VWS SDR Assembly Instructions

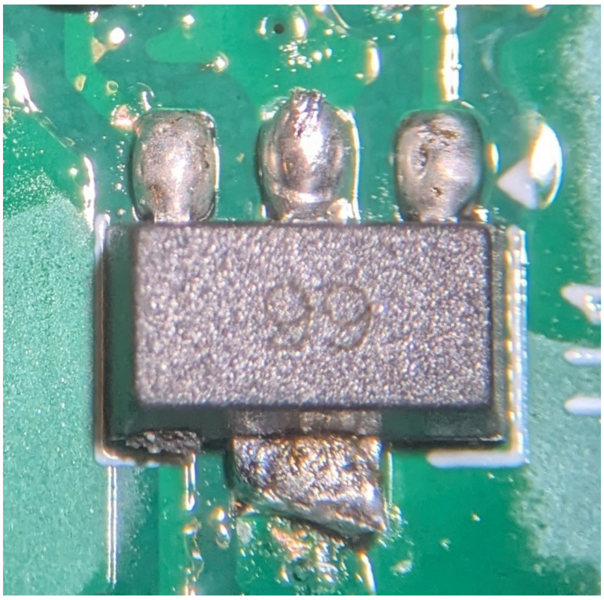
V2, 5 April 2025

When soldering surface mount components by hand, start with the lowest profile components – those that stick out the least from the PCB. Once all surface mount components are soldered, proceed to the through hole components. If a surface mount component has one end connected to ground, solder that end first.

Solder the surface mount components

1. Solder the SMD capacitors.
2. Solder the SMD resistors.
3. Solder the SMD inductors (L1 and FB).
4. Solder the semiconductors (U1 to U4)

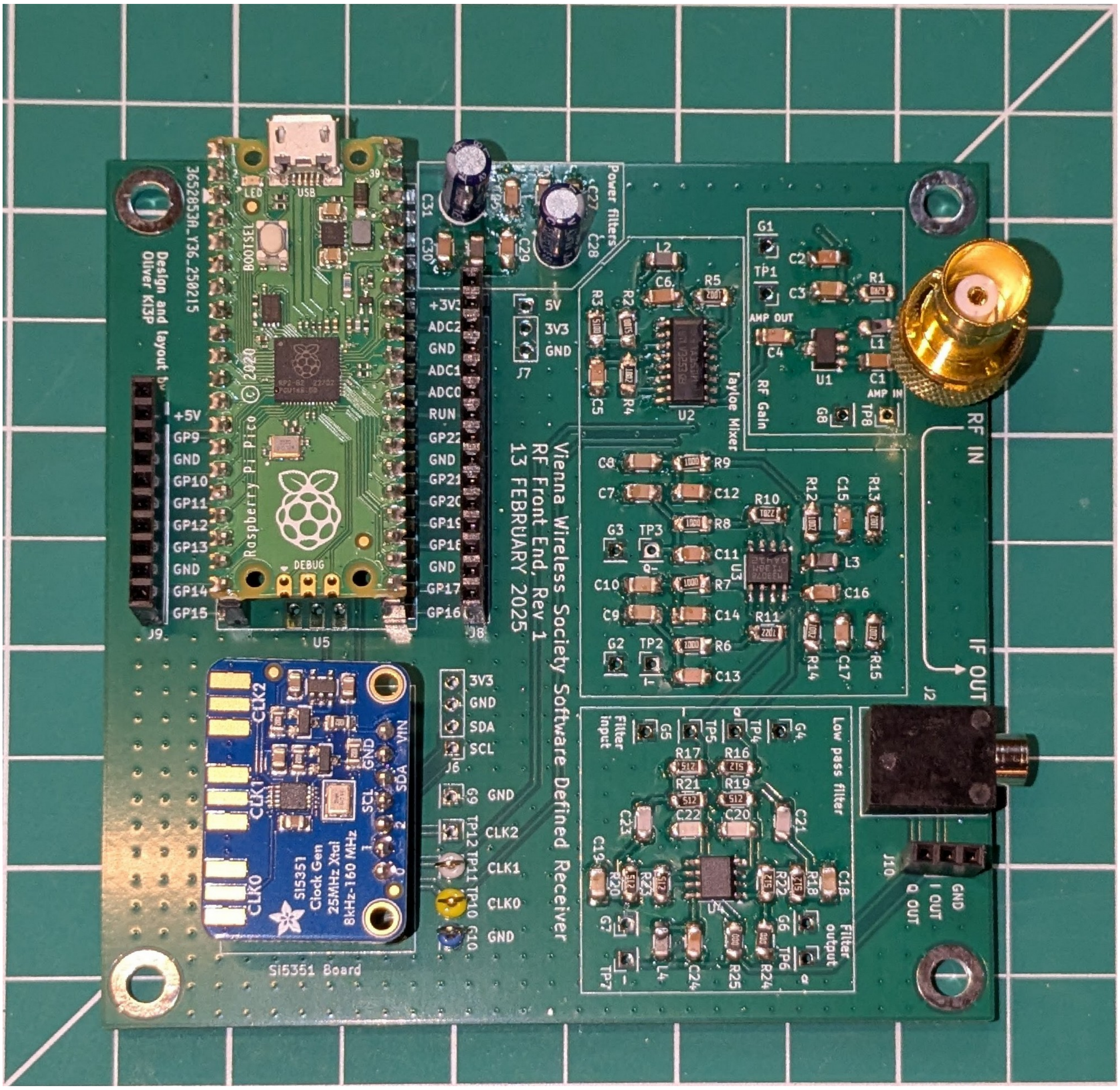
For U1, the GALI-S66 amplifier, first solder pin 1 (indicated by the triangle on the silkscreen), followed by the pin 3 pad on the opposite side of the component. Then solder the center pad, which is connected to ground. Have patience when soldering the grounded pads – you will need to apply heat for much longer to achieve good solder flow because of the heat flow from the pads into the ground plane.



