10.48047/jocaaa.2024.33.08.58

DESIGN OF A MEALY FSM FOR COMPARING TWO 4-BIT UNSIGNED NUMBERS WITH CHECK BIT INPUT

Mr.K.Ch.MALLA REDDY [(1)] , Mr.A.PRASAD [(2)] , PIDATALA HIMAJA [(3),] THIMMISETTI HEMALATHA [(4),] SANDIREDDY LAKSHMI DEVI [(5),] VENNA MOUNIKA [(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. This paper presents the design of a Mealy finite state machine (FSM) for comparing two 4-bit unsigned numbers. The numbers are received serially, with the least significant bit (LSB) arriving first. Additionally, a check-bit input (C) is used to indicate when the last bit of the 4-bit numbers has been received. The FSM compares the two numbers and asserts one of three output signals at the end of the comparison process: GT (greater than), LT (less than), or EQ (equal). The design ensures that the circuit will output the appropriate comparison result once the entire pair of 4-bit numbers has been received, indicated by the check-bit input. The paper includes a Verilog implementation of the FSM, along with a testbench to validate the functionality of the comparison operation

I.INTRODUCTION

Finite State Machines (FSMs) are fundamental components in digital design and are widely used in various applications, including control systems, communication protocols, and data processing. They serve as the backbone of sequential logic, enabling systems to transition between states based on input signals. Among the two primary types of FSMs—Mealy and Moore—the Mealy machine is distinguished by its ability to produce outputs that depend not only on the current state but also on the input signals. This characteristic makes Mealy machines highly efficient and responsive, which is essential in real-time applications.

This project focuses on designing a Mealy FSM (Finite State Machine) specifically tailored to compare two 4-bit unsigned numbers with an additional check bit input. The problem of comparing binary numbers is fundamental in digital systems, and the integration of a check bit adds another layer of complexity and functionality. The objective of this project is to develop a robust, efficient, and optimized FSM that can perform comparison operations accurately and promptly.The comparison of binary numbers is a critical function in numerous digital systems, ranging from arithmetic logic units (ALUs) to complex decision-making circuits. In embedded systems, comparing numbers is often required for data validation, error detection, and control logic. A fast and efficient comparison mechanism ensures better system performance and responsiveness.

The inclusion of a check bit further enhances the capability of the system by adding a verification mechanism, which is crucial in applications requiring error detection or validation of the input data. In communication systems, for instance, data integrity is paramount, and the check bit can significantly improve reliability by allowing the system to detect discrepancies during transmission. Similarly, in control applications, where precise and accurate comparisons dictate system actions, incorporating a check bit provides an additional safeguard against errors.

The choice of a Mealy FSM for this project is motivated by its inherent efficiency in producing quick outputs based on state and input combinations. Compared to Moore FSMs, which generate outputs solely based on states, Mealy machines respond more promptly as their outputs depend on both the state and the input signals. This characteristic aligns well with the real-time requirements of comparison operations, making the Mealy FSM an ideal choice for this application.

10.48047/jocaaa.2024.33.08.58

II.EXISTING SYSTEM

A 4-bit binary number can represent values ranging from 0 to 15. The comparison between two such numbers, denoted as A and B, involves determining whether A is greater than B (A > B), equal to B (A = B), or less than B (A < B). The 4-bit comparator performs these comparisons simultaneously and provides three distinct output signals that indicate the result of the comparison.

To achieve accurate comparison results, the comparator circuit utilizes various logic gates, including AND, OR, NOT, and XNOR gates. These gates are arranged in a way that the most significant bit (MSB) has the highest priority in decision-making. This means that if the MSB pair (A3, B3) are different, the comparator decides the output based solely on these bits without considering the lower bits.

