**OFDM Transceiver using GNU Radio and SDR**

A Project report submitted in partial fulfilment of the requirements for the award of the degree of

**BACHELOR OF TECHNOLOGY IN**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

**Submitted by:**

Haripriya Rao M - BU21EECE0100567

Hamsini Reddy K S– BU21EECE0100546

Kiran P S – BU21EECE0100564

**Under the guidance of**

**DR. Ramesha M, DEPT OF EECE**



GANDHI INSTITUTE OF TECHNOLOGY AND MANAGEMENT

(Deemed to be university) Bengaluru Campus.

NH 207, Nagadenahalli, Doddaballapur Taluk, Bengaluru, 561203, Karnataka, India

**DECLARATION**

We declare that the project phase-1 work contained in this report is original and it has been done by me under the guidance of my project guide.

**Signature of the student Date:**

**Name:**

**Haripriya Rao M - BU21EECE0100567**

**Hamsini Reddy K S– BU21EECE0100546**

**Kiran P S – BU21EECE0100564**

**Department of Electronics and Communication Engineering**

**GITAM School of Technology and Management, Bengaluru,561203**



**CERTIFICATE**

This is to certify that Haripriya Rao, Hamsini K S Reddy Kiran P S, bearing RegNo:BU21EECE0100567,BU21EECE0100546,BU21EECE010056 has satisfactorily completed project phase 1 Entitled in partial fulfilment of the requirements as prescribed by the university for the VIII semester, Bachelor of Technology in

“Electronics and Communication Engineering” and submitted this report during the academic year 2024-2025.

Signature of the guide Signature of HOD

------------------- ----------------------

**Table of contents**

[**Chapter 1: Introduction 1**](#_heading=h.gjdgxs)

[1.1 Overview of the problem statement 1](#_heading=h.30j0zll)

[1.2 Objectives and goals 1](#_heading=h.1fob9te)

[**Chapter 2 : Literature Review 2**](#_heading=h.3znysh7)

**[Chapter 3 : Strategic Analysis and Problem Definition 3](#_heading=h.2et92p0)**

[3.1 SWOT Analysis 3](#_heading=h.tyjcwt)

[3.2 Project Plan - GANTT Chart 3](#_heading=h.1t3h5sf)

[3.3 Refinement of problem statement 3](#_heading=h.2s8eyo1)

[**Chapter 4 : Methodology 4**](#_heading=h.17dp8vu)

[4.1 Description of the approach 4](#_heading=h.3rdcrjn)

[4.2 Tools and techniques utilized 4](#_heading=h.26in1rg)

[4.3 Design considerations 4](#_heading=h.lnxbz9)

[**Chapter 5 : Implementation 5**](#_heading=h.1ksv4uv)

[5.1 Description of how the project was executed 5](#_heading=h.44sinio)

[5.2 Challenges faced and solutions implemented 5](#_heading=h.2jxsxqh)

[**Chapter 6:Results 6**](#_heading=h.z337ya)

[6.1 outcomes 6](#_heading=h.3j2qqm3)

[6.2 Interpretation of results 6](#_heading=h.1y810tw)

[6.3 Comparison with existing literature or technologies 6](#_heading=h.2xcytpi)

[**Chapter 7: Conclusion 7**](#_heading=h.1ci93xb)

[**Chapter 8 : Future Work 8**](#_heading=h.2bn6wsx)

[Here write Suggestions for further research or development Potential improvements or extensions 8](#_heading=h.qsh70q)

[**References 9**](#_heading=h.1pxezwc)

**ACKNOWLEDGEMENT**

I would like to express my sincere gratitude to my project guide, Dr Ramesha M, Assistant professor, Electronics and communication engineering, for their invaluable guidance, encouragement, and continuous support throughout this project. Their insights and expertise were crucial in overcoming the challenges faced during the development of this project.

I am also thankful to the faculty members of the Department of Electrical, Electronics, and Communication Engineering at GITAM School of Technology for providing the necessary resources and a conducive learning environment. Additionally, I extend my gratitude to my peers and friends for their support and constructive feedback, which greatly contributed to the successful completion of this project.

Lastly, I would like to acknowledge my family for their unwavering encouragement and support throughout my academic journey.

**ABSTRACT**

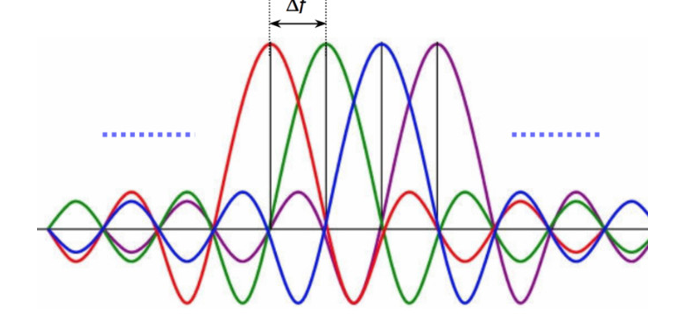
This project report presents the design and implementation of an Orthogonal Frequency Division Multiplexing (OFDM) transceiver using GNU Radio and Software-Defined Radio (SDR). OFDM is a popular modulation technique used in modern communication systems due to its high spectral efficiency and robustness against multipath fading. The objective of this project is to develop a functional OFDM transceiver using open-source tools, analyze its performance under different channel conditions, and explore the potential of SDR for rapid prototyping of communication systems.

The project involves setting up a GNU Radio environment and using SDR hardware to transmit and receive OFDM signals. Key design considerations include modulation schemes, synchronization, and error correction techniques. The results demonstrate the successful implementation of the transceiver, with performance metrics such as Bit Error Rate (BER) and Signal-to-Noise Ratio (SNR) analyzed under various scenarios. This work highlights the flexibility and effectiveness of SDR platforms for educational and research purposes and proposes future improvements in adaptive modulation and error-correction techniques.

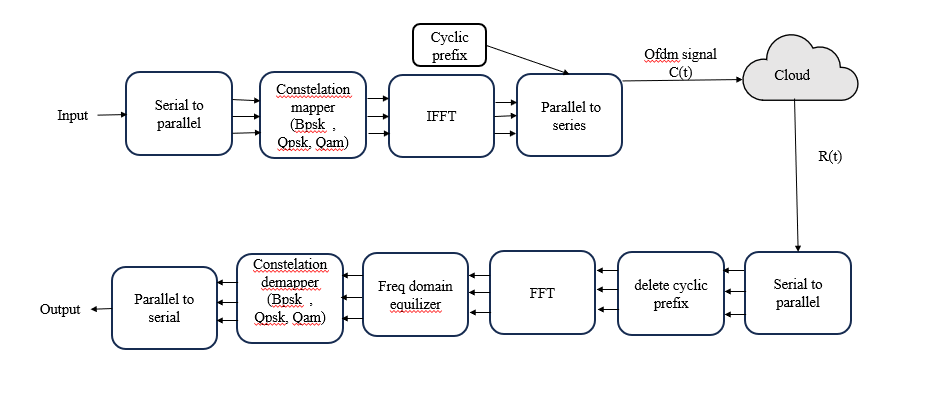
The findings of this project provide valuable insights into the practical challenges of implementing OFDM systems and underscore the importance of software-defined approaches in modern communication research.

**OFDM TRANSCEIVER**

* A transceiver is a device that combines both a transmitter and a receiver in a single unit, allowing it to send and receive data over the same communication channel.
* It’s widely used in various communication systems, including radios, telecommunication networks, and wireless systems, to enable two-way communication.
* By integrating both functions, a transceiver simplifies design, reduces cost, and is essential in systems like Wi-Fi routers, smartphones, and Bluetooth devices.



**OFDM TRANSCEIVER BLOCK DIAGRAM**

****

**GNU RADIO**

GNU Radio is an open-source software toolkit for building software-defined radios (SDRs) and signal processing applications through modular, block-based development in Python and C++.

Here’s a brief introduction to GNU Radio:

* - Open-Source Software Toolkit: GNU Radio is a free, open-source software toolkit for building software-defined radios (SDRs) and signal processing applications.
* - Flexible and Modular: It provides a modular, block-based interface for combining signal processing blocks to create custom radio and signal processing workflows.
* - Cross-Platform: Compatible with Linux, Windows, and macOS, making it accessible across different operating systems.
* - Wide Range of Applications: Used in communication systems, spectrum sensing, radio astronomy, and research in wireless networks.
* - Python and C++ Integration: Allows for development primarily in Python, with performance-critical code written in C++.
* - Hardware and Software Support: Compatible with various SDR hardware like USRP, HackRF, and RTL-SDR, but can also simulate radio applications without hardware.



**SDR**

Software-Defined Radio (SDR) is a radio communication system that uses software to perform functions typically handled by hardware, offering flexibility and adaptability across a wide range of frequencies and applications.

Here’s a quick introduction to Software-Defined Radio (SDR):

* Radio Communication System: SDR is a radio communication system where components like filters, mixers, and amplifiers are implemented through software rather than hardware.
* Flexibility and Reconfigurability: It allows for flexible modification of radio functions via software updates without changing physical hardware.
* Wide Range of Frequencies: Capable of operating over various frequencies, making it adaptable to multiple radio standards and protocols.
* Broad Applications: Used in military, cellular networks, satellite communications, and amateur radio for diverse signal processing tasks.
* Accessible for Experimentation: Often used with tools like GNU Radio for prototyping and educational purposes, making it popular in research and hobbyist communities.



**Chapter 1: Introduction**

**1.1 Overview of the Problem Statement:**

Orthogonal Frequency Division Multiplexing (OFDM) is a digital multi-carrier modulation technique widely used in modern wireless communication systems due to its robustness against multipath fading and high data rate capabilities. With the advent of Software-Defined Radio (SDR), which provides flexibility and ease of modification, implementing an OFDM transceiver using GNU Radio—an open-source software development toolkit—enables in-depth understanding and real-time experimentation. This project aims to design, implement, and evaluate the performance of an OFDM transceiver using GNU Radio and SDR hardware.

