

0. Boot computer from USB

- 0.0. Insert the USB drive in your PC.
- 0.1. Turn on your PC and keep pressing **F12** until bios GUI opens.
- 0.2. Select the USB as boot option.
(add picture)
- 0.3. Connect to the internet network advised.
(optional: add more instructions)

1. IIO Oscilloscope

- 1.1 Open a terminal (ctrl + alt + t)
- 1.2 Type "osc" and press enter

2. Transmit and receive a complex sinusoid with Python (Pluto)

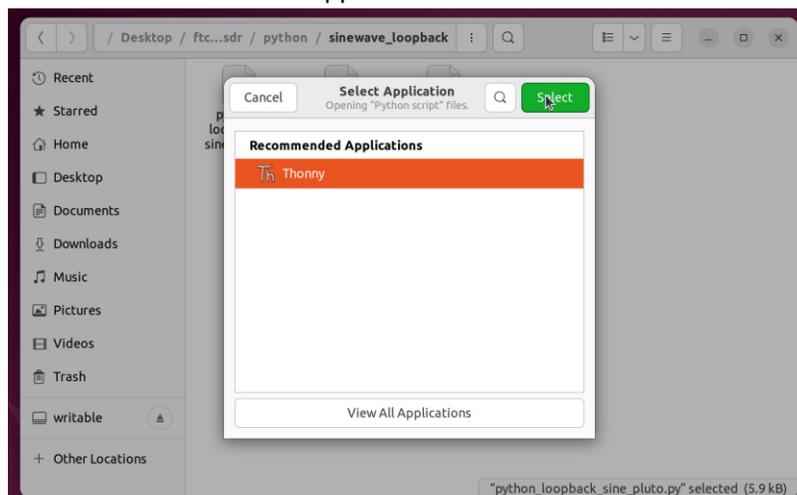
- 2.1. Make the following setup using Pluto and connect it to your PC:



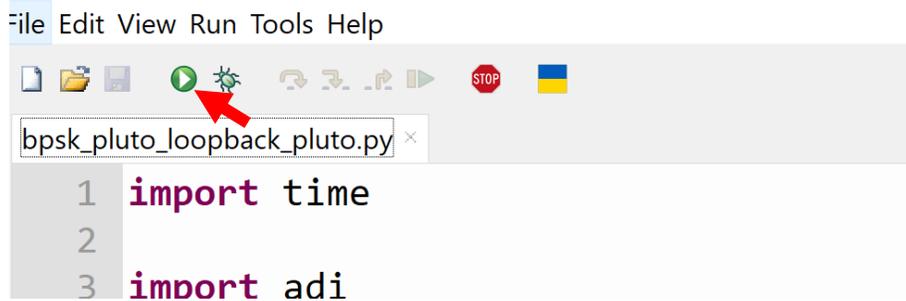
- 2.2. Open **FTC2023_SDR** folder from your desktop.

- 2.3. Go to **python** -> **sinewave_loopback** subfolder.

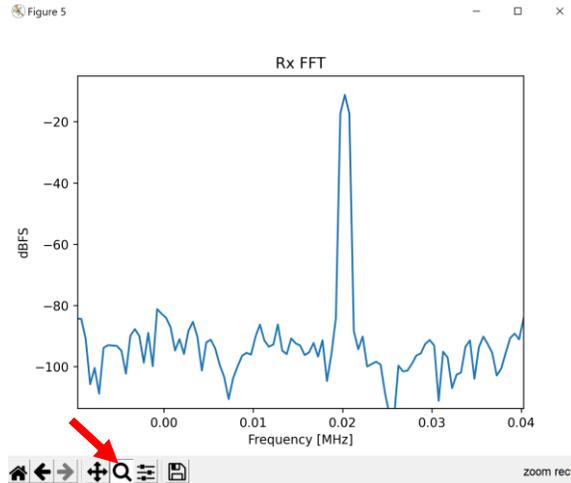
- 2.4. Right click on **python_loopback_sine_pluto.py** -> **Open With Other Application** and **select Thonny** from the Recommended Applications list.