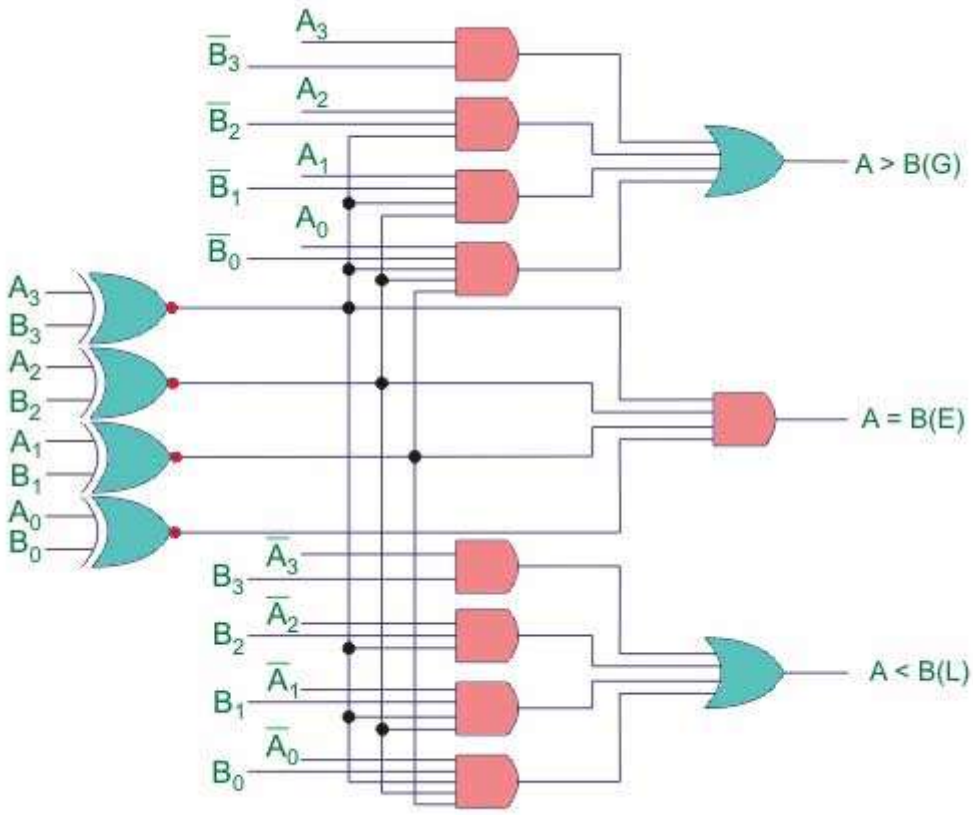

Fig: 1. 4 bit comparator

Structure of the 4-Bit Magnitude Comparator

The structure of the 4-bit magnitude comparator can be divided into three main sections, corresponding to the three output conditions:

1. **A > B (Greater Than)**
2. **A = B (Equal To)**
3. **A < B (Less Than)**

Greater Than (A > B)

The greater-than logic is designed to evaluate whether A is larger than B, starting from the most significant bit pair (A3, B3) and progressing toward the least significant bit pair (A0, B0). If A3 = 1 and B3 = 0, the output (A > B) is asserted immediately, regardless of the lower bits. If both bits are equal, the comparator proceeds to check the next

10.48047/jocaaa.2024.33.08.58

lower bit pair (A2, B2), and so on. The logic follows the principle of priority encoding, where the highest bit difference takes precedence.

The circuit utilizes a combination of AND and OR gates to achieve this comparison. The AND gates check individual bit combinations, while OR gates aggregate the results to produce the final greater-than output. The use of NOT gates enables the negation of specific bits where necessary to establish the comparison.

Equal To (A = B)

The equal-to condition is achieved through the use of XNOR gates. Each pair of corresponding bits (Ai, Bi) is compared using an XNOR gate, which outputs a high (1) only if both bits are equal. The outputs of all XNOR gates are then combined using an AND gate to determine if all bits match. Only if all pairs are equal does the AND gate produce a high output for the equality condition.

This segment of the comparator ensures that even a single mismatch in any bit pair results in a low (0) output, indicating inequality. The use of XNOR gates makes the equality check simple and efficient, as it directly outputs the equivalence of each bit pair.

