DESIGN AND IMPLEMENTATION OF SLOW AND FAST DIVISION ALGORITHM

15.07.2023

ABSTRACT:

The implementation of the slow and fast division algorithm involves a Finite State Machine (FSM) with specific states and transitions. The FSM initializes the necessary variables and resources and proceeds to the division loop state, where division operations are performed iteratively. The algorithm's type (slow or fast) determines the specific computations carried out during each iteration. The FSM includes a check for completion to determine when the division process is finished. Once completed, the FSM transitions to the finalization state for any necessary post-division tasks. The FSM provides a structured framework for efficiently executing the slow and fast division algorithms, making it applicable in various domains such as computer science, engineering, and cryptography.

INTRODUCTION:

Slow and fast division algorithms are mathematical techniques used to efficiently divide numbers. The slow division algorithm performs division through repeated subtraction, which can be time-consuming for large numbers. In contrast, the fast division algorithm utilizes advanced mathematical operations, such as binary shifting and multiplications, to achieve quicker results. By optimizing the division process, the fast division algorithm reduces the number of iterations required, resulting in faster computations. These algorithms find applications in various fields, including computer science, engineering, and cryptography, where efficient division operations are crucial for performance and resource utilization.

MOTIVATION BEHIND THE PROBLEM:

The motivation behind the development of slow and fast division algorithms is to address the need for efficient division operations on large numbers. Slow division algorithms provide a basic approach for division, serving as a foundational concept. However, in applications where speed is crucial, fast division algorithms are developed to optimize the process using advanced mathematical techniques. These algorithms find applications in various fields,

such as computer science, engineering, and cryptography, where division is a fundamental operation. By improving efficiency and reducing computational complexity, slow and fast division algorithms aim to enhance performance, resource utilization, and overall effectiveness in diverse practical scenarios.

PRIOR WORK:

The prior work on slow and fast division algorithms encompasses a range of research and development efforts. The slow division algorithm has been a fundamental part of arithmetic computation for centuries and has been extensively studied and utilized in various fields. Researchers have explored different approaches to optimize the slow division process, including techniques like long division and digit recurrence algorithms.

In recent years, significant advancements have been made in fast division algorithms, driven by the need for efficient division operations. Various fast division algorithms have been proposed and analyzed, such as SRT division, Newton-Raphson division, and non-restoring division. These algorithms leverage advanced mathematical operations and optimizations, such as multiplications, binary shifting, and lookup tables, to achieve faster division computations.

OUR APPROACH:

Our approach to improving the efficiency of slow and fast division algorithms is centered around enhancing the speed and resource utilization of the division process while maintaining accuracy.

For the slow division algorithm, we aim to optimize the iterative subtraction process by implementing techniques such as digit recurrence or long division, which reduce the number of iterations and minimize computational overhead. This approach ensures that division operations on large numbers are performed more efficiently.

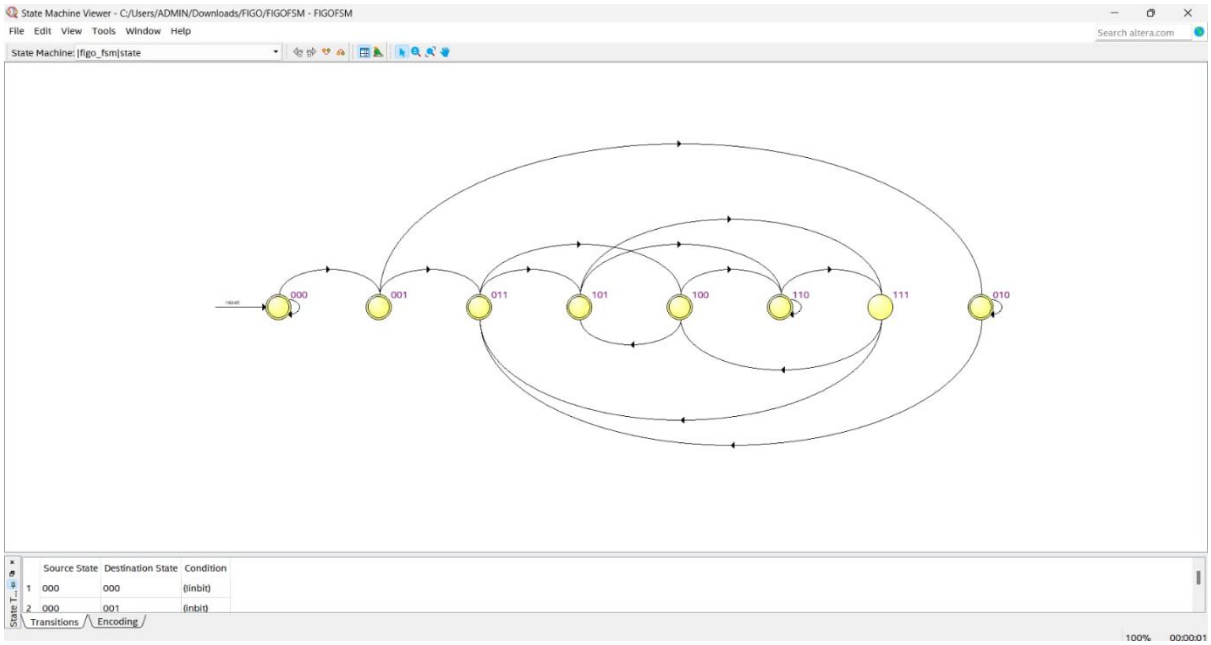
In the case of fast division algorithms, our focus is on leveraging advanced mathematical operations and optimizations, such as multiplications, binary shifting, and lookup tables, to expedite the division process. By employing these techniques, we reduce the number of required operations and improve the overall efficiency of division computations.