2.5. To run the script, press the green round button from the top left corner in Thonny IDE as shown below.



2.6. To zoom on a plot, you can use the zoom option as depicted below. Encircle holding left click the area you want to zoom. On this example, if you zoom on Figure 5 on the received signal, you should see a harmonic at 20KHz.



2.7. To stop running the python script (if you want to run it again or run another script), close all the tabs by right clicking on taskbar icon -> Quit x windows as depicted below or close them manually one by one.



3. Transmit and receive a complex sinusoid with Python (Jupiter or Talise). Wait your turn 😊.

3.1. From `FTC2023_SDR` -> `python` -> `sinewave_loopback`, open `python_loopback_sine_jupiter.py` (for Jupiter) or `python_loopback_sine_talise_zu11eg.py` (for Talise) using Thonny as described in 2.4.

3.2. Observe in the script where only the “Configure SDR” code section was changed in `python_loopback_sine_jupiter.py` and `python_loopback_sine_talise_zu11eg.py` compared to `python_loopback_sine_pluto.py` from 2.

Jupiter (in `python_loopback_sine_jupiter.py`):

```
6 #####
7 # Configure SDR #####
8 #####
9 sdr = adi.adrv9002(uri="ip:10.48.65.212") # Create Radio
10
11 frequency = 15360 # 15.360 kHz sinewave to be transmitted
12 amplitude = 2**14
13 center_freq = 915000000 # Hz
14 sample_rate = sdr.rx0_sample_rate
15 print("sample rate: ", sample_rate)
16 num_samps = int((20*sample_rate)/frequency) # number of samples per call to rx()
17
18 sdr.rx_ensm_mode_chan0 = "rf_enabled"
19 sdr.rx_ensm_mode_chan1 = "rf_enabled"
20 sdr.tx_hardwaregain_chan0 = -5
21 sdr.tx_hardwaregain_chan0 = 5
22 sdr.tx_ensm_mode_chan0 = "rf_enabled"
23 sdr.tx_cyclic_buffer = True
24 sdr.tx_buffer_size = num_samps # number of samples per call to tx()
25 sdr.rx_buffer_size = num_samps
26 sdr.tx0_lo = center_freq
27 sdr.rx0_lo = center_freq
28 #####
29 #####
```

Talise (in `python_loopback_sine_talise_zu11eg.py`):

```
7 # Configure SDR #####
8 #####
9 center_freq = 915000000 # Hz
10
11 # Configure properties Talise
12 sdr = adi.adrv9009_zu11eg("ip:10.48.65.182") # Create Radio
13
14 frequency = 245760 # 245.760 kHz
15 amplitude = 2**14
16 sample_rate = sdr.rx_sample_rate
17 num_samps = int((20*sample_rate)/frequency) # number of samples per call to rx()
18 print("num_samps: ", num_samps)
19
20 sdr.rx_enabled_channels = [2]
21 sdr.tx_enabled_channels = [0]
22 sdr.trx_lo = center_freq
23 sdr.trx_lo_chip_b = center_freq
24 sdr.tx_cyclic_buffer = True
25 sdr.gain_control_mode_chan0 = "manual"
26 sdr.gain_control_mode_chan1 = "manual"
27 sdr.tx_hardwaregain_chan0 = -10
28 sdr.tx_hardwaregain_chan1 = -80
29 sdr.tx_hardwaregain_chan0_chip_b = -80
30 sdr.tx_hardwaregain_chan1_chip_b = -80
31 sdr.gain_control_mode_chan0_chip_b = "manual"
32 sdr.rx_hardwaregain_chan0_chip_b = 10
33 sdr.rx_buffer_size = num_samps
34 sdr.tx_buffer_size = num_samps
35
36 print("Syncing")
37 sdr.mcs_chips()
38 print("Done syncing")
39 print("Calibrating")
40 sdr.calibrate_rx_qec_en = 1
41 sdr.calibrate_rx_qec_en_chip_b = 1
42 sdr.calibrate_tx_qec_en = 1
43 sdr.calibrate_tx_qec_en_chip_b = 1
44 sdr.calibrate_rx_phase_correction_en_chip_b = 1
45 sdr.calibrate_rx_phase_correction_en = 1
46 sdr.calibrate = 1
47 sdr.calibrate_chip_b = 1
48 print("Done calibrating")
```

- 3.3. Insert the ip given for Jupiter on line 12 (in `python_loopback_sine_jupiter.py` or `python_loopback_sine_talise_zu11eg.py`) to remotely connect to Jupiter or Talise.

