Two-Way Digital Paging System Using Software Defined Radios

Ratnayake R.M.S.H.(230548R)
Tennakoon U.G.R.B.(230629R)
Disssanayake R.K.T.(230164K)
Shehan M.N.N.(230613M)

Department of Electronic & Telecommunication Engineering
University of Moratuwa, Sri Lanka

September 17, 2025

Presentation Structure

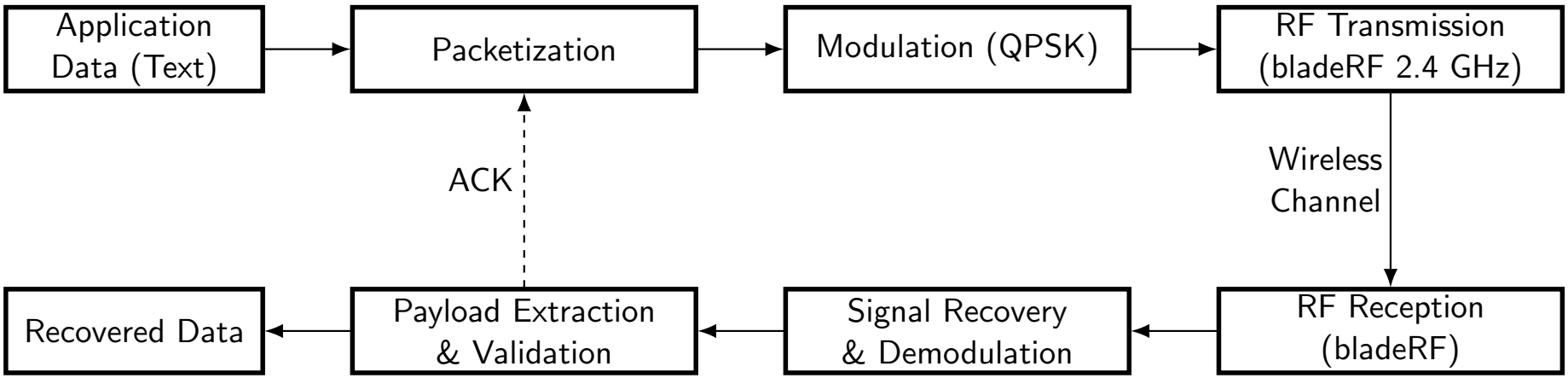
Abstraction

This project demonstrates the design of a **two-way digital paging system** using **GNU Radio** and **BladeRF software-defined radios**. The system ensures **reliable wireless messaging** through QPSK modulation, addressing, framing, and acknowledgment mechanisms. **CRC-based error detection** enhances data integrity, while a real-time GUI visualizes transmission and reception. The prototype showcases the **practical power of SDR platforms** in building robust and extensible digital communication systems.

Requirements

- Short message delivery using digital modulation (BPSK/QPSK).
- Unique user addressing (receiver responds only to its ID).
- Acknowledgment (ACK) mechanism for reliable communication.
- CRC-based error detection and rejection of corrupted messages.
- Basic user interface (console or GUI) for composing messages.

