THE DESIGN AND IMPLEMENTATION OF AN ASYNCHRONOUS RADIX-2 NON-RESTORING 32-B/32-B RING DIVIDER*

Jen-Shiun Chiang and Jun-Yao Liao #

Department of Electrical Engineering
Tamkang University
Tamsui, Taipei, Taiwan

ABSTRACT

Division operation is very important in the computer system. Conventionally synchronous techniques are applied to implement the divider. In this paper we will propose a new asynchronous architecture for the divider. In this asynchronous scheme, the architecture is of simplicity and is very easy for the VLSI implementation. By this asynchronous architecture, we use TSMC's 0.6um SPDM process to design a 32-b/32-b radix-2 non-restoring divider. The HSPICE simulation shows that this divider can finish a 32-b/32-b division operation in between 3.7ns to 160.2ns.

1. INTRODUCTION

The divider is a very important device in the computer CPU. Conventionally people use synchronous design technique to design the divider [1]. In the synchronous system, we need a global clock to activate the operation of the system. The global clock scheme used in the synchronous circuits makes the design of a system be simple [2]. However, there are several drawbacks in the synchronous system with global clock signals [3, 4]. First, the long path of the system clock signals may cause clock skew, and that may cause very severe problems and the system may thus be malfunctioned. Second, the clock frequency of the system clock is decided by the longest period among the states. In order to increase the clock frequency, people have to do a lot of efforts to separate the states to be almost equal operation time, and this may reduce the design efficiency and increase the design time. Third, nowadays people like to use dynamic devices to design circuits [5], and the charge and discharge of the circuits consume power. The higher the frequency of the clock is, the more the power consumes. Fourth, the VLSI technology progresses significantly nowadays, and the implementation processes are dependent prone, and that means the circuit works in one VLSI implementation process may not function properly in another VLSI process [4]. Therefore, the same circuit has to be redesigned in different implementation processes. This characteristic reduces the life time of a product.

Besides the synchronous design technique, there is another design technique — asynchronous design technique. The asynchronous circuit [6-9] is activated by the input event, therefore, there is no need of system clocks. There are several approaches to design an asynchronous circuit [4], such as bounded delay models, micropipelines [2], delay insensitive circuits, and quasi delay insensitive circuits [10]. By the above approaches, the asynchronous circuit is activated by the input event and the handshaking signals, so we do not need to take special care of the system clock path routing to avoid the clock skew problems. Therefore, the asynchronous designs are implementation process independent [4]. There is no system clock in the asynchronous circuits, the power consumption which we worry about in the synchronous circuits is thus prevented.

The divider proposed in this paper is in the asynchronous manner, and there is no system clock signal to activate the division operation. We use a single stage ring to finish the radix-2 non-restoring division [11,12], so the hardware is small. About the asynchronous controller, we use the very commonly used devices, such as exclusive-or gates, latches, and pseudo NMOS technique [13] to implement this 32-b/32-b divider. In order to increase the operation speed, the conditional carry-selection adder (CCSA) [14] is adopted. This divider is designed in a parallel-in serial-out manner and takes only 4,051 MOS transistors. The HSPICE simulation shows this divider works well.

2. RADIX-2 NON-RESTORING DIVISION

Division is the most difficult operation in the computer arithmetic. Basically the division algorithm can be classified as multiplicative and subtractive approaches. In the multiplicative approach, we try to find the multiplicative inverse to calculate the quotient. On the other hand, we subtract the divisor from the partial

* This paper is supported by the National Science Council of the Republic of China under grant NSC 87-2215-E-032-004

Jun-Yao Liao is with the Holtek Corporation, Science Based Industrial Park, Hsin-Chu, Taiwan.

0-7803-4455-3/98/$10.00 © 1998 IEEE

remainder (dividend) recursively to find the quotient and remainder. Here we will concentrate ourselves to the subtractive approach. In the subtractive idea the algorithm is the same as the division method that we were taught in the elementary school. Suppose that there are two n-digit numbers, X and D, which represent the dividend and divisor respectively. By the division operation we can find a n-digit quotient and a n-digit remainder denoted as Q and R respectively. The mathematical representations of X, D, Q, and R are as following [12],

$$R^{(j+1)} = r \times R^{(j)} - q_{j+1} \times D \qquad (1)$$

Where q_{j+1} is the $j+1$th digit of the quotient.

