**IMPLEMENTATION AND FUNCTIONAL VERIFICATION OF COMMUNICATION PROTOCOLS**

**Submitted**

**By**

**S Venkatesh - BU21EECE0100308**

**P Sai Chaithanya - BU21EECE0100208**

**G Chethan Sree Royal - BU21EECE0100529**

**Under the Guidance of**

**Dr S Karthick**

**Associate Professor,**

**Department of Electrical, Electronics and Communication Engineering,**

**GITAM Deemed to be University**

**Doddaballapura-561203, Bengaluru Campus,**

**Karnataka, India.**

**(Duration: 05/12/2024 to 15/03/2025)**



**Department of Electrical, Electronics and Communication Engineering**

**GITAM School of Technology**

**GITAM**

**(DEEMED TO BE UNIVERSITY)**

**(Estd. u/s 3 of the UGC act 1956)**

**NH 207, Nagadenehalli, Doddaballapur taluk, Bengaluru-561203 Karnataka, INDIA.**

**DECLARATION**

**I/We declare that the project work contained in this report is original and it has been done by me under the guidance of my project guide Dr S Karthick.**

**Name:**

**Date: Signature of the Student**

**Department of Electrical, Electronics and Communication Engineering**

**GITAM School of Technology, Bengaluru-561203**

**A logo with text on it

AI-generated content may be incorrect.**

**CERTIFICATE**

**This is to certify that S Venkatesh, P Sai Chaithanya and G Chethan Sree Royal bearing BU21EECE0100308, BU21EECE0100208 and BU21EECE0100529 has satisfactorily completed Major Project Entitled in partial fulfillment of the requirements as prescribed by University for VIIIth semester, Bachelor of Technology in “Electrical, 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)

# Chapter 1: Introduction

Reliable and efficient data exchange depends fundamentally on communication protocols when dealing with embedded systems as well as automotive networks and industrial applications. The goal of this project involved both implementing and verifying the operation of UART (Universal Asynchronous Receiver-Transmitter) and FlexRay frame protocols. The automotive and embedded domains use these protocols extensively because they offer robustness as well as scalability and efficiency in real-time communication networks.

Both UART and FlexRay protocols were implemented with Xilinx Vivado 2016.2 through which the teams designed and verified their solutions using Verilog HDL. The verification process included testbench creation and waveform analysis as well as data transmission verification tests. Our research on I2C, SPI and CAN communication protocols evolved into the latest project covered at the IEEE International Conference on Emerging Smart Computing and Informatics (ESCI-2025 7th edition) through a paper presentation. The Best Paper Award during the session affirmed our protocol verification and implementation work based on the research paper submission.

1.1 Overview of the problem statement

The effective operation of both embedded devices and automotive components depends on dependable communication links between electronic control units (ECUs) and peripheral devices. Building data exchange through protocols UART and FlexRay takes place frequently because verifying both protocols' correct functionality and performance demands strict verification methods. The correct design of hardware depends heavily on functional verification because protocol implementation errors trigger data loss alongside signal degradation and total system breakdown.

## 1.2 Objectives and goals

* Design and implement UART and FlexRay communication protocols in Verilog using Vivado 2016.2, ensuring accurate signal transmission and data integrity.
* Develop testbenches to verify protocol functionality through waveform analysis, ensuring reliable data communication.
* Extend prior research on I2C, SPI, and CAN protocols, integrating findings to enhance the understanding of embedded and automotive communication systems.
* Achieve a fully functional and verified implementation of UART and FlexRay with optimized performance and error-free communication.