Block Diagram



Transmitter

Message Transmitter Block

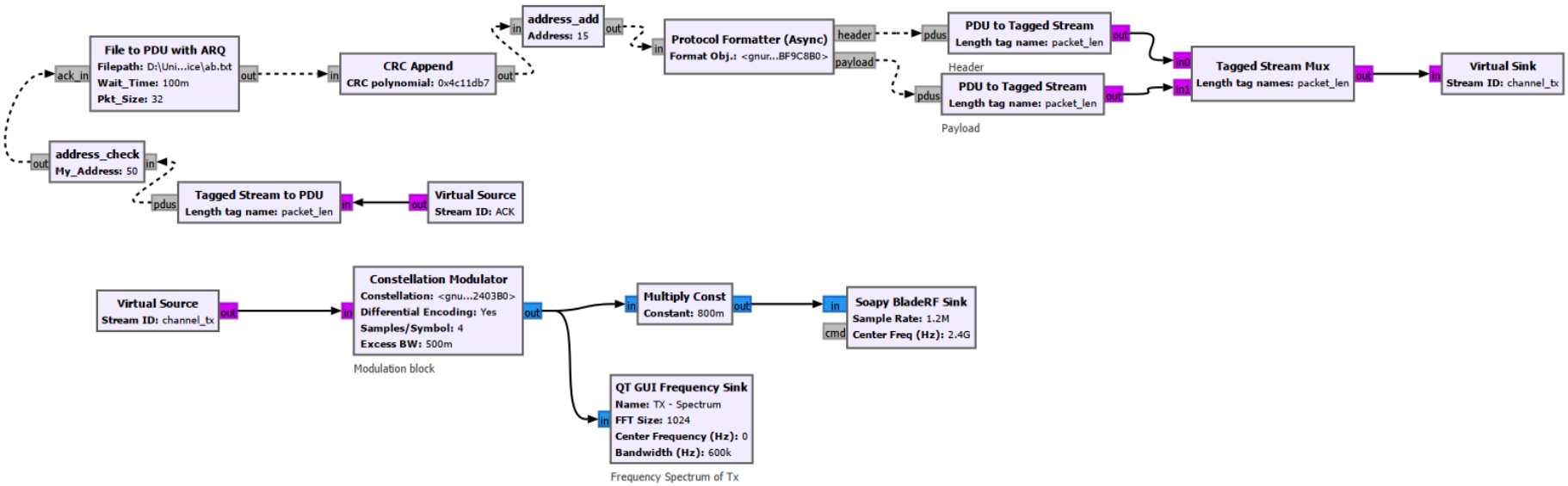


Figure: Message Transmitter Block

Acknowledgement Receiver Block

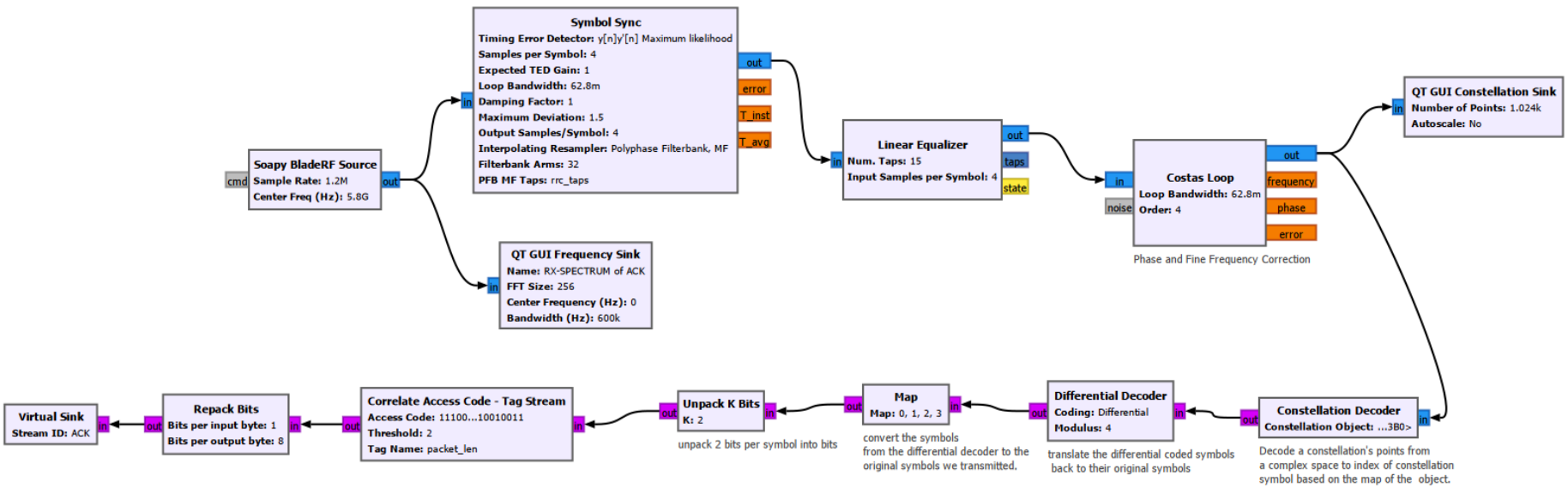


Figure: Acknowledgement Receiver Block

Blocks & Descriptions

Message Transmitter Block

- **Message Source:**[1] Custom Python block that reads the input file/message, segments it into packets, and prepares data for transmission. It also processes acknowledgment (ACK) messages from the receiver.
- **CRC Append:** Adds a Cyclic Redundancy Check (CRC) code to each packet, enabling error detection at the receiver and ensuring corrupted messages are discarded.
- **Protocol Formatter:** Frames each packet with a header (containing sync word, addressing, etc.) and payload, ensuring proper synchronization and identification.
- **Address Add:**[1] Add the destination address to the frame

[1] Custom Python Block

Blocks & Descriptions (Contd...)

- **PDU $\leftrightarrow$ Tagged Stream Conversion:** Converts packet-based data (PDUs) into tagged streams for modulation, and back again at the receiver. This allows GNU Radio blocks to handle both continuous streams and discrete packets.
- **Stream Mux:** Combines headers and payloads into a single stream for transmission.
- **QPSK Modulator:** Maps digital bits into complex QPSK symbols for RF transmission, providing bandwidth efficiency and robustness.

Acknowledgement Receiver Block

- **Symbol Synchronization:** Aligns the received signal samples with symbol timing to reduce inter-symbol interference.
- **Linear Equalizer:** Compensates for channel distortions and multipath effects, improving signal quality.

Blocks & Descriptions (Contd...)

- **Costas Loop:** Corrects carrier frequency and phase offsets in the received signal, enabling proper demodulation.
- **QPSK Decoder:** Converts received QPSK symbols back into digital bits.
- **Differential Decoder:** Removes phase ambiguity introduced during modulation/demodulation.
- **Bit Mapping + Unpack/Repack:**[1] Reconstructs the bitstream into meaningful packets, ready for higher-layer processing.
- **Address Check:**[1] Checking whether the destination address is correct and passing though it.
- **ACK Feedback Path:**[1] Ensures reliable delivery by informing the transmitter whether a message was correctly received.

