Lab 9: End To End Communication

Wadhwani Electronics Lab

Department of Electrical Engineering
Indian Institute of Technology, Bombay.

Aim of the experiment

- The goal of this experiment is to have you perform end to end data transmission.
- In Lab 5, 6, 7, 8 digital modulation has been done from primitive blocks. This lab you will be using specialised blocks of GNU radio.
- We will send a text file using three digital modulation techniques -

1. BPSK
2. Differential BPSK
3. DIfferential QPSK

Pre-lab Work

- Make sure that you revise the previous lab experiments.
- Read the supporting material uploaded along with this document.
- Study about the additional blocks whose information is given.

Hints

You can employ inbuilt blocks:

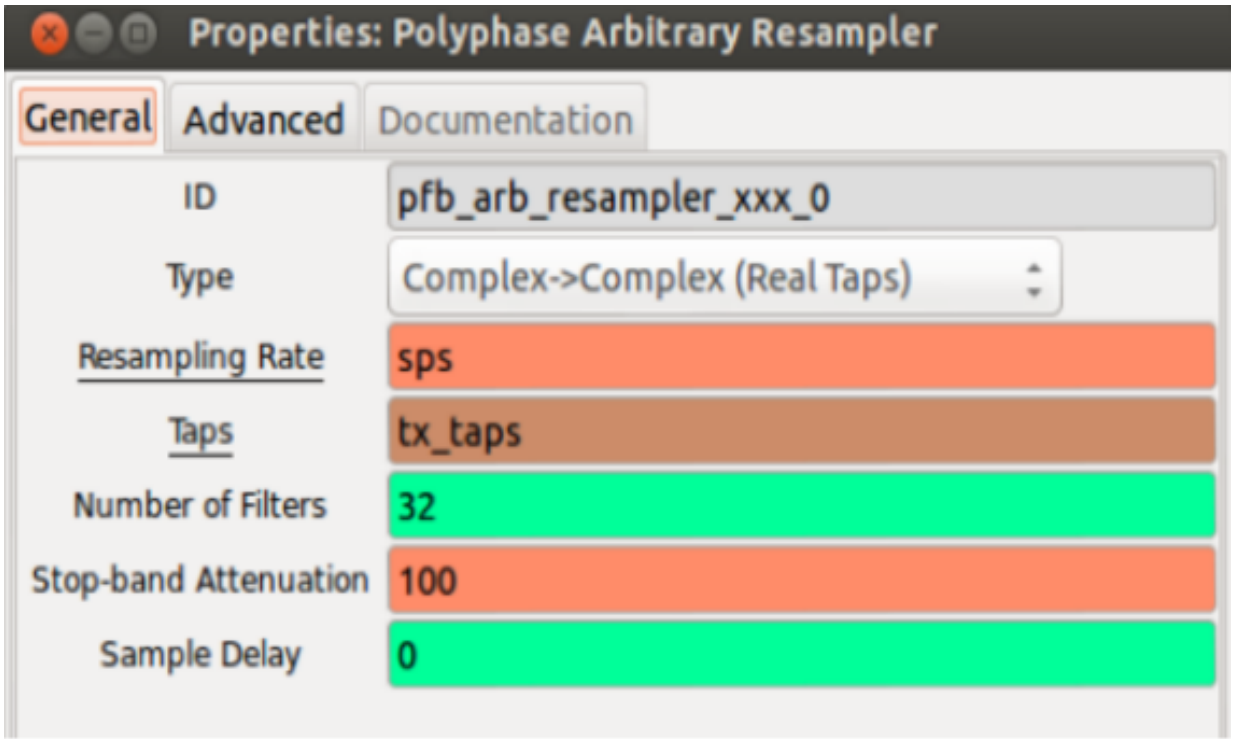
- Polyphase Arbitrary Resampler for pulse shaping and up-converting sps.
- Polyphase clock for Matched filtering and down-converting sps.
- Coastas loop block for Phase and frequency synchronization

Float To UChar

To Change the format of data stream from float to UChar (Byte Format)

Polyphase Arbitrary Resampler

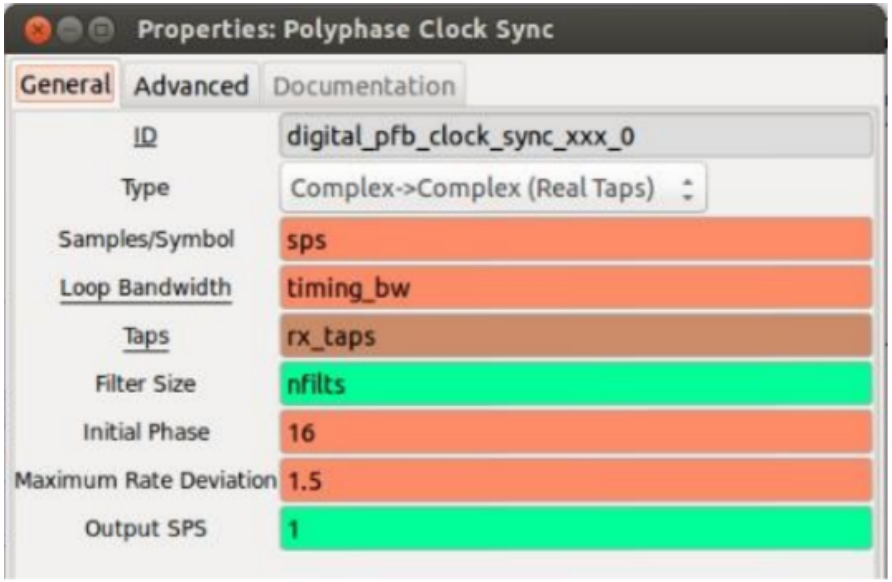
- This block is used in building the Transmitter