Solder the through-hole components

1. Solder J2, the IF out connector.
2. Solder J3, the header pins for the Si5351 board.
3. If you want to use header sockets, solder them for the GPIO breakouts and the Pico. Otherwise, solder the header pins for the Pico.
4. Solder J1, the RF input SMA connector.

5. Solder C28 and C31, the through-hole capacitors.
6. Connect or solder the Pico and Si5351 boards.



Power on and test

The board should draw roughly 80 mA of current from the USB cable. Measure this with a USB current meter if you have one.

Pico sketch

If you have a VWS-SDR kit, the Pico has been pre-loaded with the appropriate Arduino sketch. To get the latest version, download it from the following link:
https://github.com/KI3P/VWS-SDR/tree/main/

Click the green box that says CODE and select Download Zip and save the file on the computer. Unzip the file to wherever you want – you'll need to navigate there to update the sketch.

Installing and configuring the Arduino IDE

If you don't have it first download and install the Arduino IDE:

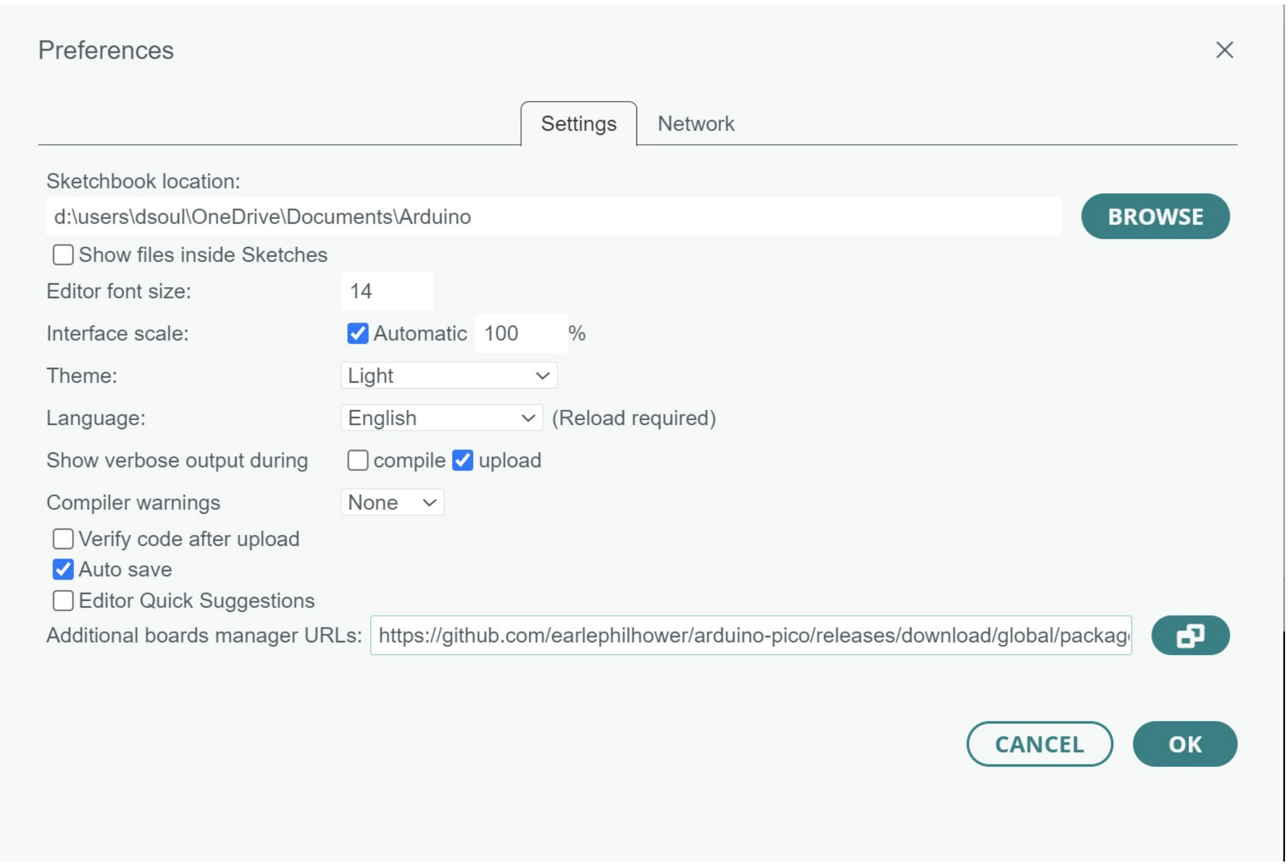
https://www.arduino.cc/en/software

Start the Arduino IDE.

Install the board manager for the Raspberry PI Pico.

In the File menu select Preferences. A dialog box will open. In the field "Additional Board Manager URLs:" cut and paste the following link and click OK.

https://github.com/earlephilhower/arduino-pico/releases/download/global/package_rp2040_index.json



Preferences

Sketchbook location: d:\users\dsoul\OneDrive\Documents\Arduino

□ Show files inside Sketches

Editor font size: 14

Interface scale: ☑ Automatic 100 %

Theme: Light

Language: English (Reload required)

Show verbose output during □ compile ☑ upload

Compiler warnings None

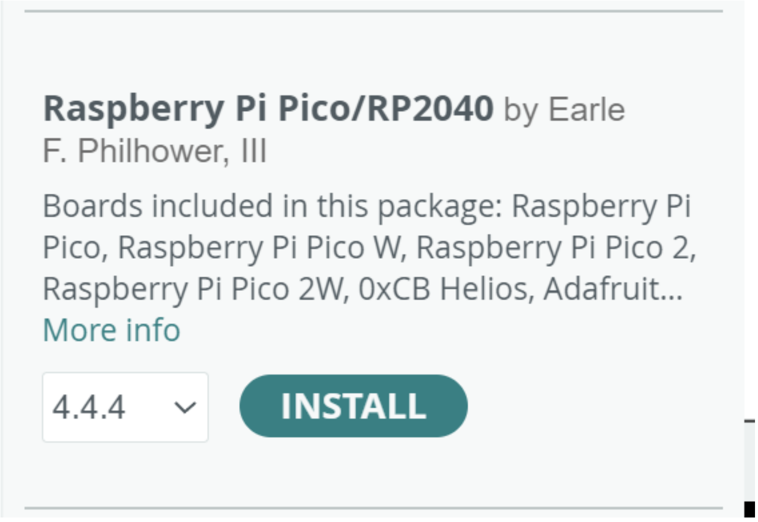
□ Verify code after upload

☑ Auto save

□ Editor Quick Suggestions

Additional boards manager URLs: https://github.com/earlephilhower/arduino-pico/releases/download/global/packag