Jupiter (in `python_loopback_sine_jupiter.py`) :

```
12 sdr = adi.adrv9002(uri="ip:10.48.65.212") # Create Radio
```

Talise (in `python_loopback_sine_talise_zu11eg.py`):

```
12 sdr = adi.adrv9009_zu11eg("ip:10.48.65.182") # Create Radio
```

- 3.4. Run the script as indicated in 2.5. In this example, a sinusoid at 15.360 KHz is transmitted and received. This frequency is chosen to have an integer number of periods of the sine at the sample rate of 15.36 Msps. More specifically, line 16 (depicted below) should return an integer number.

```
16 num_samps = int((20*sample_rate)/frequency) # number of samples per call to rx()
```

- 3.5. To zoom on the generated Figures, use the instructions given at 2.6.

- 3.6. To stop running the python script or if you want to run it again, use the instructions given at 2.7.

4. Doppler Radar with GNU Radio (Pluto)

- 4.1. Make the following setup using Pluto and connect it to your PC:

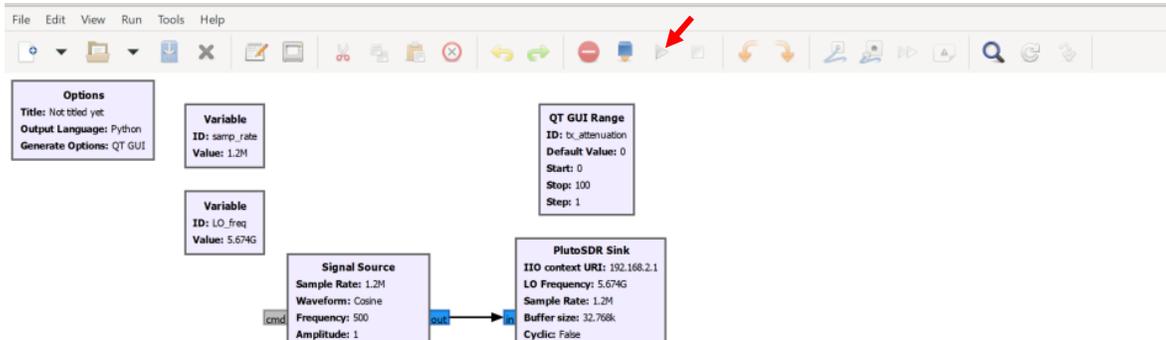


- 4.2. Open the terminal by pressing at the same time Ctrl + Alt + T.

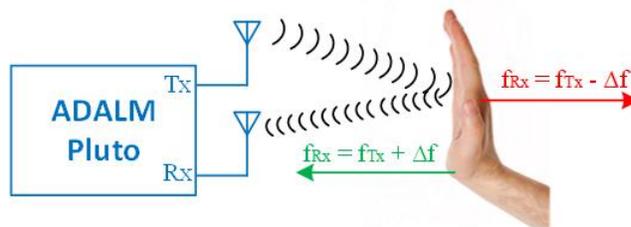
- 4.3. Type on the terminal `gnuradio-companion` and press Enter.

4.4. In gnuradio companion, click on the **File** -> **Open**, and open from **Desktop** -> **FTC2023_SDR** -> **gnuradio, doppler_radar.grc** file.

4.5. To run the flowgraph, press the grey arrow from the top function bar as indicated below.



4.6. Move your hand toward and away from the antennas and observe how the sound and the frequency of the received harmonic changes in the real time plots.



