Dean&Francis

ISSN 2959-6157

Comparative Research of Mainstream High-Performance Dividers

Yingjie Hua[1],

Xianshu Luo[2]

and Haokun Yang[3,*]

[1]School of Integrated Circuit Engineering, University of Electronic Science and Technology of China, Chengdu, China
[2]School of Mechanical and Electrical Engineering, Chengdu University of Technology, Chengdu, China
[3]School of Electrical Engineering & Electronics, University of Liverpool, Liverpool, United Kingdom

*Corresponding author: sghyan20@liverpool.ac.uk

Abstract:

The implementation of modern electronic systems relies on mathematical operations to solve problems. Of the three most basic arithmetic operation algorithms, the implementation of division has received less attention and has a higher level of difficulty. Delays in the division algorithm will lead to a reduction in overall arithmetic efficiency. Therefore, it is imperative to study and summarise high-performance dividers. In this paper, based on the existing research, five high-performance divider algorithms, namely the Pencil-and-Paper Method, Restoring Division, Non-Restoring Division, Newton-Raphson Method and Pipelined Method, have been summarised. Researchers have comparatively analysed the principles of each division algorithm, discussed the range of application of each algorithm and given an analysis of their respective advantages and disadvantages. As result, the simulation of the Restoring Division, Non-Restoring Division and Pipelined Method have been implemented. Based on the simulation results, the researchers concluded that different division algorithms need to be used in different application scenarios and discussed their specific application scenarios and comparisons between the algorithms.

Keywords: Non-Restoring division; restoring division; Newton-Raphson method; divider; vivado

1. Introduction

Since John Mauchly and J. Presper Eckert invented the world's first general-purpose electronic calculator, the ENIAC, in 1946, computer technology has advanced by leaps and bounds [1]. However, computers still process and manipulate data through mathematical operations, and almost all information processing and computational tasks can be translated into mathematical problems. Of the three most basic arithmetic operation algorithms, the fact that the division operation takes much longer than multiplication and addition, as well as the fact that division is a relatively less common operation, results in a challenging implementation of the divider algorithm [2][3][4]. However, according to S.F. Oberman and M.J. Flynn, delays in the divider can lead to lags in the overall computation time, and the implementation of new

high-performance divider algorithms can greatly improve system performance [2]. Due to the growing development of artificial intelligence and autonomous driving, the need for arithmetic power has been greatly increased, making the development of new high-performance dividers a necessity.

It is generally accepted that divisors can be classified into five main categories, digit recurrence, functional iteration, very high radix, table look-up, and variable latency [2]. The SRT algorithm is one of the more classical and common digit recurrence algorithms [3][5]. Due to the large number of parameters involved in the design of the divider, the implementation of the divider varies for different requirements [2][6]. In this paper, researchers will analyse and compare the recent algorithms for dividers in terms of computational efficiency, energy consumption, and design area. In Chapter 2, the three classical Digit Recurrence Algorithms, Restoring Division, Non-Restoring Division, and Pencil-and-Paper Method, are highlighted and their respective advantages and disadvantages are analysed. In Chapter 3, researchers have summarised and generalised the high-performance divider algorithm designs such as Newton-Raphson Iteration and Pipelined Method. This study compares and analyses the performance differences by simulating the existing mainstream divider designs. While summarising the high-performance divider algorithms, researchers also explored the future development direction of the high-performance divider, including further optimising the parallelism of the algorithms and their application in heterogeneous platforms, etc., aiming to provide an overview and reference for the research and development of high-performance divider.

2. Analysis of Three Classical Digit Recurrence Algorithms

2.1 Pencil-and-Paper Method

The Pencil-and-Paper Method is an algorithm based on the process of manual written division, which gradually corrects the error and approximates the result by estimating the value of the quotient bit by bit, thus it is a very classical Digit Recurrence Method. The principle is to start from the highest digit of the divisor, judge each bit of the quotient approach it step by step, and update the remainder by subtraction. If the remainder is greater than the divisor, the quotient of that bit is one and the operation of subtracting the remainder from the divisor is performed, and then the divisor is shifted to the right by one, if the remainder is less than the divisor, the quotient of that bit is zero, the remainder remains unchanged and the divisor is shifted to the right by one. This is repeated to get the desired quotient and remainder. The sign bit of the quotient is handled separately, and the result is obtained by the XOR operation. This algorithm is robust, but because it takes twice the word length to compute, it is usually more time-consuming, especially in high-precision computations or hardware implementations, as it requires corrections for each computation. However, this method can be parallelised to improve its computational power due to its simple structure. In a sense, this process, based on manual written division, was a precursor to restoring division.