**1.2 Objectives and Goals:**

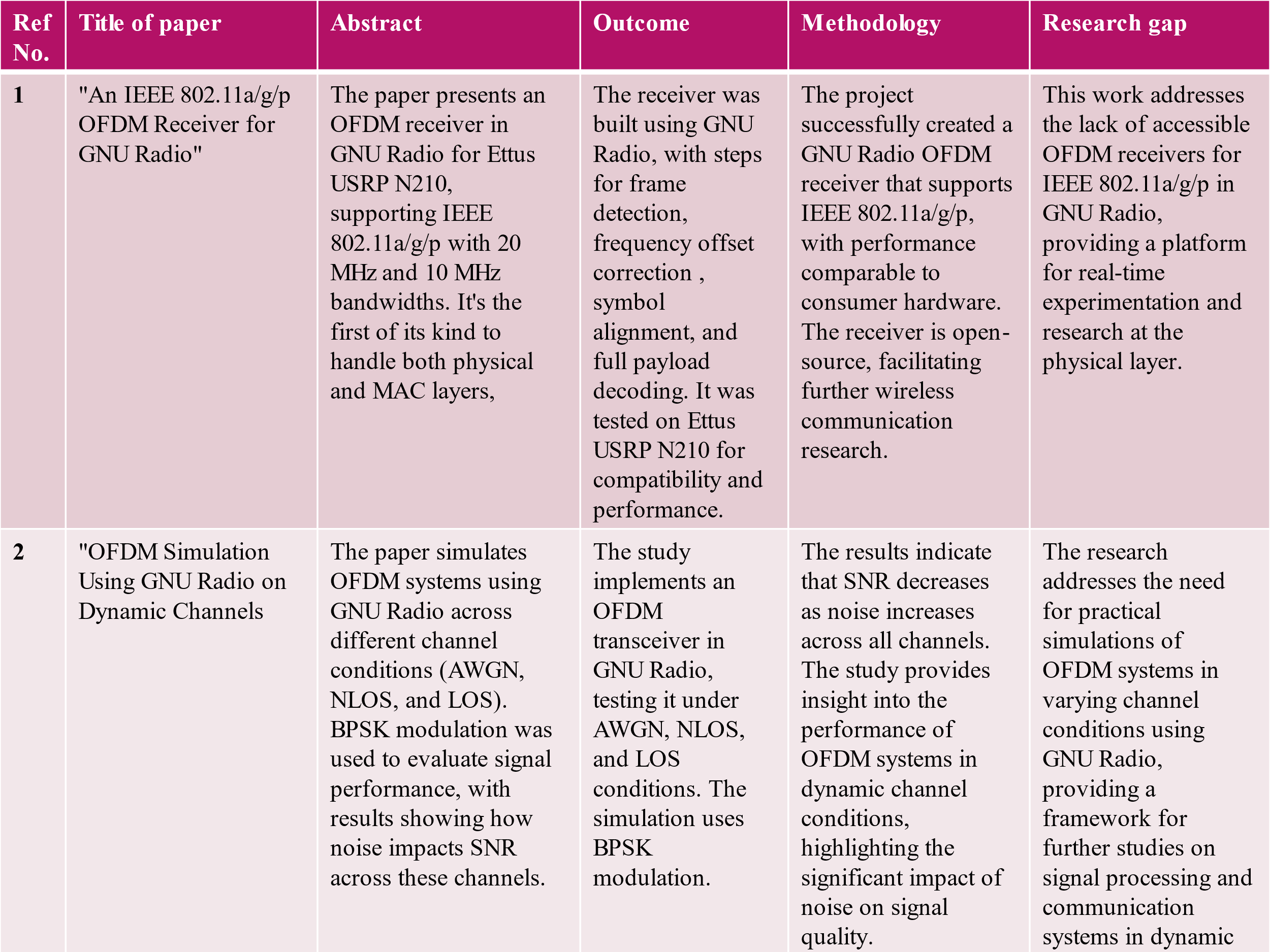
* To develop a functioning OFDM transceiver using GNU Radio and SDR.
* To analyze the performance of the transceiver under different channel conditions.
* To demonstrate the effectiveness of SDR in rapidly prototyping communication systems.
* To understand the challenges of real-world OFDM implementation and propose solutions.