Next, go to Tools -> Boards -> Board manager and search for Pico and a list of board managers will appear. The one you want is "Raspberry PI Pico/RP2040 by Earle F. Philhower III". Click INSTALL.



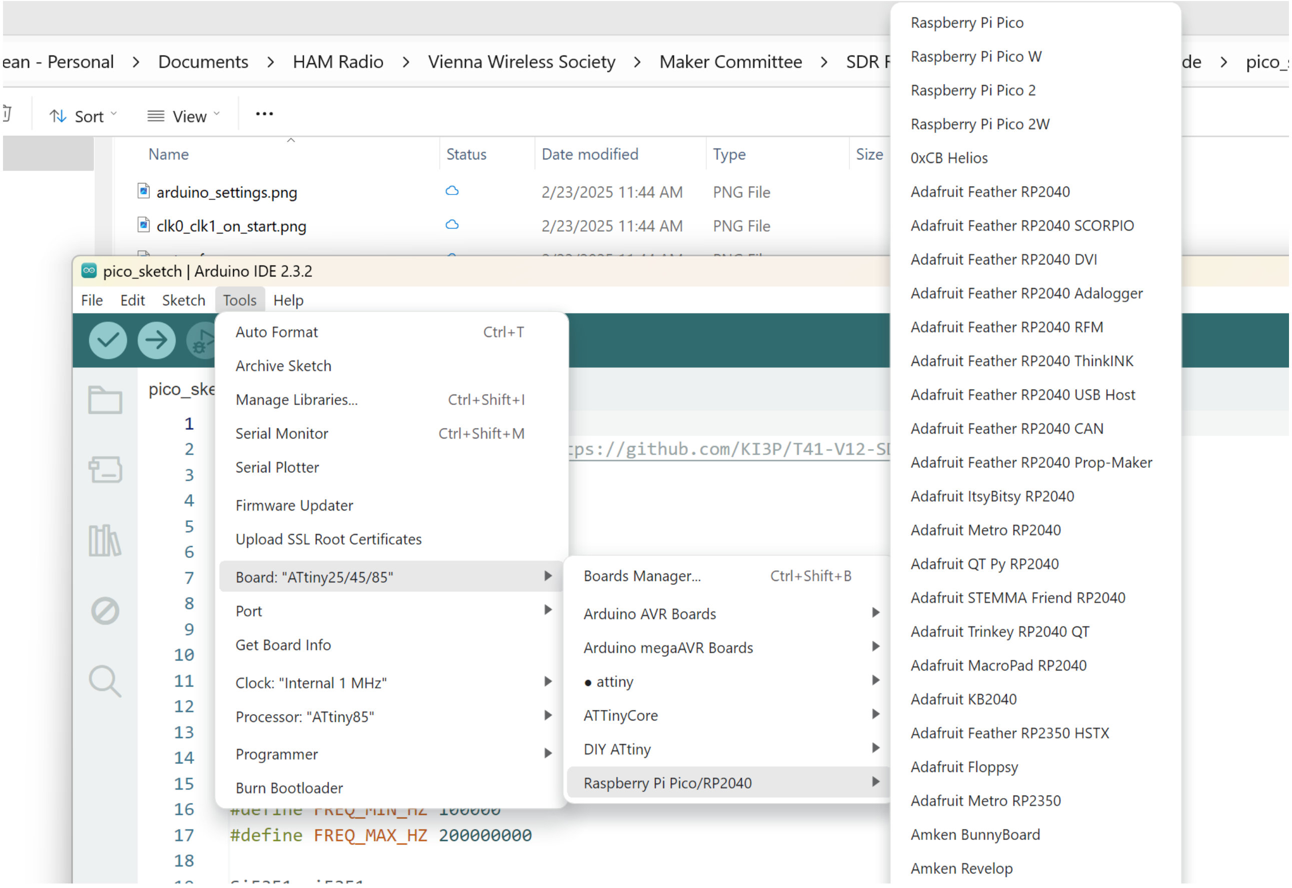
Raspberry Pi Pico/RP2040 by Earle F. Philhower, III

Boards included in this package: Raspberry Pi Pico, Raspberry Pi Pico W, Raspberry Pi Pico 2, Raspberry Pi Pico 2W, 0xCB Helios, Adafruit... More info

Now open the sketch. Navigate on your computer to the folder where you unzipped the code archive, find and open the pico sketch folder, then double click on pico_sketch.ino and the sketch will open in the Arduino IDE.

If you have not done so already, connect the Pico to your computer via a USB cable.

In the Arduino IDE go to Tools -> Board -> Board Manager -> Raspberry Pi Pico/RP2040 and select Raspberry Pi Pico.