Less Than (A < B)

The less-than condition follows a similar logic to the greater-than condition but works in the opposite direction. If A3 = 0 and B3 = 1, the less-than output (A < B) is asserted instantly. If the most significant bits are equal, the comparator continues checking the next lower bit pair (A2, B2). The AND-OR logic structure employed here ensures that the most significant unequal bit dictates the final outcome.

Similar to the greater-than logic, the less-than section employs AND, OR, and NOT gates in a configuration that prioritizes higher bit comparisons. The circuit maintains symmetry between the greater-than and less-than logic structures to simplify design and maintain consistency.

III PROPOSED SYSTEM

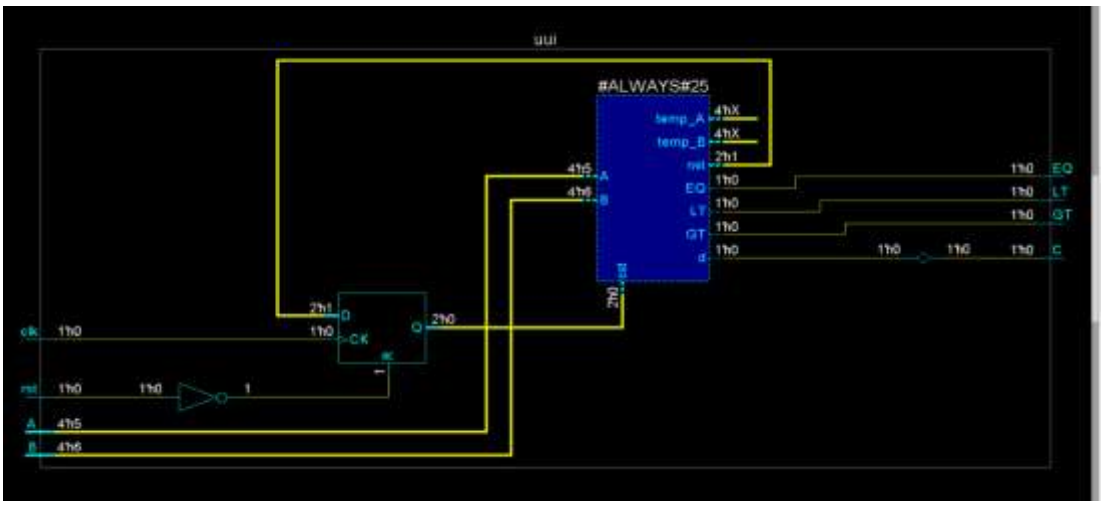

Fig:2.internal architecture of 4 bit comparator

A digital monitoring and comparison system designed to evaluate two input values, referred to as "A" and "B." These inputs represent numerical data or parameters whose relative magnitude needs to be determined. The system is structured to operate sequentially and efficiently through a series of functional blocks, which process the data step by step to generate comparative results.

At the top level, two distinct data sources feed the input values "A" and "B" into the system. These values are initially processed through a "Check bit input" block labeled as "C," which ensures the validity and integrity of the input data. This validation step is crucial for maintaining reliable comparisons, as it discards or flags invalid data. Upon receiving a valid 4-bit input, the system progresses to the next phase.

10.48047/jocaaa.2024.33.08.58

The validated data then enters the "Finite State Machine" (FSM) block, which serves as the control logic of the system. The FSM governs the sequential operation, coordinating the transition between states based on the input signals and control conditions. This systematic control ensures that the data flows in a structured manner through the subsequent stages of the system.

After the FSM processing, the data proceeds to the "Comparator" block. This unit performs the fundamental task of comparing the magnitudes of "A" and "B" and determining their relationship. The output of the comparator can indicate one of three possible states: equality (A = B), greater-than (A > B), or less-than (A < B). The ability to precisely determine these states is essential for applications that require decision-making or classification based on data comparison.

Once the comparison is made, the output from the comparator is forwarded to the "Monitor" block. The monitor collects and organizes the comparative data to provide meaningful results, which are then categorized into one of three output states. These states are labeled as "Equal" (indicating that A equals B), "Greater" (indicating that A is greater than B), and "Lesser" (indicating that A is less than B). Each output is clearly labeled and directed to indicate the nature of the comparison outcome.