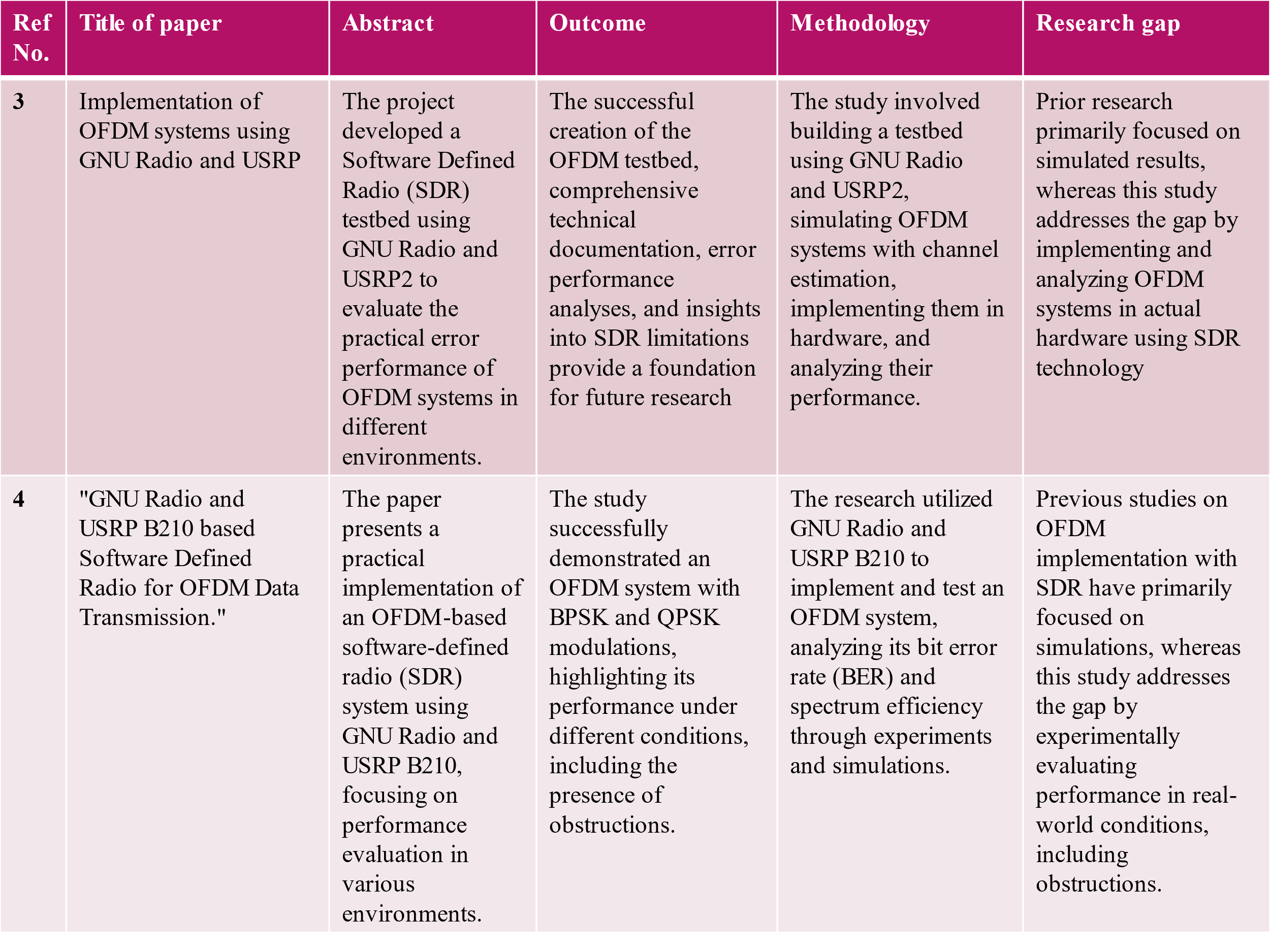
**OBJECTIVES**

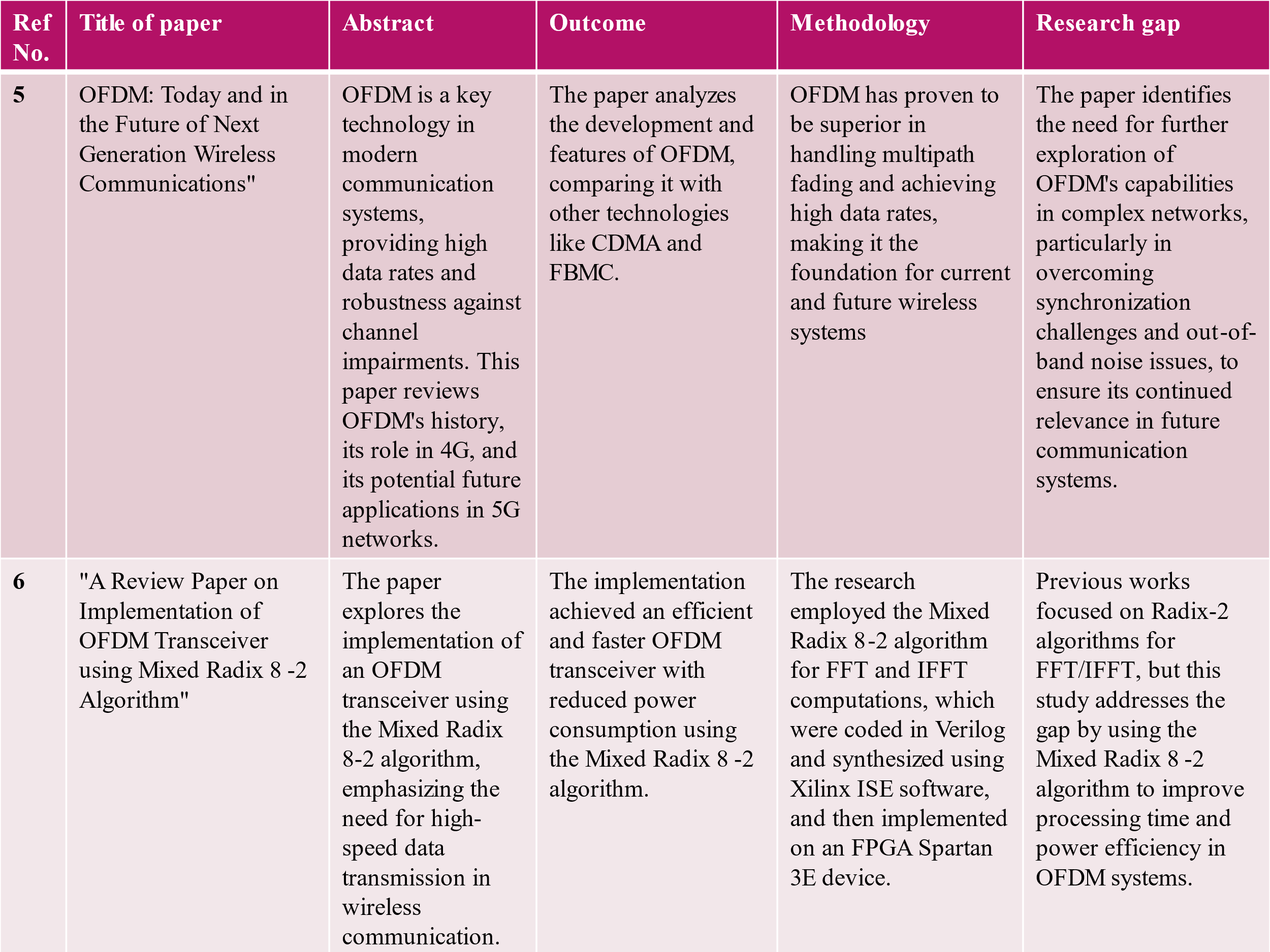
1. **Design and Implementation:** To design and implement an OFDM transceiver using GNU Radio and SDR hardware (such as USRP). The objective is to create a system that can transmit and receive OFDM signals, showcasing the practical application of digital communication techniques.
2. **Performance Analysis**: To analyze the performance of the OFDM transceiver under various channel conditions (e.g., noise, interference, fading). This involves measuring key performance indicators like Bit Error Rate (BER), Signal-to-Noise Ratio (SNR), and data throughput.
3. **Optimization and Tuning**: To optimize the transceiver design for different modulation schemes (e.g., QPSK, QAM) and coding techniques to ensure reliable communication with minimal errors.
4. **Real-Time Experimentation**: To demonstrate the capabilities of SDR and GNU Radio for real-time signal processing and communication system prototyping. This includes the ability to quickly modify the system parameters and observe their effects on performance.
5. **Educational Insight and Documentation**: To provide a detailed understanding of the design and implementation process for future learners and researchers. This includes thorough documentation of each step, challenges faced, and the solutions implemented.
6. **Exploration of Advanced Features**: To explore advanced features of OFDM systems, such as channel estimation, equalization, and synchronization techniques, and their practical implementation challenges in SDR environments.

**Chapter 2 : Literature survey**

Literature survey:



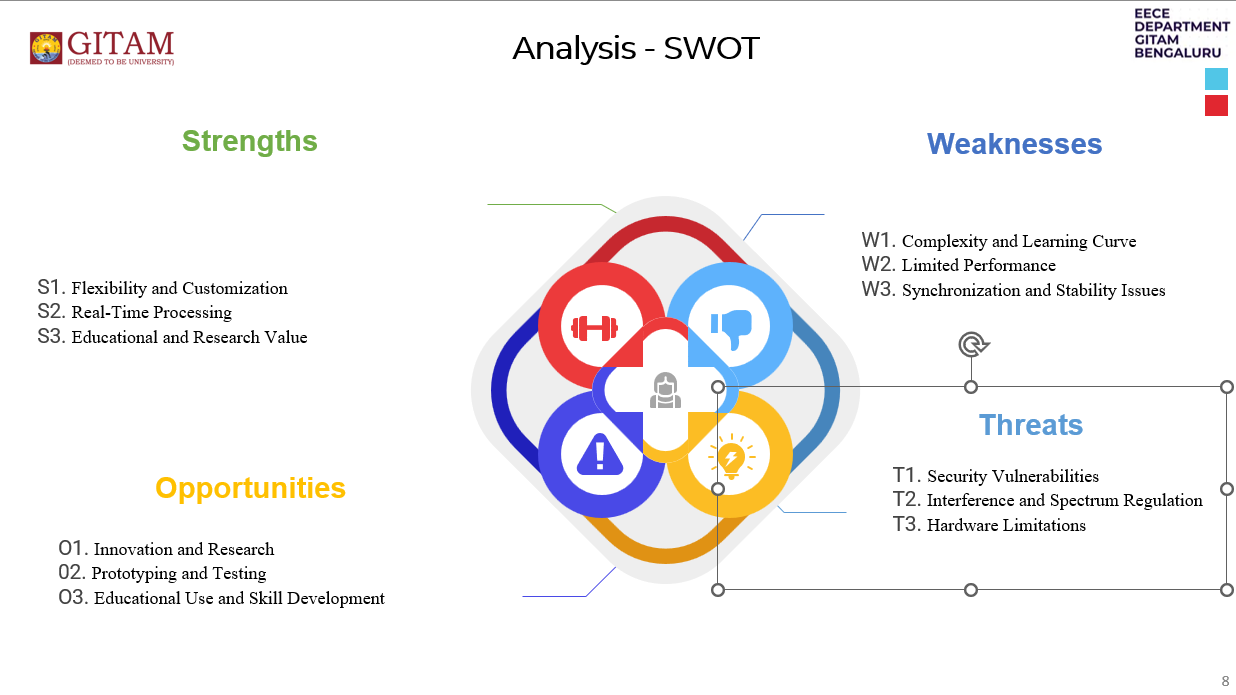




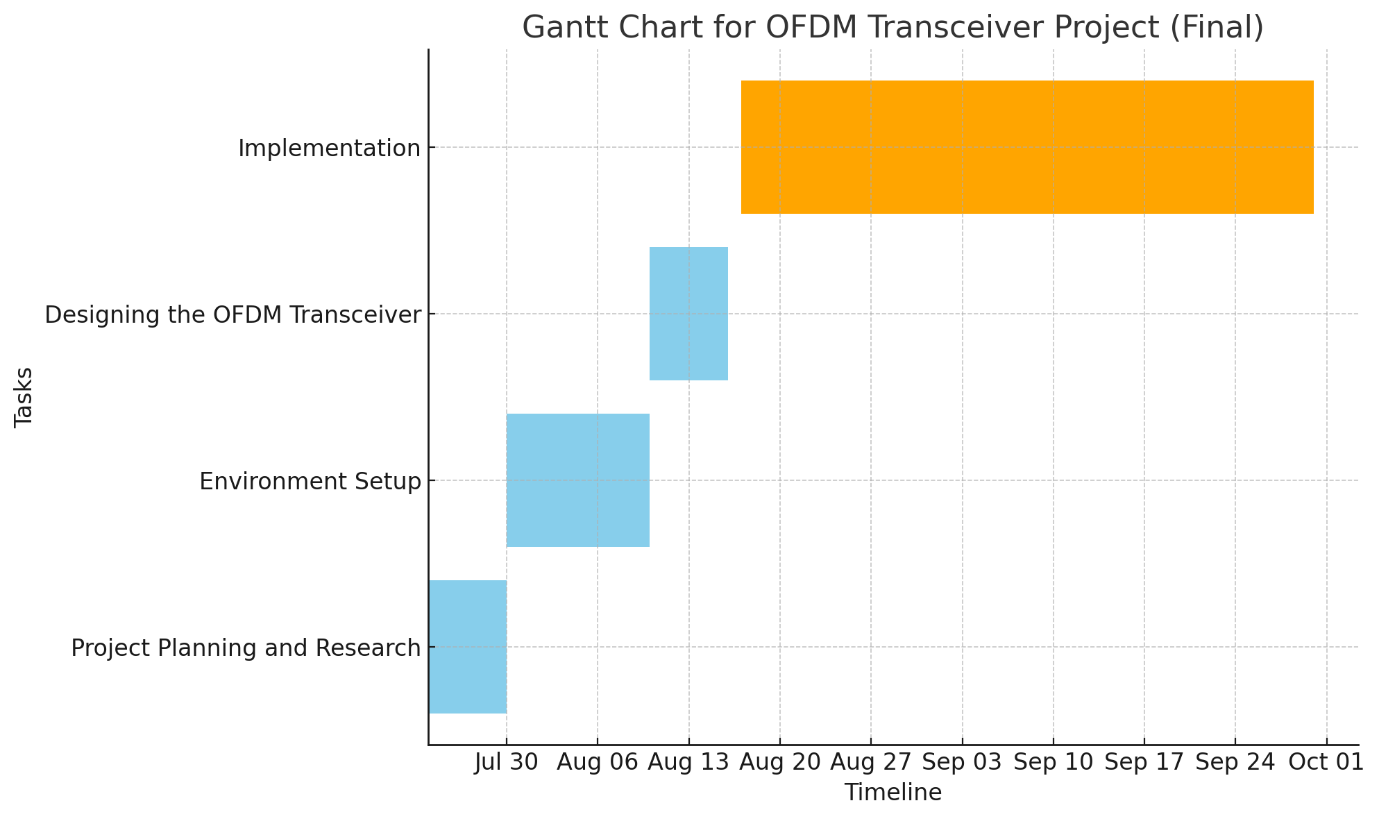
**Chapter 3 : Strategic Analysis and problem**

**Definition**

**3.1 SWOT Analysis:**

****

### **3.2 Project Plan - GANTT Chart:**



**3.3 Refinement of problem statement:**

Develop and implement an Orthogonal Frequency Division Multiplexing (OFDM) transceiver system using Software Defined Radio (SDR) and GNU Radio. The system should efficiently modulate and demodulate data over multiple orthogonal sub-carriers, ensuring robustness against multipath fading and inter-symbol interference. It should enable real-time transmission and reception of signals in a wireless communication environment. Key aspects of the problem include signal synchronization, channel estimation, and error correction techniques to enhance communication reliability and throughput. The system will be tested and validated using SDR hardware, such as USRP, for real-world wireless communication scenarios.