2.2 Restoring Division

Restoring Division is also known as Standard Long Division [3]. It is similar to the Pencil-and-Paper Method in that it constantly subtracts the divisor and adjusts the digits in the quotient and the restores according to the positive or negative result of the subtraction to find the final quotient and the remainder step by step. In this algorithm, both the dividend and the divisor are operated as complements. In a single digit, the dividend is subtracted from the divisor and the result is determined to be positive or negative, if the result is positive, the quotient is one in that digit and the result is shifted to the left, if the result is negative, the quotient is zero in that digit and the remainder is restored by adding the divisor again. By repeating the above steps until the quotient has the same number of digits as the dividend and the divisor, and the remainder is obtained by multiplying the result with 2^{-x} step-by-step, where x is the number of left shifts and the quotient consists of a combination of the quotient values of each digit, as shown in the flowchart in Fig. 1. In this figure, f is the recovery flag bit, n is the number of bits in dividend and divisor. q is the quotient value, m is the remainder, i is the counting variable and Cn is the sign bit.

ISSN 2959-6157

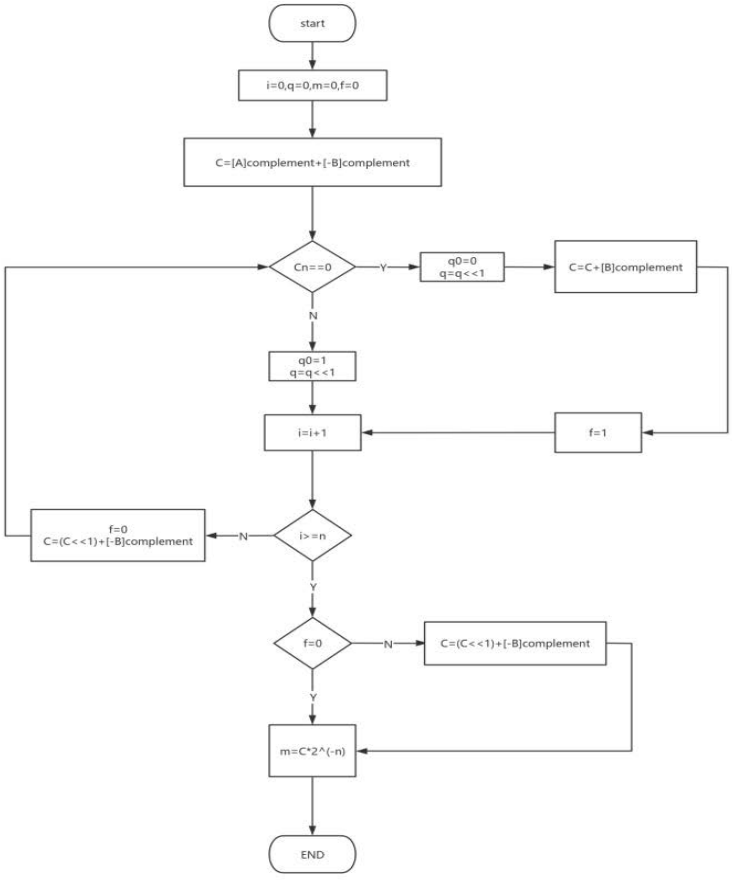

Fig. 1 Restoring division algorithm flowchart

Restoring Division is different from the Pencil-and-Paper Method in that it shifts the remainder left each time instead of the divisor, so that the number of bits in the division is the same as the divisor and the remainder, and the efficiency of the calculation will be improved. However, in the calculation process, if the sign bit is one, and at this time the remainder is negative, there will be added to the operation of the divisor to restore the remainder of the operation. This step of the operation greatly reduces the efficiency of the calculation, so there is the introduction of the Non-Restoring Division to solve the problem.

