

## **Part I**

# **Software Defined Radio (SDR)**

# Workshop 5: Digital Modulation with SDR and GNU-radio

---

**Objective:** Constructing a digital modulation and demodulation scheme using GNU radio

**Outcome:** After successfully completing the experiment, the student will be able to

1. Identify the structure of a digital modulation/demodulation scheme
2. Identify impact of noise and hardware imperfection for a digital modulation/demodulation scheme

**Equipment and Components Required:**

1. Laptop or PC (Ubuntu OS is preferred) with following software
  - GNU radio
2. Blade RF SDR
3. RTL-SDR

Additional resources:-

- [Blade RF getting started guide](#)
- Setting up the environment
  1. [Bladerf pybombs installation guide](#)
  2. [Gr-osmosdr installation guide](#)
- GNU-radio tutorials
  1. [GRC, Sources, Sinks, Audio & GUI Blocks](#)
  2. [Hysteresis, Noise, Thresholds & Bit Error](#)
  3. [Types, Channel Selection & Graphical Sinks](#)
  4. [Phase-Shift Keying \(PSK\), Constellations & Auto-correlation](#)
  5. [FFT, Phase calculation, Vectors & GNU Plot](#)
  6. [Tutorial 1 | UVic ECE Communications Labs \(mistic-lab.github.io\)](#)  
[SDR Tutorials \(ettus.com\)](#)

## 5.1 Pre-Lab

This lab is continuation of the lab 04. Here, we are moving from simulation environment to SDR-based implementation

1. Setup GNU-radio environment based on the guidance given in Moodle (refer Blade RF SDR and GNU Radio under "Useful material for SDR" section)
2. Complete the [Guided Tutorial PSK Demodulation](#).

The following exercises will be graded.

**Task 1.** Change the noise voltage (a) from 0 to 0.25 and (b) from 0 to 0.70 and draw the constellation diagram. Comment on your observations.

**Task 2.** Change the frequency offset (a) from 0 to 0.0025 and (b) from 0 to 0.08 and draw the constellation diagram. Comment on your observations.

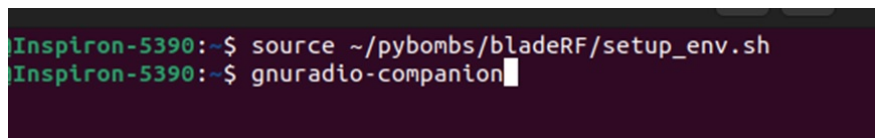
**Task 3.** Change the timing offset from 1 to 1.0008 and draw the constellation diagram. Comment on your observations.

## 5.2 Laboratory Experiment:-PSK Modulation and De-Modulation with SDR (Graded)

This lab is based on the "Guided Tutorial PSK Demodulation" available at [https://wiki.gnuradio.org/index.php?title=Guided\\_Tutorial\\_PSK\\_Demodulation](https://wiki.gnuradio.org/index.php?title=Guided_Tutorial_PSK_Demodulation). In this lab, we are replacing the simulation environment with SDR-based transmitter and receiver. First, we are going to replicate the pre-lab.

### 5.2.1 Transmitting a Signal

**Task 4.** The first step involves transmitting the QPSK signal. To do this, we generate a bit stream and modulate it onto a complex constellation. The Constellation Modulator block, with settings like Constellation Rect. Object, controls the transmitted signal. The Constellation parameter of the Constellation Modulator is linked to the Constellation Rect. Object (qpsk\_const), despite its display name. This constellation object defines how symbols are encoded. The modulator block can use this modulation scheme with or without differential encoding. The modulator expects packed bytes, so a random source generator supplies bytes ranging from 0 to 255. When selecting the number of samples per symbol, aim to keep it small (minimum of 2). This value helps match the desired bit rate to the hardware sample rate. We'll use a value of 4 (QPSK), which is more than necessary but aids signal visualization in various domains. Now, open the GNU-radio software. For that open a terminal by pressing "CTRL + ALT + T" or by right-clicking on the desktop and selecting Open in Terminal (Fig 5.1). Next type "gnuradio-companion" and enter to open GNU-radio. Now you will get a window like figure 5.2. If you are using PyBOMBS, you need to activate the prefix before typing above command. You can do this using this command: "source /my\_gnuradio/setup\_env.sh". Here, you need to replace /my\_gnuradio with the path to your prefix. Now, build the transmission block as shown in figure 5.3.



```
Inspiron-5390:~$ source ~/pybombs/bladeRF/setup_env.sh
Inspiron-5390:~$ gnuradio-companion
```

Figure 5.1: GNU-radio command line.