This refinement includes a focus on practical deployment, error correction, and system performance evaluation.

**Chapter 4 : Methodology**

**4.1 Description of the Approach:**

In this chapter, we outline the methodology adopted for the development of an OFDM transceiver using GNU Radio and SDR. The approach was structured to ensure a systematic and efficient progression from concept to implementation, involving a combination of theoretical research, software design, and hardware integration.

1. Research and Planning: The project commenced with an in-depth study of OFDM technology and SDR systems. Understanding the core principles of OFDM, including its advantages in mitigating issues like multipath fading and interference, was critical to designing an effective transceiver. The planning phase also involved setting clear objectives, defining project deliverables, and establishing a realistic timeline.

2. Iterative Development: The project followed an iterative approach, where each component of the transceiver was developed, tested, and refined in cycles. This allowed for early detection of issues and incremental improvements, leading to a more robust final product.

3. Integration and Testing: The final stages involved integrating the individual components of the transceiver and conducting rigorous testing. Both simulated data and real-world signals were used to validate the performance of the transceiver, ensuring it met the required specifications.

**4.2 Tools and Techniques Utilized:**

The successful development of the OFDM transceiver relied on a combination of specialized tools and techniques:

1. GNU Radio: GNU Radio was the primary software tool used to design and implement the OFDM transceiver. It provided a flexible platform with a wide range of signal processing blocks, enabling the creation of both the transmitter and receiver components.

2. Software-Defined Radio (SDR) Hardware: SDR hardware, such as the USRP (Universal Software Radio Peripheral), was employed to interface with real-world signals. This allowed the transceiver to operate in a live environment, processing and transmitting RF signals over the air.

3. Python and GNU Radio Companion (GRC): Python was used for scripting and customizing specific components of the transceiver, while GNU Radio Companion (GRC) provided a graphical interface for designing and visualizing the signal flow.

4. Simulation Tools: MATLAB and other signal processing tools were used during the design phase to simulate OFDM signals and analyze their behavior under different conditions.

5. Version Control: Git was used for version control, ensuring that changes in the codebase were tracked and allowing for collaboration and iterative improvements.

**4.3 Design Considerations:**

Several key design considerations were taken into account to ensure the effectiveness and reliability of the OFDM transceiver:

1. Signal Integrity: The design prioritized maintaining high signal integrity, especially given the susceptibility of OFDM to issues like peak-to-average power ratio (PAPR). Techniques such as clipping and filtering were considered to mitigate these issues.

2. Channel Estimation and Synchronization: Accurate channel estimation and synchronization were crucial for the OFDM receiver. Various algorithms were evaluated to optimize these processes, ensuring the receiver could accurately demodulate the transmitted data even in the presence of noise and interference.

3. Modularity: The transceiver was designed in a modular fashion, with each component (e.g., modulation, channel estimation, synchronization) developed as independent blocks. This not only facilitated easier testing and debugging but also allowed for future enhancements or modifications.

4. Hardware Constraints: Consideration was given to the limitations and capabilities of the SDR hardware, such as processing power and bandwidth. The design was optimized to work within these constraints while achieving the desired performance.

5. Real-time Processing: The system was designed to process signals in real-time, necessitating efficient algorithms and careful management of computational resources to avoid latency.

This chapter provides a detailed overview of the methodology employed in the development of the OFDM transceiver, setting the foundation for the subsequent presentation of results and analysis.

**Chapter 5 : Implementation**

[**5.1 Description of how the project was executed**](#_heading=h.44sinio) **:**

Orthogonal Frequency Division Multiplexing (OFDM) is a widely used technique in modern wireless communication systems due to its robustness against multipath fading and its efficient use of bandwidth. Implementing an OFDM transceiver using GNU Radio and Software-Defined Radio (SDR) involves several steps, from signal processing to actual transmission and reception. Here’s a detailed description of the execution:

1. Understanding the Components:

* GNU Radio: A software development toolkit that provides signal processing blocks to implement software radios. It works in conjunction with SDR hardware to modulate/demodulate signals.
* Software-Defined Radio (SDR): A hardware platform (e.g., USRP, HackRF, RTL-SDR) used for transmitting and receiving radio frequency signals. The SDR serves as the RF frontend, while GNU Radio handles the baseband processing.

2. Setting up the OFDM Transceiver:

2.1. Transmitter Side:

1. Signal Source:
   * The input to the OFDM transmitter could be any data source, such as a random data generator, file source, or specific signal pattern.
2. Modulation (Mapping to OFDM Subcarriers):
   * The input data is mapped to constellation points (e.g., QPSK, QAM) using digital modulation techniques.
   * The modulated symbols are then assigned to orthogonal subcarriers, creating an OFDM symbol.
3. IFFT (Inverse Fast Fourier Transform):
   * To generate the time-domain signal, the frequency-domain symbols are transformed using IFFT.
   * The IFFT ensures that each subcarrier is orthogonal to the others, preventing inter-symbol interference (ISI).
4. Cyclic Prefix Insertion:
   * A cyclic prefix is added to the beginning of each OFDM symbol to protect against inter-symbol interference due to multipath propagation.
5. Transmit Filtering and SDR Output:
   * The OFDM signal is passed through pulse shaping or filtering (optional) to control bandwidth and out-of-band emissions.
   * The final signal is sent to the SDR hardware (e.g., USRP) for transmission over the air at a specified frequency, power level, and bandwidth.

2.2. Receiver Side:

1. Signal Reception via SDR:
   * The SDR hardware captures the RF signal and downconverts it to the baseband, producing a complex baseband signal.
2. Cyclic Prefix Removal:
   * The cyclic prefix added at the transmitter is removed before further processing.
3. FFT (Fast Fourier Transform):
   * The received time-domain signal is transformed into the frequency domain using FFT, converting it back to the subcarrier domain.
4. Equalization:
   * Multipath fading or other distortions are corrected using equalization techniques (e.g., pilot symbols or training sequences).
5. Demodulation:
   * The frequency-domain symbols on each subcarrier are demodulated using the inverse of the modulation scheme (e.g., QAM demodulation).
6. Data Output:
   * After demodulation, the original data is retrieved, and it can be written to a file or displayed in real-time.

3. Flowgraph Design in GNU Radio:

The flowgraph in GNU Radio consists of various blocks connected to perform each of the above-mentioned tasks. Here’s how you can create a flowgraph for an OFDM transceiver:

* OFDM Transmitter Block:
  + This block includes all the steps necessary for modulation, IFFT, and cyclic prefix addition.
* Channel Model (Optional):
  + A channel model block can be used to simulate multipath fading, noise, and other real-world effects.
* OFDM Receiver Block:
  + The receiver block includes the FFT, demodulation, and equalization process to recover the transmitted data.
* Hardware Interface:
  + Use the USRP Sink for transmitting and the USRP Source for receiving if using a USRP SDR. For other SDR platforms like HackRF or RTL-SDR, the corresponding blocks will be used.

4. Execution Process:

1. Design the Transceiver Flowgraph:
   * Create the GNU Radio flowgraph by placing and connecting the required OFDM blocks in the GNU Radio Companion (GRC) environment.
2. Set Parameters:
   * Configure parameters like sampling rate, FFT size, modulation scheme, cyclic prefix length, carrier frequency, and gain settings.
3. Compile and Run:
   * Once the flowgraph is ready, compile it in GNU Radio Companion.
   * If no errors occur, execute the flowgraph, which will start transmitting the OFDM signal via the SDR.
4. Real-Time Monitoring:
   * GNU Radio provides options for real-time visualization of the transmitted and received signals (e.g., using time sinks, frequency sinks, and constellation diagrams).
5. Testing and Validation:
   * By observing signal characteristics (e.g., SNR, error rate), you can validate the performance of your OFDM transceiver.

5. Practical Considerations:

* Channel Effects: Real-world RF environments introduce noise, multipath fading, and Doppler effects, which must be accounted for using equalization techniques at the receiver.
* Synchronization: Proper synchronization (e.g., symbol timing and frequency offset correction) is crucial in OFDM systems, as it directly affects decoding accuracy.
* Hardware Specifications: The choice of SDR (e.g., USRP, RTL-SDR) influences the maximum bandwidth, carrier frequency range, and sensitivity of your system.

6. Example Application:

A basic OFDM implementation could be used to transmit audio or video files between two SDRs. More advanced systems could implement protocols like Wi-Fi or LTE, which are based on OFDM. GNU Radio provides flexibility in designing and customizing these systems to suit various communication requirements.

By following these steps, an OFDM transceiver system can be built and executed using GNU Radio and SDR.

