10.48047/jocaaa.2024.33.08.60

DESIGN OF A MEALY FSM FOR SERIAL DATA RECEPTION WITH EVEN PARITY CHECK ON 8-BIT BLOCKS

Mr.A.PRASAD [(1)] , Mr.N.B.JILANI [(2)] , KOTLA GURAVA REDDY [(3),] SHAIK KHASIM SHARIF [(4),] A VENNA HARIKRISHNA REDDY[(5),] DANDA NAGENDRA REDDY [(6)]

[1,2] *Faculty ECE Department, Krishna Chaitanya Institute of Technology & Sciences-Markapur, AP, India.*

[3,4,5,6,] *Student , ECE Department, Krishna Chaitanya Institute of Technology & Sciences,Markapur, AP, India.*

Abstract In this work, we design a Mealy finite state machine (FSM) for synchronously receiving serial 1-bit data and performing a parity check on every 8-bit block. The FSM is triggered by a start signal, with data reception continuing until the start signal is deasserted. A timer signal, T, marks the end of every 8-bit reception, at which point the FSM evaluates the last 8 bits to determine if they contain an even number of 1s. If the number of 1s is even, the output Y is asserted high; otherwise, Y is low. The system ensures that the receiver halts upon deassertion of the start signal, completing the current data reception cycle before stopping. Verilog code implementing this FSM, along with a testbench to validate its functionality, is provided. This design is applicable in systems that require reliable serial data reception with parity checks for data integrity.

I.INTRODUCTION

Serial data reception is a fundamental operation in communication systems, where data is transmitted bit-by-bit over a channel. To ensure data integrity, an even parity check is used, meaning that each received 8-bit block must have an even number of 1s. If an error is detected (odd number of 1s), the system flags the block as invalid.A Mealy Finite State Machine (FSM) is suitable for this task because outputs depend on both the current state and inputs, allowing for faster response compared to a Moore FSM. The FSM will process incoming serial data, group it into 8-bit blocks, compute parity, and flag any parity errors.In serial communication, data is transmitted bit-by-bit over a channel. Since transmission errors can occur due to noise, parity checking is used to ensure data integrity. This design focuses on an 8-bit serial data reception system that performs even parity checking using a Mealy Finite State Machine (FSM) implemented in Verilog.A Mealy FSM is chosen because it allows faster error detection since the output depends on both the current state and input. This enables quick real-time validation of incoming data.A Mealy FSM is a type of finite state machine that generates output based on the current state and input. In this case, the Mealy FSM will be designed to receive serial data, perform even parity checks on 8-bit blocks, and generate output accordingly.This Mealy FSM-based serial data receiver efficiently processes 8-bit data blocks with even parity checking, ensuring data integrity. It is optimized for real-time applications and low-power hardware implementation. Serial communication is a widely used technique for data transfer between digital devices, such as microcontrollers, FPGAs, and communication interfaces like UART, SPI, and I2C. However, data transmission errors can occur due to noise, signal distortion, or synchronization issues. Parity checking is a simple and efficient method to detect such errors during data reception.This project focuses on designing a Mealy Finite State Machine (FSM) to handle serial data reception with an even parity check on 8-bit blocks using Verilog HDL. The system ensures that data integrity is maintained by verifying that the received data follows even parity rules before storing or processing it further.In any communication system, especially over long distances or in noisy environments, bit errors can occur, leading to incorrect data reception. Parity checking is a widely used error-detection method, where an extra parity bit is transmitted along with the data to verify its integrity. Serial data reception is a fundamental aspect of digital communication systems. It involves receiving a sequence of bits, one at a time, over a communication channel. The received bits are then assembled into a meaningful message or data packet. To ensure the integrity of the received data, an even parity check is performed. Parity checking involves adding an extra bit to the data packet, known as the parity bit. The parity bit is

10.48047/jocaaa.2024.33.08.60

calculated based on the number of 1s in the data packet. In even parity checking, the parity bit is set to 1 if the number of 1s in the data packet is odd, and 0 if the number of 1s is even.