  - **tx taps** are the Root Raised Cosine FIR filter tap coefficients generated by the FIR design utility.
    tx_taps = firdes.root_raised_cosine(gain,sps*symb_rate,symb_rate,excess_bw,ntaps)

Polyphase Clock Sync

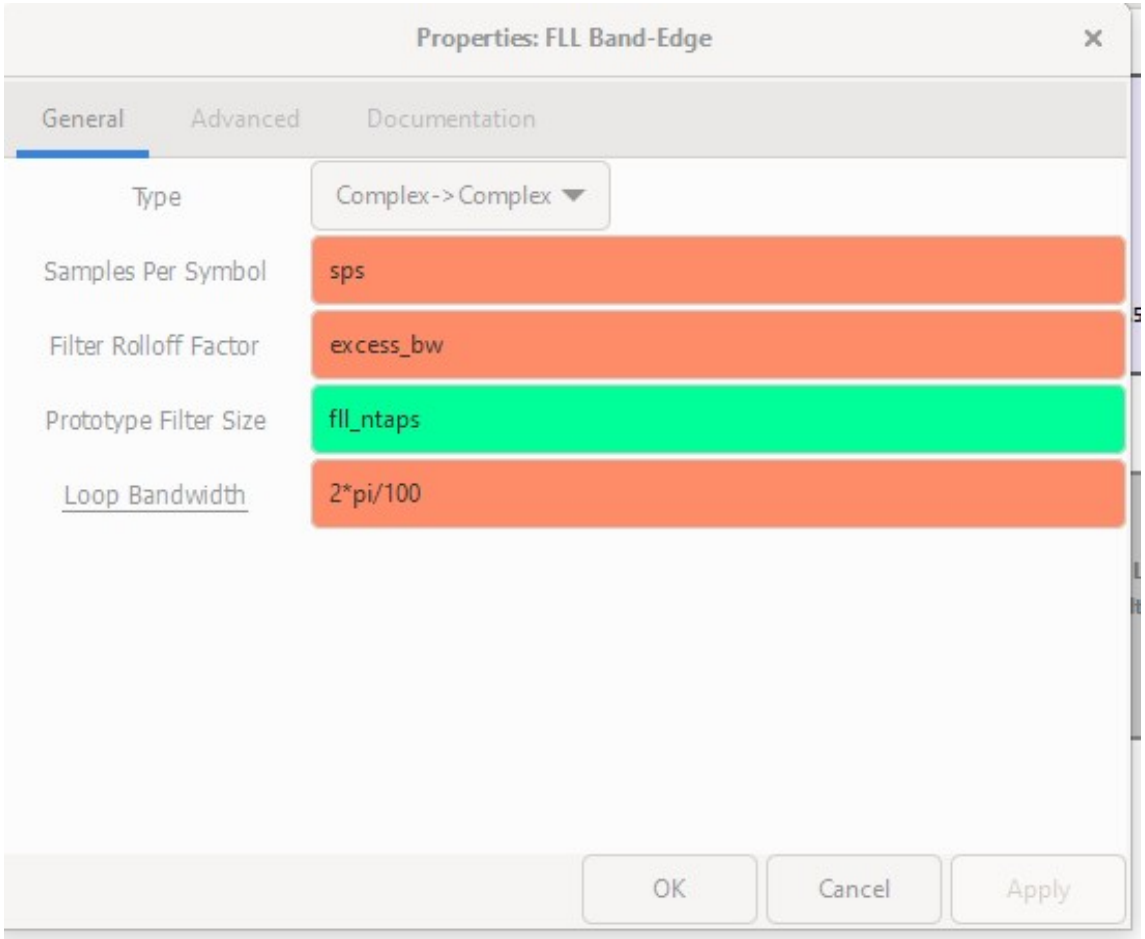
- This block is used in building the receiver



- **timing bw** is 2*pi/100 (normalised loop bandwidth for Polyphase Clock Synchronizer) and **nfilts** is equal to 32.
- **rx_taps** = firdes.root_raised_cosine(gain,nfilts*sps*symb_rate,symb_rate,excess_bw,ntaps)