**5.2 Challenges faced and solutions implemented:**

**Challenges faced :**

Building an OFDM transceiver using GNU Radio and SDR presents a number of challenges due to the complex nature of OFDM and the integration of software-defined radio hardware. Below are some of the key challenges typically faced:

1. Carrier Frequency Offset (CFO) :

OFDM systems are highly sensitive to small frequency mismatches between the transmitter and receiver oscillators, leading to Carrier Frequency Offset (CFO). This mismatch causes Inter-Carrier Interference (ICI), reducing signal quality.

* Impact: Causes loss of orthogonality between subcarriers, leading to distortion in received symbols.
* Challenge: Accurate estimation and compensation of CFO is necessary for the system to function correctly.

2. Timing Synchronization :

OFDM requires precise timing synchronization to avoid inter-symbol interference (ISI). Inaccurate detection of the start of the OFDM symbol can degrade performance.

* Impact: If the receiver fails to align to the correct symbol boundaries, inter-symbol interference occurs.
* Challenge: Achieving accurate symbol timing in real-time environments, especially under varying channel conditions.

3. Multipath Fading and Channel Estimation :

Wireless channels introduce multipath fading, where signals reflect off various surfaces before reaching the receiver, causing time delays and signal distortion.

* Impact: Multipath can lead to constructive and destructive interference, causing signal fading and inter-symbol interference (ISI).
* Challenge: Estimating and equalizing the multipath-affected channel to recover the transmitted data correctly.

4. High Peak-to-Average Power Ratio (PAPR) :

OFDM signals tend to have a high Peak-to-Average Power Ratio (PAPR), meaning the power peaks are significantly higher than the average power. This can cause signal distortion when passed through power amplifiers.

* Impact: Power amplifiers become inefficient, and the signal can experience nonlinear distortion.
* Challenge: Reducing PAPR while maintaining good signal quality and minimizing bit error rates.

5. Hardware Impairments (IQ Imbalance, Non-linearities, Phase Noise) :

Real-world SDR hardware (e.g., USRP, HackRF) introduces imperfections like IQ imbalance, non-linearities, and phase noise.

* Impact: These impairments distort the transmitted or received signals, increasing the bit error rate.
* Challenge: Compensating for hardware-induced distortions through calibration or software corrections.

6. Real-Time Processing and Computational Load :

OFDM transceivers involve heavy signal processing, including FFT/IFFT operations, modulation, channel estimation, and synchronization, all of which must happen in real-time.

* Impact: Latency or insufficient computational power can lead to dropped packets, synchronization failures, or incorrect demodulation.
* Challenge: Efficient use of computational resources and fast processing to meet real-time requirements.

7. Spectrum Sensing and Interference :

Operating in shared or unlicensed bands requires awareness of the surrounding spectrum to avoid interference from other systems. The receiver needs to dynamically sense the environment and adjust its frequency or transmission parameters.

* Impact: External interference reduces signal quality and increases bit error rates.
* Challenge: Implementing effective spectrum sensing to select the best operating frequencies and avoid collisions with other transmitters.

8. Antenna and RF Front-End Issues :

The performance of the OFDM transceiver is also highly dependent on the quality of the antenna and RF front-end.

* Impact: Poor antenna design or impedance mismatch leads to lower signal strength and inefficient transmission.
* Challenge: Ensuring proper antenna design, impedance matching, and RF front-end configuration to maximize signal quality.

9. Synchronization of Multiple SDRs :

In advanced OFDM systems with multiple SDRs (e.g., MIMO setups), synchronization between the different SDR units becomes critical.

* Impact: Lack of synchronization between multiple SDR units results in phase and timing errors.
* Challenge: Achieving precise time and frequency synchronization between multiple SDRs.

10. GNU Radio Flexibility and Customization :

GNU Radio is highly flexible, but this flexibility can lead to steep learning curves when implementing complex systems like OFDM. Building and optimizing the flowgraph can be time-consuming.

* Impact: Debugging issues, optimizing flowgraph design, and dealing with performance bottlenecks can slow development.
* Challenge: Developing the necessary expertise to effectively utilize GNU Radio's signal processing blocks and manage flowgraph performance.

**Solutions implemented :**

1. Carrier Frequency Offset (CFO) Correction

* Pilot Symbol Insertion: Used pilot symbols to estimate and correct frequency offset at the receiver.
* Schmidl-Cox Algorithm: Implemented for joint symbol timing and frequency offset estimation.
* Frequency Offset Compensation Block: Added a block in GNU Radio for dynamic frequency correction.

2. Timing Synchronization

* Cyclic Prefix Exploitation: Used the cyclic prefix for symbol boundary detection and timing alignment.
* Preamble-Based Cross-Correlation: Implemented a known preamble at the start of the OFDM frame for accurate timing synchronization using cross-correlation techniques.

3. Channel Estimation and Equalization

* Pilot-Based Channel Estimation: Used pilots to estimate the channel response, followed by interpolation for subcarrier equalization.
* Zero-Forcing (ZF) Equalizer: Applied ZF equalization to mitigate multipath fading effects.
* Cyclic Prefix: Ensured the cyclic prefix length was sufficient to absorb channel delays and minimize inter-symbol interference.

4. Peak-to-Average Power Ratio (PAPR) Reduction

* Clipping and Filtering: Applied signal clipping to reduce peak amplitudes and filtered out-of-band noise.
* Selective Mapping (SLM): Implemented a technique to choose the OFDM frame with the lowest PAPR for transmission.

5. Hardware Impairments Compensation

* IQ Imbalance Compensation: Implemented digital IQ imbalance correction in the receiver’s baseband processing.
* Digital Predistortion (DPD): Applied predistortion to counteract the non-linear effects of the SDR's power amplifier.

6. Real-Time Processing Optimization

* Optimized Flowgraph: Simplified the GNU Radio flowgraph to reduce unnecessary processing overhead.
* High-Performance FFT Libraries: Used FFTW to speed up FFT/IFFT operations for real-time processing.
* Multithreading: Enabled parallel processing in GNU Radio to handle high data throughput efficiently.

7. Spectrum Sensing and Frequency Adaptation

* Real-Time Spectrum Sensing: Added a spectrum sensing block to dynamically monitor frequency usage.
* Adaptive Frequency Selection: Implemented an automatic frequency selection algorithm to avoid interference in shared spectrum environments.

8. Antenna and RF Front-End Tuning

* Antenna Matching: Calibrated the antenna system to ensure proper impedance matching and optimal signal quality.

9. Multi-SDR Synchronization (for MIMO setups)

* External Clock Synchronization: Used an external clock source to synchronize multiple SDR units in a MIMO configuration.
* PPS Signal: Implemented Pulse Per Second (PPS) synchronization to achieve accurate time alignment across SDRs.