$j = 0, 1, 2, \ldots, n-1$ is the iteration number.

R_j is the partial remainder at iteration j.

r is the radix number.

The final quotient is represented as

$$Q = q_1 q_2 q_3 \ldots q_n$$

Due to the complexity and the hardware cost, we use radix-2, i.e., r=2, for our design. Therefore, equation (1) can be rewritten and represented in equation (2) as follows [12].

$$R^{(j+1)} = 2 \times R^{(j)} - q_{j+1} \times D \qquad (2)$$

In the hardware design we have to check the subtraction at each step to decide the quotient in that digit. There are two ways to find the quotient of the current digit. One is the restoring method, and the other is the non-restoring method. The non-restoring quotient map is shown in Figure 1. By this method, there is no need to add divisor to restore the previous partial remainder. However, the quotient in the non-restoring scheme is represented in the signed bit (digit) format. Therefore, after we finish the division process, the non-restoring method needs an additional step to convert the signed bit format to the binary number representation. Since we do not need to check the polarity of the partial remainder to do the restoring of the partial remainder. Therefore, the speed of the non-restoring division algorithm is faster than the speed of the restoring division algorithm [11,12]. For the benefit of the speed and hardware cost, we use normalized radix-2 non-restoring division algorithm to build our divider.

3. THE ARCHITECTURE OF THE ASYNCHRONOUS DIVIDER

The architecture of the divider is shown in Figure 2. The basic components of this divider consists of multiplexers, counters, adders, shift registers, and latches. The operating procedures are described in next paragraph. Suppose that the divisor and dividend are ready, we give a starting signal to start the divider. According to the non-restoring division algorithm, the multiplexer of the divisor will select the positive or negative divisor which depends on the sign of the result of the 32-b adder (partial remainder). If the partial remainder is positive the quotient bit is set to 1, and the multiplexer selects the negative divisor for the next iteration. On the other hand, the quotient bit is set to 0, and the multiplexer will select the positive divisor for the next iteration. The GC (generate complete) gate has two sets of inputs, and each set is with 32-bit length. When both of the two sets are the same, the output of the GC gate is set to 0, otherwise, the output of the GC is set to 1. The detail circuit design of GC will be described in Section IV. The GC gate and the feedback latch together will generate a self-timed clock signal. The self-timed clock signal is used to trigger the counter and the feedback latch. Originally the output of GC is 0, as soon as the starting signal is triggered, then the output of GC becomes 1. This GC signal will activate the counter and make the counter to count 1 up. The feedback latch is designed as a level triggered latch. As soon as the counter counts, the multiplexer in the dividend side will select the result from the feedback latch. When the adder finishes summation operation, the two sets of inputs of GC are the same and cause the output of GC to be 0, and the feedback latch stops latching data. At this moment the 31st bit of the partial remainder will select either positive or negative divisor to the adder and do the addition operation. By the way, the 31st bit of the partial remainder is shifted into the MSB of the shift register to form the current bit of the quotient. As soon as the addition operation of the adder starts, the output of GC changes to 1, and the feedback latch starts latching the partial remainder, and the counter counts again. Recursively, the operation will repeat 32 times, and the counter will count 32 times and then set the ripple carry bit of the counter to 1 and the division procedure stops. Finally the quotient will be found in the output latch.

