Proceedings of the 5th International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/120/2025.19329

A Study of High Performance Dividers

Jiaqi Zhao[1,a,*]

[1]*School of Optoelectronics and Communication Engineering, Xiamen University of Technology, Xiamen, 361000, China*
a. zyl990602@outlook.com
**corresponding author*

Abstract: This paper explores the design of a high performance divider aimed at improving the performance of integer and floating point division operations to enhance the overall performance of the microprocessor. In the field of high-performance computing (HPC) and scientific computing, the design of floating-point divisors is critical because division is one of the most complex operations in floating-point arithmetic. The article firstly introduces the standard and arithmetic rules of floating-point numbers, analyses the basic principles of floating-point division, and compares the advantages and disadvantages of several classical floating-point division algorithms. In particular, the design of a double-precision floating-point divider based on Goldschmidt's algorithm is investigated in order to solve the problem of the complexity of the double-precision floating-point division arithmetic process and the large delay. The article also discusses current technology trends and practical application requirements for high-performance divider design, including applications in digital signal processing, communication systems, image processing, and other fields. In addition, the article discusses key factors such as the selection of the initial value of the iteration, the determination of the number of iterations, and the error control, and evaluates the advantages and disadvantages of different hardware implementations such as the SRT algorithm, the CORDIC algorithm, and the Radix-8 algorithm. Finally, the article looks at future trends in high-performance dividers, including the application of new materials and processes, the development of novel algorithms, and the integration of hardware accelerators.

Keywords: High Performance Dividers, Goldschmidt algorithm,SRT algorithm, CORDIC algorithm, Hardware implementation.

1. Introduction

In the field of high-performance computing and scientific computing, the performance of floating-point dividers is critical, as it is directly related to the accuracy and stability of the system. As technology advances, the requirements for processor performance are no longer limited to speed, but also include power consumption and chip area. The aim of this paper is to provide an overview of the current technology and application requirements for the design of high-performance dividers, and to explore optimization options to advance the technology and meet practical applications. Tingting He, in their study proposed a high performance double precision floating point divider design based on Goldschmidt algorithm [1]. This study significantly improves the performance and accuracy of the divider by optimizing the iterative process and error control mechanism. The design and

© 2025 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/120/2025.19329

implementation of a SIMD floating-point divider based on the SRT-8 algorithm was investigated by Ziyao Deng [2].This research reduces the number of iterations and increases the speed of computation through parallel processing techniques and algorithm optimization, while reducing the hardware resource requirements. Jingcun Wang and Yingbo Wang implemented the FPGA design of a complex divider based on CORDIC algorithm [3].This study utilizes the CORDIC algorithm's rotation operation to replace the traditional multiplication and addition operations, reducing hardware resource consumption and increasing the speed of operations. Xujun Li, Na Shi and Kejun Long proposed the design of an exact floating-point divider based on polynomial approximation algorithm [4].The design achieves a highly accurate division operation with an error of less than 0.5 ULP by reducing the number of iterations and optimizing the hardware structure. These research cases demonstrate the diversity and activity in the field of high-performance divider design, and also point to directions for future research, including the application of new materials and processes, the development of novel algorithms, and the integration of hardware accelerators.

This paper will analyze the strengths and limitations of different division algorithms and evaluate their advantages and disadvantages in terms of hardware implementation with a view to providing guidance for design. Through these studies, it is hoped to provide insight into the future development of high-performance dividers, as well as technical support for practical applications.

2. Technological developments

In the early days, the design of the divider was mainly focused on the optimization of the algorithm. For example, the Newton-Raphson algorithm and the Goldschmidt algorithm are two commonly used iterative algorithms that reduce the number of computations by iterative approximation, thus increasing speed. However, these algorithms still face challenges in hardware implementation, especially in terms of area and power consumption constraints. So modern algorithms were derived. Goldschmidt's algorithm is an iterative based floating point division algorithm which reduces the number of computations by iterative approximation. The advantage of this algorithm is that it enables high-speed floating-point division, but its disadvantage is that it requires a large look-up table and a complex error control mechanism. The SRT (Sh Subbase Radix) algorithm is an efficient numerical division algorithm that reduces the number of iterations by choosing a suitable base. The advantages of the SRT algorithm are simple hardware implementation and low area and power consumption, but its disadvantage is that it may require larger hardware resources in some cases. The design of a floating point divider based on logarithmic conversion makes use of logarithmic and antilogarithm conversions to simplify the division operation. The advantage of this approach is that it can significantly reduce the computational complexity. But the disadvantage is that it requires accurate logarithmic and inverse logarithmic tables. These may increase the demand on hardware resources.