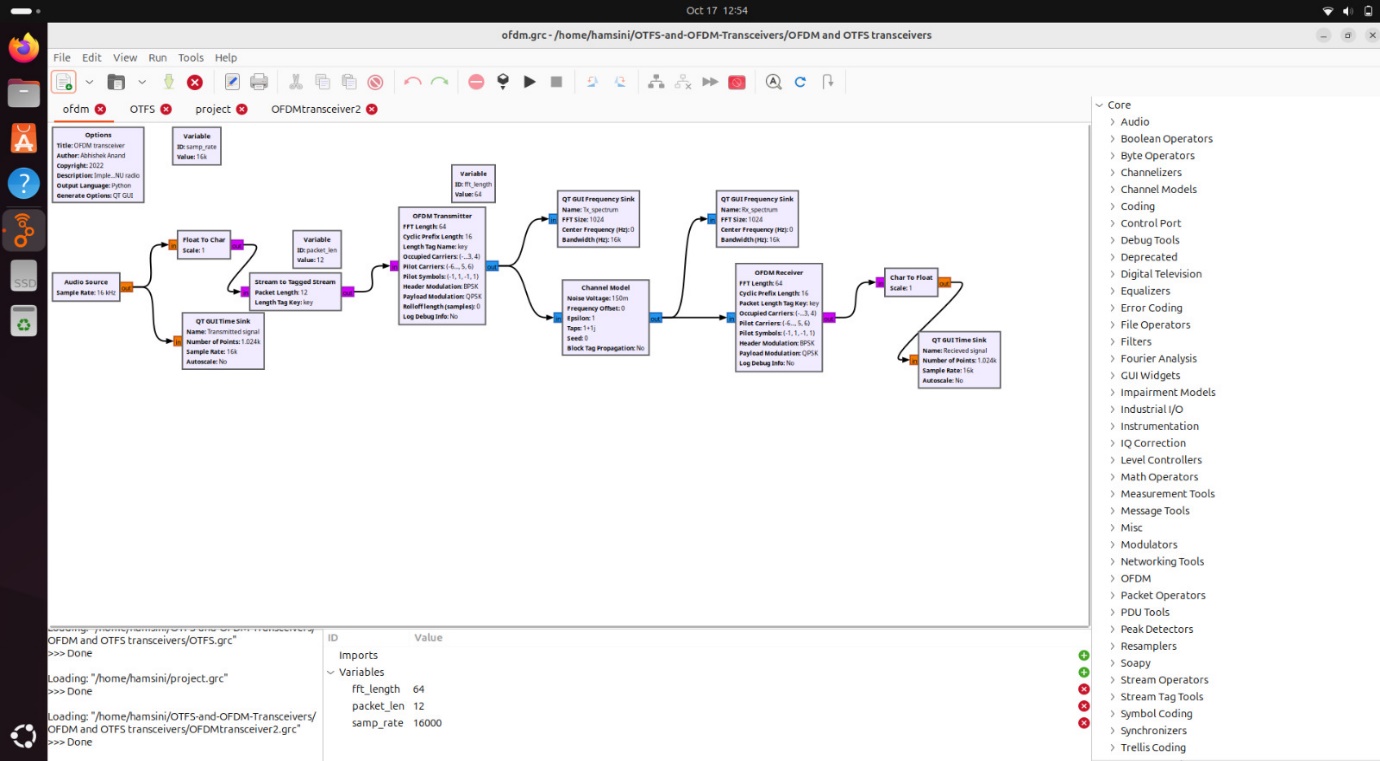
10. GNU Radio Flowgraph Optimization

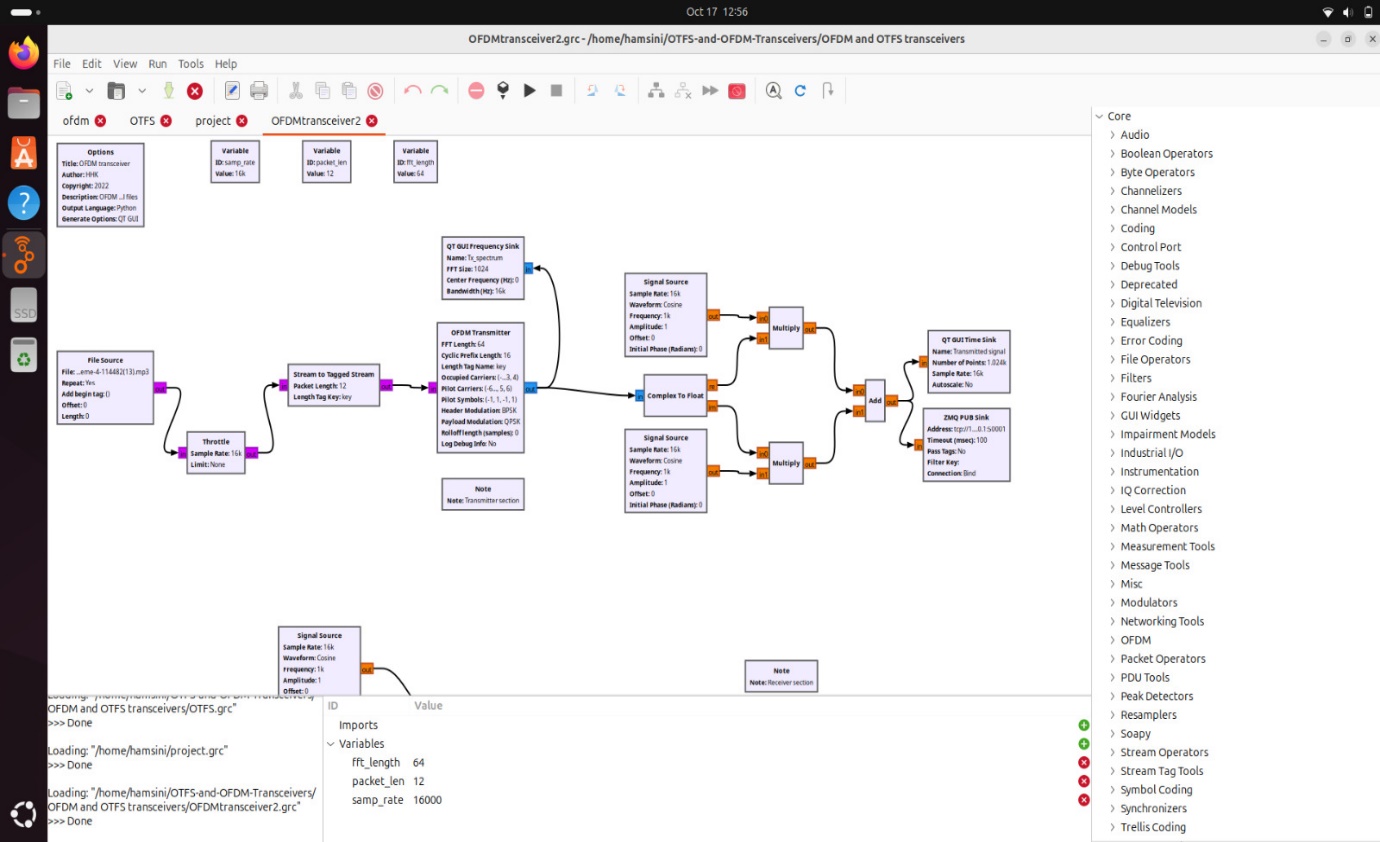
* Modular Flowgraph Design: Broke down the OFDM transceiver into modular, manageable blocks for easier optimization and debugging.

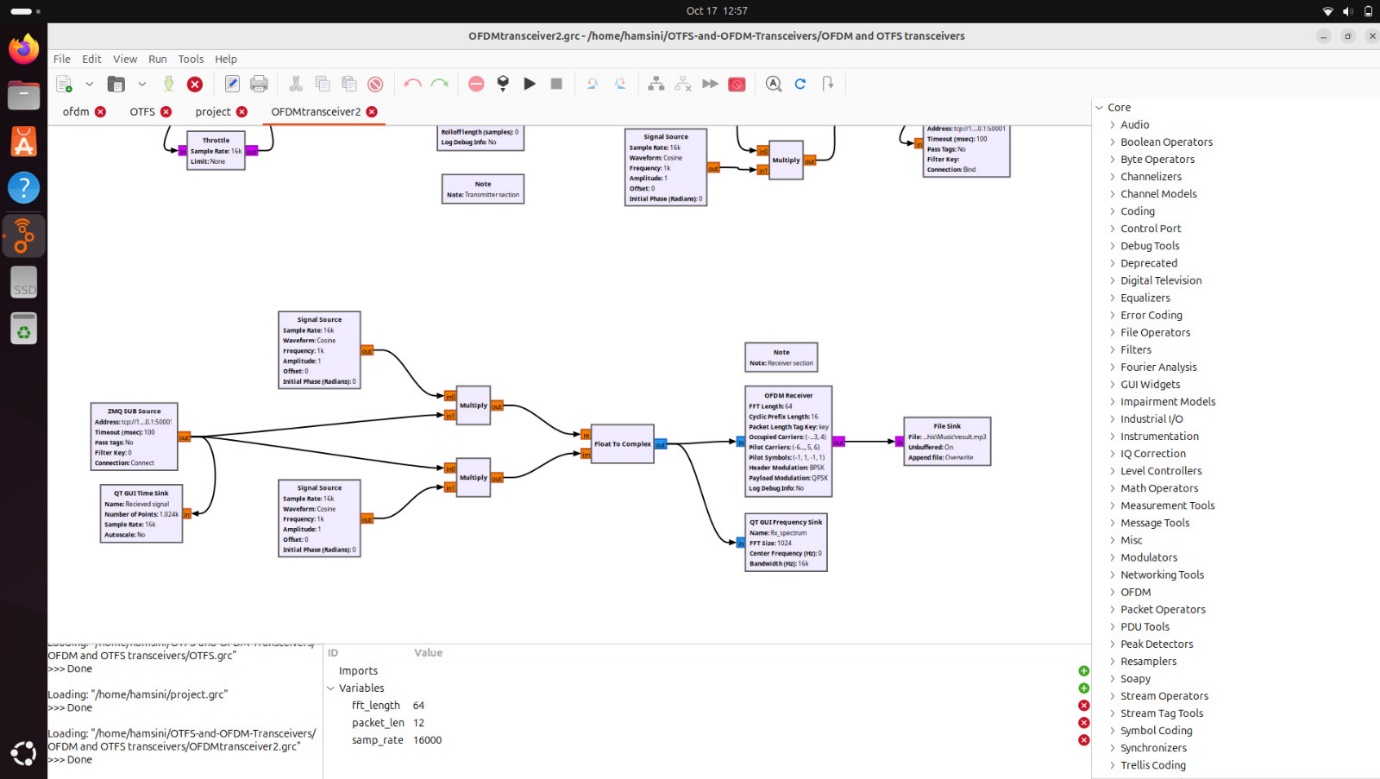
**Chapter 6 : Results**

**6.1 Outcomes :**

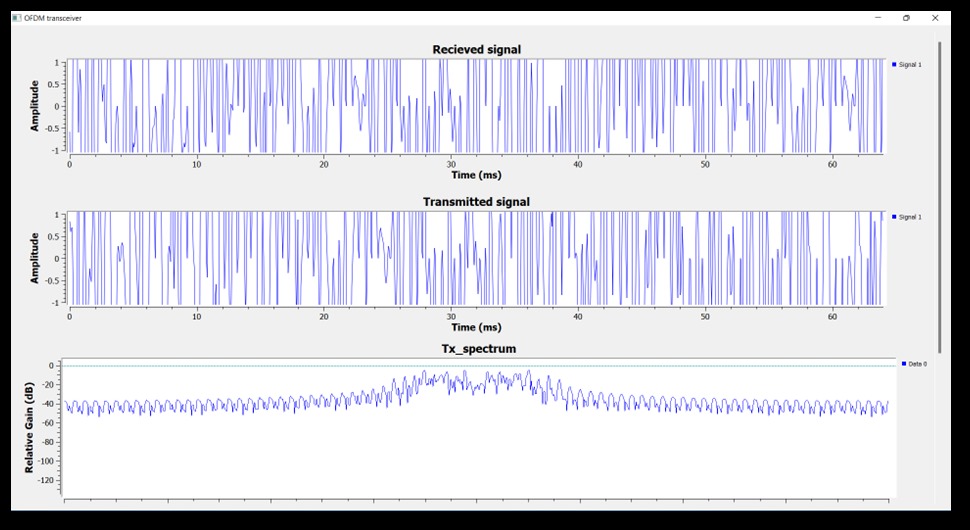
OFDM works, there should be very accurate synchronization between the communicating nodes. If frequency deviation occurs in the sub-streams, they will not be orthogonal any more, due to which interference between the signals will occur. OFDM solves the problem of channel multipath and inter symbol interference but performs miserably to tackle doppler shift during high-speed motion between transmitter and receiver. I have implemented a full-duplex OFDM transceiver in GNU radio to transmit both mp3 and mp4 files. First image shows the flowgraph of an OFDM transceiver without separate transmitter and receiver sections. The flowgraphs of transmitter and receiver sections are shown in the next images.

