

Raspberry Pi 4 – Based Multi-mode Beacon using Wav Files for Signal Generation

Roger Rehr W3SZ 5/21/2024

Goal: Multi-mode beacon, including WSJT modes, CW, and SSB using audio files to generate the signals.

Model: Use a USB GPS dongle to GPS time-align the Raspberry Pi 4, so that the timing of the audio signals can be set precisely and accurately. Use a python script to control the timing and selection of the audio signals to be sent from the Raspberry Pi to the transmitter.

Hardware used:

Raspberry Pi 4, 4 GB \$55 <https://www.adafruit.com/product/4295>

RPi Case \$6 <https://www.adafruit.com/product/4301>

5V @ 4A power supply <https://www.amazon.com/dp/B097P2NLVH>

16 GB Micro Ultra San Disk many choices on Amazon and elsewhere, on the order of \$5-6 or less.

VK-162 G-Mouse USB GPS Dongle

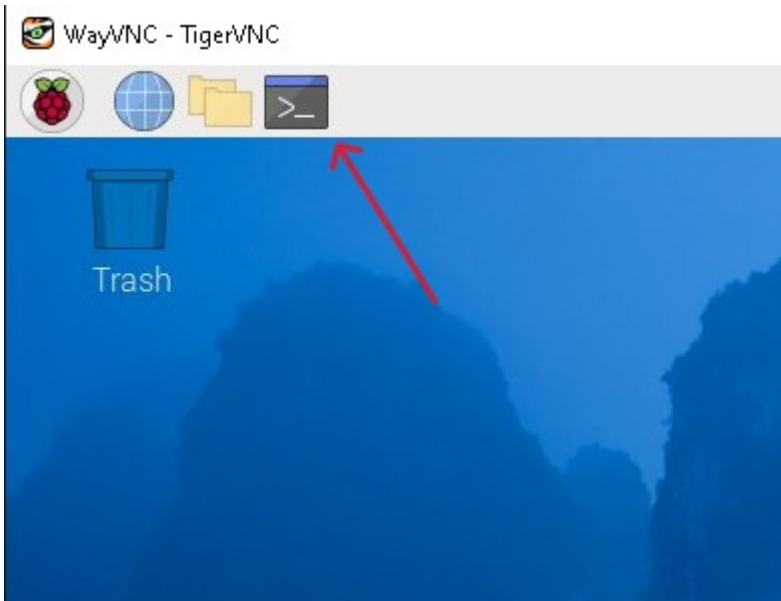
<https://www.amazon.com/Navigation-External-Receiver-Raspberry-Geekstory/dp/B078Y52FGQ/>
\$17

I. Initial Setup:

1. Follow the setup guide for the Raspberry Pi starting at this URL:
<https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up/1>
2. At step 2, I recommend installing Raspberry Pi OS using Raspberry Pi Imager, following instructions at: <https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up/2>
3. Although the RPi will run “headless” without monitor, keyboard, or mouse when it is controlling the beacon, you do need to connect up to these peripherals while setting it up. The 3rd setup page describes these connection details: <https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up/3>
4. Page 4 completes the setup process: <https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up/4>

II. Update Operating System:

1. Left-click the “Terminal” icon at the top left of the Desktop to open a terminal window as shown below:



Then type **sudo apt update** and hit “Enter”

```
roger@raspberrypi1: ~  
File Edit Tabs Help  
roger@raspberrypi1:~ $ sudo apt update
```

After the update completes, type **sudo apt upgrade** and hit “Enter”.

III. Add GPS Software and Configure it:

This recipe is taken from Mike Richards, G4WNC at: <https://photobyte.org/raspberry-pi-stretch-gps-dongle-as-a-time-source-with-chrony-timedatectl/> . Mike’s procedure worked for me with just a couple of modifications needed, changing “python” to “python3” in a couple of cases. These changes are reflected in the text below.

1. At the command prompt, type **sudo apt -y install gpsd gpsd-clients python3-gps chrony python3-gi-cairo**
2. Type **sudo nano /etc/default/gpsd** and then edit so that you have the lines shown in white below present:

```
## Devices gpsd should collect to at boot time.
# They need to be read/writeable, either by user gpsd or the group
#

# Other options you want to pass to gpsd

# Automatically hot add/remove USB GPS devices via gpsdctl
#START_DAEMON="true"
USBAUTO="true"
DEVICES="/dev/ttyACM0"
GPSD_OPTIONS="-n"
```

Note that “START_DAEMON=“true” is commented out. If it is NOT commented out, your GPS dongle will likely not work.

Type **ctl-o** and hit “Enter” to save the modified file, and then type **ctl-x** to close the file.

Now that you are back at the command prompt, type **sudo reboot** to reboot.