Receiver

Message Receiver Block

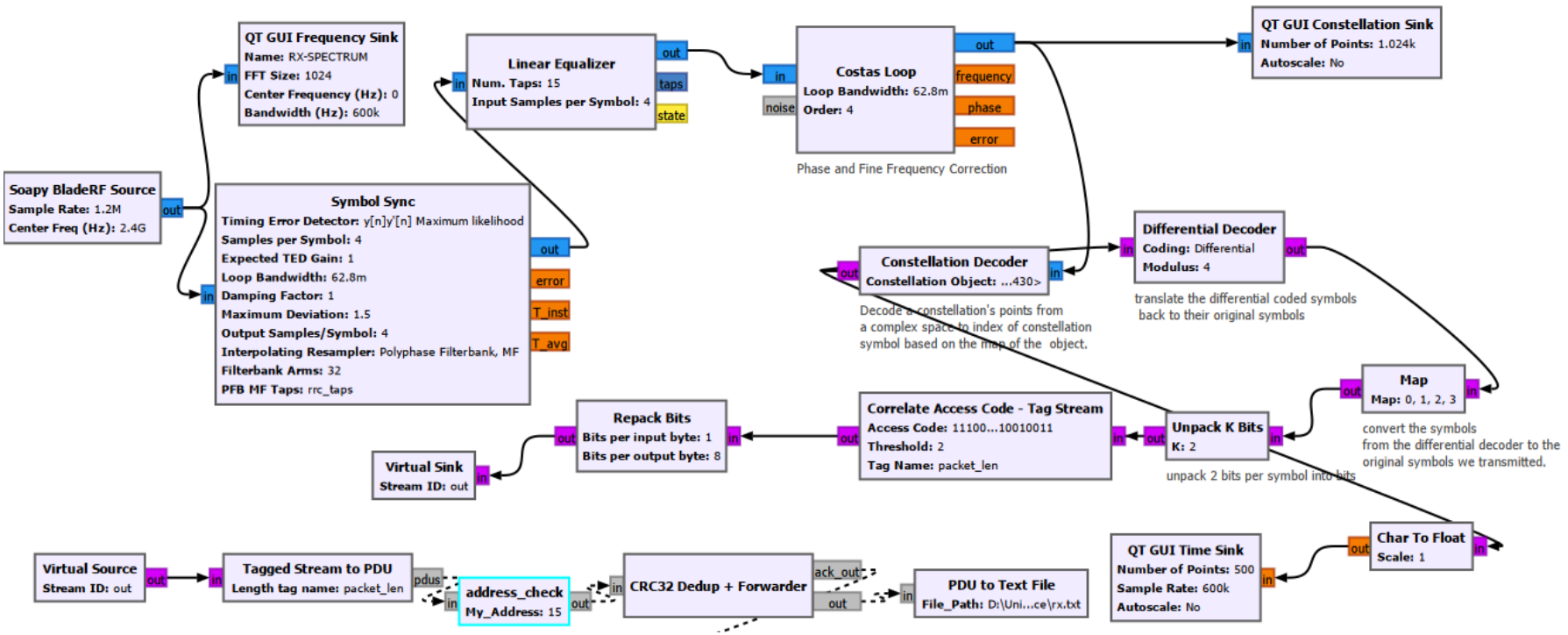


Figure: Message Receiver Block Diagram

Acknowledgement Transmitter Block

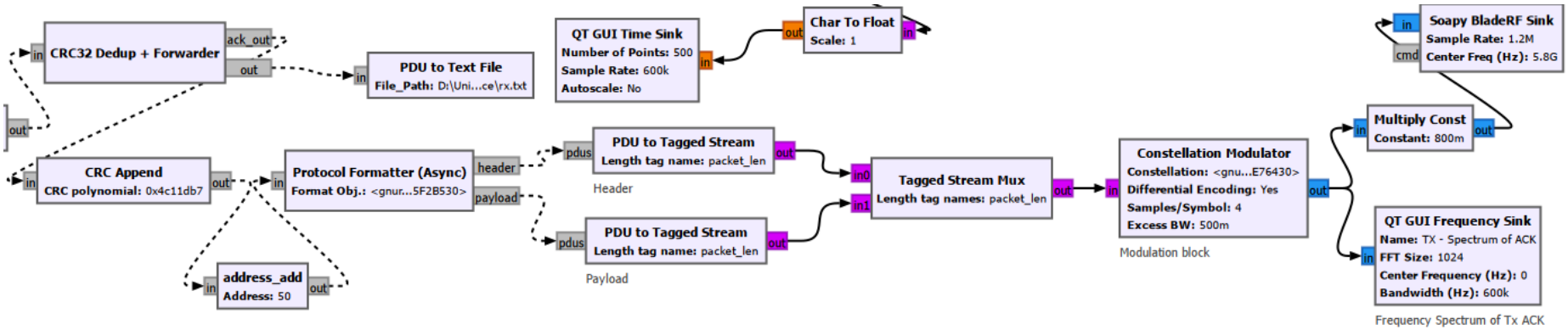


Figure: Acknowledgement Transmitter Block

Other than blocks in transmitter, we only used,

- **CRC 32 Dedup + Forwarder:**[1] Receives packets, checks their CRC32 for integrity, forwards the payload only if it's not a duplicate, and always sends an acknowledgment (ACK) for valid packets. Essentially, it acts as a CRC validator, deduplicator, and forwarder.

Selected Methodologies

Frame structure

Preamble (4 B)	DST Address (1 B)	Sq ID + Payload ($\leq$ 32 B)	CRC 32 (4 B)

Figure: Message Frame

Preamble (4 B)	DST Address (1 B)	0xAA + Sq ID (2 B)	CRC 32 (4 B)