5. Binary Shift Keying in Python (Pluto)

5.1. Make the following setup using Pluto and connect it to your PC:

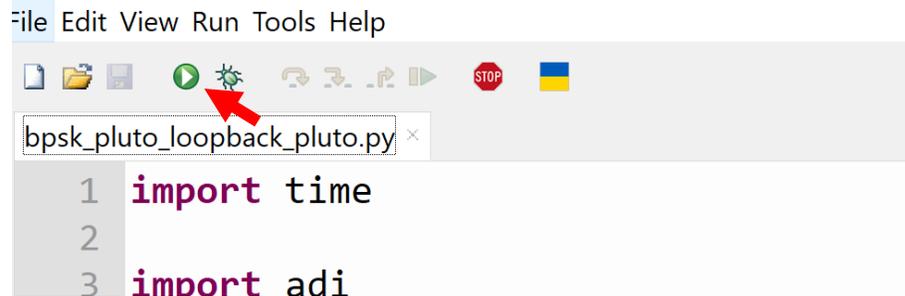


5.2. Open **FTC2023_SDR** folder from your desktop.

5.3. Go to **python** -> **bpsk_loopback** subfolder

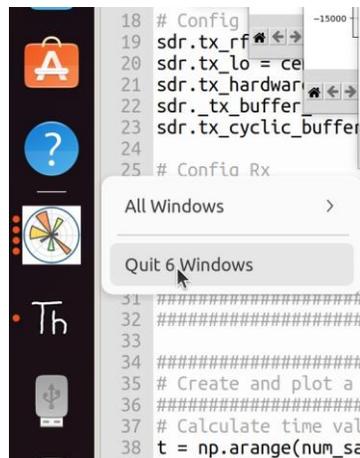
5.4. Right click on **bpsk_pluto_loopback_pluto.py** -> **Open With Other Application** and **select Thonny** from the Recommended Applications list.

5.5. To run the script, press the green round button from the top left corner as shown below.



5.6. To zoom on a plot, use the instructions from 2.6.

5.7. To stop running the python script (if you want to run it again), close all the tabs by **right click on taskbar icon** -> **Quit x windows** as depicted below or close them manually one by one.



6. Binary Shift Keying in Python (Jupyter and Talise)

6.1. From **FTC2023_SDR** -> **python** -> **bpsk_loopback** folder, open **bpsk_pluto_loopback_jupiter.py** (for **Jupyter**) or **bpsk_pluto_loopback_talise_zu11eg.py** (for **Talise**) using **Thonny** as in 2.4.

6.2. Observe in the script that only the Configure SDR portion was changed in **bpsk_pluto_loopback_jupiter.py** and **bpsk_pluto_loopback_talise_zu11eg.py** compared to **bpsk_pluto_loopback_pluto.py** from 5.

Jupyter (in **bpsk_pluto_loopback_jupiter.py**):

```

8 #####
9 # Configure SDR #####
10 #####
11 # Configure properties #####
12 sdr = adi.adrv9002(uri="ip:10.48.65.212") # Create Radio
13
14 center_freq = 915000000 # Hz
15 sample_rate = sdr.rx0_sample_rate
16 print("sample rate: ", sample_rate)
17
18 sdr.rx_ensm_mode_chan0 = "rf_enabled"
19 sdr.rx_ensm_mode_chan1 = "rf_enabled"
20 sdr.tx_hardwaregain_chan0 = -20
21 sdr.rx_hardwaregain_chan0 = 0
22 sdr.tx_ensm_mode_chan0 = "rf_enabled"
23 sdr.tx_cyclic_buffer = True
24 sdr.tx_buffer_size = 1024 # number of samples per call to tx()
25 sdr.rx_buffer_size = 32768
26 sdr.tx0_lo = center_freq
27 sdr.rx0_lo = center_freq
28
29 fs = int(sdr.rx0_sample_rate)
30 #####