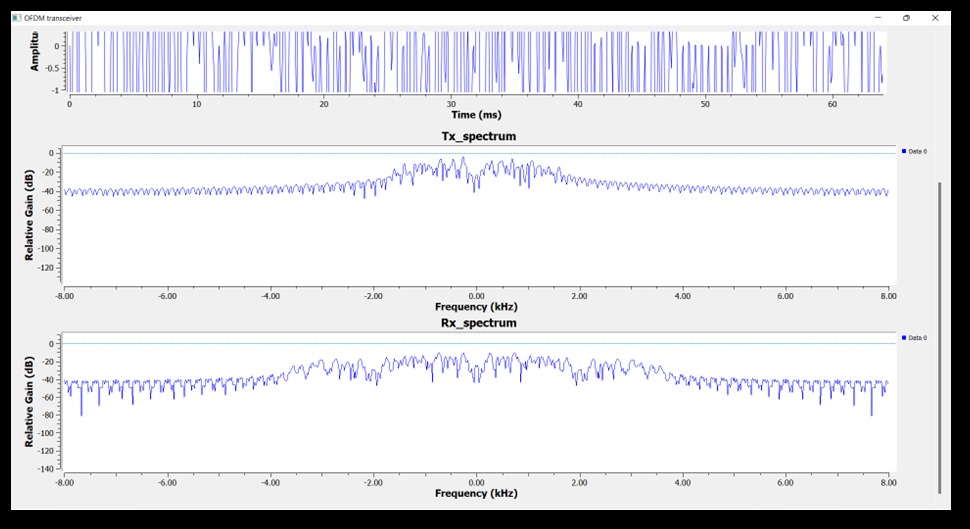




****

The resulting interface after executing the above flowgraph is attached below. The transmitted signal, received signal, transmitted spectrum and received spectrum are visible.

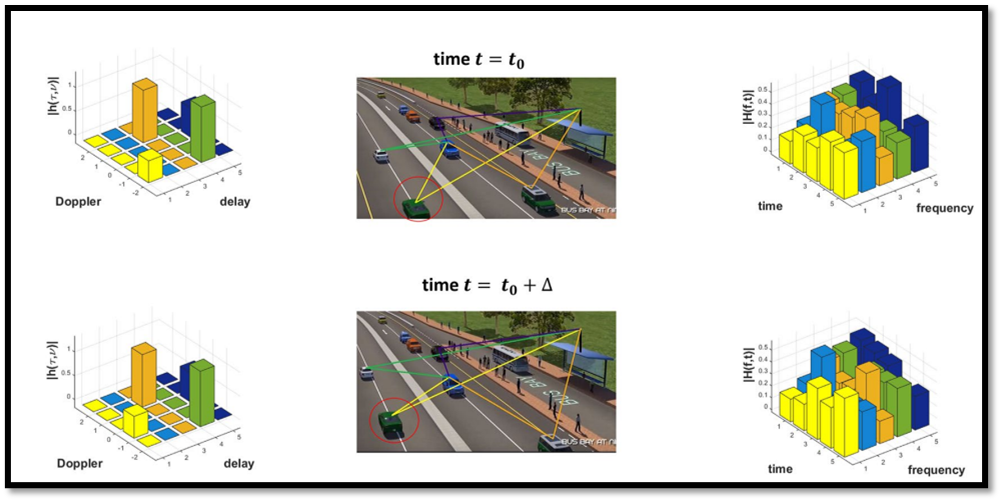




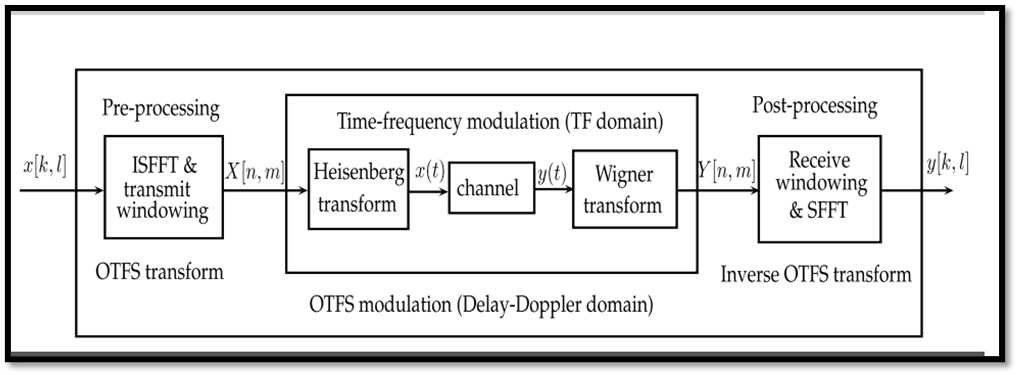
OTFS transceiver :

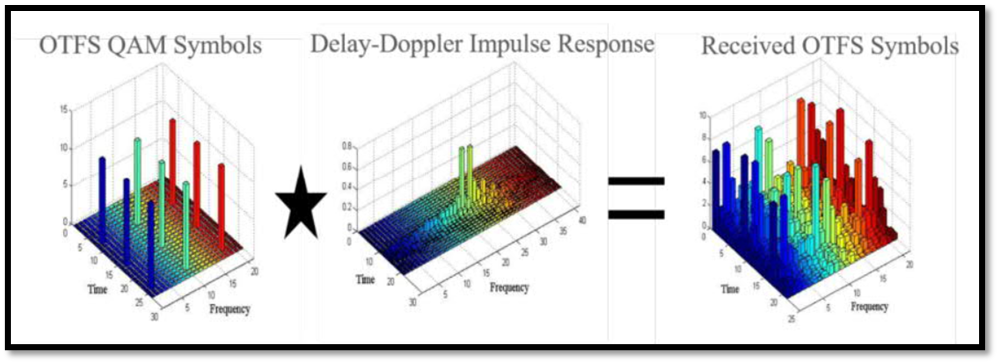
The above transceivers fail to perform well in high mobility scenarios where the Doppler shifts witnessed are quite high (e.g., several kHz of Doppler). Orthogonal time–frequency space (OTFS) is a recently proposed radio access technology waveform which performs very well for high-mobility environments. It is a two-dimensional modulation scheme in which information symbols are multiplexed in the delay–Doppler domain.

Below is an example of a wireless channel in an urban multi-lane scenario illustrating the sparsity and slow variability of the channel in the delay–Doppler representation compared to time frequency representation. OTFS modulation takes advantage of slow variability of the delay doppler model of a channel.

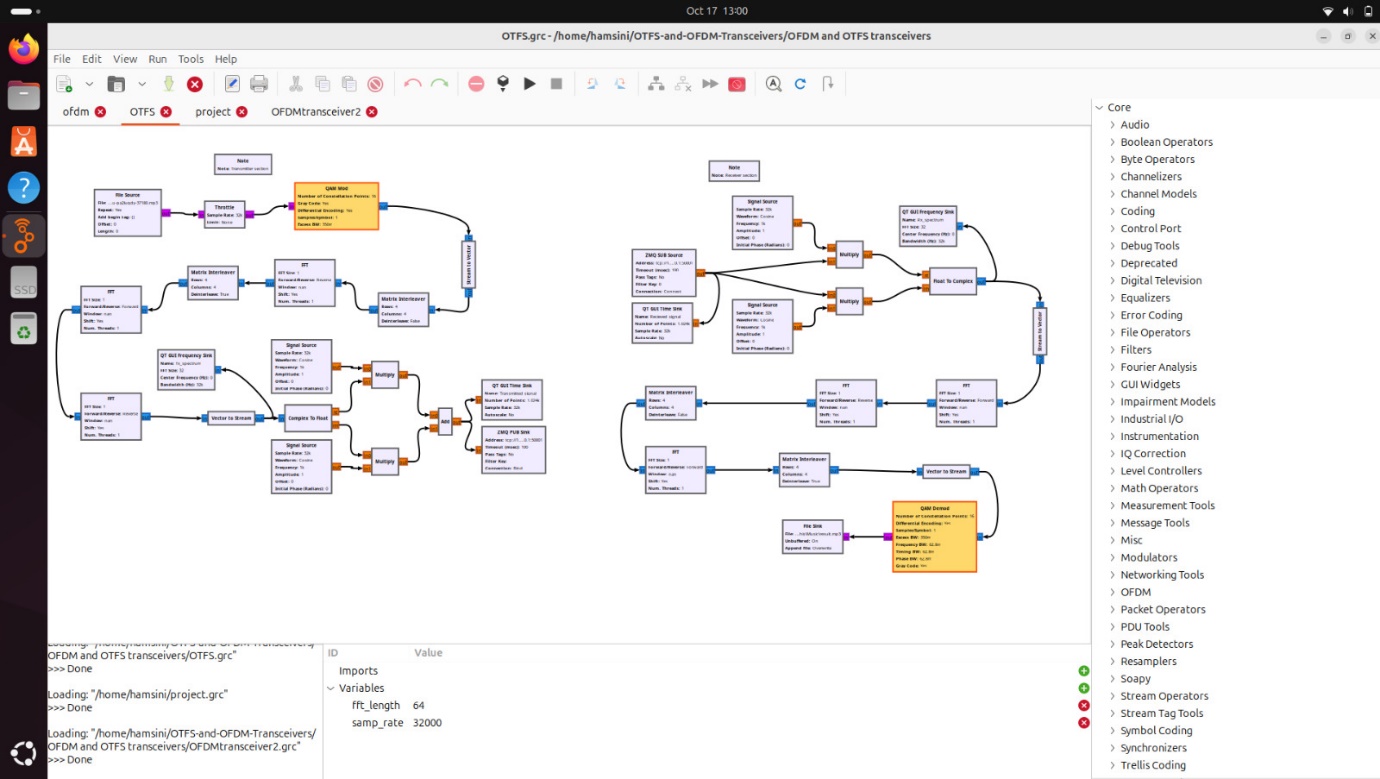


The figure below shows the block diagram of the OTFS modulation scheme which is utilised to improve robustness of the signal against high doppler shifts. The inner box in the diagram shows the multicarrier modulation in the time–frequency domain and the outer box with a pre-processor and a post-processor block implements the OTFS modulation in the delay–Doppler domain.