# Chapter 2 : Literature Review

1. Cho et al. (2021) propose an FPGA-based ECU for remote reconfiguration in automotive systems, addressing the limitations of traditional microcontroller-based ECUs. Their approach enhances performance, scalability, and security while enabling over-the-air (OTA) updates and fault detection. The study demonstrates improved efficiency, real-time adaptability, and cybersecurity compared to conventional systems. By integrating reconfigurable computing, this work contributes to the advancement of intelligent and connected vehicle architectures.
2. Shanker and Fahmy (2015) present an extensible FlexRay communication controller for FPGA-based automotive systems, addressing the need for high-speed and reliable in-vehicle communication. The proposed design enhances flexibility and scalability, enabling efficient integration with evolving automotive networks. By leveraging FPGA technology, the controller supports real-time data exchange with improved fault tolerance and adaptability. This work contributes to the development of robust automotive communication systems, essential for modern intelligent and autonomous vehicles.
3. Avinash et al. (2012) explore the FlexRay protocol for automotive applications, highlighting its advantages over traditional communication protocols like CAN and LIN. The study emphasizes FlexRay’s high-speed, deterministic, and fault-tolerant nature, making it suitable for safety-critical automotive systems. The authors discuss its implementation and benefits in real-time vehicle control, contributing to the advancement of robust and efficient in-vehicle communication networks.
4. The FlexRay Consortium (2005) provides the FlexRay Requirements Specification (Version 2.1), defining the high-speed, deterministic, and fault-tolerant communication standard for automotive applications. FlexRay improves upon traditional protocols like CAN and LIN by offering higher bandwidth and reliability, making it suitable for safety-critical and real-time vehicle control systems. This specification serves as a foundation for developing advanced automotive communication networks, supporting the evolution of intelligent and autonomous vehicles.
5. Shreejith and Fahmy (2014) propose an extensible FlexRay communication controller for FPGA-based automotive systems, enhancing real-time data exchange and fault tolerance. The design improves flexibility and scalability, making it suitable for evolving in-vehicle networks. By leveraging FPGA technology, the controller ensures high-speed, deterministic communication, contributing to the development of reliable and adaptive automotive communication systems.

# 

# Chapter 3: Strategic Analysis and Problem Definition

# 3.1 SWOT Analysis

Organizations use the specific methodology of SWOT (Strengths, Weaknesses, Opportunities, and Threats) analysis to assess project feasibility together with challenges and potential improvements.

**Strengths:**

* The development includes the implementation of industry-standard UART and FlexRay communication protocols through Verilog programming and usage of Vivado 2016.2.
* Reliable system data is guaranteed by conducting functional verification through tests and analyzing waveforms in simulations.
* The study extends work previously shown at ESCI-2025 which establishes the researcher as an expert in protocol validation procedures.

**Weaknesses:**

* The performance requirements related to complex protocol implementation and testing steamed from the available time.
* The main emphasis in this work is on simulation-based verification because hardware validation exists at limited levels.

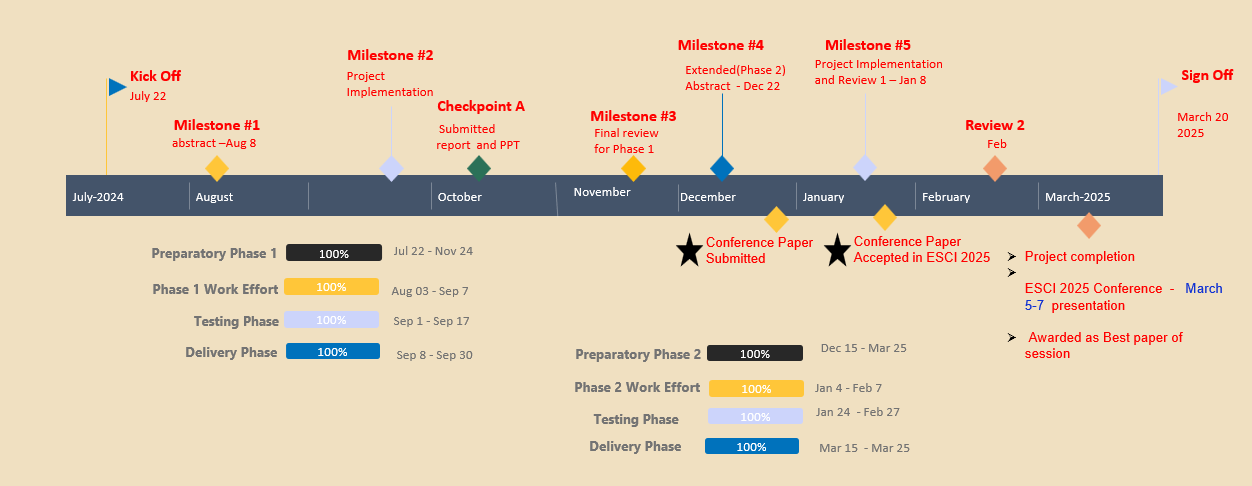
**Opportunities:**

* An opportunity exists to integrate hardware components on Field Programmable Gate Arrays for improving real-time practical validation.
* The system finds potential applications in automotive safety-critical sectors.

