

A Pipelined Division for Fixed Operation Using User-defined Floating Point

Pengfei Yang*, Daolu Zha*, Xi Jin*

*Department of Physics, University of Science and Technology of China, Hefei, Anhui Province, China

yangpf@mail.ustc.edu.cn, ustczdl@mail.ustc.edu.cn, jinxj@ustc.edu.cn

Abstract—A pipelined division arithmetic for fixed operation using user-defined floating point is proposed in this paper. Division operation is difficult in fixed operation, the dividend A is firstly converted into user-defined floating point, then the inverse is implemented in the inverter unit. B/A will complete with a fixed multiplication. Linear approximation theory and Newton-Raphson iteration are used in inverter unit.

The major advanced of this arithmetic is that it easily combines with fixed operation, we can proposed user-defined floating point according to the range of dividend A and the accuracy can easily acquire. In this paper, inverter unit for A range to [0.5, 2) with 23 decimal bits is designed. It is compiled and implemented both on FPGA and Synopsys Design Compiler.

Keywords—division, linear approximation, Newton-Raphson iteration, FPGA, Synopsys Design Compiler

I. INTRODUCTION

Floating point division is the most crucial arithmetic. Compared with other floating point arithmetic, division is much more complex and spends much resources. The hardware implementation of a high-performance division is not an oversimplified project. There are three major arithmetic for implementing a high-performance division: Taylor Series (TS), Digit Recurrence (DR) and Functional Iteration Algorithm (FIA). TS have low latency and high precision but also spends much more resource and power. The SRT algorithm is the briefest implementation of division based on DR, however, it has a linear convergence which leads to a much long latency. The most famous FIA is the Newton-Raphson iteration. It is convenient to achieve a high-performance implementation of division with a quadratic convergence and a certain periodicity which are important for a high-speed and high-performance pipelined implementation of division.

Our research is focused on the implementation of a pipelined division using user-defined floating point. This is used in a fixed point system. Linear approximation is used to get the initial root and Newton-Raphson iteration are applied to increase precision in inverter unit.

The rest of this paper is organized as follows: In section II, we present the theory of user-define floating point and then introduce the linear approximation that is used in this paper, we also provide a background knowledge on Newton-Raphson iteration. In section III, show the architecture of our algorithm. The hardware implementation is presented in section IV. The conclusion and results compared with related work is provided in section V.

II. BACKGROUND

In this section, a brief introduction of the theory used in this paper is presented, include: user-define floating point, linear approximation and New-Raphson iteration.

A. User-defined Floating Point

The standard format for single-precision floating point for IEEE-754 format floating point is as below:



Figure 1. Representation of single precision

floating point for IEEE-754 format

The real value is present as the following formula:

$$x_r = (-1)^{S_x} * (1 + m_x) * 2^{e_x}$$

Where:

$$m_x = \frac{M_x}{L}; L = 2^{23}$$

$$e_x = E_x - B; B = 127$$

It show that S_x indicate the symbol of x_r , e_x determine the scope and the precision of x_r depends on m_x .

A user-defined floating point means make a customization of S_x, e_x, m_x according to the x_r , for example, it does not need S_x bit, and only 1 bit e_x , 23 bits m_x when $x_r \in [0.5, 2)$ with 23 decimal bits.

B. Linear Approximation

Linear equation $g(x) = kx + b$ where $k = \pm 2^n, n$ is an integer is used to approximate the objective equation $f(x) = 1/x$ when $x \in (x_1, x_2) \ni (1, 2)$, the error equation is:

$$h(x) = f(x) - g(x) = \frac{1}{x} - kx - b$$

The derivative of $h(x)$ is that:

$$\frac{\partial h}{\partial x} = \frac{-1}{x^2} - k$$

The derivative of $h(x)$ should be closed to zero as far as possible to make the $h(x)$ have small change in the range of

$x \ni (x_1, x_2)$. It means that k should be approximate equal to $\frac{-1}{x^2}$ which can help confirming the value of k .

Considering that:

$$q(x) = f(x) - kx$$

$q(x)_{max}$ and $q(x)_{min}$ denote the maximum and minimum value of $q(x)$ when $x \ni (x_1, x_2)$, and the minimum error $|h(x)|$ can be indicated as below:

$$\Delta = \frac{q(x)_{max} - q(x)_{min}}{2}$$

Where

$$b = \frac{q(x)_{max} + q(x)_{min}}{2}$$

Now we get the linear equation $g(x) = kx + b$, $g(x)$ is related to floating point. $y = \frac{1}{x} \in (0.5, 1)$ when $x \ni (1, 2)$, it is the same as $g(x)$. The linear equation $g(x)$ is presented as:

$$2^{-1} * (1 + m_y) = k * 2^0 * (1 + m_x) + b$$

Put it in order:

$$M_y = 2kM_x + R$$

$$R = (2b - 1) * 2^l$$

l is the amount of mantissa bit.