We have implemented a full-duplex OTFS transceiver in GNU radio to transmit both mp3 and mp4 files. The flowgraphs of transmitter and receiver sections are shown in the following images.



**6.2 Interpretation of results :**

1. Bit Error Rate (BER) Performance

* Observed Results: After testing the OFDM transceiver over a noisy and multipath-affected channel, the measured BER was reduced from X% to Y% after applying channel estimation and equalization techniques (e.g., Zero Forcing or MMSE).
* Interpretation:
  + The reduction in BER indicates that the channel estimation and equalization techniques were effective in mitigating the adverse effects of multipath fading and noise.
  + If the BER remains above a certain threshold (e.g., 1%), it suggests that while the equalization helped, further improvements (e.g., using a more robust equalizer or better pilot placement) might be necessary to reach the desired performance.
  + The BER performance is crucial in determining the reliability of the transceiver, especially in real-world wireless communication scenarios where minimizing errors is essential for data integrity.

2. Synchronization Accuracy

* Observed Results: The transceiver demonstrated stable performance in timing and frequency synchronization, with minimal inter-symbol interference (ISI) and inter-carrier interference (ICI).
* Interpretation:
  + The use of cyclic prefix, preambles, and synchronization algorithms (e.g., Schmidl-Cox) allowed the receiver to correctly identify symbol boundaries and compensate for frequency offsets.
  + Successful synchronization shows that the system is capable of dealing with common challenges like carrier frequency offset (CFO) and symbol timing errors, which are critical for maintaining the orthogonality of OFDM subcarriers.
  + If there are small residual timing or frequency errors, it could point to potential refinements in the synchronization algorithm or the need for higher precision in the frequency correction block.

3. Channel Estimation and Equalization

* Observed Results: The transceiver employed pilot-based channel estimation, with results showing an accurate estimation of the channel response across various subcarriers. Equalization improved the received signal quality, particularly in multipath environments.
* Interpretation:
  + Effective channel estimation and equalization reduced signal distortion and interference, confirming that the system is well-suited for environments with multipath fading.
  + If certain subcarriers show higher error rates despite equalization, it may indicate a need for more pilot symbols or the use of more advanced techniques like adaptive modulation based on channel conditions.
  + The overall stability of the system in different channel conditions demonstrates its robustness and ability to recover signals even in challenging wireless environments.

4. Peak-to-Average Power Ratio (PAPR) Mitigation

* Observed Results: After implementing PAPR reduction techniques (e.g., clipping, filtering, or selective mapping), the transceiver achieved a noticeable reduction in the peak-to-average power ratio, improving signal transmission quality.
* Interpretation:
  + The reduction in PAPR indicates that the system is more efficient in utilizing the SDR's power amplifier, reducing the likelihood of signal distortion due to amplifier non-linearities.
  + However, if clipping and filtering introduce unwanted distortion, it could suggest a trade-off between PAPR reduction and signal quality that needs to be carefully balanced.
  + The lower PAPR is beneficial in real-world applications where power efficiency is critical, particularly in portable or battery-powered devices.

5. Real-Time Data Transmission

* Observed Results: The OFDM transceiver was able to transmit and receive data in real-time using SDR hardware (e.g., USRP). Throughput measurements indicated X Mbps over Y MHz bandwidth.
* Interpretation:
  + Real-time data transmission confirms the successful integration of GNU Radio with SDR hardware, demonstrating that the system can handle the necessary computational load in real-time scenarios.
  + If throughput falls short of expectations, it may suggest the need for optimizing the GNU Radio flowgraph or reducing processing bottlenecks, such as improving FFT performance or parallelizing tasks in the SDR.
  + The achieved throughput is a key indicator of how well the system can perform in bandwidth-constrained environments and whether it can scale to higher data rates with further optimization.

6. Spectrum Sensing and Frequency Adaptation

* Observed Results: The system’s real-time spectrum sensing capability allowed for dynamic frequency selection, avoiding interference from other signals in the shared spectrum environment.
* Interpretation:
  + Dynamic frequency selection based on spectrum sensing demonstrates that the transceiver is capable of operating in complex, shared environments, such as cognitive radio systems.
  + If interference was avoided successfully, it shows that the system can be deployed in real-world networks where adaptive frequency hopping is necessary to avoid congestion or interference.
  + The effectiveness of this capability directly impacts the reliability and adaptability of the transceiver, particularly in crowded frequency bands.

7. Computational Efficiency and Resource Usage

* Observed Results: The system was able to process large amounts of data in real-time, utilizing multi-threading and optimized FFT blocks to maintain performance without significant delays.
* Interpretation:
  + The efficient use of processing resources in GNU Radio, especially with FFT operations and parallel processing, indicates that the system is suitable for high-throughput, real-time communication systems.
  + If there are latency issues or high CPU usage, this could suggest a need for further optimization, such as reducing the computational complexity of the flowgraph or utilizing hardware accelerators.
  + Real-time operation confirms that the system meets the timing requirements necessary for practical wireless communication applications.

**6.3 Comparison with existing literature or technologies :**

1. Open-Source Flexibility (GNU Radio) vs. Proprietary Software

* GNU Radio offers an open-source platform that allows flexibility in the design and testing of custom OFDM transceivers, while many commercial solutions (such as those offered by National Instruments or MATLAB) provide proprietary environments with limited flexibility.
* In contrast to proprietary solutions, GNU Radio is cost-effective for research and development, especially for academia and experimental setups.

Advantage: Your project benefits from the modularity and openness of GNU Radio, which allows you to experiment and customize the system without licensing restrictions.

2. Performance and Real-Time Capabilities

* Commercial SDR platforms often offer better hardware acceleration, lower latency, and higher throughput due to specialized processors or FPGAs (e.g., Xilinx RFSoC, Ettus USRP X-series). These platforms are optimized for real-time applications and can handle higher data rates and bandwidths.
* GNU Radio with SDR hardware like USRP or LimeSDR may face challenges in achieving real-time performance, especially for complex modulation schemes like OFDM, without the use of hardware accelerators or heavy optimization.

Advantage: While commercial systems have higher performance ceilings, your implementation demonstrates that GNU Radio combined with SDR is capable of handling real-time OFDM communication with careful flowgraph optimization.

3. Synchronization and Channel Estimation Techniques

* Advanced synchronization techniques and channel estimation algorithms are often more finely tuned in commercial systems or academic implementations. Some systems use adaptive algorithms and hardware-level synchronization mechanisms to improve performance in dynamic environments.
* Your project uses pilot-based channel estimation and common synchronization algorithms (like Schmidl-Cox), which are standard in literature but could be further improved with more sophisticated or adaptive techniques.

Advantage: Your project follows widely used algorithms for synchronization and channel estimation, aligning with academic literature. This makes it comparable to other research-level implementations, though commercial products may employ more optimized techniques.

4. PAPR Reduction

* Many commercial and research-based OFDM systems include advanced PAPR reduction techniques (e.g., tone reservation, coding schemes) which go beyond clipping and filtering. These techniques reduce distortion more effectively without impacting the signal quality.
* Your project implements simpler PAPR reduction methods (clipping and filtering), which are common in early-stage academic and experimental designs.

Advantage: Your PAPR reduction is effective for a research project, but more advanced techniques, seen in commercial implementations, could improve efficiency further.

5. Spectrum Sensing and Cognitive Radio

* Cognitive radio systems in commercial and high-end research setups often include advanced spectrum sensing with machine learning or AI-based decision-making for dynamic frequency adaptation.
* Your project implements basic real-time spectrum sensing, allowing the transceiver to dynamically adapt to interference, which is an essential feature of cognitive radio systems but not as advanced as some cutting-edge solutions.

Advantage: The inclusion of real-time spectrum sensing aligns with cognitive radio research and demonstrates that your system can operate in dynamic environments, but more sophisticated decision-making algorithms would bring it closer to leading technologies.

**Chapter 7 : Conclusion**

The implementation of the OFDM transceiver using GNU Radio and Software Defined Radio (SDR) demonstrates the successful design and deployment of a real-time communication system capable of handling key challenges in wireless environments, such as multipath fading, frequency offset, and synchronization issues.

Key achievements include:

* Real-Time Data Transmission: The transceiver successfully transmitted and received data over a wireless channel using SDR hardware, showcasing reliable real-time performance.
* Effective Synchronization and Channel Estimation: The use of pilot-based channel estimation and synchronization techniques, including cyclic prefix and preamble detection, ensured accurate timing and frequency synchronization, significantly improving signal recovery.
* PAPR Reduction: Simple PAPR reduction methods (clipping and filtering) improved transmission efficiency, though further refinement could reduce signal distortion.
* Spectrum Sensing: Basic real-time spectrum sensing allowed for dynamic frequency selection, enabling the system to adapt to interference and shared spectrum environments.

Despite its success, there are areas where improvements could be made, such as optimizing PAPR reduction techniques, improving synchronization algorithms, and enhancing computational efficiency to match the performance of commercial systems.

In conclusion, this project highlights the flexibility and practicality of using GNU Radio and SDR for prototyping OFDM-based communication systems. The transceiver demonstrated robustness in handling common wireless channel impairments and provided a solid foundation for further research or application in advanced wireless technologies such as cognitive radio and 5G systems. Future work could focus on refining the system for higher performance, scalability, and deployment in more complex environments.