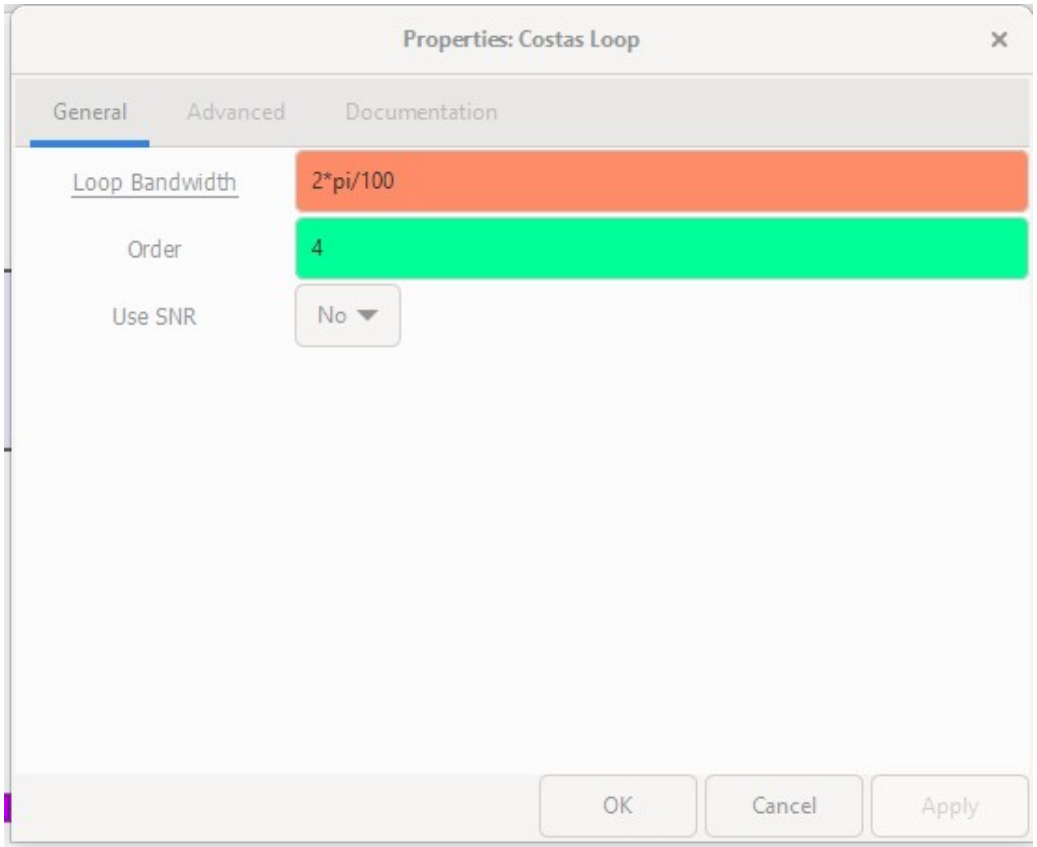
FFL Band-Edge

- The frequency lock loop derives a band-edge filter that covers the upper and lower bandwidths of a digitally-modulated signal



Costas Loop

- This block can be used for Phase and frequency synchronization



- **Loop Bandwidth**:-The loop bandwidth is the equivalent noise bandwidth of a PLL described earlier and should be adjusted accordingly. Internal 2nd order loop bandwidth (2pi/100)
- **Order**:-The loop order depends on the modulation scheme: 2 for BPSK, 4 for QPSK and 8 for 8 -PSK.

Task-1

- Implement a simple Bpsk modulation scheme to transmit and receive a message signal.
  - sps=8 , Gain =3 ,Num_taps=11*sps
  - Symbol rate = 50k , Sample rate=sps$\times$symbol rate
  - alpha = 1
- Use Polyphase Arbitrary Resampler for pulse shaping and upconverting sps.
- Modulate the signal with carrier of 500 Khz and sample rate of 8×10^6.
- Add Gaussian noise = 0.1.
- Introduce frequency offset of 1Hz in the demodulation.
- Implement the Match filtering and sps downconversion with Polyphase clock Sync.
- Use Costas Loop for phase synchronisation.
- You should be able to recover the message signal and observe the stabilised Constellation plot.

Differential modulation & Task-2

- Task-1 assumed you have a perfect phase sync between the transmitter and the reciever. But what if you had a 180° phase shift between the transmitter and the reciever? Your constellation diagram would look perfect but your demodulated signal would be wrong!
- One solution to this problem is to not encode the information in the phase value of the output symbol, but in the phase differences.
- Repeat the whole task-1 with minor differences in introducing the differential blocks for DBPSK transmission.
- You should be able to receive the transmitted message correctly.
- Some information about Differential encoder and Decoder are given in next slide.

Differential Encoder and Decoder

- These blocks are to be used to perform Differential BPSK and Differential QPSK modulation schemes.
- The modulus value changes correspondingly, 2 for BPSK and 4 for QPSK.

Task-3 Differential QPSK

- As done in DBPSK repeat it for DQPSK.
- Make sure to use two bits of information in a symbol. So you might have to pack bits after unpacking the bits. Same goes while in demodulation of the signal.
- You should be able to receive the transmitted message correctly.