The structured layout of the system ensures that the entire process, from input validation to final output, is streamlined and logically organized. The modular design allows for scalability and integration into larger systems where data comparison and decision-making are necessary. The system's accuracy and reliability are maintained through proper synchronization and control flow managed by the FSM. Additionally, the clear distinction between validation, comparison, and monitoring phases ensures robustness and precision in operations.

The entire design showcases a systematic approach to digital comparison and monitoring, emphasizing accuracy and modularity. Its application could range from simple comparison tasks in digital circuits to more complex data evaluation processes in embedded systems. The well-organized structure and precise control mechanisms contribute to its effectiveness and reliability, making it a versatile solution for applications requiring efficient data comparison and monitoring.

IV. RESULTS AND ANALYSIS DISCUSSION

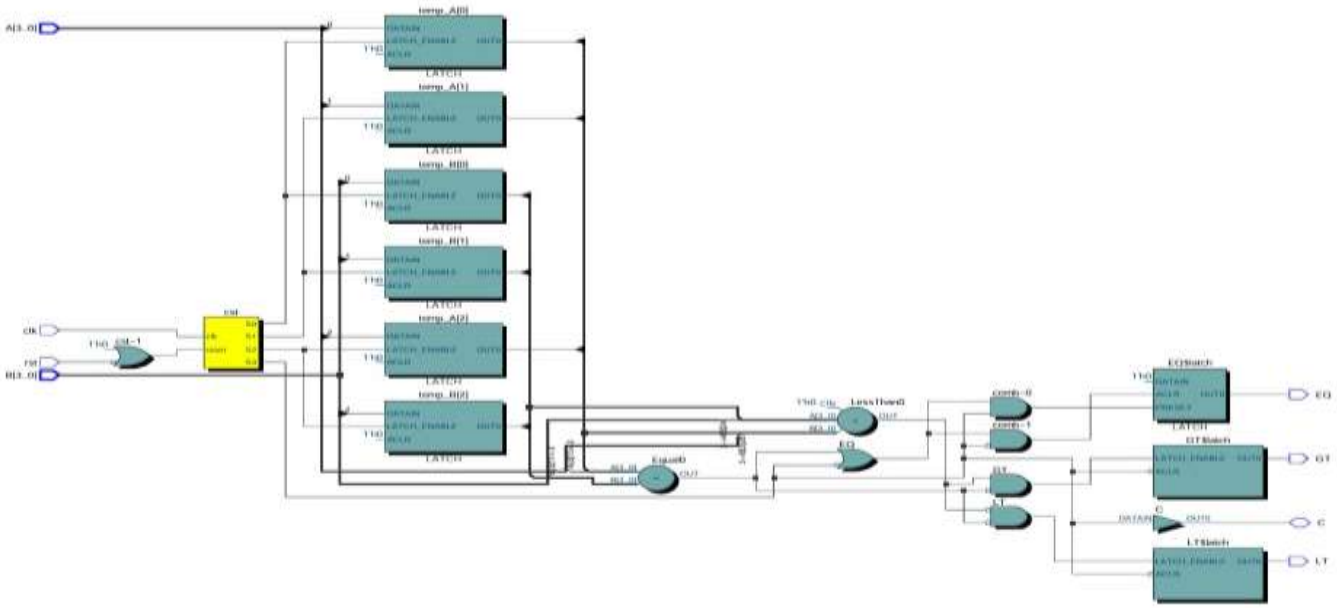

fig:3.schematic for 4 bit unsigned comparison with checkbit

10.48047/jocaaa.2024.33.08.58

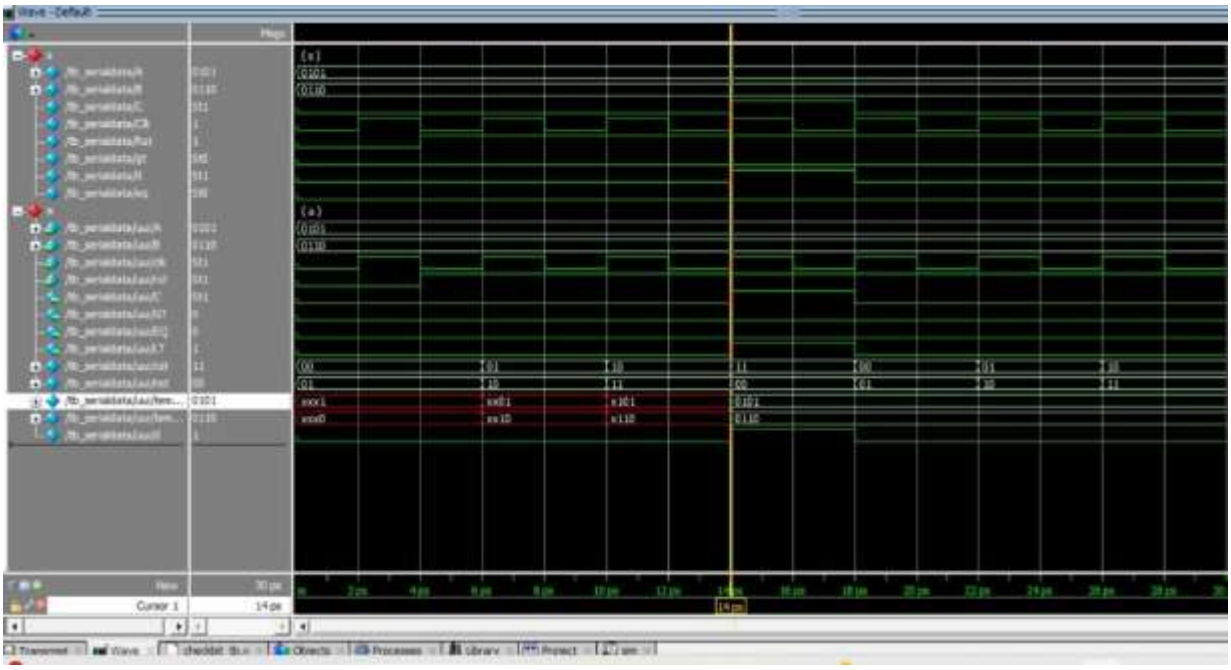

Fig:4. Waveform with checkbit

V CONCLUSION

The design and implementation of a Mealy Finite State Machine (FSM) for comparing two 4-bit unsigned numbers with an additional check bit input have been successfully accomplished. This project demonstrates the effectiveness of