C. Newton-Raphson Iteration

Newton-Raphson iteration is a very efficient iteration method to approximate the root of a non-linear equation. To approximate the root of the equation $f(x) = 0$, $f'(x)$ is the derivative of equation $f(x)$. Given the initial approximation x_0 , then a better approximation x_{n+1} is:

$$x_{n+1} = x_n - \frac{f(x)}{f'(x)}$$

Apply to the equation $f(y) = \frac{1}{y} - x = 0$, the derivative of $f(y)$ is:

$$f'(y) = \frac{-1}{y^2}$$

The iteration equation is:

$$y_{n+1} = y_n - \frac{\frac{1}{y_n} - x}{\frac{-1}{y_n^2}}$$

Put it in order is

$$y_{n+1} = (2 - y_n * x) * y_n$$

The first-order Newton-Raphson method has a quadratic convergence, and it always converges effectively.

III. PROPOSED PIPELINED DIVISION ARCHITECTURE

In our design, the positive division A is sent to Floating point transfer unit to transfer into user-defined floating point parameter E and M. Then the inverter unit calculates the division according to the M. result from inverter and dividend B complete the multiplication in the fixed multiplication. The normalization unit will give the real result of B/A according to E finally.

The architecture of our algorithm is shown as blow:

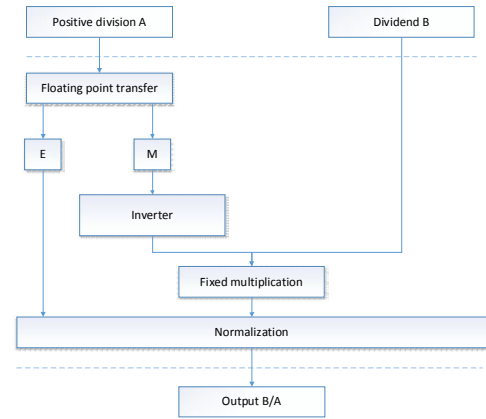


Figure 2. Architecture of proposed algorithm

In our implementation, division A is a 24 bit fixed number with 23 decimal bits and A has a range (0.5, 2). So the E of A only has 1 bit, if the highest bit of A is 1, E is 1, to the contrary E is 0, and the M(23 bits) is obtained easily.

The result from fixed multiplication is shifted in normalization unit to get the correct result of B/A according to E.

The inverter unit is the most important and complex unit in the architecture, the detail architecture of this unit is shown as below:

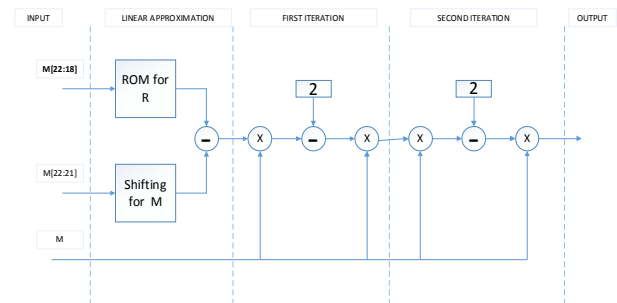


Figure 3. Architecture of the inverter unit

The inverter unit is designed to hunt for the root of $1/x$ when x is in a range (1, 2) the same as most of the division algorithm. there is only one input that is 23 bits mantissa of x , the same as most single precision floating division method.

In the linear approximation section, the range of (1, 2) is averagely divided into 32 small range and encoded by M[22:18], the proper value of R(11 bits) is store in a 32x11 ROM, and the k depends on M[22:21]: k is -1 when M[22:21]

is 2'b00, k is -1/4 when $M[22:21]$ is 2'b11, and k is -1/2 in the rest.

M is firstly sent to the linear approximation section, a high precision initial approximation root is calculated by the below mathematical expression:

$$2kM_x + R$$

The precision of the root will increase further through twice Newton-Raphson iteration according to following iteration equation:

$$y_{n+1} = (2 - y_n * x) * y_n$$

IV. IMPLEMENTATION AND RESULT INTRODUCTION

The Verilog description of the proposed algorithm was successfully organized. The algorithm was simulated to ensure correction performance.

In proposed inverter unit, (1,2) had been averagely divided in to 32 small ranges. Figure 4 shows maximum error in every different ranges of linear approximation only, and maximum error in whole (1,2) is 5.24E-3, it achieved 8.57 bits precision.