**Threats:**

* FlexRay systems might experience synchronization problems due to their high-speed data transmission needs.
* Debugging complexities in multi-protocol integration.

### 3.2 Project Plan - GANTT Chart



##### 3.3 Refinement of problem statement

Embedded and automotive systems require flawless communication between electronic control units (ECUs) as their operational success depends on this functionality. Verification of both UART and FlexRay protocols is essential to prevent data loss together with synchronization errors and signal integrity problems. The analysis continues by defining the following refined issue:

* The design requires implementation of UART and FlexRay using Verilog under Vivado 2016.2.
* A testbench should be used for functional verification procedures which must be examined through waveform analysis.
* The solution needs to address particular challenges from UART using baud rate correction along with FlexRay requiring timing constraint resolution.

# Chapter 4: Methodology

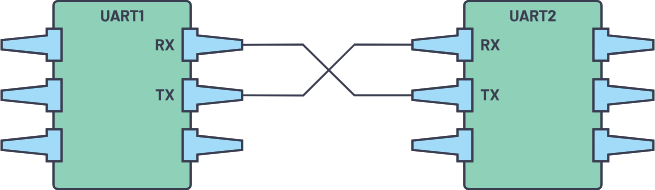
## 4.1 Description of the approach

1. **UART Protocol**

Embedded systems, microcontrollers, and computers mostly use UART as a form of device-to-device hardware communication protocol. Among the available communication protocols, UART uses only two wires for its transmitting and receiving ends.

By definition, UART is a hardware communication protocol that uses asynchronous serial communication with configurable speed. Asynchronous means there is no clock signal to synchronize the output bits from the transmitting device going to the receiving end.

Interference:

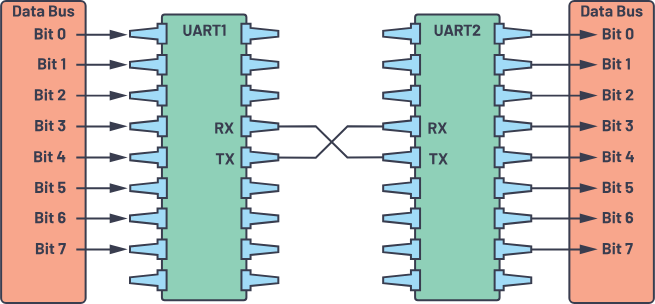


***Figure 1. Two UARTs directly communicate with each other.***

The two signals of each UART device are named:

* Transmitter (Tx)
* Receiver (Rx)

The main purpose of a transmitter and receiver line for each device is to transmit and receive serial data intended for serial communication.

 ***Figure 2. UART with data bus.***

The transmitting UART is connected to a controlling data bus that sends data in a parallel form. From this, the data will now be transmitted on the transmission line (wire) serially, bit by bit, to the receiving UART. This, in turn, will convert the serial data into parallel for the receiving device.

The UART lines serve as the communication medium to transmit and receive one data to another. Take note that a UART device has a transmit and receive pin dedicated for either transmitting or receiving.

For UART and most serial communications, the baud rate needs to be set the same on both the transmitting and receiving device. The baud rate is the rate at which information is transferred to a communication channel. In the serial port context, the set baud rate will serve as the maximum number of bits per second to be transferred.

|  |  |
| --- | --- |
| **Wires** | **2** |
| Speed | 9600, 19200, 38400, 57600, 115200, 230400, 460800, 921600, 1000000, 1500000 |
| Methods of Transmission | Asynchronous |
| Maximum Number of Masters | 1 |
| Maximum Number of Slaves | 1 |

Table 1 summarizes what we must know about UART.