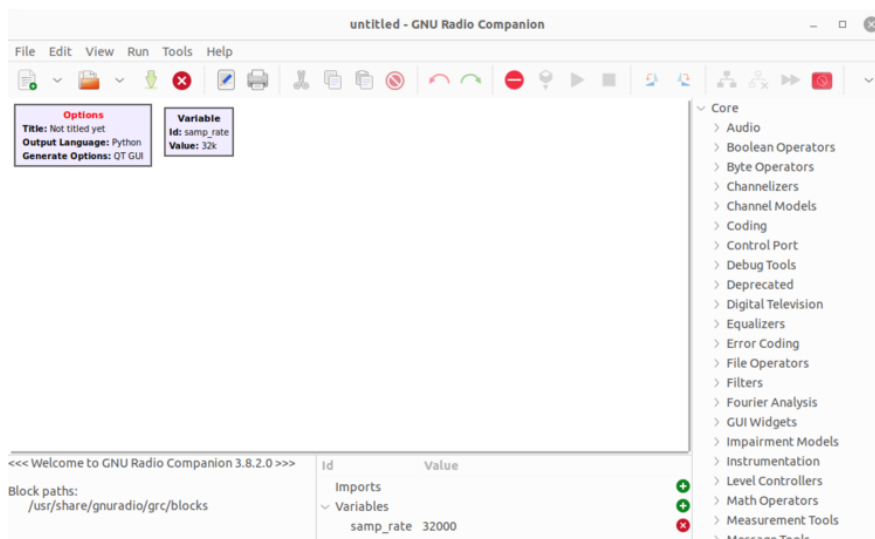


Figure 5.2: GNU-radio window.

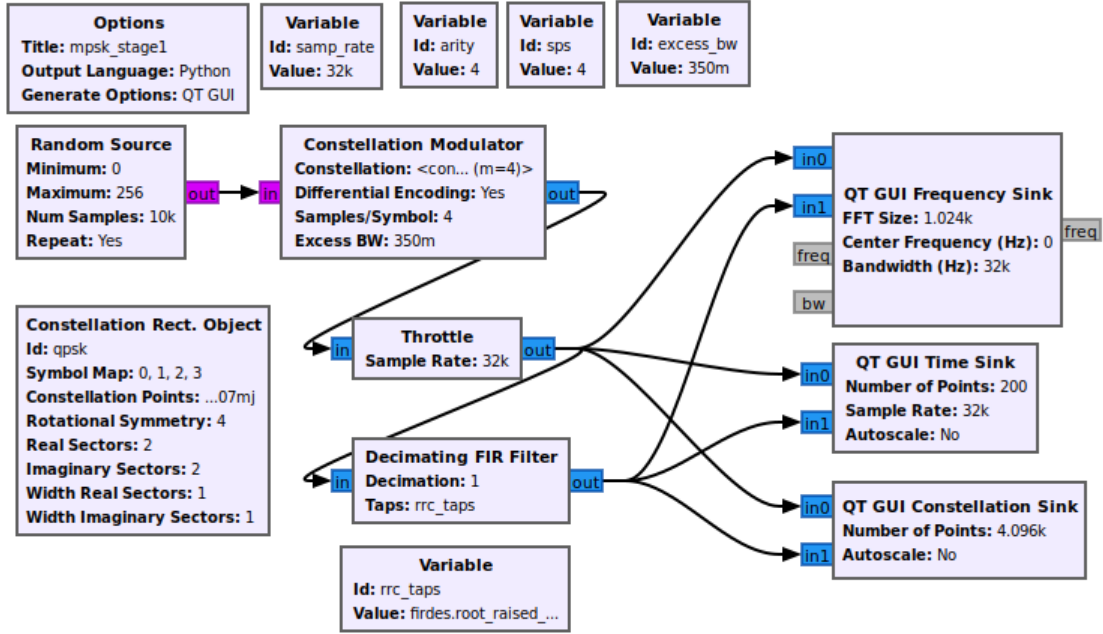


Figure 5.3: GNU-radio transmission block.