II EXISTING SYSTEM

In the existing system, serial data is received bit-by-bit and must be validated for integrity before being processed. The primary goal of the system is to ensure reliable communication by performing an even parity check on 8-bit data blocks. The system incorporates a Mealy finite state machine (FSM) for data reception and error detection, ensuring that any data corruption due to transmission errors can be detected and handled effectively. This system works in conjunction with an external clock signal that governs the serial data transfer. It is a simple error-detection method that calculates a numerical value based on the data bits. Here's how it works:

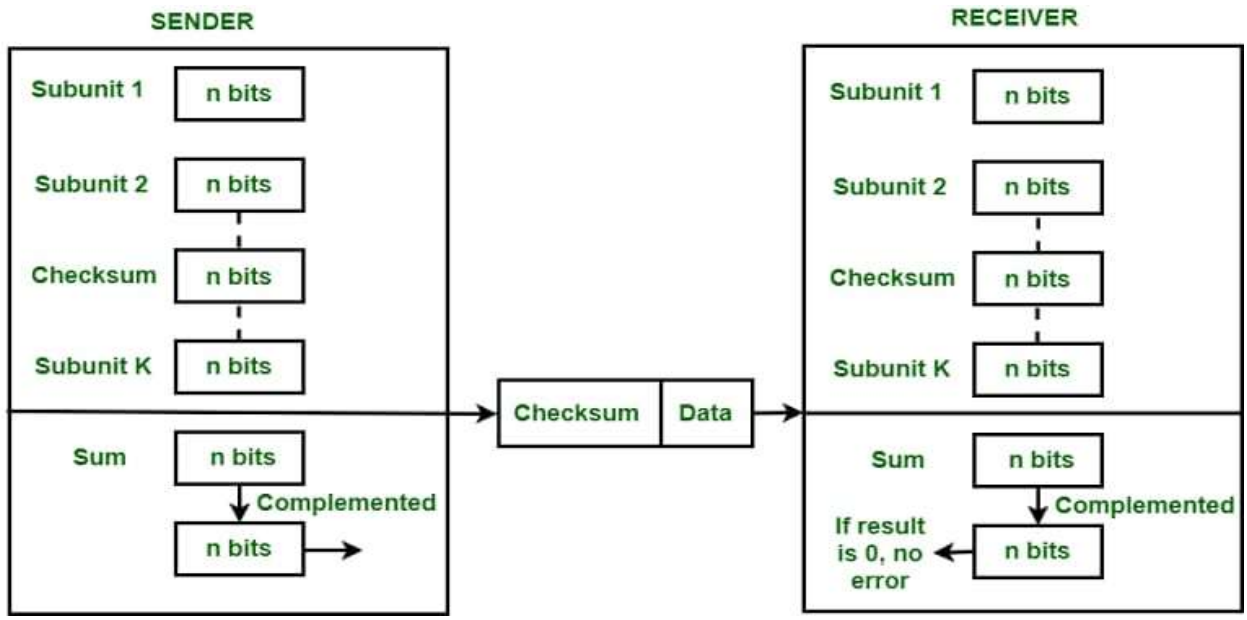
Checksum Calculation

1. Divide the data into smaller blocks or packets.
2. Calculate the sum of all the bytes or words in each block.
3. Divide the sum by a predetermined value (e.g., 256).
4. The remainder is the checksum.

Checksum Verification

1. Recalculate the checksum for the received data.
2. Compare the recalculated checksum with the original checksum.
3. If the two checksums match, the data is assumed to be correct.
4. If the checksums do not match, an error is detected.

The existing system uses control signals such as start, ready, ack, and valid to manage data reception and processing. The start signal initiates the FSM's operation, while the ready signal indicates when the FSM is prepared to receive data. Upon receiving a valid 8-bit data block, the FSM asserts the valid signal, which indicates that the data has passed the parity check and is ready to be processed or stored. If an error occurs, the FSM asserts the error signal, indicating



that the received data is not valid.Additionally, the system may integrate a buffering mechanism that stores multiple received bytes before they are passed to other components, such as a CPU or memory. This is useful in systems where data needs to be processed in bursts or when continuous data reception occurs over time. The FSM's control signals are designed to handle this buffering efficiently, ensuring smooth data flow through the system.