The UART interface does not use a clock signal to synchronize the transmitter and receiver devices; it transmits data asynchronously. Instead of a clock signal, the transmitter generates a bitstream based on its clock signal while the receiver is using its internal clock signal to sample the incoming data. The point of synchronization is managed by having the same baud rate on both devices. Failure to do so may affect the timing of sending and receiving data that can cause discrepancies during data handling. The allowable difference of baud rate is up to 10% before the timing of bits gets too far off.

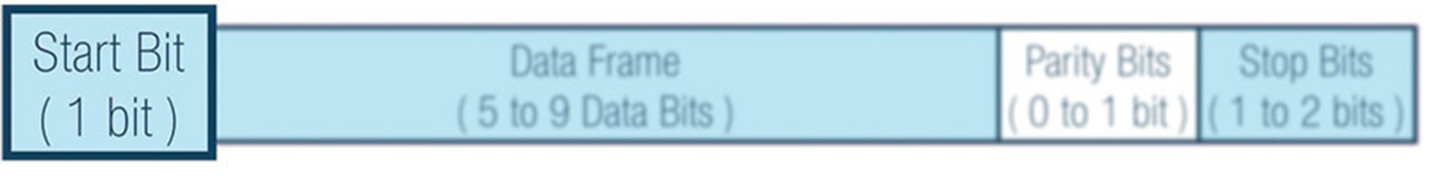
### **Data Transmission**

### In UART, the mode of transmission is in the form of a packet. The piece that connects the transmitter and receiver includes the creation of serial packets and controls those physical hardware lines. A packet consists of a start bit, data frame, a parity bit, and stop bits.

 ***Figure 3. UART packet.***

**Start Bit**

The UART data transmission line is normally held at a high voltage level when it’s not transmitting data. To start the transfer of data, the transmitting UART pulls the transmission line from high to low for one (1) clock cycle. When the receiving UART detects the high to low voltage transition, it begins reading the bits in the data frame at the frequency of the baud rate.

 ***Figure 4. Start bit.***

#### **Data Frame**

The data frame contains the actual data being transferred. It can be five (5) bits up to eight (8) bits long if a parity bit is used. If no parity bit is used, the data frame can be nine (9) bits long. In most cases, the data is sent with the least significant bit first.

 ***Figure 5. Data frame.***

#### **Parity**

Parity describes the evenness or oddness of a number. The parity bit is a way for the receiving UART to tell if any data has changed during transmission. Bits can be changed by electromagnetic radiation, mismatched baud rates, or long-distance data transfers.

After the receiving UART reads the data frame, it counts the number of bits with a value of 1 and checks if the total is an even or odd number. If the parity bit is a 0 (even parity), the 1 or logic-high bit in the data frame should total to an even number. If the parity bit is a 1 (odd parity), the 1 bit or logic highs in the data frame should total to an odd number.

When the parity bit matches the data, the UART knows that the transmission was free of errors. But if the parity bit is a 0, and the total is odd, or the parity bit is a 1, and the total is even, the UART knows that bits in the data frame have changed.

 ***Figure 6. Parity bits.***

#### **Stop Bits**

To signal the end of the data packet, the sending UART drives the data transmission line from a low voltage to a high voltage for one (1) to two (2) bit(s) duration.

 ***Figure 7. Stop bits***

**Advantages:**

* Hardware complexity is low.
* As this is one to one connection between two devices, software addressing is not required.
* Due to its simplicity, it is widely used in the devices having 9 pin connector.

**Disadvantages:**

* It is suitable for communication between only two devices.
* It supports fixed data rate between devices wanting to communicate otherwise data will be garbled.

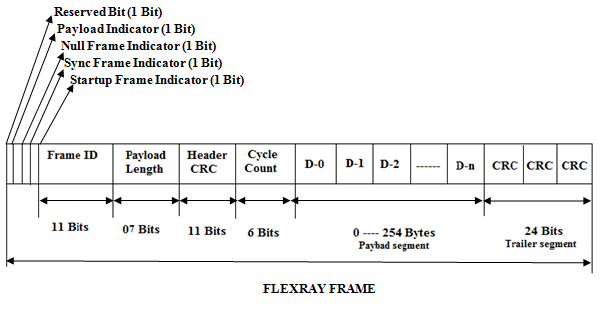
1. **FlexRay Frame Protocol**

FlexRay is an automotive network communications protocol developed through the FlexRay Consortium to govern on-board automotive computing. It is designed to be faster and reliable than CAN and TTP, but it is more expensive. Visual Sim FlexRay library enables a system designer to construct models of complex standard and non-standard FlexRay topologies. The graphical model of the FlexRay topology can contains any number of nodes that transmit across the static and dynamic slots. The FlexRay consortium disbanded in 2009. The FlexRay standard is a set of ISO standards ISO 17458-1 to 17458-5.