**Task 5.** Now, connect output of the constellation modulator to blade rf transmitter block (blade rf sink or soaphy sdr) as shown in figure 5.4. Set the transmitter frequency to a open band (i.e., 2.4 GHz band or 5 GHz band)

**Do not run the code** until you complete the receiver section. Next, we are going to implement the receiver side.

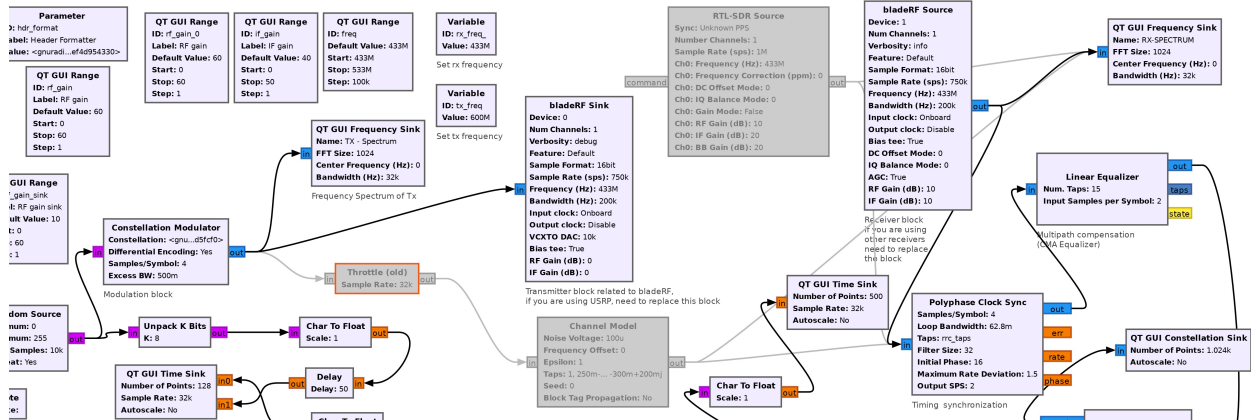


Figure 5.4: GNU-radio transmission block.

## 5.2.2 Receiving a Signal

**Task 6.** build the transmission block as shown in figure 5.5. Here, we need to replace the component in the black box with the SDR transmitter and receiver. For receiving, it is possible to use RTL SDR as well.

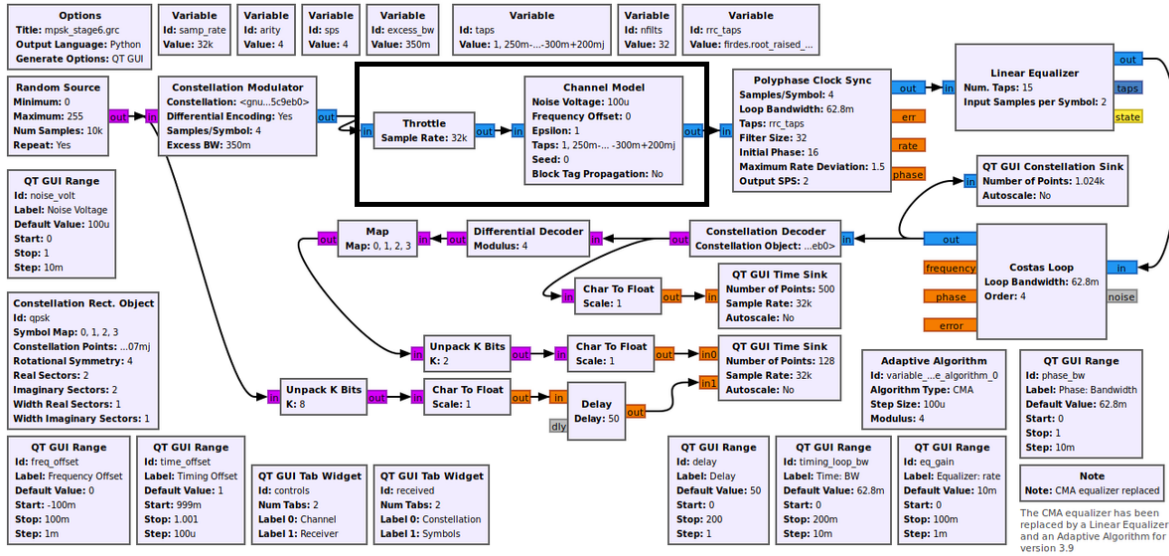


Figure 5.5: Transmitter and receiver block without SDR.