```

Talise (in `bpsk_pluto_loopback_talise_zu11eg.py`):

```

8 #####
9 # Configure SDR #####
10 #####
11 # Configure properties Talise
12 sdr = adi.adrv9009_zu11eg("ip:10.48.65.182") # Create Radio
13 sdr.rx_enabled_channels = [2]
14 sdr.tx_enabled_channels = [0]
15 sdr.trx_lo = 914500000
16 sdr.trx_lo_chip_b = 914500000
17 sdr.tx_cyclic_buffer = True
18 sdr.gain_control_mode_chan0 = "slow_attack"
19 sdr.gain_control_mode_chan1 = "slow_attack"
20 sdr.tx_hardwaregain_chan0 = -20
21 sdr.tx_hardwaregain_chan1 = -80
22 sdr.tx_hardwaregain_chan0_chip_b = -80
23 sdr.tx_hardwaregain_chan1_chip_b = -80
24 sdr.gain_control_mode_chan0 = "slow_attack"
25 sdr.gain_control_mode_chan1 = "slow_attack"
26 sdr.gain_control_mode_chan0_chip_b = "slow_attack"
27 sdr.gain_control_mode_chan1_chip_b = "slow_attack"
28 sdr.rx_buffer_size = 32768
29 sdr.tx_buffer_size = 1024
30
31 fs = int(sdr.rx_sample_rate)
32 #####

```

- 6.3. Insert the ip given for Jupiter (in `bpsk_pluto_loopback_jupiter.py`) or Talise (`bpsk_pluto_loopback_talise_zu11eg.py`), on line 12 to remotely connect to Jupiter.

Jupiter (in `bpsk_pluto_loopback_jupiter.py`):

```

12 sdr = adi.adrv9002(uri="ip:10.48.65.212") # Create Radio

```

Talise (in `bpsk_pluto_loopback_talise_zu11eg.py`):

```

12 sdr = adi.adrv9009_zu11eg("ip:10.48.65.182") # Create Radio

```

- 6.4. Run the script as indicated in 2.5.
- 6.5. To zoom on the generated Figures, use the instructions given at 2.6.
- 6.6. To stop running the python script or if you want to run it again, use the instructions given at 2.7.

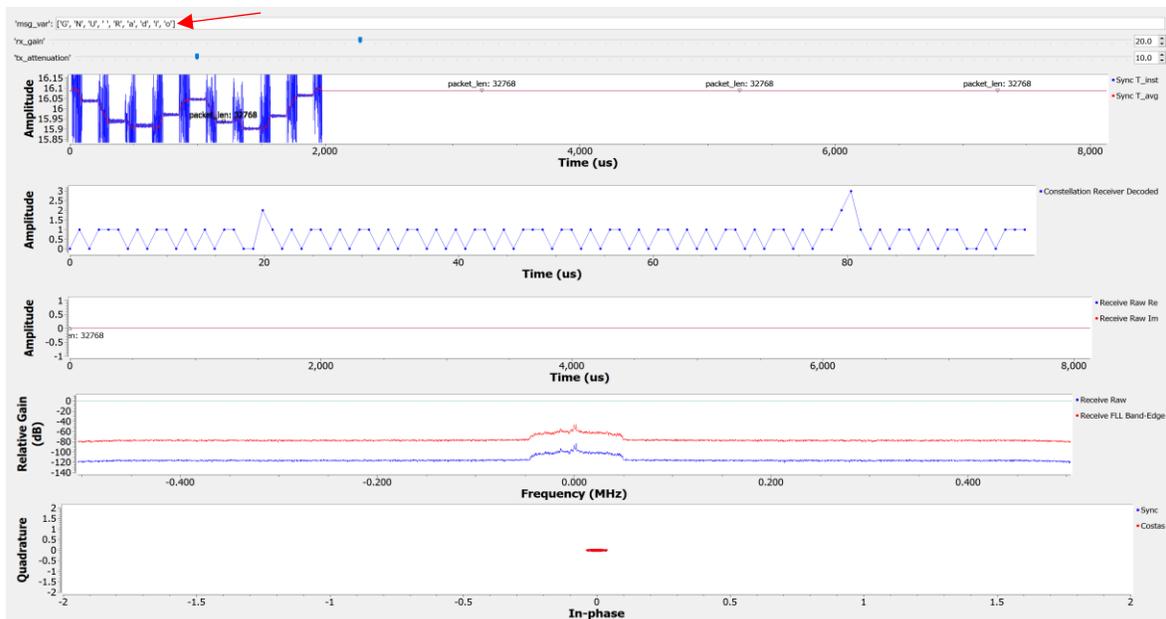
7. Quadrature Shift Keying in GNU Radio (Pluto)