Name	Status	Date modified	Type	Size
arduino_settings.png		2/23/2025 11:44 AM	PNG File	
clk0_clk1_on_start.png		2/23/2025 11:44 AM	PNG File	

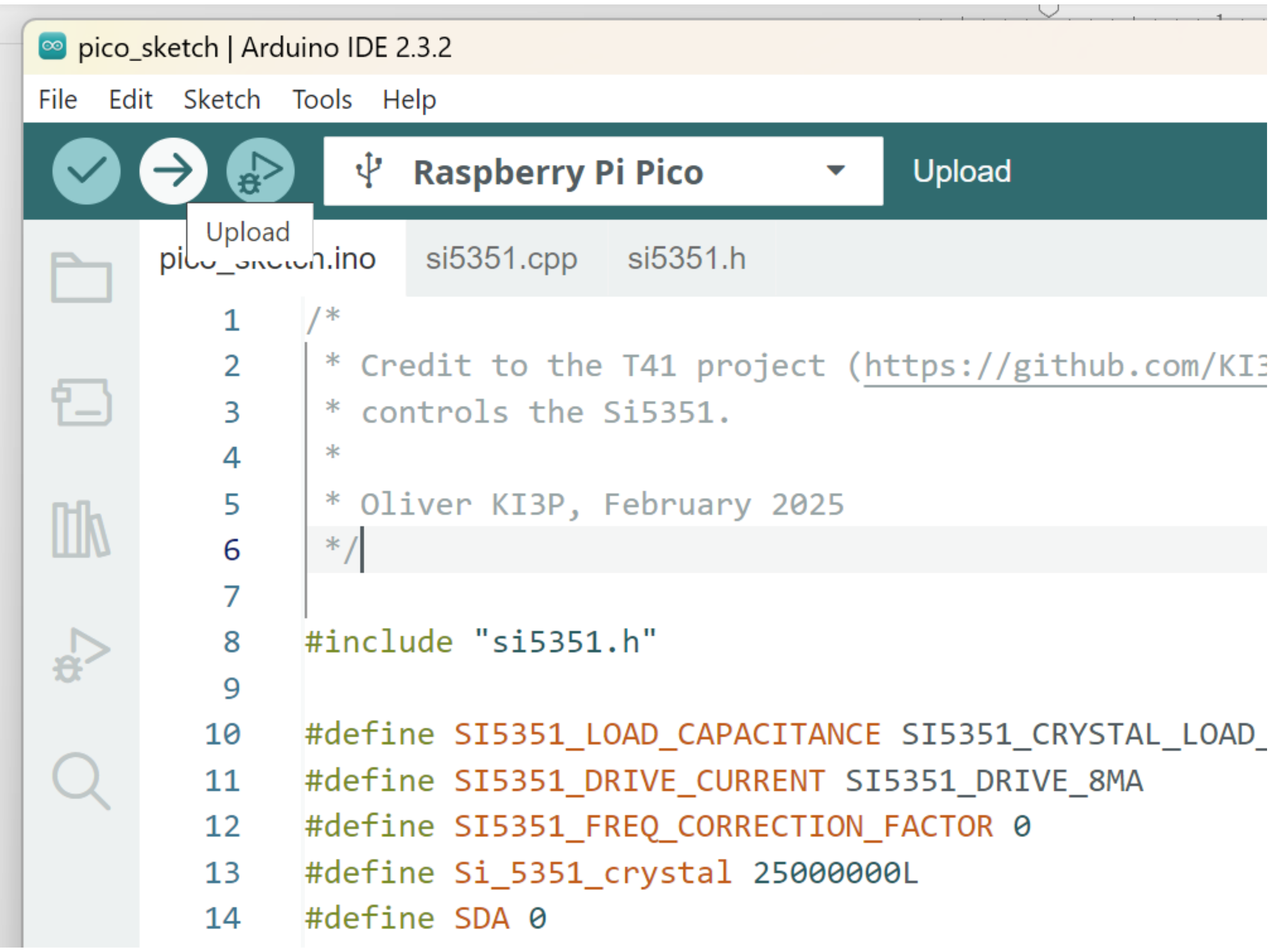
pico_sketch | Arduino IDE 2.3.2

```
#define FREQ_MAX_HZ 200000000
```

ean - Personal > Documents > HAM Radio > Vienna Wireless Society > Maker Committee > SDR F
de > pico_
Sort
View
...
File
Edit
Sketch
Tools
Help
pico_ske
tps://github.com/KI3P/T41-V12-SI
Auto Format Ctrl+T
Archive Sketch
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter
Firmware Updater
Upload SSL Root Certificates
Board: "ATtiny25/45/85"
Port
Get Board Info
Clock: "Internal 1 MHz"
Processor: "ATtiny85"
Programmer
Burn Bootloader
Boards Manager... Ctrl+Shift+B
Arduino AVR Boards
Arduino megaAVR Boards
● attiny
ATTinyCore
DIY ATtiny
Raspberry Pi Pico/RP2040
Raspberry Pi Pico
Raspberry Pi Pico W
Raspberry Pi Pico 2
Raspberry Pi Pico 2W
0xCB Helios
Adafruit Feather RP2040
Adafruit Feather RP2040 SCORPIO
Adafruit Feather RP2040 DVI
Adafruit Feather RP2040 Adalogger
Adafruit Feather RP2040 RFM
Adafruit Feather RP2040 ThinkINK
Adafruit Feather RP2040 USB Host
Adafruit Feather RP2040 CAN
Adafruit Feather RP2040 Prop-Maker
Adafruit ItsyBitsy RP2040
Adafruit Metro RP2040
Adafruit QT Py RP2040
Adafruit STEMMA Friend RP2040
Adafruit Trinkey RP2040 QT
Adafruit MacroPad RP2040
Adafruit KB2040
Adafruit Feather RP2350 HSTX
Adafruit Floppsy
Adafruit Metro RP2350
Amken BunnyBoard
Amken Revelop

Open the sketch in the Arduino IDE and configure Arduino to have the settings shown below under the Tools menu. For “Port” select whatever COM port appears when you plug the Pico into your computer with a USB cable:
* Tools -> Port and select the COM port connected to the Pico.

To compile and upload the sketch to Pico, click the Right arrow on the top toolbar in the IDE.