Figure: Acknowledgement Frame

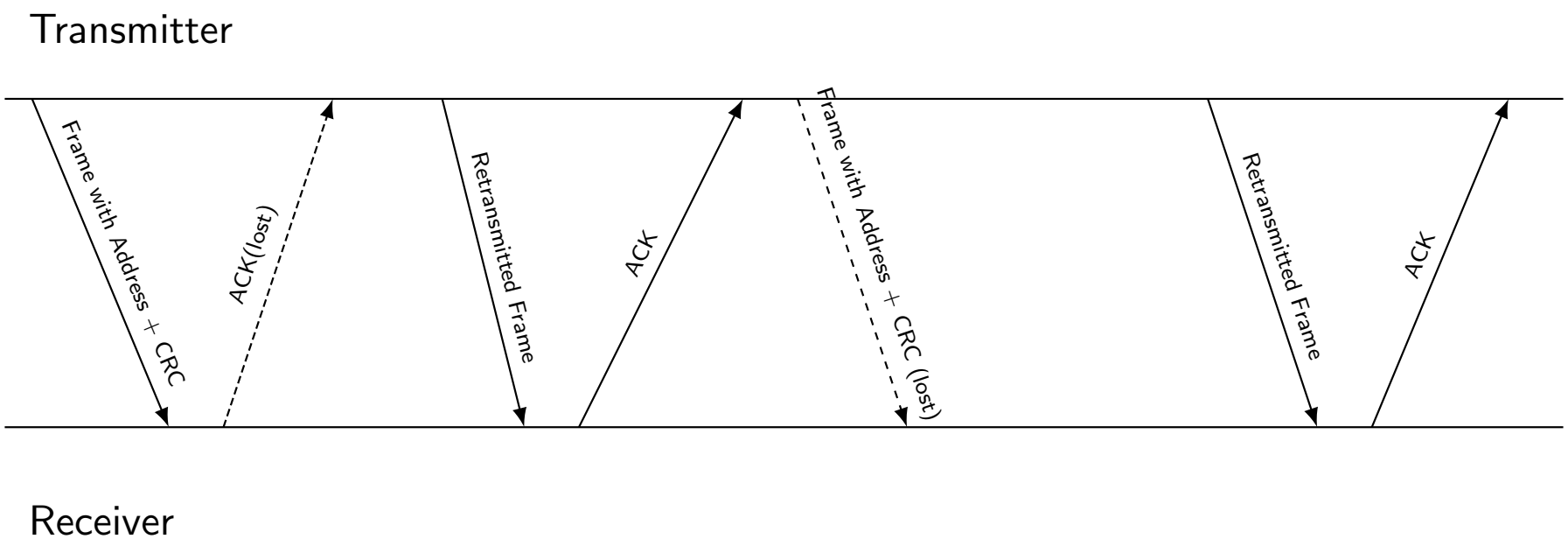
Maximum text file size[2] $=(2^8 - 1) \times N$ Bytes, where N is the number of data bytes in payload.

[2]For this frame design $N = 31$. Therefore, Maximum text file size = 7905 Bytes.

ARQ Mechanism

We are using a **Stop-and-Wait ARQ** system:

- Step 1: Each frame carries an **address**.
- Step 2: The receiver checks the **address**.
- Step 3: If correct, it sends an **ACK** back.
- Step 4: The sender waits for the ACK. If it is missing, the sender **retransmits** the same frame.



How ARQ Mechanism Visible in GNU Radio?

```
[CRC32] OK (Packet 20)
[Forward] Packet 20 duplicate, not forwarded
[FilePDU] Timeout waiting for ACK of 0x15, retrying...
[FilePDU] Packet 0x15 sent (try 2)
[CRC32] OK (Packet 21)
[Forward] UTF-8 decode error: 'charmap' codec can't encode character '\u2192' in
position 20: character maps to <undefined>
[TextFile] Wrote 31 chars to D:\University\Semester 3\EN2130\gnupractice\rx.txt
[FilePDU] Timeout waiting for ACK of 0x15, retrying...
[FilePDU] Packet 0x15 sent (try 3)
[CRC32] OK (Packet 21)
[Forward] Packet 21 duplicate, not forwarded
[FilePDU] ACK received for packet 0x15
[FilePDU] Packet 0x16 sent (try 1)
[CRC32] OK (Packet 21)
[Forward] Packet 21 duplicate, not forwarded
[FilePDU] Timeout waiting for ACK of 0x16, retrying...
[FilePDU] Packet 0x16 sent (try 2)
[CRC32] OK (Packet 22)
[Forward] UTF-8 decode error: 'charmap' codec can't encode character '\u2192' in
position 20: character maps to <undefined>
[TextFile] Wrote 7 chars to D:\University\Semester 3\EN2130\gnupractice\rx.txt
[FilePDU] Timeout waiting for ACK of 0x16, retrying...
[FilePDU] Packet 0x16 sent (try 3)
[CRC32] OK (Packet 22)
[Forward] Packet 22 duplicate, not forwarded
[FilePDU] ACK received for packet 0x16
```