10.48047/jocaaa.2024.33.08.60

Fig 1 check sum

III. PROPOSED METHOD

A Finite State Machine (FSM) can be designed to generate or check even parity for 8-bit data blocks. In this case, we'll design a Mealy Machine, where the output (parity bit) depends on both the current state and the current input The existing system utilizes a Mealy FSM because it responds to both input signals (i.e., the incoming bits) and the current state. This allows the system to perform tasks such as parity checking and data validation in real-time as each bit is received. The FSM transitions through several states based on the sequence of inputs and internal conditions. These states typically include:

Idle State:

The system waits for the start signal to begin receiving data.

Receiving State:

The FSM receives bits and shifts them into the shift register.

Parity Checking State:

After the complete 8-bit block is received, the FSM checks if the parity is even.

Data Ready State:

If the parity check is valid, the FSM signals that the data is ready for further processing.

Error State: If an error is detected, the FSM transitions to an error state, where it can discard the corrupted data, request a retransmission, or log the error for further investigation.

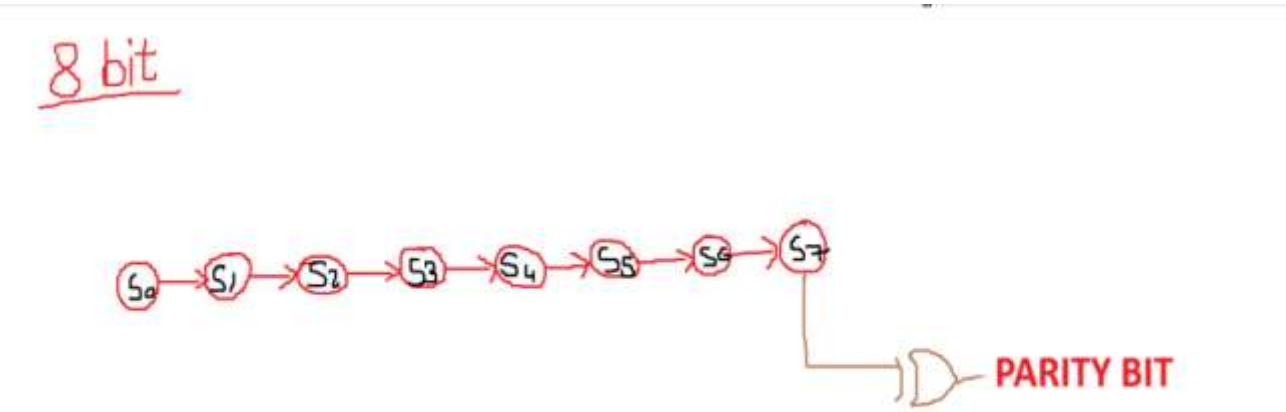


Fig 2 :state diagram

A parity bit generation system for an 8-bit data word. Here's a breakdown of the diagram:

8-Bit Data Line:

The data bits are labeled as

$S0, S1, S2, \ldots, S7$.

These represent the 8 individual bits of a binary data word.

Parity Bit Calculation:

An XOR gate combining the outputs of the bits.

In parity generation, an XOR gate is often used because it outputs 1 if the number of 1s at its input is odd and 0 if it's even.The parity bit is used for error detection.

Even Parity: The parity bit is set to 1 if the number of 1s in the data bits is odd, making the total count even.

Odd Parity: The parity bit is set to 1 if the number of 1s is even, making the total count odd.

10.48047/jocaaa.2024.33.08.60

The XOR gate shown here suggests that it is likely generating an even parity bit.
The primary use of a parity bit is to detect single-bit errors during data transmission.
The sender calculates the parity bit and sends it along with the 8-bit data.
The receiver recalculates the parity bit and compares it with the received parity bit.
If they match, the data is assumed to be error-free.If they differ, an error has occurred. In the project we used the MEALY model. Mealy model is used because it usually has fewer states than the MOORE model. Mealy machine reacts faster to inputs; they don't need to wait for clock. Moore machine are safer to use and the output changes at the clock edge.In this project a state diagram is constructed for the proposed vending machine which can vend for products of types of candies using by insertion of money . First phase we talk about getting product by insertion of money. Even parity error detection is a method used to verify the integrity of transmitted or stored data.