FlexRay supports high data rates, up to 10 Mbit/s, explicitly supports both star and party line bus topologies, and it consist of two independent data channels for fault-tolerance. The bus operates on a time cycle, divided into two parts: the static segment and the dynamic segment. The static segment is pre-allocated into slices for individual communication types, providing a stronger real-time guarantee than its predecessor CAN. The dynamic segment operates more like CAN, with nodes taking control of the bus as available, allowing event-triggered behavior.

**OVERVIEW FLEXRAY FRAME STRUCTURE**

An overview of the general FlexRay frame format is illustrated in Figure. The FlexRay frame format divides into three segments: Header, Payload, and Trailer.



**Header:**

The FlexRay header segment is 5 bytes long. It consists of 9 parts. Each part has the different function.

1. Reserved bit (1 bit) – It is reserved for future protocol use
2. Payload preamble indicator (1 bit) – It indicates the extension of vector information in the frames payload segment. If the frame is transmitted in the static segment, this position indicates the presence of a network management vector at the beginning of the payload. If the frame is transmitted in the dynamic segment, this position indicates the presence of a frame ID at the beginning of the payload.
3. Null frame indicator (1 bit) – It is to identify the frame is empty. ”0” means the payload segment contains no valid data; “1” means the payload segment contains valid data.
4. Sync frame indicator (1 bit) – It is to check the frame is a sync frame. If “0” means no synchronization for node; and “1” means all receiving nodes are used for synchronization.
5. Startup frame indicator (1 bit) – It indicates frame is a startup frame. “0” means this frame is not a startup frame; “1” means this frame is a startup frame.
6. Frame ID (11 bits) - A frame ID is unique on each channel in CC. The frame ID ranges from 1 to 2047. 0 is invalid.
7. Payload length (7 bits) - This part is used to indicate the size of the payload segment. The value of the payload length position is set to the number of payload bytes divided by 2. Its range is from 0 to 254 bytes.
8. Header CRC (11 bits) - This part contains a cyclic redundancy check code (CRC), the startup frame indicator, the frame ID, and the payload length. The header CRC of transmitted frames is computed offline and provided to the Communication Controller by means of configuration. It is not computed by transmitting CC. The CC calculates the received frame’s header CRC in order to check that the CRC is correct.
9. Cycle count (6 bits) - This part indicates the value of cycle counter, from the transmitting node's at the time of frame transmissions.

**Payload:**

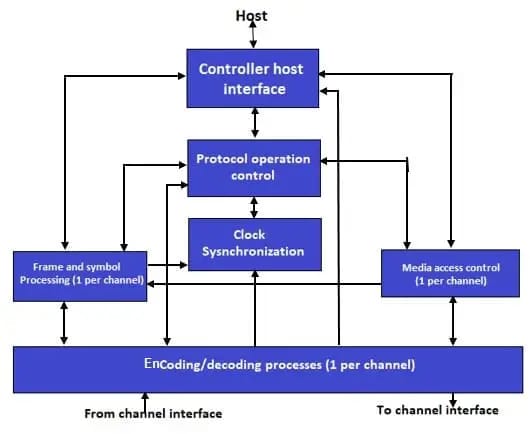
The Flex Ray payload segment contains 0 to 254 bytes data (0 to 127 two-byte words). It is used to notice that the payload segment contains even number of bytes

**E. Trailer:**

The Flex Ray trailer segment consists of 24-bit cyclic redundancy check code (CRC) for the frame. It is computed with the data in the header segment and the payload segment of the frame .