2.3 Non-Restoring Division

The Non-Restoring Division continually adds and subtracts dividends through the relationship between the divisor and the remainder, and determines the quotient based on the result of the operation, while allowing the remainder to be temporarily negative during the process, but then the remainder will be added in subsequent steps through the addition operation [7]. But in subsequent steps, the remainder will be naturally restored to a positive number through an addition operation. A key difference between the Non-Restoring Division and Restoring Division is that at the time of arithmetic for each digit. This algorithm does not use restoring remainder if the result of dividend minus divisor is negative. Fig. 2 illustrates the flowchart for this algorithm. where n is the number of bits in the dividend and divisor, q is the quotient, m is the remainder, i is the counting variable, and Cn is the sign bit, which performs the operation of dividing A by B.

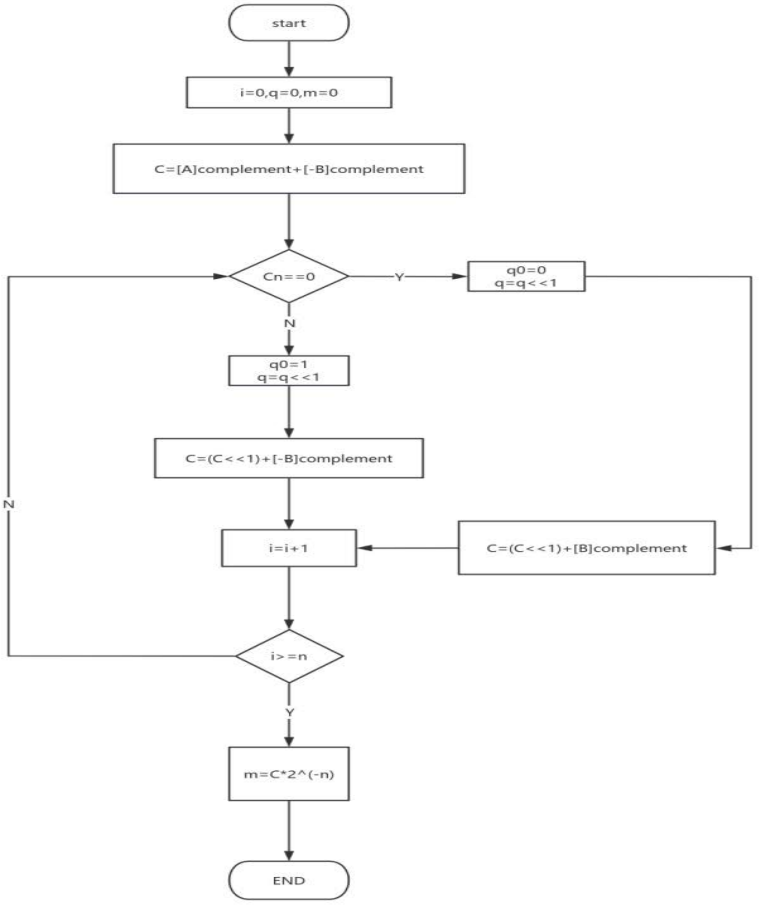

Fig. 2 Non-Restoring division algorithm flowchart

Unlike Restoring Division, Non-Restoring Division does not need to recover the negative remainder, but directly performs the left shift and complement operations, using direct addition and subtraction instead of simple subtraction, which greatly improves the arithmetic efficiency, and the steps are fixed in all cases, so this division algorithm is widely applicable. However, Non-Restoring Division must perform a subtraction operation followed by a positive or negative judgment on the remainder to output the quotient value. So, a redundant clock delay is added to the algorithm. In principle, the dividend of n-bit width requires n clocks for the shift operation to produce the result, while the Non-Restoring Division requires n+1 clocks to complete the operation. For a large number of bits of the operation, it will take a lot of time, so to address this shortcoming can be used in a number of pre-processing methods. For example, before the shift operation can be compared with the divisor to determine the value of the quotient so that the result is one clock ahead of the non-recoverable remainder method [8]. In addition, some nonlinear algorithms can be used to greatly improve the efficiency of the operation.

3. High-Performance Division Algorithms and Optimizations

3.1 Newton-Raphson Method

Based on the three Digit Recurrence Algorithms in the Chapter 2, they all perform linear operations by adding and subtracting. The efficiency of these Digit Recurrence Algorithms does not seem to be very high if some division operations with a very high number of digits are carried out. Thus, the Newton-Raphson Method can be used to optimise and improve the performance of the division algorithm.