3. Key design and implementation factors

The choice of the initial value of the iteration has an important impact on the convergence speed and final accuracy of the divider. For example, Tingting He mentioned the double lookup table method for determining the initial value of iteration in a double-precision floating-point divider based on Goldschmidt's algorithm, which saves area and improves the speed of iteration [5].In addition, the improved Goldschmidt algorithm in which a linear minimax polynomial approximation algorithm obtains an estimate of the reciprocal of the divisor with higher accuracy further optimizes the choice of the initial value of the iteration. The determination of the number of iterations is another key factor that affects the performance of the divider. Too many iterations increase the computation time, while not enough iterations may lead to insufficient accuracy. For example, in the design of a divider based on the SRT algorithm, by optimizing the algorithm structure and parallel processing techniques, the

DOI: 10.54254/2755-2721/120/2025.19329

number of iterations required can be effectively reduced, thus increasing the speed of operation. Error control is an important aspect of ensuring the performance of the divider. In Goldschmidt's algorithm, Tingting He found that the error generated during the iteration process can be effectively reduced and the accuracy of the calculation results can be ensured by a reasonable error control method [6].In addition, the optimization algorithm of the Newton iterative method also has a fast convergence rate with high accuracy, which helps to control the error. The SRT algorithm (Suan-Tung-Robust) is an efficient floating-point division algorithm widely used in modern computer systems. The algorithm increases the speed of the division operation by generating multiple quotient bits per iteration. To reduce the computational delay, the SRT algorithm can be implemented by processing quotient selection and remainder addition in parallel. For example, the design of SIMD structure based on SRT-8 algorithm mentioned by Ziyang Deng significantly reduces the computational delay and improves the frequency of iterative division by parallel processing of quotient selection and residue addition, as well as the use of the quotient digital storage technique. The performance of the SRT algorithm is affected by the size of the base [7].A larger base reduces the number of iterations and thus speeds up the computation, but at the same time increases the hardware complexity and area. For example, the optimization method proposed by Huaping Liu achieves a reduction in latency, albeit with an increase in area, for SRT algorithms with different bases, such as bases 4, 8, and 16, by executing in parallel two relatively independent parts of the traditional SRT structure (calculating the input values of the selection function table and finding the selection function table) [8].To reduce the overhead of hardware resources and save area, multiplexing strategies can be used. For example, the design mentioned by Jiyang Chen reduces hardware resource overhead by multiplexing lookup tables and adders, while reducing computational latency through parallel processing and on-the-fly conversion techniques Using redundant number representations simplifies the design and shortens the operation time. On-the-fly conversion techniques are used to improve the performance of the algorithm by quickly correcting the error during the iteration process. The design based on the SRT4 algorithm mentioned by Jiyang Chen employs these techniques [9].

4. Evaluation of hardware implementation methods

The hardware architecture and design consider different division algorithms for different application scenarios. For example, Ziyang Deng proposed that the SRT-8 algorithm is widely used in the design of floating-point dividers because of its fast iteration speed [10].The CORDIC algorithm, on the other hand, is suitable for FPGA implementation of complex dividers due to its rotation operation instead of multiplication and addition operation. The Radix-8 algorithm improves the speed of the complex divider and reduces the use of hardware resources by means of a bit-by-bit recursive algorithm and an operand pre-transformation technique. Tzu-Coc Deng proposed that hardware architectures for high-performance dividers are usually SIMD structures , parallel structures or optimized structures based on specific algorithms. These architectures are designed to increase computing speed and reduce power consumption. When designing a high-performance divider, Deng Ziyi needs to consider how to optimize the iterative division structure, how to reduce the hardware resource overhead, and how to improve the speed and accuracy of the operation .Advantages, Disadvantages and Applicable Scenarios of Different Hardware Implementation Methods SRT-8 algorithm, the advantage is that the iteration speed is fast and it is suitable for the design of a floating-point divider, the disadvantage is that it may require larger hardware resources. Jingcun Wang proposed the CORDIC algorithm with the advantages of device saving, high computational accuracy, and applicability to FPGA implementation of complex divider, and the disadvantage that it may not be applicable to all types of division operations. The Radix-8 algorithm, has the advantage of increased speed and reduced hardware resources, and the disadvantage that complex re-transformation techniques may be required. Specific use cases and performance evaluation of CORDIC algorithm in complex divider FPGA