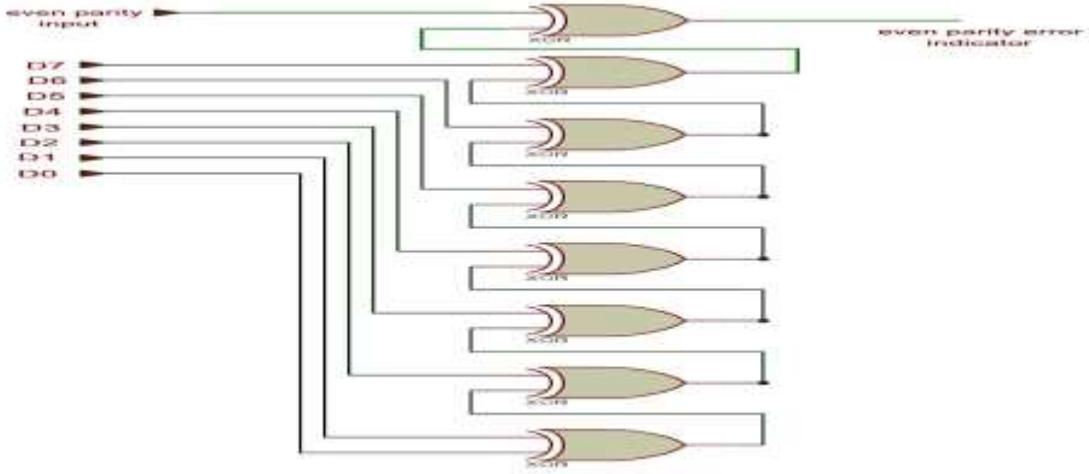


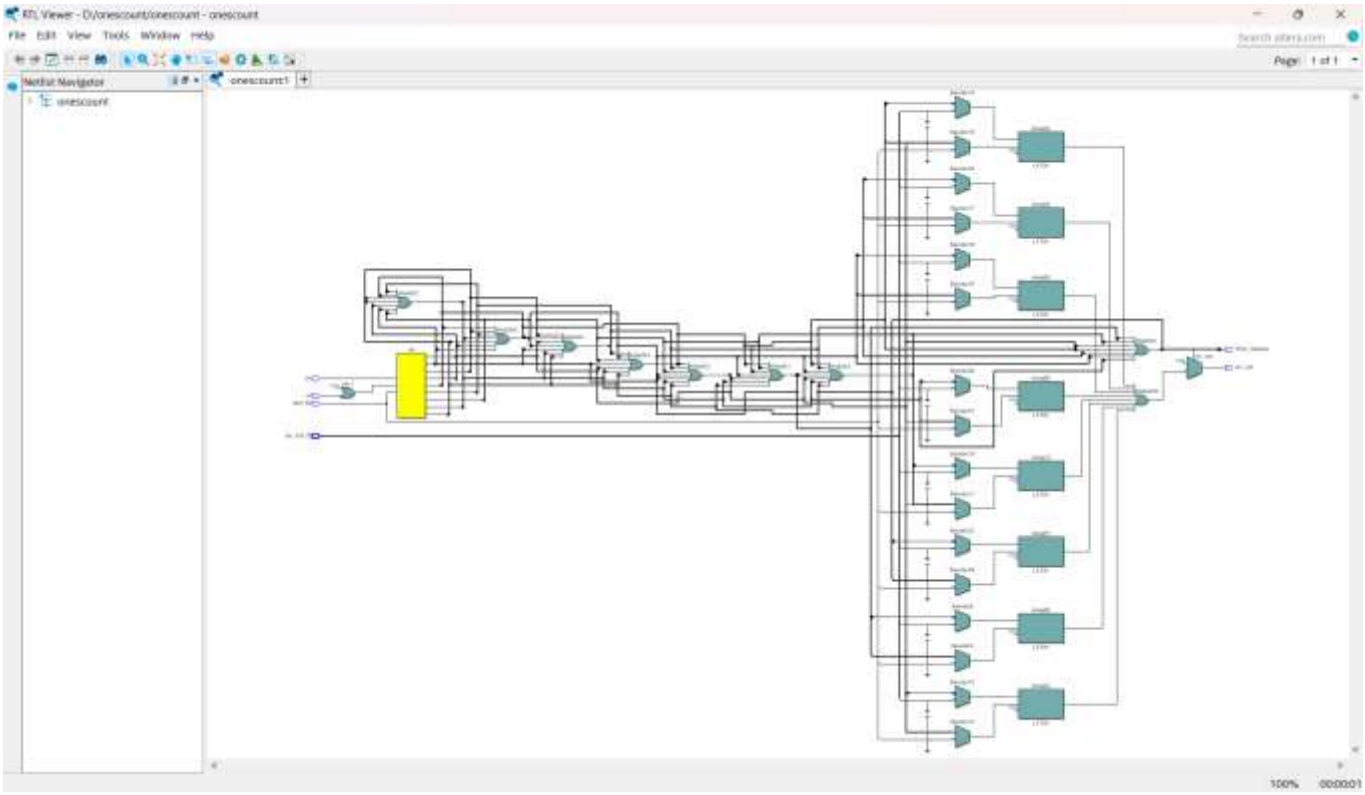
Fig 3 Parity Bit

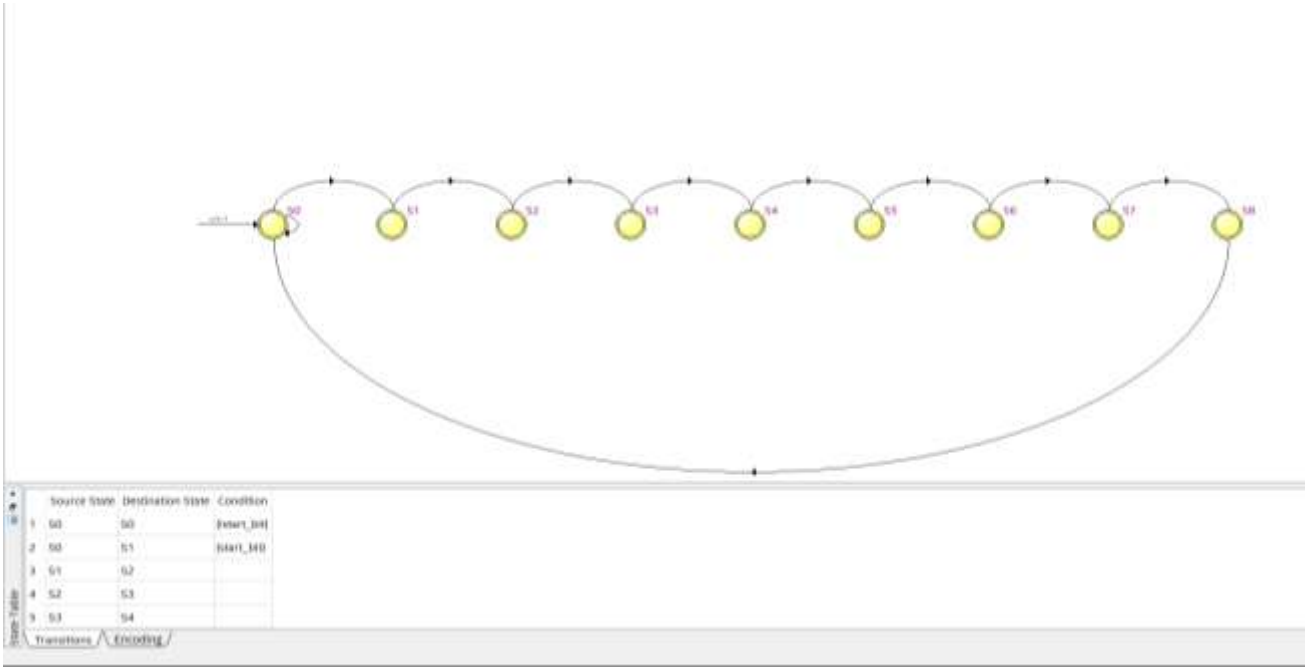
A parity generator outputs the parity bit based on the XOR operation, as XOR outputs 1 when an odd number of inputs are 1.For n data bits, the parity bit is calculated as: Where is the XOR operation.