In principle, the Newton-Raphson Method, as seen in Equation (1), rewrites the divisor into its inverse form, changes the division to multiplying by the inverse of the divisor, and then iterates the formula to converge to the true quotient and consequently improves the computational efficiency.

$$Y = \frac{A}{D} = A\frac{1}{D} \tag{1}$$

Equation (2) represents the calculation of the inverse

ISSN 2959-6157

value, which can be calculated by the iterative method of calculating the root sign of an equation. Thus, the Newton-Raphson Method has the form in Equation (3)

$$F(X) = \frac{1}{X} - D \tag{2}$$

$$X_{i+1} = X_i - \frac{F(X_i)}{F'(X_i)} = X_i - \frac{\frac{1}{X_i} - D}{\frac{1}{X_i^2}} = X_i(2 - X_i D) \tag{3}$$

The basic idea of the Newton-Raphson Method is to approximate a function using the first few terms of the Taylor series of the function at a point, and to approximate the roots of the original equation by solving for the roots of this approximated function. Specifically, if r is a root of the equation f(x) = 0, choose x0 as the initial approximation of r, and then make a tangent to the function y = f(x) at the point (x0, f(x0)), and the transverse coordinates of the intersection of this tangent and the x-axis are the first approximation of r. Repeating this process results in a sequence of approximations of r. Each iteration brings the approximation closer to the true root. After many iterations, the equation will converge to 1/D, and important for the efficient operation of the algorithm is the choice of the initial value X0, which determines the characteristics of the convergence. The value of D is scaled so that it is in the range 0.5 < D < 1, in which case the algorithm converges as fast as possible: to scale the value of D to a given range, it is sufficient to use a shift operation, both left and right [9]. So, on the basis of this method, it is possible to design a more efficient division algorithm, which has the advantage of high convergence and allows the handling of large numbers of digits; the disadvantages are also obvious: first of all, since it is the final result of the convergence, the accuracy will be affected by the number of iterations. Secondly, if you need to calculate the 64-bit division, in addition to the initialisation time, it takes at least 65 clock cycles to complete the calculation, and if an addition operation takes more than one clock cycle, the calculation time will be doubled, which is not suitable for real-time scenarios with high requirements [10]. At the same time the performance can be further improved by combining it with the pipelining idea, which will be able to process the data continuously.

3.2 Pipelined Method

Pipelined Method controls the execution of stages in a pipeline by using a state machine. In contrast to methods that compute each part of the function separately, the researchers implemented the pipelined process by re-splicing the 1bit data corresponding to the bits of the single-step computation of the remainder and the original dividend as a new single-step dividend input to the next level of the single-step division computation unit.

For single-step division calculations, the single-step dividend bit width needs to be 1bit more than the original divisor bit width in order not to overflow. For pipelining purposes, registers are required at the output to store the original divisor and dividend information. The result of the single-step operation is the new 1-bit quotient data and remainder, and the new 1-bit quotient data needs to be shifted and summed with the quotient result of the previous cycle in order to get the final division result. The remainder of the single-step calculation and the 1bit data of the corresponding bits of the original dividend are respliced and input to the next level of single-step division calculation unit as the new single-step divided number. The information of dividend, divisor and quotient is also passed to the next level.

4. Simulation Results

In order to verify the rationality and operational feasibility of the above algorithms, this paper used the software Vivado 2019.2 to write corresponding programs for the Restoring Division, Non-Restoring Division and Pipelined Method. Based on each of the principles, researchers wrote the corresponding testbench to derive the simulation images.

The Restoring Division counterpart defines a total of eight ports, namely clk, rst, start, divide, divisor, quotient, remainder, finish_s, where clk is the clock signal, the low effective rst is responsible for controlling the reset, and the start signal controls whether the divider the start signal controls whether or not the divider performs an operation. Dividend, divisor, quotient and remainder represent the dividend, divisor, quotient and remainder of the division operation. Finally finish_s is used to judge whether the divider completes the operation or not, if it does, then finish_s is set to 1, and the result of quotient and remainder is updated, and the end signal, quotient and remainder are cleared in the next clock cycle. To simulate Fig 3 as an example, the results shown in the figure for one of the three groups of data in the test, three groups of data dividend and divisor corresponds to 9 and 4, 15 and 3, 14 and 4, respectively, the resulting quotient and remainder corresponds to quotient and remainder, the results of the 2 and 1, 5 and 0, 3 and 2. The results are correct, proving that the Restoring The result is correct, which proves that the principle of Restoring Division is reasonable and feasible in practice. However, it is worth noting that Restoring Division from the data input to the results of the beginning of the operation to the output delay is large, in order to input 15 divided by 3, for example, from the input data