DOI: 10.54254/2755-2721/120/2025.19329

implementation. The CORDIC (Coordinate Rotation Digital Computer) algorithm has a wide range of applications in FPGA implementations of complex dividers, especially in the field of digital signal processing (DSP).The CORDIC algorithm implements multiplication and division of complex numbers through a series of rotational operations, which significantly reduces the hardware resources required and improves the speed of operations compared to the traditional multiplication and division operations. Specifically for the complex divider application case, the CORDIC algorithm is used as an alternative to the traditional multiplication and addition operations. In complex division, the CORDIC algorithm adjusts the angle of the complex numbers through a rotation operation to achieve the division operation. Wang Yingbo suggests that the advantage of this approach is that it requires only basic addition, subtraction and shift operations without the need for complex multipliers or lookup tables, which greatly simplifies the hardware design and reduces resource consumption. For performance evaluation, implementations of CORDIC algorithms on FPGA typically exhibit high computational speed and low hardware resource consumption. For example, in the design of a complex divider based on the CORDIC algorithm, the multiplication and addition operations are replaced by using the CORDIC rotation operation, and the result is finally obtained by two shift operations. In addition, an improved version of the CORDIC algorithm further optimizes the performance of the algorithm. For example, by improving the judgement equations for the single iteration angle and the currently selected angle, the number of iterations can be reduced while maintaining the same computational accuracy, thus increasing the speed of the FFT operation. This improvement not only increases the efficiency of the algorithm, but also reduces the hardware area consumption and improves the computational accuracy.

In graphics processors, floating-point dividers are used to perform complex graphics rendering algorithms such as ray tracing and shadow calculations. In digital signal processing, dividers are used for tasks such as filter design and signal analysis. In addition, Huimin Du proposed that high performance dividers are also used in embedded systems and mobile devices to increase processing speed and reduce power consumption. Performance evaluation typically includes metrics such as computing speed, accuracy, area, and power consumption. Computing speed is usually measured in terms of the number of floating point operations performed per second (FLOPS).Accuracy, on the other hand, is assessed by comparing the difference between the calculated result and the actual value, and commonly used error metrics include ULP(unit in the last position).Tina Ho mentioned that area and power consumption, on the other hand, are important metrics for measuring the cost of hardware implementation, especially important in mobile and embedded devices. These data show that different algorithms and design strategies can significantly affect the performance of the divider. The results of the performance evaluation are important for the design and implementation of the divider. Firstly, they help designers to select or optimize algorithms to meet the needs of a particular application. Secondly, by comparing the performance of different designs, the allocation and optimization of hardware resources can be guided. Finally, performance evaluation results can also be used to predict how a new design will perform in real-world applications, thereby accelerating the product development process. In conclusion, performance evaluation of high-performance dividers is a multidimensional process involving a variety of techniques and methods. By continuously optimizing the algorithms and design, the performance of the divider can be significantly improved to meet the increasing computational demands.

The latest use cases of high performance dividers in the field of graphics processing and digital signal processing are mainly in the following areas.3D image processing algorithm that implements a three-stage pipeline divider, which is a new type of divider with smaller area and faster speed than the traditional divider while satisfying the same accuracy. Digital signal processing (DSP) multiplication is one of the main operations. For example, novel approximate 4:2 compressors based on carbon nanotube field effect transistors (CNFETs) are proposed for image multiplication

DOI: 10.54254/2755-2721/120/2025.19329

applications. These designs offer significant improvements in power consumption, delay, power delay product (PDP), and transistor counting over previous approximate compressors. Laplace Transform Detection for Image Processing Applications Using an Ultra-Fast Current Mode Approximate Full Adder. These adders use diodes to stabilize the voltage, resulting in a shorter critical path, increased computing speed, and a reasonable compromise between power delay product (PDP) and other important evaluation factors.