A Flex Ray communication system consists of number of FR nodes and physical transmission medium (FR bus) for interconnecting all of the Flex Ray nodes. FR nodes are also called as ECU, which is connected to a Flex Ray bus via a FR interface. Flex Ray interface consist of communication Controller and one/two bus drivers depending up on the number of channels. Basically in FR communication system, as two channels for designing it can choose between single channel and dual channel configuration. Based on the design the communication channels operate at the data rate of 10 Mbit/sec.

**FlexRay Protocol Working Principle:**



4.2 Tools and techniques utilized

Tools:

* Vivado 2016.2 functions as the FPGA design and simulation tool for UART and FlexRay implementations which includes coding and synthesizing and verification features.
* Verilog HDL serves as the Hardware description language to design communication protocols while testbenches get developed in this language.
* Simulation and verification of protocol functionality happens in Verilog using testbenches for waveform analysis.
* Waveform Analyzer functioned to both analyze signal shifts and confirm the correctness of data transfers.

Techniques:

* During this phase developers constructed UART and FlexRay modules with Verilog focusing on the simulation of genuine communication situations.
* Testing through functional verification produced testbenches to check proper data transfers which resulted in error-free operation.
* The system used waveform analysis for verifying protocol times during synchronization and establishing data integrity.

#### 4.3 Design considerations

* The selection of UART and FlexRay protocols occurred because both have extensive application in embedded and automotive systems for dependable data transmission.
* Hardware Description Language (HDL): Used Verilog for efficient design and simulation of communication protocols.
* Verification of the simulation took place in Vivado 2016.2 which demonstrated proper synthesis functionality and correct operation.
* Baud Rate and Timing Constraints: Ensured proper baud rate selection in UART and managed time-triggered communication in FlexRay for synchronization.
* The system includes error detection through parity bits implemented in UART while FlexRay adds cycle count validation to maintain data integrity.

# Chapter 5: Implementation

# 5.1 Description of how the project was executed

* Looped the development process by implementing UART and FlexRay protocols through Verilog programming in Vivado 2016.2.
* Developed testbenches for functional verification and waveform analysis.
* The UART module received simulation tests which validated proper baud rate operations and transmission processes.
* Development of the FlexRay frame showed proper function for synchronization combined with error detection features.
* The team examined waveforms to verify both the protocol operations alongside proper data transfer success.

### 5.2 Challenges faced and solutions implemented

* An adjustment of UART settings solved the synchronization problems caused by baud rate mismatch.
* FlexRay Timing Constraints → Optimized clock settings for precise data transfer.
* The team solved signal transmission errors through waveform analysis for clear detection and error correction purposes.
* Limited Hardware Validation → Focused on robust simulation and verification techniques.

