:

```
sudo reboot
```

```
ssh pi@raspberrypi
```

```
sudo i2cdetect -y 1
```

You should see UU instead of 0x68



```
pi@raspberrypi:~ $ sudo i2cdetect -y 1
    0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
10:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
20:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
30:  --  --  --  --  --  --  --  --  --  --  --  3c  --  --  --  --
40:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
50:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
60:  --  --  --  --  --  --  --  UU  --  --  --  --  --  --  --  --
70:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
pi@raspberrypi:~ $
```

Verify the Pi shows the correct time:

```
date
```

```
Tue  9 Oct 20:51:40 UTC 2018
```

Note that the time **MUST** be in **UTC** time.

If the date/time is not correct, be sure the Pi can access the Internet so it can set its time.

Now run the RTC install script again:

```
cd /home/pi/RaspiWWV
sh installRTC.sh
```

Answer with the same RTC chip you have.
This time, the script will disable the old fake hardware clock and replace with the RTC.
It will also set the time of the RTC clock to match the system clock.

Check the time for the System clock vs the hardware RTC clock:

```
timedatectl status
```

```
pi@raspberrypi:~ $ timedatectl status
Local time: Tue 2018-10-09 21:23:29 UTC
Universal time: Tue 2018-10-09 21:23:29 UTC
RTC time: Tue 2018-10-09 21:23:29
Time zone: Etc/UTC (UTC, +0000)
Network time on: yes
NTP synchronized: yes
RTC in local TZ: no
```

FYI:

You can also manually set the RTC clock to the Pi's current System time:

```
sudo hwclock -w
```

You should only have to set it once, unless there is a problem with the battery.

To read the RTC clock's time:

```
sudo hwclock -r
```

If wifi is available, the Raspi will update the system clock and the RTC clock to the Internet Network Time Protocol (NTP) servers so as to correct any drift.

For this to occur, the Pi must be rebooted while connected to a WiFi network.

Audio Setup

(from <https://learn.adafruit.com/adafruit-max98357-i2s-class-d-mono-amp/raspberry-pi-usage>)

NOTE: THIS ASSUMES YOU ARE USING THE ADAFRUIT MAX98357 I2S AMPLIFIER

[HTTPS://WWW.ADAFRUIT.COM/PRODUCT/3006](https://www.adafruit.com/product/3006)

Shutdown and unplug power from the Pi and install the audio card:

```
sudo shutdown now
```

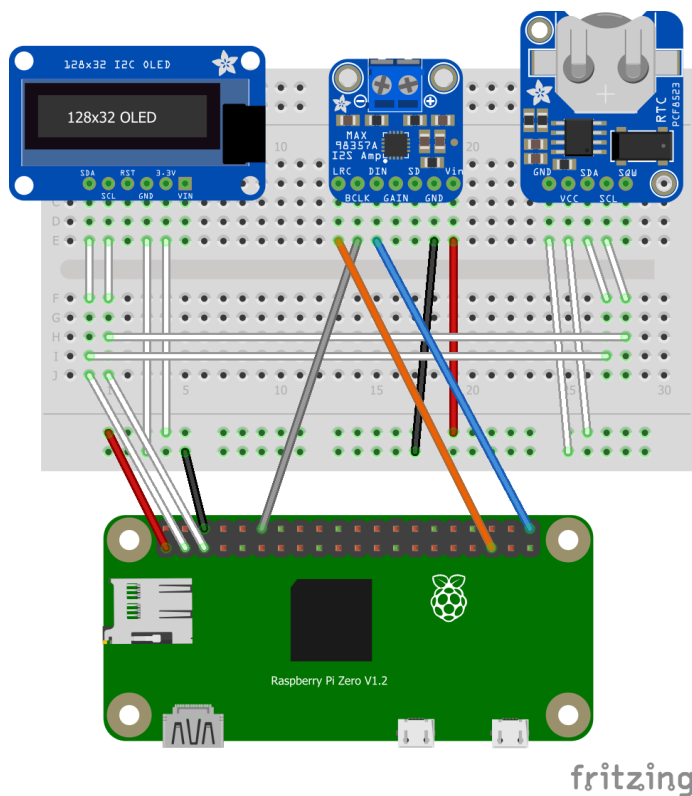


Wire as follows:

- **Amp Vin** to Raspi Zero Pi **Pin 1 +3.3V**
- **Amp GND** to Raspi Zero Pi **GND**
- **Amp DIN** to Raspi Zero Pi **Pin 40 (GPIO 21)**
- **Amp BCLK** to Raspi Zero Pi **Pin 12 (GPIO 18)**
- **Amp LRCLK** to Raspi Zero Pi **Pin 35 (GPIO 19)**
-
- **Speaker** to Amp Audio connector

For wiring simplicity sake, I have used the +3.3v lead to power all the modules.