- 7.1. Make the following setup using Pluto and connect it to your PC:



7.2. In gnu-radio companion, open `qpsk_point_to_point_rx_console.grc` from `FTC2023_SDR` -> `gnuradio` -> `qpsk_point_to_point_console`.

Run the flowgraph and observe the received message stored in “`msg_var`” variable as depicted below.



8. Receiving QPSK modulated Video with SDRs (Pluto)

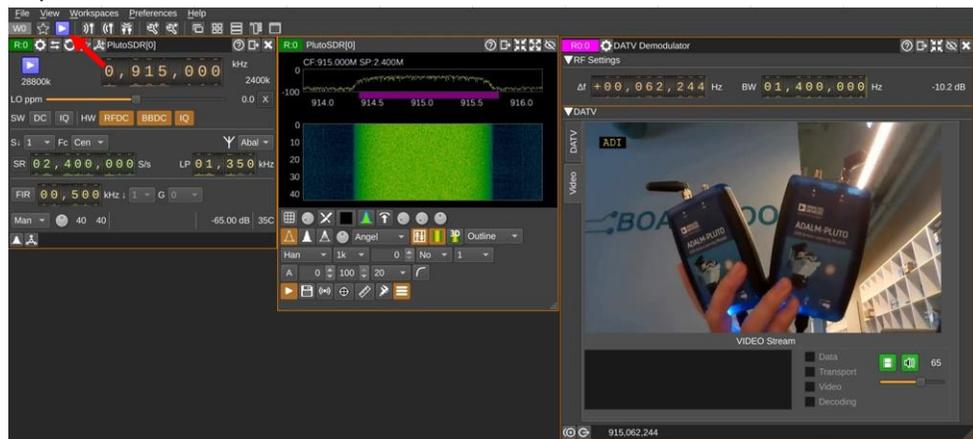
8.1. Make the following setup using Pluto and connect it to your PC:



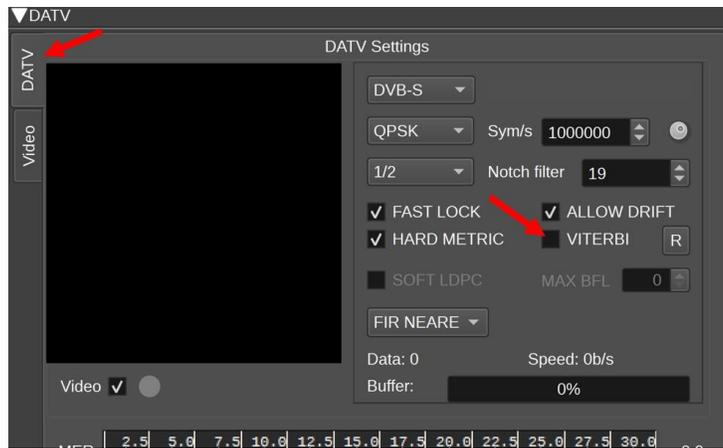
8.2. Open the terminal by pressing at the same time Ctrl + Alt + T.

8.3. Type on the terminal **sdrrangel** and press enter to open the application.

8.4. The application should be preconfigured. Run the program by pressing the purple arrow as depicted below.



8.5. On DATV window, **check and then uncheck Viterbi** to synchronize the qpsk data **while the program is running** (as shown below).



9. Spectrum Paint (Pluto)

9.1. Make the following setup using Pluto and connect it to your PC:



9.2. Open in gnuradio-companion **paint_tx.grc** from **FTC2023_SDR** -> **gnuradio** -> **spectrum_paint** and run the flowgraph. Observe the waterfall received.