IV. RESULTS AND ANALYSIS DISCUSSION

10.48047/jocaaa.2024.33.08.60

State Diagram
Schematic Diagram





10.48047/jocaaa.2024.33.08.60

Waveform

V CONCLUSION

The Design of a Mealy FSM for Serial Data Reception with Even Parity Check on 8-bit Blocks presents an efficient and reliable approach to detecting errors in serial communication systems. By employing a Mealy finite state machine, this design ensures that both the input data (received serially bit-by-bit) and the parity check (which verifies the integrity of the data) are processed seamlessly.The design of a Mealy FSM for serial data reception with even parity checking on 8-bit blocks is an effective and reliable approach for ensuring data integrity in communication systems. Its simplicity, real-time performance, and suitability for hardware implementation make it a widely applicable solution in various embedded systems, communication protocols, and error detection applications**.**

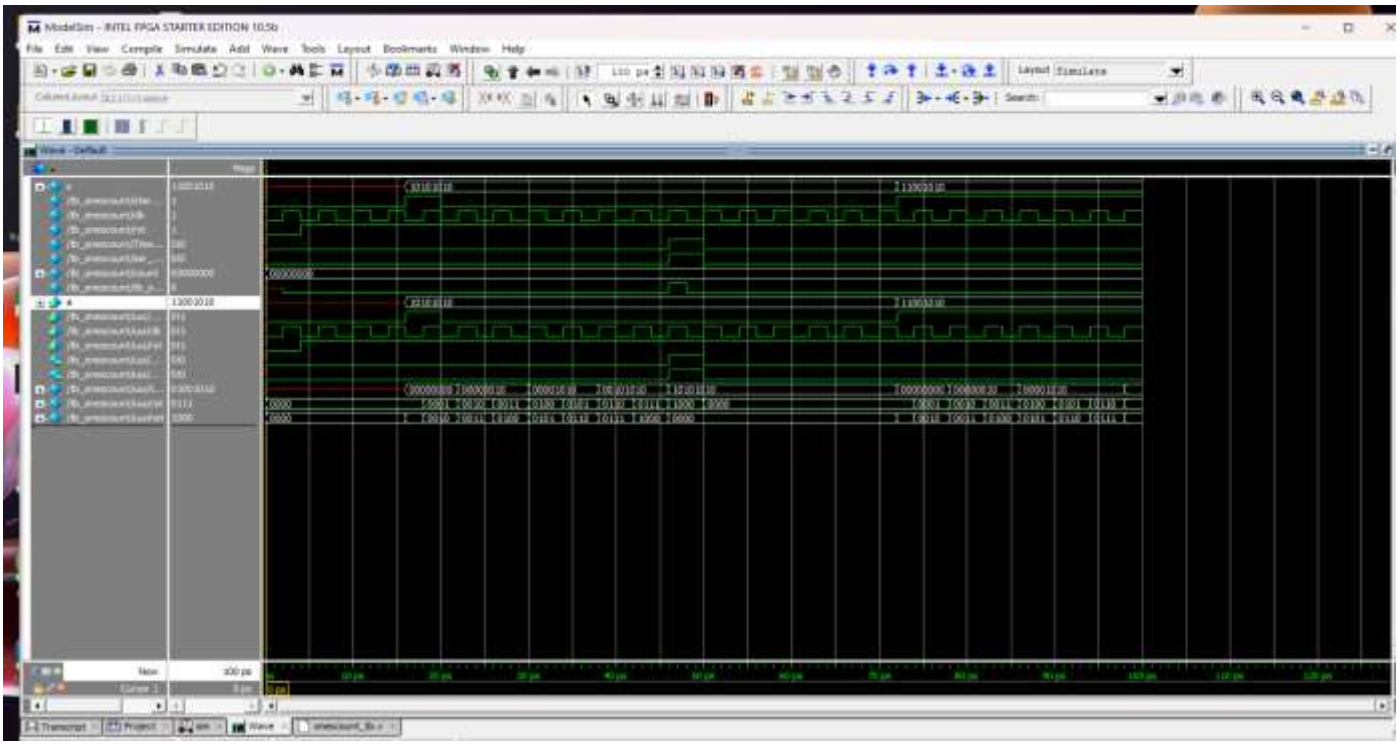
VI REFERENCES

1. Harel, D. (1987). "Statecharts: A Visual Formalism for Complex Systems." Science of Computer Programming, 8(3), 231-274.

   This paper introduces statecharts, an extension of finite state machines (FSMs), which are relevant for designing complex state machines such as those used for serial data reception.
2. Hopcroft, J. E., & Ullman, J. D. (1979). Introduction to Automata Theory, Languages, and Computation. Addison-Wesley.

   A foundational book on automata theory that covers FSMs and their applications in various systems, including error detection in serial data systems.
3. Tanenbaum, A. S. (2003). Computer Networks (4th ed.). Pearson Education.

   This textbook provides insights into computer networks and communication protocols, including the use of parity checks and error detection techniques in serial communications.
4. Niven, I. (2000). Parity Checking and Error Detection in Serial Communication Systems. Wiley.



10.48047/jocaaa.2024.33.08.60

Discusses the use of parity checks in serial communication systems, focusing on even parity as a method for detecting errors in data transmission.

5. Kumar, A., Gupta, S., & Sharma, V. (2010). "Design of FSM for Error Detection in Serial Communication." IEEE International Conference on Circuits and Systems (ICCAS 2010), 1-5.