After the RPi has rebooted, again open a terminal window, and then type:

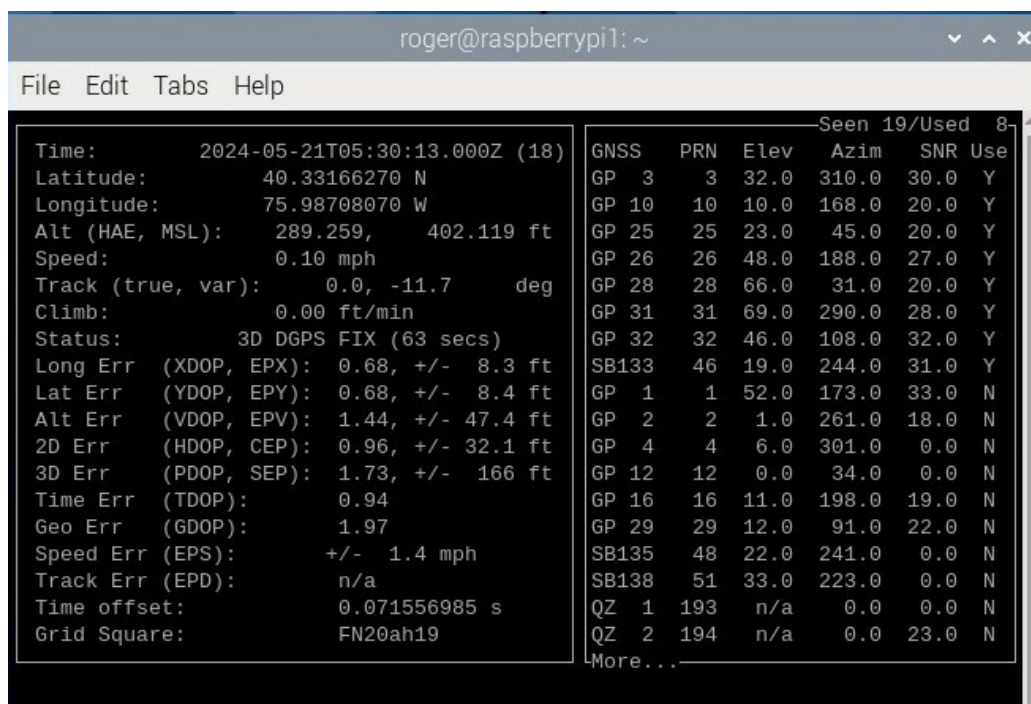
systemctl is-active gpsd

The terminal should respond with the word “active” on the next line.

Then type **systemctl is-active chronyd**

The terminal should again respond with the word “active” on the next line.

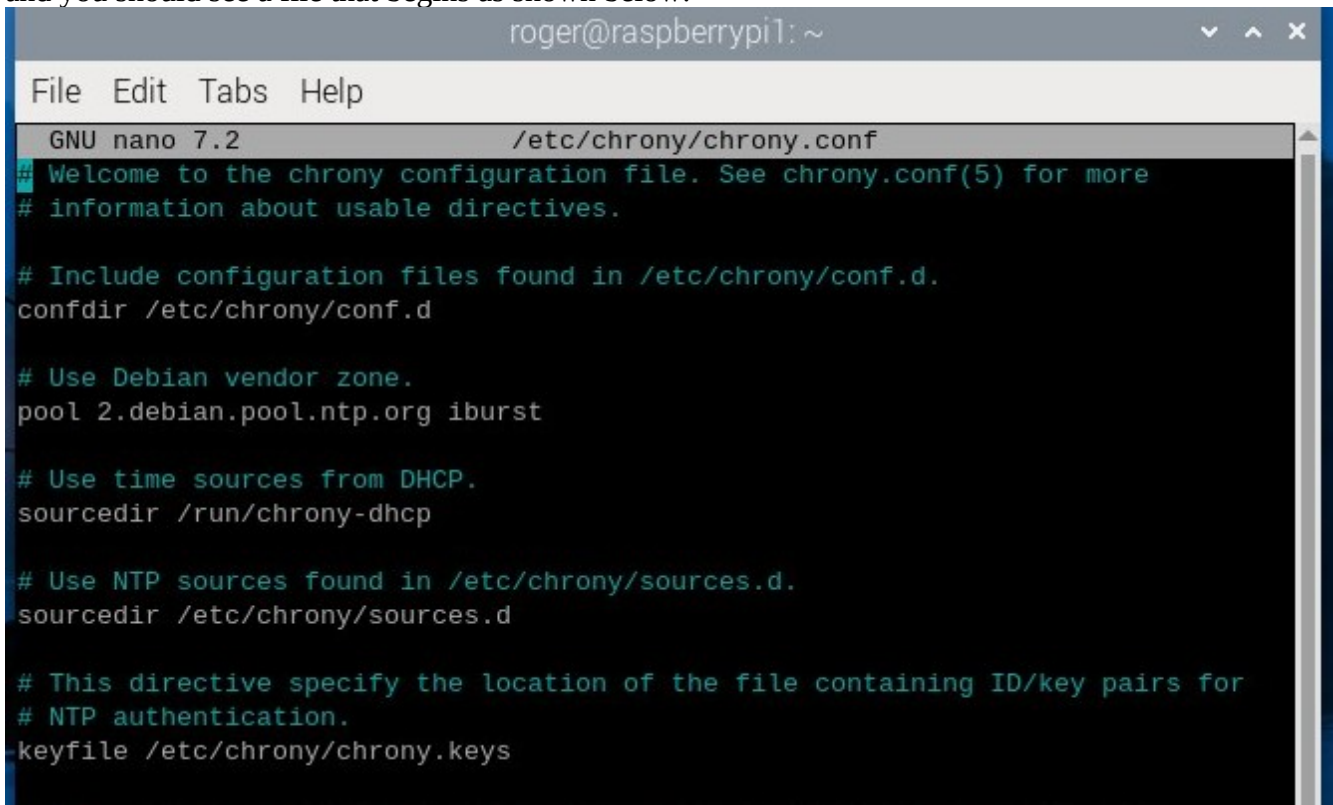
Then type **cgps -s** and the terminal window should appear similar to below:



The screenshot shows a terminal window titled "roger@raspberrypi1: ~" with a menu bar (File, Edit, Tabs, Help). The output of the 'cgps -s' command is displayed in two columns. The left column shows various GPS status and error metrics, and the right column shows a table of satellite data.

GNSS	PRN	Elev	Azim	SNR	Use
GP	3	32.0	310.0	30.0	Y
GP	10	10.0	168.0	20.0	Y
GP	25	23.0	45.0	20.0	Y
GP	26	48.0	188.0	27.0	Y
GP	28	66.0	31.0	20.0	Y
GP	31	69.0	290.0	28.0	Y
GP	32	46.0	108.0	32.0	Y
SB133	46	19.0	244.0	31.0	Y
GP	1	52.0	173.0	33.0	N
GP	2	1.0	261.0	18.0	N
GP	4	6.0	301.0	0.0	N
GP	12	0.0	34.0	0.0	N
GP	16	11.0	198.0	19.0	N
GP	29	12.0	91.0	22.0	N
SB135	48	22.0	241.0	0.0	N
SB138	51	33.0	223.0	0.0	N
QZ	1	193	n/a	0.0	N
QZ	2	194	n/a	0.0	N

Next, type **sudo nano /etc/chrony/chrony.conf**
and you should see a file that begins as shown below:



```
roger@raspberrypi1: ~
File Edit Tabs Help
GNU nano 7.2 /etc/chrony/chrony.conf
# Welcome to the chrony configuration file. See chrony.conf(5) for more
# information about usable directives.

# Include configuration files found in /etc/chrony/conf.d.
confdir /etc/chrony/conf.d

# Use Debian vendor zone.
pool 2.debian.pool.ntp.org iburst

# Use time sources from DHCP.
sourcedir /run/chrony-dhcp

# Use NTP sources found in /etc/chrony/sources.d.
sourcedir /etc/chrony/sources.d

# This directive specify the location of the file containing ID/key pairs for
# NTP authentication.
keyfile /etc/chrony/chrony.keys
```

modify the chrony configuration file by scrolling to the bottom of the file and adding the following line to the end of the configuration file:

refclock SHM 0 offset 0.5 delay 0.2 refid NMEA

Type **ctl-o** and hit “Enter” to save the modified file, and then type **ctl-x** to close the file.

Check that all is well by typing:

chronyc sources -v

You should see a window that looks something like below, and “NMEA” should be somewhere on the list. If you have internet access, then there will be other time sources as well.

```

roger@raspberrypi1:~ $ chronyc sources -v

.-- Source mode '^' = server, '=' = peer, '#' = local clock.
/ .- Source state '*' = current best, '+' = combined, '-' = not combined,
/      'x' = may be in error, '~' = too variable, '?' = unusable.
||
||                               .- xxxx [ yyyy ] +/- zzzz
||      Reachability register (octal) -.      | xxxx = adjusted offset,
||      Log2(Polling interval) --.      |      | yyyy = measured offset,
||                               \      |      | zzzz = estimated error.
||                               |      |      \
MS Name/IP address             Stratum Poll Reach LastRx Last sample
=====
#x NMEA                          0  4  377   19  -431ms[ -431ms] +/- 100ms
^* time.cloudflare.com           3  8  377  174  +240us[ +344us] +/- 15ms
^+ static.190.111.161.5.cli>     4  8  377  104  -867us[ -867us] +/- 17ms
^+ t1.time.gq1.yahoo.com         2  7  377  111  +300us[ +300us] +/- 40ms
^- 208.67.75.242                 3  8  377  111  +155us[ +155us] +/- 100ms
roger@raspberrypi1:~ $

```

IV. Add and Modify Beacon-Related Files

The following files will need to be added to your home directory which is /home/yourusername. For example, my home directory is /home/roger:

[bash script](#)

[BeaconPlayAudio3.py](#)

Your audio files. In this example, these are [W3SZ Beacon CW.wav](#), [W3SZ Beacon SSB3.wav](#), and [Q65 60C 140.wav](#).