# Chapter 6:Results

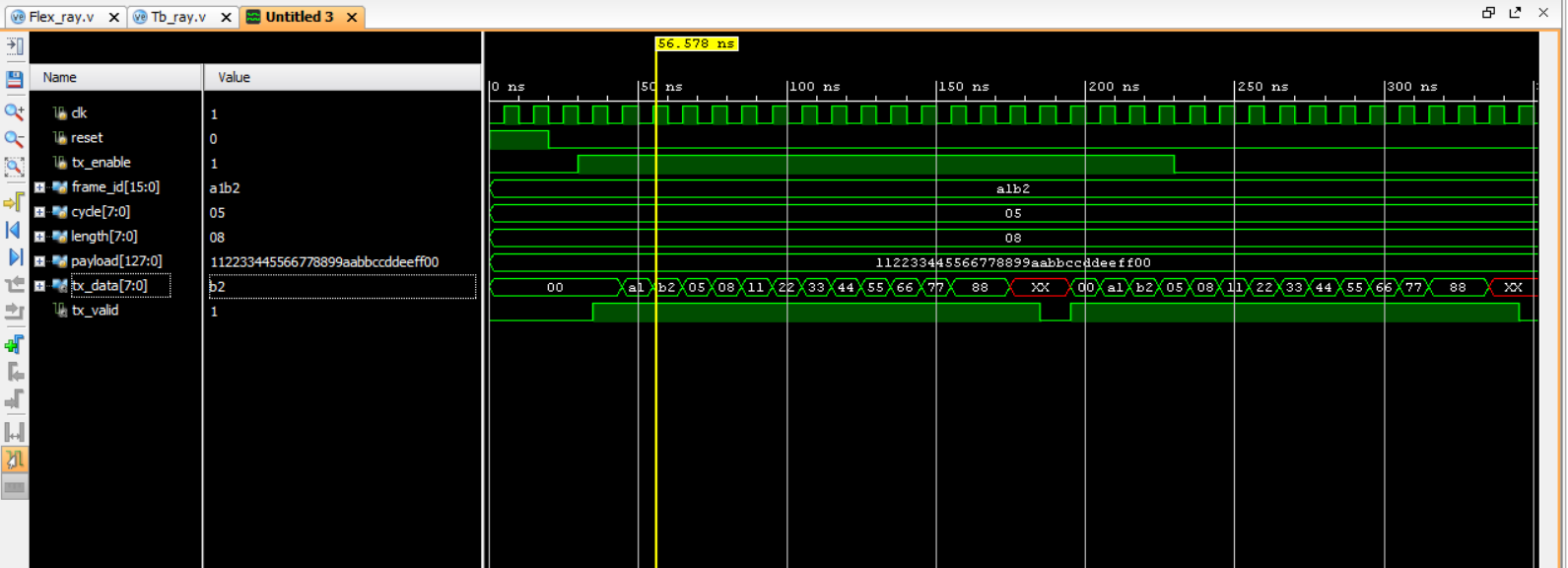
# 6.1 outcomes

**i)** **UART Waveforms**

A screenshot of a computer

Description automatically generated

**ii)** **FlexRay Waveforms**



6.2 Interpretation of results

1. **UART**

During the simulation of UART protocol in Vivado 2016.2 developers succeeded in maintaining a consistent clock signal for operational stability. The (w\_TX\_Active) signal brings valid data transmission indications to the system while (w\_TX\_Serial) changes accompany the sending of bits to the network. The (w\_TX\_Byte[7:0]) byte transmitted through observation leads to successful data transmission when it matches (w\_RX\_Byte[7:0]). The RX data validity is confirmed through the (w\_RX\_DV) signal while (o\_TX\_Done) indicates that transmission is complete. Analysis of communication intervals supports the correct baud rate configuration. Any discrepancies between (w\_TX\_Byte[7:0]) and (w\_RX\_Byte[7:0]) will require investigating potential signal integrity issues as well as baud rate errors.

1. **FlexRay**

Simulation results from Vivado 2016.2 prove that data transmission is operational when the clock signal (clk) remains steady and reset (reset) returns to disabled state. Data transfer begins through the active tx\_enable while frame\_id contains a1b2 as its 16-bit value. The cycle count value (cycle[7:0]) settles at 05 to ensure proper synchronization and the payload length (length[7:0]) is set to 08 bytes. The payload (payload[127:0]) contains the anticipated sequence 112233445566778899aabbccddeeff00 while the data transmitted through (tx\_data[7:0]) properly starts with b2 thereby demonstrating byte-by-byte data movement. The transmission valid (tx\_valid) signal shows assertion for confirming correct output data. The data transmission matches the correct frame structure yet the presence of XX code at the waveform end hints at possibly incomplete data transfer or storage problems or data corruption issues. Furthers testing must take place to verify that proper data reception functions correctly.

6.3 Comparison with existing literature or technologies

The project evaluates UART and FlexRay protocol implementation together with protocol verification compared to current communication systems and investigations of I2C, SPI, and CAN. UART positions itself as an established asynchronous serial protocol that uses FlexRay as a high-speed protocol specialized for automotive and real-time operations. UART provides an easier communication method yet does not support multidevice operations at the same level as FlexRay which outperforms CAN across fault tolerance measures along with bandwidth and synchronization capabilities.

The automotive industry currently uses CAN as its main in-vehicle communication standard while FlexRay attracts automotive manufacturers with its time-triggered structure and fault-tolerant capabilities for high-performance systems. Previous studies at ESCI-2025 about I2C, SPI, and CAN examined error detection and verification systems which align with the study’s work concerning UART and FlexRay error detection and verification systems. The research extends previous studies by implementing waveform examination and timing assessment methods which results in better transmission quality and system reliability.