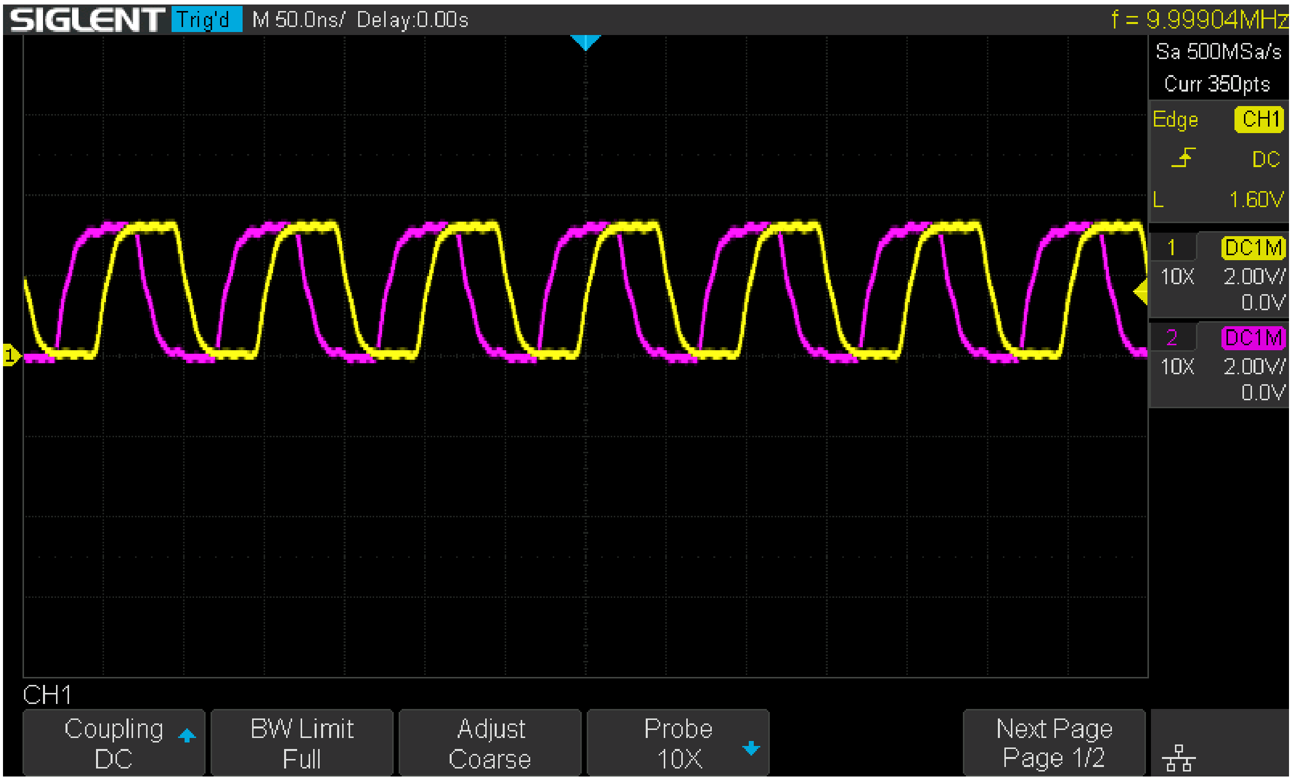


```
/*
 * Credit to the T41 project (https://github.com/KI3
 * controls the Si5351.
 *
 * Oliver KI3P, February 2025
 */

#include "si5351.h"

#define SI5351_LOAD_CAPACITANCE SI5351_CRYSTAL_LOAD_
#define SI5351_DRIVE_CURRENT SI5351_DRIVE_8MA
#define SI5351_FREQ_CORRECTION_FACTOR 0
#define Si_5351_crystal 25000000L
#define SDA 0
```

That’s it. You now have the latest sketch on the Raspberry PI Pico and you are ready to test and are all set for future updates.

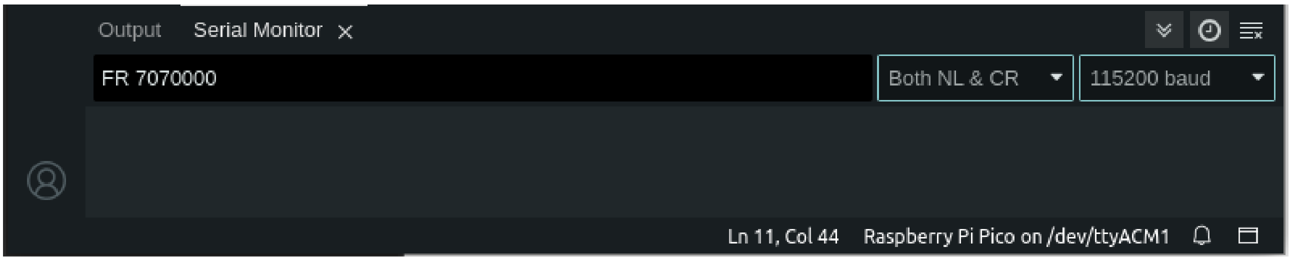
Testing

The Pico sketch configures the Si5351 to produce 10 MHz tones on start. You can confirm this by connecting an oscilloscope to the CLK0 and CLK1 outputs where you should see something like the screen shot below.



You can manually change the frequency using the Serial Monitor window in the Arduino. The Pico communicates at a baud rate of 115200 and expects commands in the form: “FR 123456” where 123456 is the frequency in Hz.