using a Mealy FSM for real-time comparison tasks, especially in applications requiring rapid response and error detection, the project successfully achieved the goal of designing a compact and optimized Mealy FSM for comparing two 4-bit unsigned numbers with a check-bit input. The system's robustness and efficiency make it ideal for digital applications requiring quick comparisons and error detection. Future improvements may include extending the FSM for larger bit comparisons or integrating additional error-correction mechanisms to further enhance system reliability.

VI REFERENCES

1. Digital Design" by Morris Mano and Michael Ciletti*
2. "Logic and Computer Design Fundamentals" by Charles H. Roth Jr. and Larry L. Kinney*
3. "Finite State Machines in Hardware: Theory and Design" by Volnei A. Pedroni*
4. *"Fundamentals of Digital Logic with Verilog Design" by Stephen Brown and Zvonko Vranesic*
5. *"Digital Systems: Principles and Applications" by Ronald J. Tocci, Neal S. Widmer, and Gregory L. Moss*

10.48047/jocaaa.2024.33.08.58

6. *"Verilog HDL: A Guide to Digital Design and Synthesis" by Samir Palnitkar*
7. *"VHDL: Programming by Example" by Douglas L. Perry*
8. *"Synthesis and Optimization of Digital Circuits" by Giovanni De Micheli*
9. *"Introduction to Logic Synthesis using Verilog HDL" by Robert B. Reese and Mitchell A. Thornton*
10."Digital Design with RTL Design, VHDL, and Verilog" by Frank Vahid*
11.Advanced Digital Design with the Verilog HDL" by Michael D. Ciletti*