and start to the finish signal reversal through a total of about 11 clock cycles. Therefore, although the structure of Restoring Division divider is simpler than that of Non-Restoring Division divider, the overall delay of the system is larger due to the fact that Restoring Division divider needs to recover the remainder at each step of the arithmetic process, and there are more iterative steps.

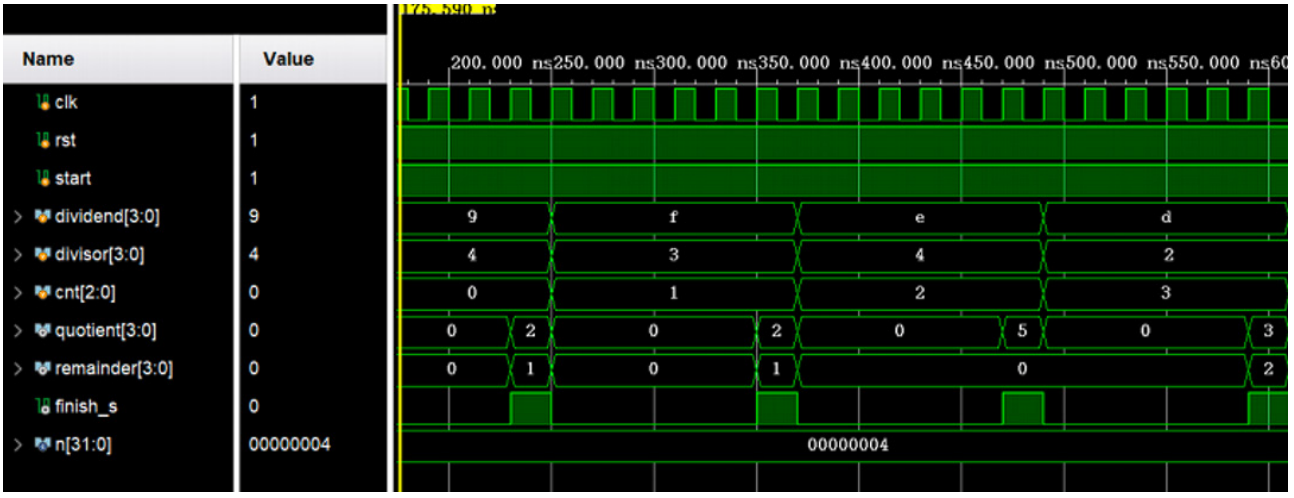

Fig. 3 Simulation results of the restoring division

The interface definition of Non-Restoring Division is exactly the same as that of Restoring Remainder. Unlike the Restoring Division test procedure, in order to make the test results more intuitive, the test procedure adds a corresponding delay processing for the inputs, so that the output results correspond to the outputs one by one, and therefore the output results will not be updated in the next round of arithmetic operations as in the Restoring Division, but rather will be output in sync with the results of the current round of arithmetic operations. Take the simulation Fig 4 as an example, in the figure dividend and divider's inputs are 12 and 5, 9 and 4, 10 and 3 respectively, the output quotient and remainder are 2 and 2, 2 and 1, 3 and 1. The result is correct, which proves that the principle of the method of unrecoverable remainder is reasonable and feasible for practical operation.

It is worth noting that in the test Non-Restoring Division divider, the system clock and the test Restoring Division consistent, in the operation of the Non-Restoring Division divider, its operation speed is faster, from start flip to 1 to finish In the Non-Restoring Division divider operation, its operation speed is faster, from start flip-flop to 1 to finish after only 5 clock cycles, the latency is about half of the Restoring Division, and this gap will be further magnified in the order of magnitude of time in the division operation of higher bit widths. The low latency at the same time Non-Restoring Division divider requires more logic units and complex control mechanisms, compared to Restoring Division divider hardware resource consumption is higher, in the actual design of digital circuits also need to make more trade-offs.