The implementation stands out because UART maintains its necessity for basic serial communication, yet FlexRay achieves better results for fast automotive networks thus meeting embedded and vehicular system requirements.

# Chapter 7: Conclusion

The execution of UART and FlexRay protocols in Vivado 2016.2 shows their applicable dependability features coupled with efficiency when used for embedded systems and automobiles. Through UART simulations the correct data transmission occurred thanks to proper baud rate setup together with waveform analysis for ensuring error-free communication. The FlexRay simulation demonstrated correct transmission of frames and synchronization but additional work remains to solve undefined XX values during payloads.

UART exists as a core serial communication method but FlexRay delivers improved speed alongside fault tolerance that enables superior data transfer determinism thus making it an effective alternative to CAN in automotive networks. The current study expands upon former studies of I2C, SPI, and CAN by exploring additional understandings of protocol confirmation protocols and measurement of real-time performance.

# Chapter 8: Future Work

#### The expansion of this project will result in producing an IEEE paper which details UART and FlexRay protocol implementation together with functional verification testing. The paper conducts a comprehensive breakdown of protocol design while examining simulation outcomes and verification strategies to deliver scientific contributions for embedded as well as automotive communication systems.

#### This research needs additional support through a comparative study between I2C, SPI, and CAN protocols which will demonstrate how the proposed approach performs better regarding latency performance and error management while keeping data integrity higher. The implementation of hardware systems on FPGA will be carried out to validate realistic operational performance that exceeds simulation capabilities. Improvements in error detection techniques along with buffer management strategies and power optimization methods will be investigated for improved protocol efficiency.

#### The paper maintains strict IEEE standards while presenting its content through a structured presentation method with appropriate citations and experimental proof and proposed research directions for the future. The research project seeks acceptance by a respected IEEE conference or journal as part of its mission to advance embedded system communication protocols.

#### 

# 

# References

1. Kwonneung Cho , Jeongeun Kim , DoYound Choi , Yung Hyun Yoon , Jung Hwan Oh and Seung Eun Lee “An FPGA-Based ECU for Remote Reconfiguration in Automotive Systems” *Micromachines* 2021, *12*(11), 1309; <https://doi.org/10.3390/mi12111309>.
2. Shreejith Shanker and Suhaib A Fahmy “Extensible FlexRay Communication Controller for FPGA-Based Automotive Systems” IEEE transactions on vehicular technology, vol. 64, no. 2, february 2015.
3. Avinash K R,P Nagaraju, Surendra S,Shivaprasad S, ”FlexRay protocol based an automotive application”, International journal of Emerging technology and Advanced Engineering (IJETAE),vol.2,no.5,May 2012.
4. Flexray Consortium: FlexRay Requirements Specification (Version 2.1), 2005.URL <http://www.flexray.com/>.
5. Shanker Shreejith Student Member, IEEE and Suhaib A. Fahmy Senior Member “Extensible FlexRay Communication Controller for FPGA Based Automotive Systems” IEEE Transacation on Vehicular technology, vol.6, no.99, May 2014.
6. C. He, Y. Xia, and L. Wang, “A universal asynchronous receiver transmitter design,” Int’l Conf. on Elect. Comm. and Control (ICECC 2011), Ningbo, China, Sept. 2011.
7. [B. Priyanka](https://ieeexplore.ieee.org/author/37088862926); [M Gokul.](https://ieeexplore.ieee.org/author/37088878633); [A Nigitha.](https://ieeexplore.ieee.org/author/37088879658); [J.T Poomica.](https://ieeexplore.ieee.org/author/37088881836) “Design of UART Using Verilog And Verifying Using UVM” .DOI:[10.1109/ICACCS51430.2021.9441789](https://doi.org/10.1109/ICACCS51430.2021.9441789).Electronic ISSN: 2575-7288.
8. Kashyap B, Ravi V. Universal Verification Methodology Based Verification of UART Protocol. InJournal of Physics: Conference Series 2020 Dec 1 (Vol. 1716, No. 1, p. 012040). IOP Publishing.
9. Agrawal RK, Mishra VR. The design of high speed UART. In2013 IEEE Conference on Information & Communication Technologies 2013 Apr 11 (pp. 388-390). IEEE.