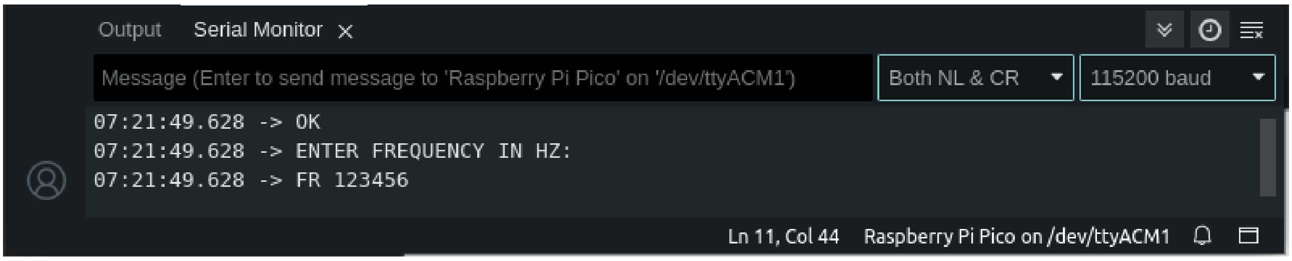
For instance, to tune the center of the band to 7070 kHz, enter the following command:



```
FR 7070000
```

Both NL & CR
115200 baud

You should get the reply “OK” and an invitation to enter a new frequency. The CLK0 and CLK1 outputs should now show an output at 7070 kHz.


Output
Serial Monitor
Message (Enter to send message to 'Raspberry Pi Pico' on '/dev/ttyACM1')
Both NL & CR
115200 baud
Ln 11, Col 44
Raspberry Pi Pico on /dev/ttyACM1

```
07:21:49.628 -> OK
07:21:49.628 -> ENTER FREQUENCY IN HZ:
07:21:49.628 -> FR 123456
```

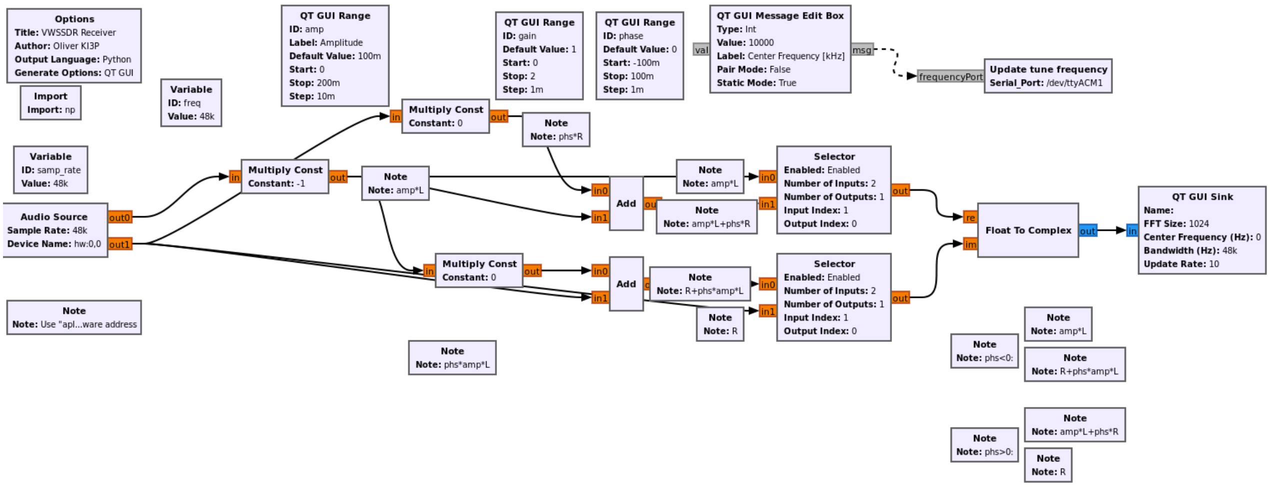
Demodulation

Connect the IF output (J2) to the microphone input on the sound card of your computer. We will use gnuradio to test whether the SDR is producing signals that can be demodulated. The gnuradio sketch can be found at the following link:

https://github.com/KI3P/VWS-SDR/tree/main/code/gnuradio_KI3P

To install gnuradio on your computer follow these instructions:

https://wiki.gnuradio.org/index.php/InstallingGR



Options
Title: VWSSDR Receiver
Author: Oliver KI3P
Output Language: Python
Generate Options: QT GUI

Import
Import: np

Variable
ID: freq
Value: 48k

Variable
ID: samp_rate
Value: 48k

Audio Source
Sample Rate: 48k
Device Name: hw:0,0

Note
Note: Use "apl...ware address

QT GUI Range
ID: amp
Label: Amplitude
Default Value: 100m
Start: 0
Stop: 200m
Step: 10m

Multiply Const
Constant: -1

Note
Note: amp*L

Multiply Const
Constant: 0

Note
Note: phs*R

Multiply Const
Constant: 0

Note
Note: phs*amp*L

QT GUI Range
ID: gain
Default Value: 1
Start: 0
Stop: 2
Step: 1m

QT GUI Range
ID: phase
Default Value: 0
Start: -100m
Stop: 100m
Step: 1m

QT GUI Message Edit Box
Type: Int
Value: 10000
Label: Center Frequency [kHz]
Pair Mode: False
Static Mode: True

Update tune frequency
Serial_Port: /dev/ttyACM1

Add

Note
Note: amp*L

Note
Note: amp*L+phs*R

Selector
Enabled: Enabled
Number of Inputs: 2
Number of Outputs: 1
Input Index: 1
Output Index: 0

Add

Note
Note: R+phs*amp*L

Note
Note: R

Selector
Enabled: Enabled
Number of Inputs: 2
Number of Outputs: 1
Input Index: 1
Output Index: 0

Float To Complex

QT GUI Sink
Name:
FFT Size: 1024
Center Frequency (Hz): 0
Bandwidth (Hz): 48k
Update Rate: 10

Note
Note: amp*L

Note
Note: phs<0:

Note
Note: R+phs*amp*L

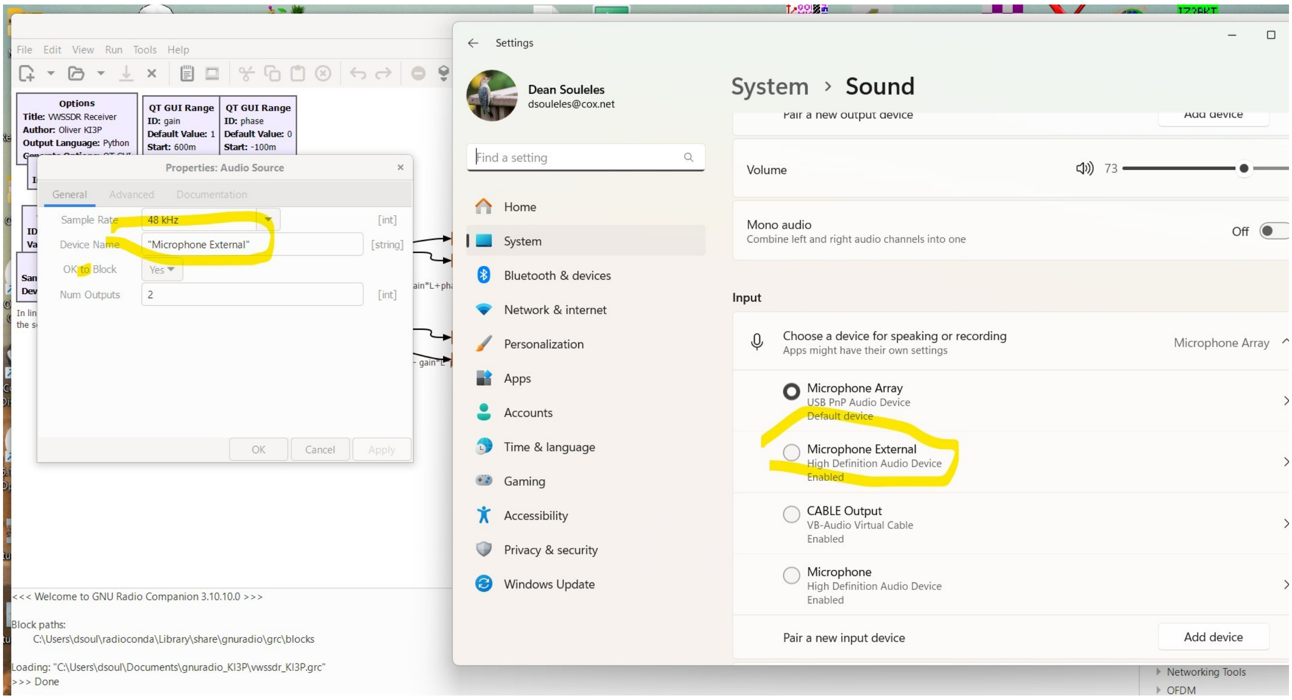
Note
Note: amp*L+phs*R

Note
Note: phs>0:

Note
Note: R

You will need to update two fields in the gnuradio sketch for your computer setup. The first is the address of the sound card in the Audio Source block on the left. By default it is set to the first audio device. This may work out of the box for your computer. If not, then you can use the “aplay -l” command in linux to find the hardware address of your sound card device.

To find the Microphone Input name in Windows, open Settings and go to System and then Sound. Look for the input devices and find the name windows uses for the device you are using and copy that into the GNU radio field. You can also rename the device to something there if that makes it easier.



Options
Title: VWSSDR Receiver
Author: Oliver KI3P
Output Language: Python

QT GUI Range
ID: gain
Default Value: 1
Start: 600m

QT GUI Range
ID: phase
Default Value: 0
Start: -100m

Properties: Audio Source

General | Advanced | Documentation

Sample Rate: 48 kHz [int]
Device Name: "Microphone External" [string]
OK to Block: Yes
Num Outputs: 2 [int]

<<< Welcome to GNU Radio Companion 3.10.10.0 >>>

Block paths:
    C:\Users\dsoul\radioconda\Library\share\gnuradio\grc\blocks

Loading: "C:\Users\dsoul\Documents\gnuradio_KI3P\vwssdr_KI3P.grc"
>>> Done

Settings

Dean Souleles
dsouleles@cox.net

System › Sound

Pair a new output device — Add device

Volume 73

Mono audio — Off
Combine left and right audio channels into one

Input

Choose a device for speaking or recording — Microphone Array
Apps might have their own settings

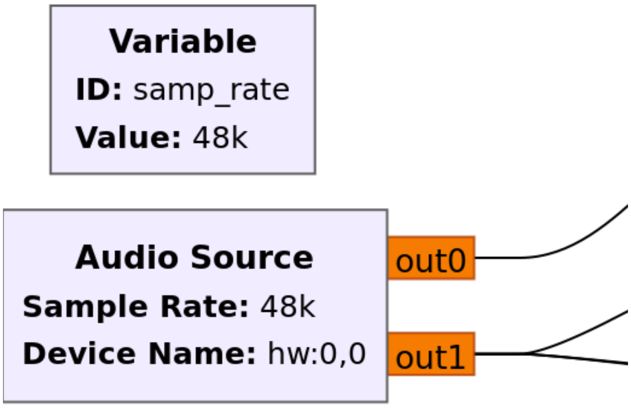
● Microphone Array
USB PnP Audio Device
Default device

○ Microphone External
High Definition Audio Device
Enabled

○ CABLE Output
VB-Audio Virtual Cable
Enabled

○ Microphone
High Definition Audio Device
Enabled

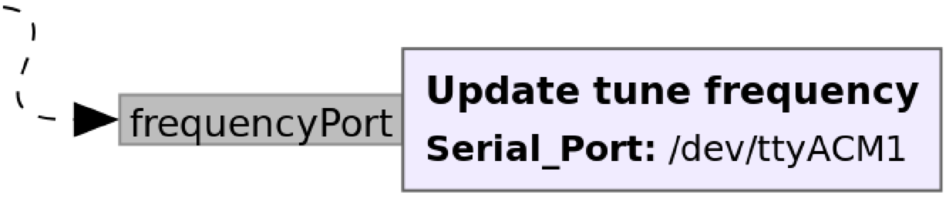
Pair a new input device — Add device



Variable
ID: samp_rate
Value: 48k

Audio Source
Sample Rate: 48k
Device Name: hw:0,0

The second field that will need to be updated is the serial COM port address of the Pico.