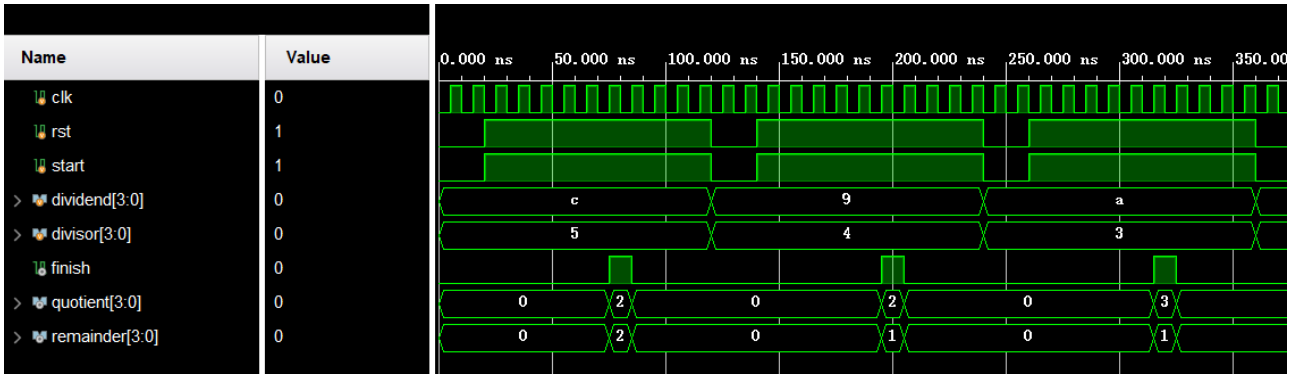

Fig. 4 Simulation results of the Non-Restoring division

From the simulation Fig 5, it can be seen that divide_ref and divisor_ref are the delay of input divide and divisor, corresponding to the output result merchant and remainder, i.e. quotient and remainder. Take the input 0x19 and 5,

ISSN 2959-6157

0x10 and 3, 0x0a and 4 as an example, the corresponding output results merchant and remainder are 5 and 0, 5 and 1, 2 and 2, respectively, and it can be seen that the results are correct and the Pipelined Method algorithm is reasonable.