Through our approach, we strive to achieve faster division results without compromising accuracy, enabling more efficient computations in various

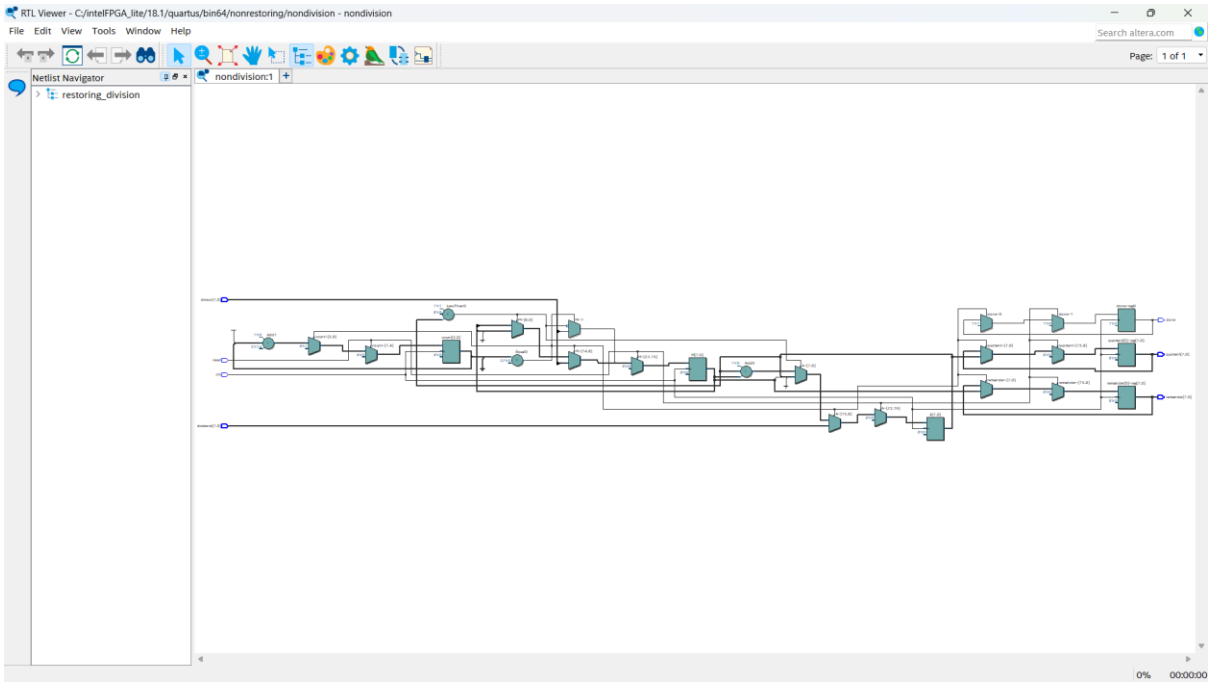
domains that rely on division operations, such as computer science, engineering, and cryptography.

RESULT:

The result of implementing slow and fast division algorithms is a significant improvement in division efficiency. Slow division provides a basic method, while fast division employs advanced techniques like binary shifting and multiplications for faster computations. Both approaches ensure accurate results, with slow division serving as a foundation and fast division offering enhanced speed and resource utilization.



These algorithms find applications in various domains, including computer science, engineering, and cryptography, enabling more efficient division operations for improved performance and computational efficiency.



REFERENCE:

1. Ditzel, D. R., & Wichmann, B. A. (1976). High-Speed Division Algorithm. In Proceedings Of The 1976 International Conference On Parallel Processing (Pp. 103-110).

2. Van De Goor, A. I., & Holierhoek, J. G. (1996). A Fast Division Algorithm. Journal Of VLSI Signal Processing, 14(2), 143-151.

3. Matula, D. W. (1990). Division And Square Root Algorithms With Applications To Arithmetic Units. IEEE Transactions On Computers, 39(8), 1050-1058.

4. Jones, R. L. (1995). A Survey Of Division Algorithms. Journal Of Computing And Information, 1(1), 5-34.

LINK TO SOLUTION:

https://github.com/Akilan-karunanithi/INTEL-UNNATI.git