The main technical challenges faced by high performance dividers are accuracy control. As the computational demands increase, so do the accuracy requirements for the divider. For example, Tingting He proposes that floating-point dividers need to support the IEEE-754 standard, which requires the dividers to ensure the accuracy of the computation results along with the speed [1].In addition, approximate computational methods, while reducing power consumption and design complexity, may sacrifice some computational accuracy. Power consumption optimization, as semiconductor technology evolves, the performance gains from traditional scaling effects gradually diminish and power consumption issues become a key challenge in design. High-performance dividers need to guarantee performance while keeping power consumption as low as possible. Lidong Xing proposed, for example, an FPGA-based low-power high-speed divider design that significantly reduces dynamic power consumption by using new algorithms and gated clocking techniques .Algorithm optimization requires continuous exploration and optimization of the division algorithm. For example, a divider based on the SRT algorithm improves the speed of integer and floating-point division operations by optimizing the design structure. In addition, a double-precision floating-point divider based on Goldschmidt's algorithm achieves higher performance through improved algorithms and hardware multiplexing methods.

5. Future development trends

The future development trend of high-performance separator new materials and new processes will be more efficient and energy-saving applications. For example, a new high-frequency, high-linearity CMOS cross-conductivity current-mode multiplier/divider utilizes an advanced CMOS process to improve bandwidth and linearity while reducing power consumption. The development of novel algorithms has become an important direction for the future. For example, a new divider based on a two-bit algorithm significantly improves the speed of the division operation by reducing the number of execution cycles. In addition, the 64-bit divider, designed based on the principles of bead work, utilizes the principles of the operating model of ancient mathematics to significantly reduce the critical path delay. Integrator for hardware accelerators have become a trend. Weisheng Wang proposed, for example, a single-precision Goldschmidt divider integrated into a SOC based on the Cortex-M3 core, which effectively improves the speed of the division operation in the image recognition process .

6. Conclusion

This paper comprehensively discusses the design of high-performance divider, the research significance of which is to enhance the performance of integer and floating-point division operations in microprocessors, which is crucial for high-performance computing, digital signal processing, communication systems, image processing and other key areas. Through in-depth analysis and comparison of different division algorithms, this paper reveals the advantages and limitations of various algorithms and proposes an optimized design structure to meet the needs of different application scenarios. The results show that by optimizing the algorithm and design, the performance of the divider can be significantly improved to meet the increasing computational demands.

DOI: 10.54254/2755-2721/120/2025.19329

Looking ahead, the trend in high-performance dividers will focus on the application of new materials and processes, the development of novel algorithms, and the integration of hardware accelerators. With the development of new materials and processes, the design of high-performance dividers will become more efficient and energy-saving. The development of novel algorithms will respond to increasing computational demands, while the integration of hardware accelerators will improve computational efficiency in specific application scenarios. These advances will not only promote the technological progress in related fields, but also provide effective technical support for practical applications, foreshadowing a brighter future for high-performance divider design.

References

[1] *He, T., Peng, Y., & Lei, Y. (2015). Design of high-performance double-precision floating-point divider based on Goldschmidt algorithm. Computer Applications.*

[2] *Deng, Z., Chen, S., & Peng, Y. (2014). Design and implementation of a SIMD floating-point divider based on SRT-8 algorithm. Computer Engineering and Science.*

[3] *Liu, H., & Hu, W. (2003). An optimization method to reduce the delay of SRT floating-point algorithm. Computer Research and Development.*

[4] *Chen, J., Peng, Y., et al. (2014). Design of double-precision floating-point division and square root based on SRT algorithm. In Proceedings of the IEEE Conference on Computer Communications.*

[5] *Wang, J., & Wang, Y. (2008). FPGA implementation of complex divider based on CORDIC algorithm. Modern Electronic Technology.*

[6] *Wang, Y. (2008). Complex divider FPGA implementation based on CORDIC algorithm. Modern Electronics Technique.*

[7] *Du, H., & Liu, T. (2013). Design and implementation of floating-point dividers in graphics processors. Journal of Xi'an University of Posts and Telecommunications.*

[8] *Li, X., Shi, N., Long, K., et al. (2023). Design of an accurate floating-point divider based on polynomial approximation algorithm. Microelectronics and Computer.*

[9] *Xing, L. (2014). FPGA-based low-power high-speed divider design. Electronic World.*

[10] *Wang, W. (2023). 32-bit SOC design of ARMV7 instruction set and its divider optimisation (Master's thesis). Beijing University of Chemical Technology.*