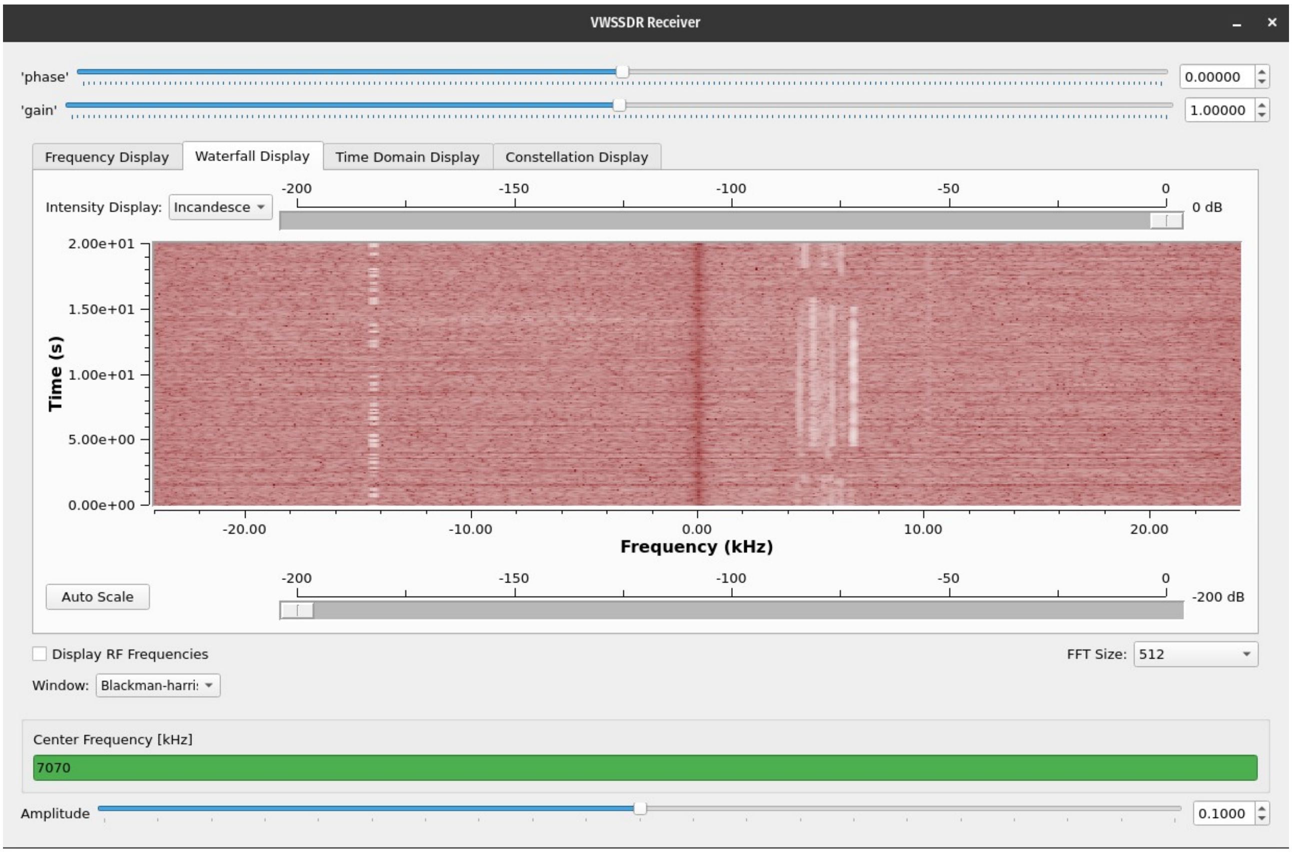


Update tune frequency
Serial_Port: /dev/ttyACM1

Enter whatever COM or serial port you used in the Pico sketch section above.

Once these fields are updated, launch the gnuradio sketch by clicking on Run → Execute or hitting the play button in the tool bar. Note that the Arduino IDE must be closed or you will get an error when both Arduino and gnuradio try to control the USB serial port.

Connect your antenna and change the center frequency to 7070 kHz in the green box. Change to the Waterfall Display tab and you should see the ever-present FT8.



VWSSDR Receiver

'phase' 0.00000

'gain' 1.00000

Intensity Display: Incandesce

Display RF Frequencies □

FFT Size: 512

Window: Blackman-harri:

Center Frequency [kHz]

7070

Amplitude 0.1000

Troubleshooting

(See separate testing guide for more)

1. Make sure the PC side of this is working. If you have a low frequency signal generator, use it to make a 5 kHz, 100 mV signal. Inject this into the audio cable that you're using to connect the SDR to your PC. You should see this tone in the gnuradio sketch.
2. If the above test works, then the problem must be on the board.
   a. Make sure that you're seeing 10 MHz signals on CLK0 and CLK1.
   b. Use a TinySA or RF signal generator to generate a -53 dBm signal at 10.005 MHz and inject it into J1 (this equates to 20dB over S9).
   c. Using an oscilloscope, start tracing the signal. I see the following:
      i. TP6: 1.1V peak to peak (380mV RMS) sine at ~8 kHz (my LO and generator have difference clocks, to the IF frequency not the expected 10 kHz)
      ii. TP4: 530 mV peak to peak (190mV RMS) sine at ~8 kHz.
      iii. TP3: ~30 mV peak to peak sine at ~8 Hz.
   d. Oscilloscopes like mine are too insensitive to probe weak RF signals, so a spectrum analyzer is needed if the problem lies in the pre-mixer part.

Signal 10.005MHz at -23dBm into J1
TP 8 & 1 -scope 100ns, 50mv scale

TP8: ??mV s/b 14mVpp
TP1: 140mVpp – 20dBm
TP3: 220mVpp @ 2.914MHz (???)