QPSK Modulation

- **Quadrature Phase Shift Keying (QPSK)** is used as the modulation scheme.
- Each symbol carries **2 bits**, mapped to one of four constellation points.
- Constellation points: $(\pm 1, \pm 1)$ with Gray coding to minimize bit errors.
- **Advantages:**
  - Bandwidth efficient (2 bits/symbol).
  - Robust against noise compared to higher-order modulations.
- Symbol synchronization is used to correctly recover symbols at the receiver.

Simulation Results

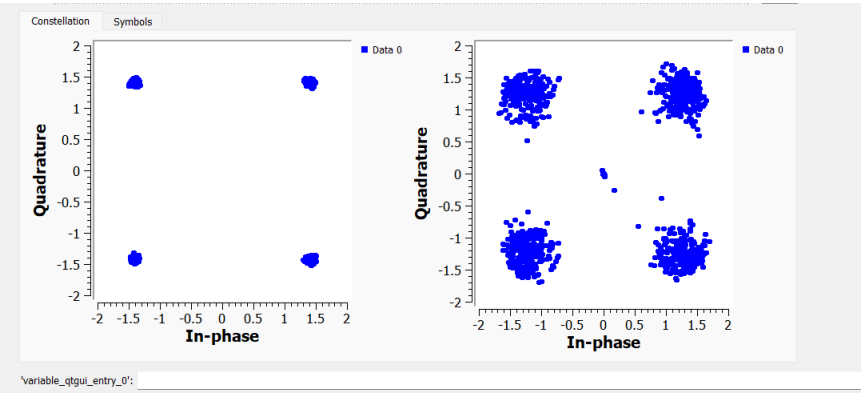


Figure: Constellation Diagrams

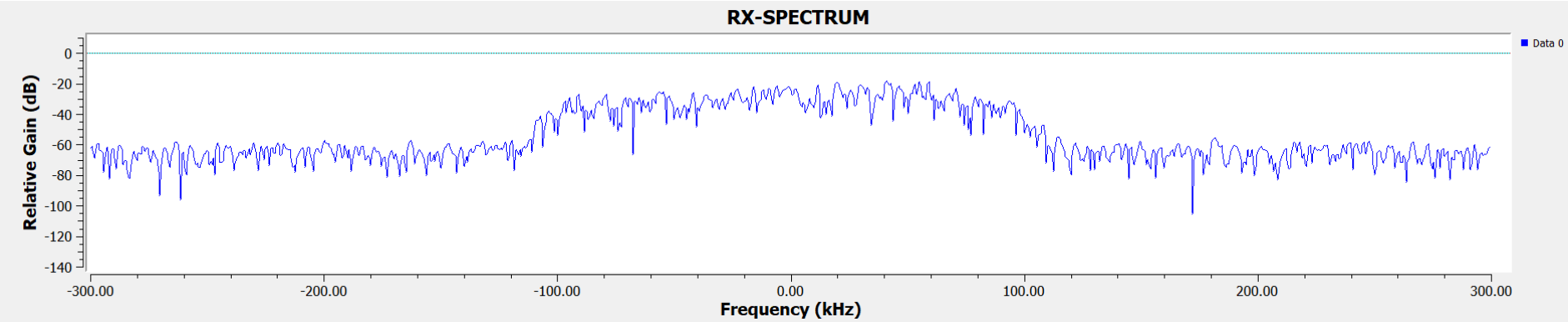


Figure: Frequency Spectrum

References


GNU Radio, "Tutorials," GNU Radio Wiki. [Online]. Available: https://wiki.gnuradio.org/index.php/Tutorials. [Accessed: Sep. 14, 2025].

Thank You!