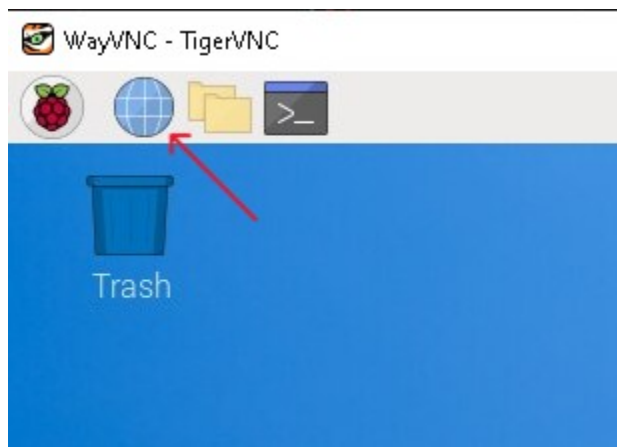
Samples of each of these files can be obtained by clicking on their embedded links above. However, if you don't yet know your way around the RPi the easiest way to put these files into the proper directory is likely to use the RPi's web browser to download the zip file at

<https://w3sz.com/pythonbeacon/webbeaconfiles.zip> then type the following commands into a terminal window, in each case substituting your username for "roger":

sudo mv /home/roger/Downloads/webbeaconfiles.zip /home/roger/

unzip -j webbeaconfiles.zip

You can start the browser from the Desktop by left-clicking the globe icon in the upper left corner of the Desktop:



The contents of the file `bash_script` are as below:

```
#!/bin/bash
/bin/sleep 30 && /usr/bin/python3 /home/roger/BeaconPlayAudio3.py &
```

Using nano as described several times above you need to change the text “roger” in this file to your username and then save the modified file.

Similarly, in the file `BeaconPlayAudio3.py` , where “roger” appears in the path for each of the sound files, you need to change it to your username.

You need to set the permissions on the bash script and python files so that they can be executed. Using a terminal window, move to your home directory where these files reside and type:

sudo chmod +x bash_script and then hit “Enter” and then type:

sudo chmod +x BeaconPlayAudio3.py and then hit “Enter.

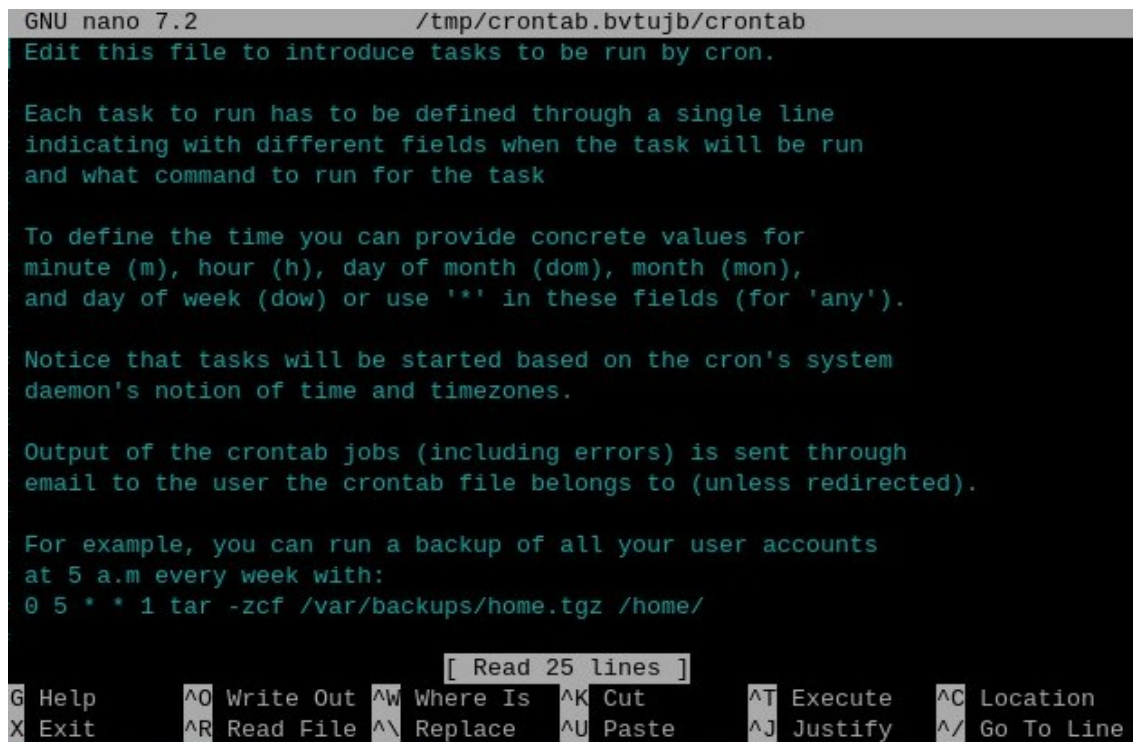
Note that `aplay` supports only `wav`, `raw`, `au`, and `voc` audio file types. It does NOT support `mp3` files. Also, remember that the `wav` file durations need to be shorter than the beacon intervals that you have defined. Here that interval is 60 seconds, and each of the 3 files is 58 seconds duration or less. Depending on the origin of your WSJT mode files, you may need use an app such as Audacity to shorten them by removing the “dead space” after the sequence has completed in order to achieve this.