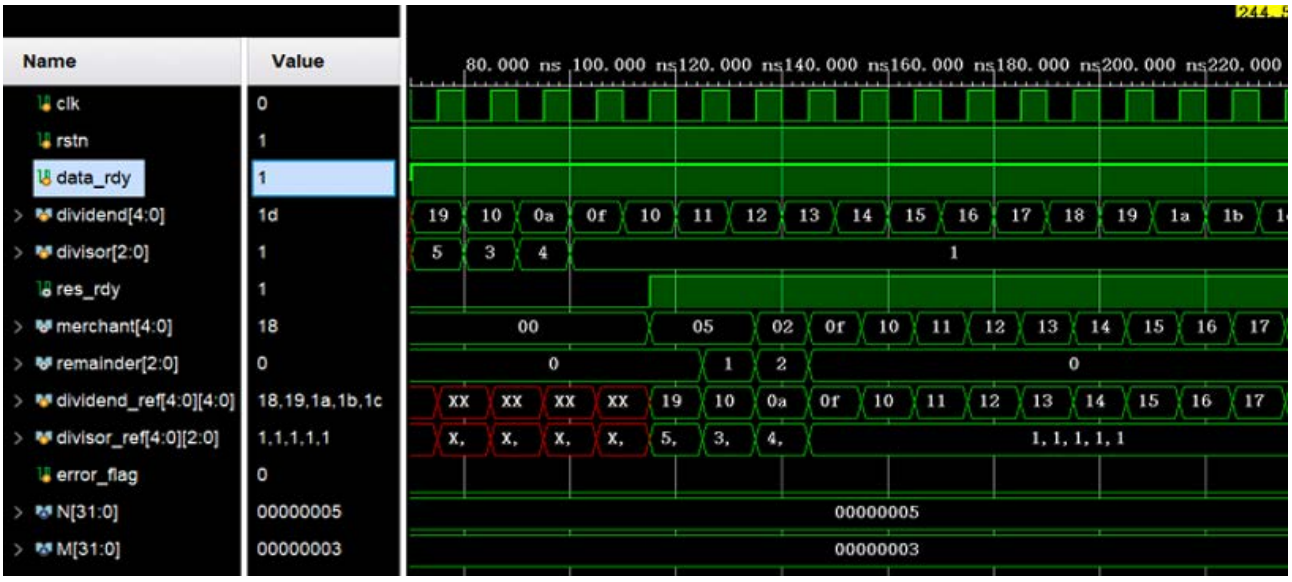

Fig. 5 Simulation results of the pipelined method

It should be noted that dividend_ref and divisor_ref are the input into dividend and divisor after delaying the number of cycles with the same bit width as dividend, and then the output results of merchant and remainder can be pipelined without delay, which is in line with the principle of pipelining algorithm. This is a major advantage of the pipelined method over other division algorithms. Parallel processing of different phases of the division operation makes the throughput of the pipelined method divider compared to a single-step divider, which is suitable for applications that require high speed and large amounts of data processing, such as image processing and scientific computing.

5. Conclusion

In this paper, researchers have reviewed and analysed various implementation methods for high-performance divider, including Restoring Division, Non-Restoring Division, Pencil-and-Paper Method, Newton-Raphson Method and Pipelined Method. The principles, code implementations and simulations of these algorithms are investigated, and it is found that different algorithms have their own advantages in terms of hardware complexity, computational efficiency and applicable scenarios. Pencil-and-Paper Method is simple but not efficient enough, but the performance can be improved by parallelising the operation. Restoring Division and Non-Restoring Division are suitable for simple hardware structures, but the performance of the former is limited by the restoring remainder process and the overall delay is larger, while the latter improves the efficiency by eliminating the restoring step but consumes higher hardware resources. The Newton-Raphson Method is very effective in dealing with division operations with a very large number of bits, but the precision of the result is affected by the number of iterations. The Pipelined Method uses a state machine to step-by-step control the execution of each stage of the pipeline, which can output a pipelined output without delay, and has a greater throughput, suitable for high-speed computing and image processing and other scenarios. Overall, different divider designs strike different balances between performance, hardware resources, and system constraints, and should be selected based on specific needs to achieve optimal performance.

Authors Contribution

All the authors contributed equally and their names were listed in alphabetical order.

References

[1] R. Eigenmann and G. Cybenko, As Eniac turns 50: perspectives on computer science support for science and engineering, in IEEE Computational Science and Engineering, 1996, 3(2): 16-17.

[2] S. F. Obermann and M. J. Flynn, Division algorithms and implementations, in IEEE Transactions on Computers, 1997, 46, (8): 833-854.

[3] Vishwas B R, Kiran V. Implementation and comparison of different non-restoring division algorithm. International Journal of Research and Review. 2022; 9(11): 70-73.

[4] U. S. Patankar, M. E. Flores and A. Koel, Division algorithms - From Past to Present Chance to Improve Area Time and Complexity for Digital Applications, 2020 IEEE Latin America Electron Devices Conference (LAEDC), San Jose, Costa Rica, 2020: 1-4.
[5] N. Neelima, A. S. Kumar, A. Jayanth, K. K. Mahitha and A. D. K. Reddy, Implementation of Efficient Restoring and Long Division Algorithm, 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), Delhi, India, 2023: 1-6.
[6] K. Kataria and S. Patel, Design Of High Performance Digital Divider, 2020 IEEE Vlsi Device Circuit And System (VLSI DCS), Kolkata, India, 2020: 1-6.
[7] K. Jun and E. E. Swartzlander, "Improved non-restoring division algorithm with dual path calculation," 2013 IEEE 56th International Midwest Symposium on Circuits and Systems (MWSCAS), Columbus, OH, USA, 2013: 1379-1382.
[8] Yang Licheng, Zhang Donghong, Zhan Siwei, et al. "Improvement of Non-Restoring Remainder Method—FPGA Implementation of Pre-Comparison Method Divider" Industrial Control Computer, 2015, 28(07): 79-80+84.
[9] O. I. Bureneva and O. U. Kaidanovich, FPGA-based Hardware Implementation of Fixed-point Division using Newton-Raphson Method, 2023 IV International Conference on Neural Networks and Neurotechnologies (NeuroNT), Saint Petersburg, Russian Federation, 2023: 45-47.
[10] D. N. Rao, G. Sai Charan, D. V. Venkata Sairam and K. S., "Posit Number Division using Newton-Raphson method," 2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT), Bhilai, India, 2021: 1-6.