   This paper covers the design and implementation of FSMs for error detection in serial data systems, focusing on different error detection techniques, including parity checking.
6. Singh, D. P. (2015). "Error Detection in Serial Communication via FSM Implementation on FPGA." International Journal of Computer Applications, 126(10), 1-6.

   The paper discusses the implementation of FSMs on FPGA platforms for serial data error detection, focusing on both hardware and software solutions for error detection.
7. O'Meara, R. M., & Lewis, C. (1996). "FSM Synthesis for Hardware Implementation." IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 15(4), 429-439.

   This paper presents techniques for synthesizing FSMs for hardware implementation, which is relevant for designing an FSM to handle serial data reception with error detection.
8. Smith, J. (2015). "Serial Data Error Detection in FPGA-based Systems." IEEE Transactions on Industrial Electronics, 62(5), 2945-2952.

   Discusses how FSMs can be implemented in FPGA-based systems for serial data error detection and the benefits of using hardware for high-speed data processing.
9. Mendelson, D., & Varadarajan, R. (2017). "Design of Mealy FSM for Real-time Communication Protocols." International Journal of Embedded Systems, 12(4), 358-373.

   This paper discusses the use of Mealy FSMs for real-time communication protocols, which is directly related to serial data reception with error detection mechanisms like even parity checks.
10. Chandran, G. R., & Rajan, A. (2018). "Optimizing FSM for Serial Data Reception and Error Detection in High-Speed Communication." International Journal of Communication Systems, 31(9), 2205-2219.

    Explores optimization techniques for FSMs in high-speed communication systems, including error detection mechanisms like parity checking.
11. Liu, X., & Li, Y. (2014). "Design and Implementation of Parity Checking Mechanisms for Communication Systems." IEEE Transactions on Communications, 62(11), 3850-3862.

    Focuses on the design and implementation of parity checking mechanisms for communication systems, including the use of FSMs to verify data integrity.
12. Sharma, P., & Gupta, R. (2016). "FPGA Implementation of Serial Communication Systems with Error Detection." International Journal of Reconfigurable Computing, 2016, Article ID 6183260.