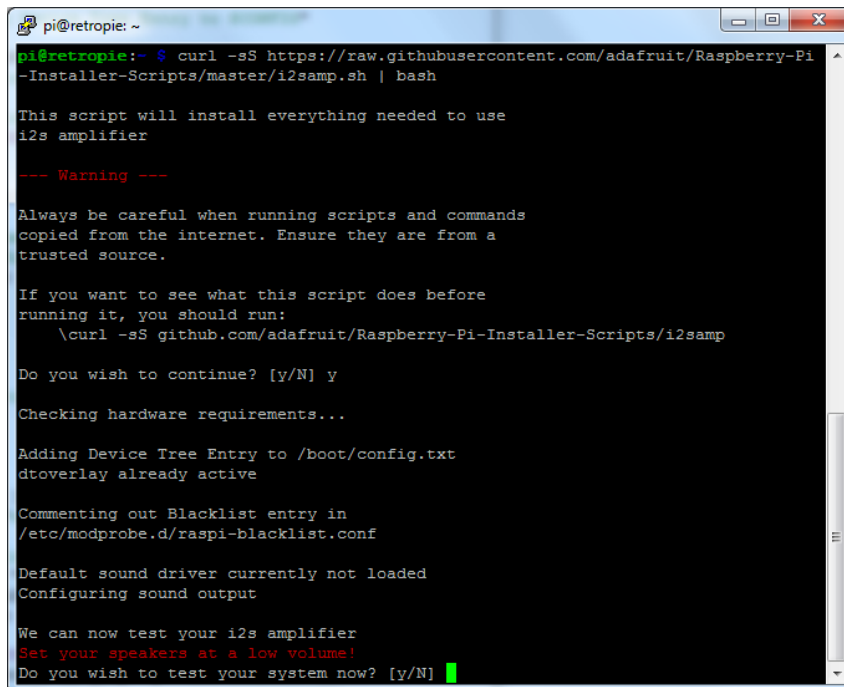
The audio amp can also be run from Pin 2, +5V instead, if you need higher volume.



Next, power up, log back in and run the Audio amp install script:

```
cd /home/pi/RaspiWWV
sh installAMP.sh
```

Answer the questions "Y"



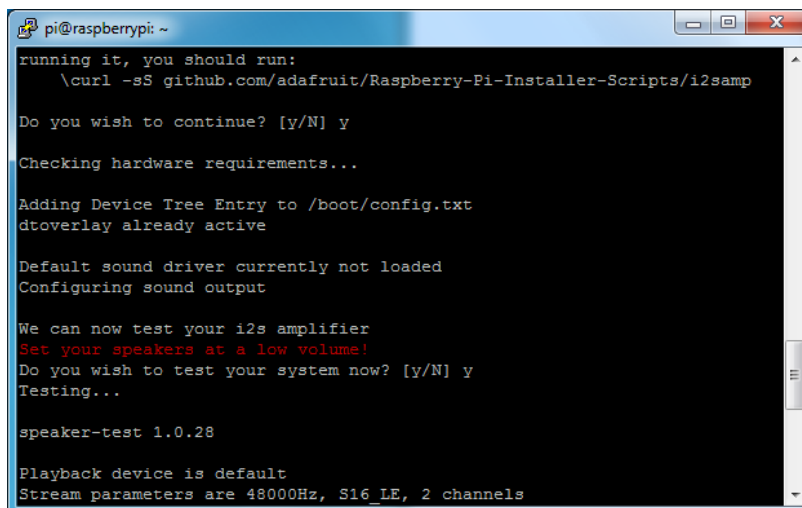
```
pi@retropie: ~  
pi@retropie:~$ curl -sS https://raw.githubusercontent.com/adafruit/Raspberry-Pi-Installer-Scripts/master/i2samp.sh | bash  
  
This script will install everything needed to use  
i2s amplifier  
  
--- Warning ---  
  
Always be careful when running scripts and commands  
copied from the internet. Ensure they are from a  
trusted source.  
  
If you want to see what this script does before  
running it, you should run:  
  \curl -sS github.com/adafruit/Raspberry-Pi-Installer-Scripts/i2samp  
  
Do you wish to continue? [y/N] y  
  
Checking hardware requirements...  
  
Adding Device Tree Entry to /boot/config.txt  
dtoverlay already active  
  
Commenting out Blacklist entry in  
/etc/modprobe.d/raspi-blacklist.conf  
  
Default sound driver currently not loaded  
Configuring sound output  
  
We can now test your i2s amplifier  
Set your speakers at a low volume!  
Do you wish to test your system now? [y/N]
```

After rebooting, log back in and re-run the script again.

```
cd /home/pi/RaspiWWV  
sh installAMP.sh
```