V. Schedule the RPi to start the beacon (by running the file `bash_script`) with each reboot

We use `crontab` to schedule the running of `bash_script` with each boot of the RPi. To do this, first type in a terminal window:

crontab -e

This will bring up nano with the `crontab` schedule file loaded. The window will initially look like this:



```
GNU nano 7.2 /tmp/crontab.bvtujb/crontab
Edit this file to introduce tasks to be run by cron.

Each task to run has to be defined through a single line
indicating with different fields when the task will be run
and what command to run for the task

To define the time you can provide concrete values for
minute (m), hour (h), day of month (dom), month (mon),
and day of week (dow) or use '*' in these fields (for 'any').

Notice that tasks will be started based on the cron's system
daemon's notion of time and timezones.

Output of the crontab jobs (including errors) is sent through
email to the user the crontab file belongs to (unless redirected).

For example, you can run a backup of all your user accounts
at 5 a.m every week with:
0 5 * * 1 tar -zcf /var/backups/home.tgz /home/

[ Read 25 lines ]
G Help      ^O Write Out ^W Where Is  ^K Cut      ^T Execute  ^C Location
X Exit      ^R Read File ^\ Replace   ^U Paste    ^J Justify  ^_ Go To Line
```

Scroll to the bottom and add the line:

@reboot /home/roger/bash_script

remembering to change “roger” to your username.

Then type **ctl-o** and hit “Enter” to save the modified file, and then type **ctl-x** to close the file.

VI. Configure Audio Output Device

Because we are starting our script at boot we need to explicitly configure the audio output to use the 3.5 mm jack. To do this, we need to determine the device number by typing, from a terminal window, **cat /proc/asound/cards** and then hitting “Enter”. Doing this will produce a response like this:

```
roger@raspberrypi1:~ $ cat /proc/asound/cards
 0 [vc4hdmi0      ]: vc4-hdmi - vc4-hdmi-0
                        vc4-hdmi-0
 1 [vc4hdmi1      ]: vc4-hdmi - vc4-hdmi-1
                        vc4-hdmi-1
 2 [Headphones    ]: bcm2835_headpho - bcm2835 Headphones
                        bcm2835 Headphones
roger@raspberrypi1:~ $
```

Of course we don’t want devices 0 or 1 as they are both hdmi devices, so we want device 2.

We need to create (or modify, if it has already been created) the configuration file for asound. To do this, type **sudo nano /etc/asound.conf**

and then write the following 2 lines into the file, changing the device number if it is not 2 for your installation:

```
defaults.pcm.card 2
```

```
defaults.ctl.card 2
```

and then type **ctl-o** and hit “Enter” to save the modified file, and then type **ctl-x** to close the file.

You should already be a member of the audio group, but to make sure of this go ahead and type in a terminal window:

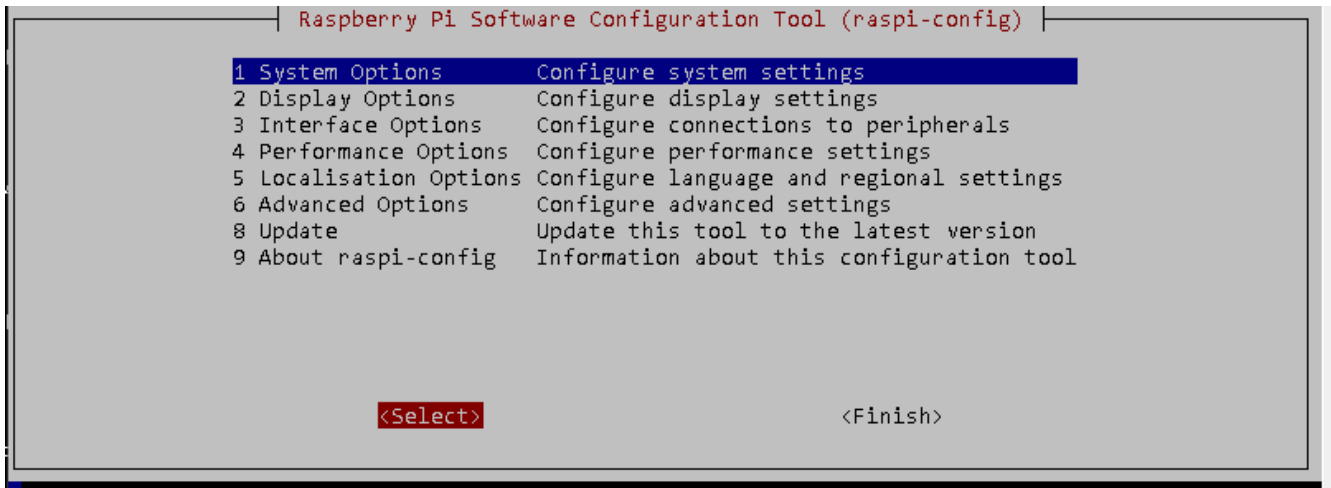
sudo adduser “username” audio where “username” is your username, which you must place in double quotes in this instance.

VII. Configure RPi to auto login to the console when booting

We saved this item until last because after you perform this step, you will boot into the console (terminal window) instead of into the Desktop. You need to have auto login enabled so that the script and audio can run automatically every time the RPi boots up. You want to log into the console rather than into the Desktop when the RPi is running headless at the beacon site to reduce overhead.

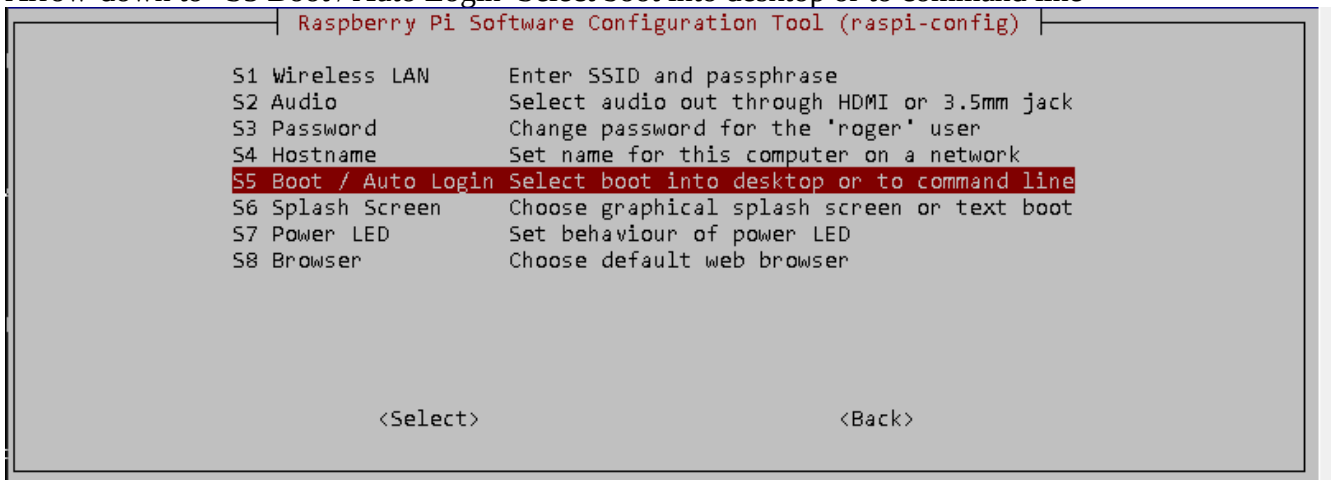
This step is done using the raspi-config utility, which is available from a terminal window, and raspi-config can be used to change this configuration detail as many times as you like.

First, from a terminal window type **sudo raspi-config** and you will see the window below:



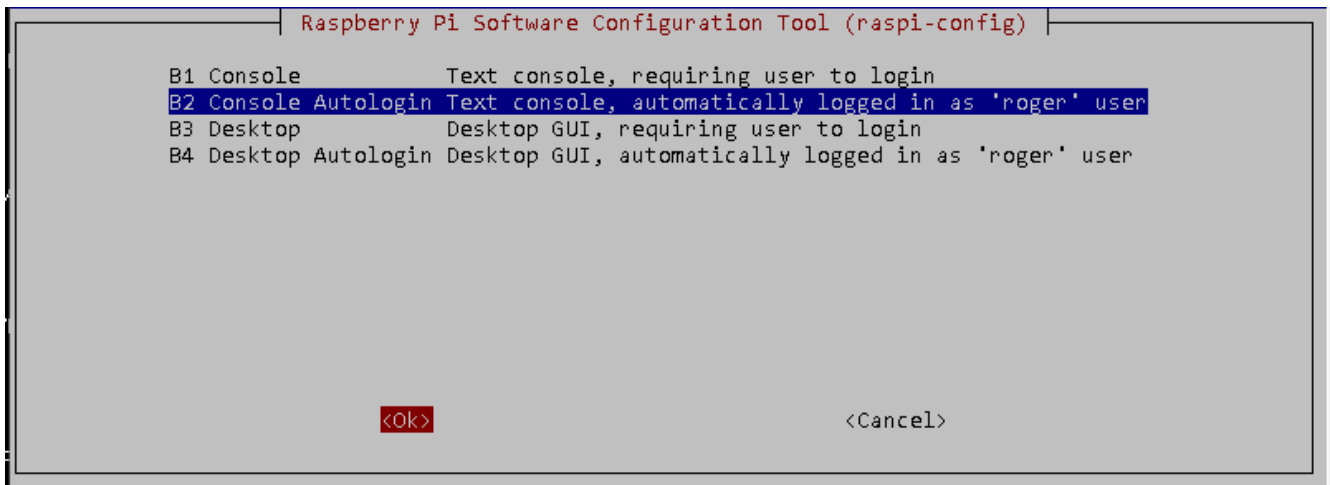
Hit “Tab” to highlight the <Select> icon, as shown above.
Then hit “Enter” and you will be presented with the window below:

Arrow-down to “S5 Boot / Auto Login Select boot into desktop or to command line”



Then hit “Tab” to highlight the <Select> icon.
Then hit “Enter”

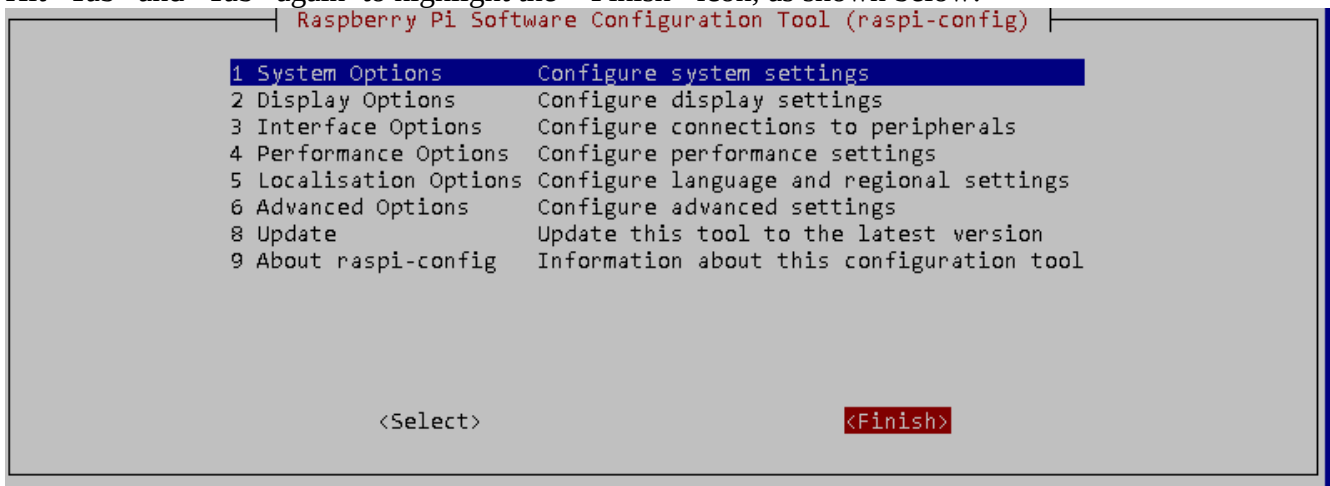
Then Arrow down to “B2 Console Autologin Text console, automatically logged in as ‘user’ user”



Then hit “Tab” to highlight the <OK> icon.
Then hit “Enter”

This will take you back to the main menu.

Hit “Tab” and “Tab” again to highlight the <Finish> icon, as shown below:



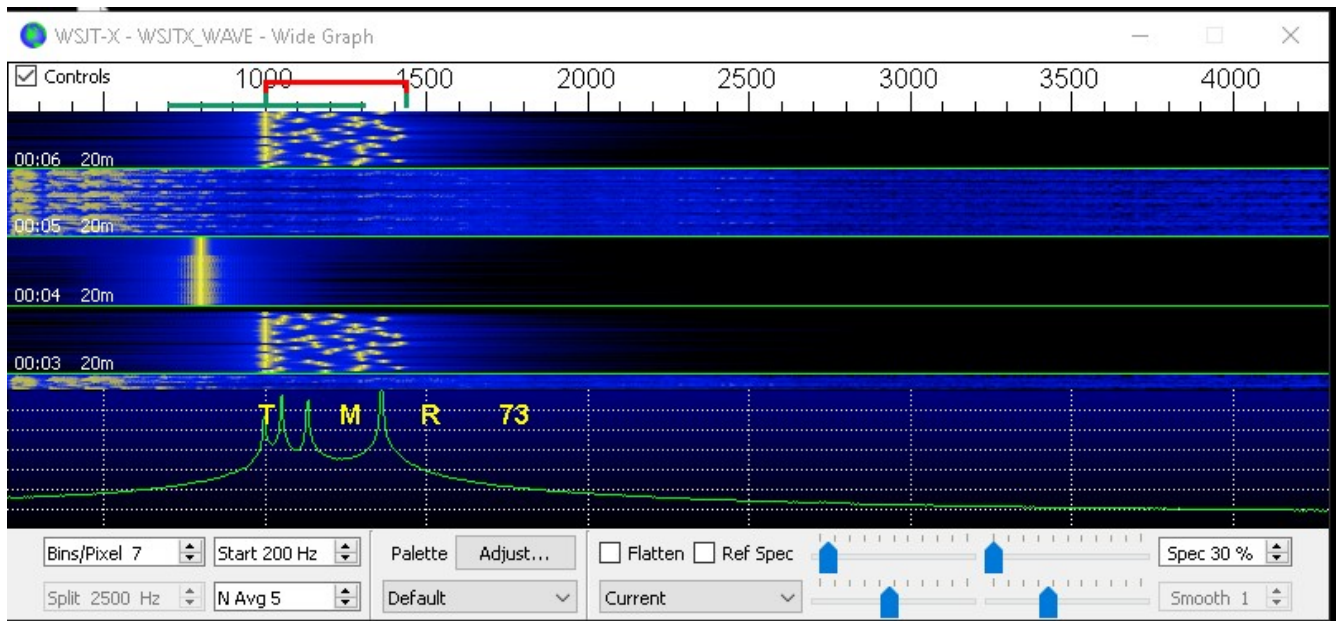
Then hit “Enter”. It may take raspi-config a few moments to save your changes before the terminal reappears.

That is all there is to setting up the RPi beacon audio player.

Simple Demonstration

I did a demonstration of this very simple script by connecting the audio output of the RPi running the script to the audio line input of a Windows 10 computer that was running an instance of WSJT-X set to decode Q65-60C using as audio source the line input.

You can see in the image below of the waterfall that on succeeding minutes the RPi beacon is sending Q65-60C then CW then SSB and then again Q65-60C:



A three-minute recording of the beacon audio output as received on the Windows 10 computer is [here](#).

The WSJT-X GUI below shows Q65-60C decodes occurring at 3 minute intervals, as expected:

WSJT-X - WSJT_X_WAVE v2.7.0-rc4 by K1JT et al.

File Configurations View Mode Decode Save Tools Help

Band Activity						Decodes containing My Call				
UTC	dB	DT	Freq	Message		UTC	dB	DT	Freq	Message
2342	10	0.1	1000	: W3SZ/B FN20AG	q0					
2345	14	0.1	1000	: W3SZ/B FN20AG	q0					
2348	11	0.1	1000	: W3SZ/B FN20AG	q0					
2351	13	0.1	1000	: W3SZ/B FN20AG	q0					
2354	3	0.2	1000	: W3SZ/B FN20AG	q0					
2357	4	0.4	1000	: W3SZ/B FN20AG	q0					
0000	4	0.4	1000	: W3SZ/B FN20AG	q0					
0003	3	0.4	1000	: W3SZ/B FN20AG	q0					
0006	5	0.4	1000	: W3SZ/B FN20AG	q0					
0009	6	0.1	1000	: W3SZ/B FN20AG	q0					
0012	6	0.1	1000	: W3SZ/B FN20AG	q0					
0015	5	0.1	1000	: W3SZ/B FN20AG	q0					
0018	7	0.1	1000	: W3SZ/B FN20AG	q0					
0021	5	0.1	1000	: W3SZ/B FN20AG	q0					
0024	9	0.1	1000	: W3SZ/B FN20AG	q0					

Log QSO Stop Monitor Erase Clear Avg Decode Enable Tx Halt Tx Tune Menus

14.074 000 Tx 1000 Hz F Tol 300 Submode C Rx 1000 Hz Max Drift 0 Report -15 T/R 60 s Sh Auto Seq CQ: None Tx6

Generate Std Msgs Next Now Pwr

2024 May 23 00:26:02

Receiving Q65-60C 46 47 2/60 WD:0m

Summary: A simple python / Raspberry Pi audio beacon controller with GPS-aligned transmit timing is described and detailed instructions for its implementation are given. The project is easily expanded to provide more complicated audio sequencing, and the RPi's GPIO could be used to generate CW rather than using the method described in this description.

Roger Rehr W3SZ
5/21/2024