The modified block diagram is shown in 5.6

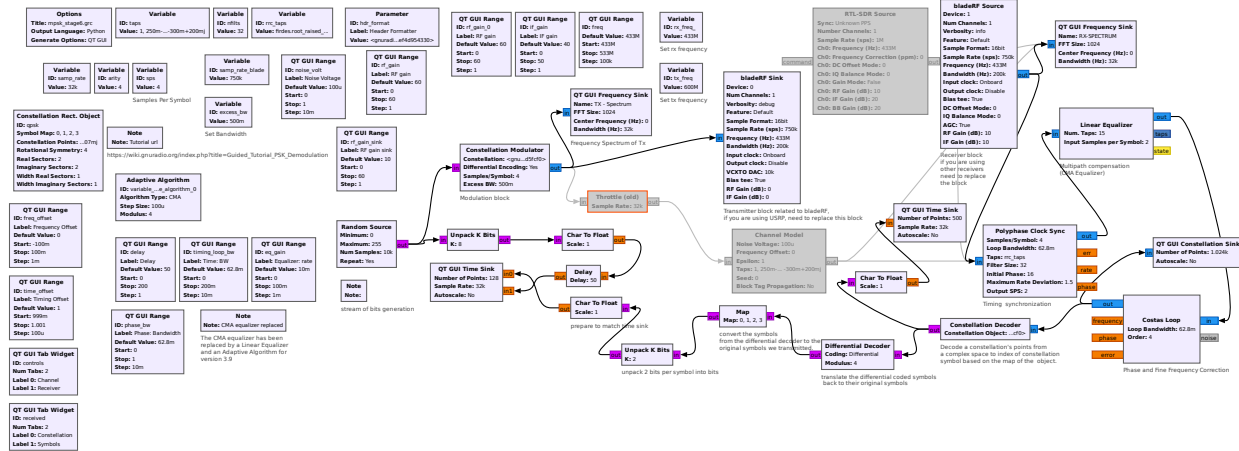


Figure 5.6: Transmitter and receiver block.

In the following, a brief summary of the main component in this lab is summarized.

## Recovering Timing

1. Timing recovery aims to find the best time to sample incoming signals, optimizing the Signal-to-Noise Ratio (SNR) and reducing Inter-Symbol Interference (ISI).
2. The polyphase clock recovery algorithm is used for this purpose.
3. It uses root raised cosine filters to up-sample and demonstrates the impact of Nyquist filtering.
4. Timing recovery applies a matched filter to satisfy the Nyquist ISI criterion, correcting for ideal sampling points.

5. Clock synchronization between transmitter and receiver is crucial due to clock imperfections, resulting in timing differences and drift.
6. Clock recovery, a part of timing recovery, synchronizes transmitter and receiver clocks using information from incoming samples.
7. The polyphase clock recovery technique uses feedback control loops to calculate the differential of the incoming signal, indicating clock offset.
8. Clock recovery uses a 2nd order control loop to acquire the correct filter phase and rate difference between clocks.

### Details of the Polyphase Clock Sync Block

1. Clock recovery typically involves feedback control loops, and Polyphase Clock Sync is a technique used in multirate signal processing for synchronization.
2. The block performs clock recovery, receiver matched filtering to mitigate ISI, and down-sampling to produce samples at 1 sample per symbol (sps).
3. It calculates the first differential of the incoming signal to determine clock offset.
4. Multiple filters with different phases are used to find the correct filter phase that corresponds to the desired timing value.
5. The control loop adjusts filters to minimize the error signal and acquire the optimal filter for sampling.
6. Using many filters achieves precise timing recovery without significant computational complexity.

### Using the Polyphase Clock Sync Block in Our Receiver

1. The Polyphase Clock Sync block with 32 filters and a loop bandwidth of  $2\pi/100$ .
2. It adapts to the incoming signal's rates to synchronize clocks and correct for initial timing offsets.
3. The Costas Loop block further refines the phase and frequency corrections.
4. Frequency offset can cause the constellation to become a circle, indicating the need for further correction.

For more information refer [https://wiki.gnuradio.org/index.php?title=Guided\\_Tutorial\\_PSK\\_Demodulation](https://wiki.gnuradio.org/index.php?title=Guided_Tutorial_PSK_Demodulation).

**Task 7.** *What is the significance of the time synchronization between the transmitter and receiver in a communication system?*

**Task 8.** *Why is it essential to achieve frequency synchronization between the transmitter and receiver in communication systems?*

**Task 9.** *What is the primary purpose and role of a raised cosine filter in communication systems?*

**Task 10.** *Now, run the code and visualize the transmitted and received constellations. Comment on your observation (are we able to recover the transmitted symbols correctly?)*

# **EN2130: Communication Design Project**

## **Software Defined Radio (SDR) - Workshop 5**

### **Task Sheet**

---

**Index No.:**

**Group No.:**

**Date:**

---

**Task 1.**

**Task 2.**

**Task 3.**

**Task 7.**

**Task 8.**

**Task 9.**

**Task 10.**