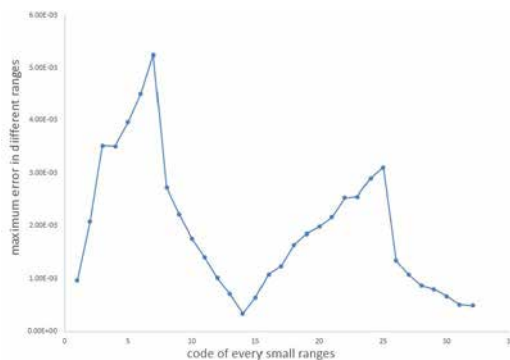


Figure 4. Error distribution of linear linear approximation only

Figure 5 shows the maximum error in every different ranges of whole inverter unit. As known that absolute value of derivative has a prominent effect on rate of convergence. Absolute value of derivative becomes smaller while the x tends to 1, it means that the root converges slowly, so, the initial approximation should be more exact. The proposed linear approximation method which is used to get the initial approximation satisfies this condition well. Maximum error in the whole (1,2) is 1.079E-7, it achieved more than 24 bits precision.

Table 1 shows the implementation result when the algorithm mapped on Xilinx Virtex7 FPGA. Compared with other single

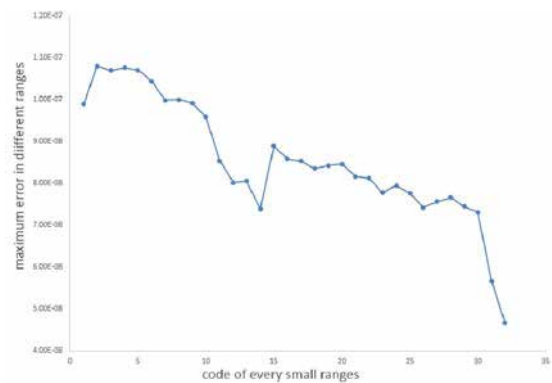


Figure 5. Error distribution of whole inverter unit

precision division algorithm I knew, proposed algorithm used less LUTs. It is significant that our algorithm only used 352 byte (32x11) ROM and it is structured by a few LUTs, do not need block memory, and it could achieve 223MHz.

TABLE 1. SYNTHESIS RESULTS ON VIRTEX-7 FPGA

Module	LUT	FF	DSP	RAM
Inverter	25	--	7	--
Total	122	59	9	--

TABLE 2. SYNTHESIS RESULTS USING SYNOPSIS DC

Voltage (V)	Temperature (°C)	Area (um ²)	Power (mW)	Frequency (MHz)
1.08	125	0.333	54.07	495
1.20	25	0.325	97.93	757

Table 2 shows the implementation result using Synopsys Design Compiler in 0.11 micron cell library. The clock transition is 0.15 ns, the set up time is 0.2 ns and input delay is 0.15 ns. It shows that our implementation had a good performance in area and power.

V. CONCLUSION AND FUTURE WORK INTRODUCTION

A pipeline division algorithm applied to fixed operation is present in this paper, and it is convenient for different range division A by using user-defined floating point. The algorithm is implemented both on Xilinx Virtex7 FPGA and Synopsys Design Compiler. The synthesis result shows that less resource are used and have a high throughput. It could be a high performance division algorithm.

A linear approximation method is proposed to achieve an accurate initial approximation efficiently. It may have advantages when merged with other iteration algorithm.

REFERENCES

- [1] Süleyman Savas, Erik Hertz, Tomas Nordström. "Efficient Single-Precision Floating-Point Division Using Harmonized Parabolic Synthesis" Computer Society Annual Symposium on VLSI .July 2017
- [2] Shaobing Huang, Li Yu, Fang-jian Han , Yiwen Luo. "A Pipelined Architecture for User-defined Floating-point Complex Division on FPGA". Conference on Electrical and Computer Engineering. June 2017
- [3] Liang-Kai Wang, Michael J. Schulte. "Decimal Floating-Point Division Using Newton-Raphson Iteration". 15th IEEE International Conference on Application-Specific Systems, Architectures and Processors, October 2004.
- [4] "Fast inverse square root," Oct. 2016, page Version ID: 747006619.[Online].Available:https://en.wikipedia.org/wiki/Fast_inverse_square_root



Pengfei Yang received his B.S. degree from University of Science and Technology of China, and he is currently a M.S. student in Department of Physics in University of Science and Technology of China, Anhui, China, under the supervision of Prof. Xi Jin. His current research work is mainly on image processing.



Daolu Zha received his Bachelor degree in Applied Physics from University of Science and Technology of China. Currently, he is pursuing a Ph.D. degree of Microelectronics and solid electronics in University of Science and Technology of China. His current research interests include computer vision, multimedia analysis and SoC design.



Xi Jin received the B.S. degree from University of Science and Technology of China, Anhui, China, and he is currently an associate professor in Department of Physics in University of Science and Technology of China, Anhui, China. His research interests include SOC design technology, VLSI design, computer-aided design methodologies for SoC system integration and FPGA-based Hardware structure design.