It will ask to test the speaker. Answer Y

You should hear "Front Left" "Front Right" repeated about 5 times then it exits.



```
pi@raspberrypi: ~  
running it, you should run:  
  \curl -sS github.com/adafruit/Raspberry-Pi-Installer-Scripts/i2samp  
  
Do you wish to continue? [y/N] y  
  
Checking hardware requirements...  
  
Adding Device Tree Entry to /boot/config.txt  
dtoverlay already active  
  
Default sound driver currently not loaded  
Configuring sound output  
  
We can now test your i2s amplifier  
Set your speakers at a low volume!  
Do you wish to test your system now? [y/N] y  
Testing...  
  
speaker-test 1.0.28  
  
Playback device is default  
Stream parameters are 48000Hz, S16_LE, 2 channels
```

You can also run the following to test your speaker:

```
sudo amixer sset PCM,0 95%  
  
aplay /usr/share/sounds/alsa/Front_Center.wav
```

If you don't have any sound or need more information, again check out the Adafruit Tutorial URL at the beginning of the Audio section, above.

NOTE: Unfortunately, the I2S audio interface on the Raspberry Pi Zero can generate pops intermittently. In actual operation, this should not occur too often.

Test everything together:

```
cd /home/pi/RaspiWWV
sh test.sh
```

Adjust the volume with either:

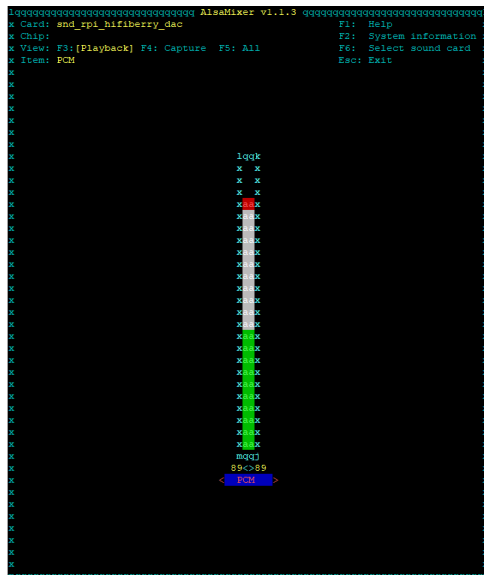
```
sudo amixer sset PCM,0 95%    # 0% to 100%
```

or

```
sudo alsamixer
```

Use the arrow keys to increase/decrease master volume.

Esc key to exit.

A screenshot of the terminal window running the alsamixer command. The window shows the ALSA Mixer v1.1.3 interface. At the top, it lists the card as 'snd_rpi_hifiberry_dac' and the chip. Below that, it shows the current view is F3: [Playback] and the current item is PCM. The main part of the window displays a volume control slider for the PCM channel, with the volume set at 89%. The slider is represented by a series of 'x' characters, with the current level highlighted in red. The bottom of the window shows the PCM channel name and the current volume level '89<>89'.

Finally, install the WWV cron file to start the WWV program every time Pi boots up:

```
cd /home/pi/RaspiWWV
sh installWWV.sh
sudo reboot
```

Reboot your Pi and after about a minute or two, you should start hearing the “WWV” time signal.

“On/Off” Button Setup (optional)

To add a safe shutdown button to your Pi, add a momentary pushbutton between pin 16 (GPIO 23) and Ground.

Pi	Desc
Pin 16 (GPIO 23)	Safely Shutdown Raspberry (Does not remove 5V power, must be done usually via the 5V USB battery on/off switch)
Pin 14 (GND)	Ground side of momentary button

Next, add the following line to bottom of /boot/config.txt:

```
sudo nano /boot/config.txt  
  
dtoverlay=gpio-shutdown,gpio_pin=23  
  
sudo reboot
```

Pushing the button for a second or two causes the Pi to safely shutdown. Once the Pi’s green LED turns off, you can unplug the power. (The button does not remove power.)

Note: Unfortunately, because this project requires I2C serial interface, it interferes with the ability to startup (turn “ON”) the Pi, so the button can only shutdown, not startup. Plugging in the power again will start the Pi.

Sources (and MANY THANKS!)

- WWV Audio – from Mxsmanic
<https://freesound.org/people/Mxsmanic/sounds/139323/>
- OLED SSD1306 Tutorial and Library
<https://learn.adafruit.com/ssd1306-oled-displays-with-raspberry-pi-and-beaglebone-black/overview>
https://github.com/adafruit/Adafruit_Python_SSD1306
- RTC Clock Tutorial
<https://learn.adafruit.com/adding-a-real-time-clock-to-raspberry-pi/wiring-the-rtc>
- Adafruit MAX98357 I2S Class-D Mono Amp Tutorial
<https://learn.adafruit.com/adafruit-max98357-i2s-class-d-mono-amp/raspberry-pi-usage>
- Lunar Occultation - <https://en.wikipedia.org/wiki/Occultation>