4. THE CIRCUIT DESIGN AND THE ASYNCHRONOUS DIVIDER

The key components of this divider are the adder and the GC gate. In order to increase the speed of the division, the conditional carry-selection adder (CCSA) [14] is used. The GC gate is designed by the pseudo NMOS approach which is shown in Figure 3. In the GC circuit, the exclusive-or gate is used to check the completion of the summation of the adder. One input of the exclusive-or gate is from the output of the feedback latch, and the other input is from the corresponding bit of the partial remainder. When both inputs of GC are different, it means that the adder has not finished summation operation yet,

and the feedback latch is still latching the output of the adder. As soon as the adder finishes summation, the exclusive-or gate of GC becomes 0 and makes the feedback latch stop latching data but hold the latched data. By the pseudo NMOS and exclusive-or gates arrangement as shown in Figure 3, if one output of the exclusive-or gate is 1, the pseudo NMOS circuit will discharge and make the output of the GC to be 1. Only all the outputs of the exclusive-or gates are 0's, then the "output" of Figure 3 becomes 0. This "0" signal of GC will cause the feedback latch to hold the latched data, and generate a series of self-timed clock signals.

By the arrangement of GC and CCSA adder, the divider can work recursively. We need only one trigger signal ("starting") to tell the divider to do the division, and no external clock is needed. When the division operation is finished, the quotient is latched, and then the system stops and idle to wait for the next sets of divisor and dividend, and the asynchronous operation mode is thus formed.

5. THE SIMULATION AND THE VLSI LAYOUT

By the architecture and the circuit design technique discussed in Section III and IV, we use TSMC's 0.6um SPDM process to design and implement a 32-b/32-b parallel-in serial-out divider. The fastest division needs only 3.7ns and the HSPICE simulation is shown in Figure 4. If we need 32 division steps, it takes 160.2ns and the HSPICE simulation is shown in Figure 5. The specification of this divider is shown in Table. 1. The VLSI layout of this 32-b/32-b radix-2 non-restoring divider is shown in Figure 6.

6. CONCLUSION

A new architecture of the 32-b/32-b radix-2 non-restoring asynchronous divider is proposed and implemented. This is a ring divider, and the asynchronous component, GC gate, is used to control the addition or subtraction procedures. The architecture of the divider is of simplicity and takes small hardware. From the simulation we find that even a very simple architecture the divider can operate asynchronously and properly. We design the divider by TSMC's 0.6μm SPDM process and this divider (32-b/32-b) is composed of 4,051 MOS transistors. Theoretically asynchronous circuits consume less power. However, it takes 157.5mW for the 32-b/32-b division. We find the reason is that pseudo NMOS architecture in the GC gate may consume most of the power. If the pseudo NMOS of the GC gate can be revised, the power consumption of this asynchronous divider may be reduced significantly.

7. REFERENCES

[1] J. B. Kuo, H. P. Chen, and H. J. Huang, " A BiCMOS dynamic divider circuit using a non-restoring iterative architecture with carry look ahead for CPU VLSI", *1993 IEEE Int. Symposium on Circuits and Systems*, pp. 2027-2030.

[2] Ivan E. Sutherland , "Micropiplines", *Commun. ACM*, vol. 32, no. 6, pp. 720-736, June 1989.

[3] Stephen H. Unger, "Hazards, critical races, and metastability", *IEEE Trans. Computers*, vol. 44, no. 6, pp. 754-768, June 1995.

[4] Scott Hauck, "Asynchronous design methodologies : an overview", *IEEE Proceeding*, vol. 83, no. 1, pp. 69-93, Jan. 1995.

[5] J. Yuan, and C. Svensson, "High-speed CMOS circuit technique", *IEEE J. Solid-State Circuits*, vol. 24, no. 1, pp. 62-70, Feb. 1989.

[6] Jacob and R. W. Brobersen, "A fully asynchronous digital signal processor using self-timed circuits", *IEEE J. Solid-State Circuits*, vol. 25, no. 12, pp. 1526-1537, Dec. 1991.

[7] Christian Piguet, "Logic synthesis of race-free asynchronous CMOS circuits", *IEEE J. Solid-State Circuits*, vol. 26, no. 3, pp. 371-380, March 1991.

[8] Ted E. Willians and Mark A. Horowitz, "A zero-overhand self-timed 160-ns 54-b CMOS divider", *IEEE J. Solid-State Circuits*, vol.26, no.11, pp.1651-1661, Nov. 1991.

[9] Marc Renaudin, Bachar Ei Hassan, and Alain Guyot, "A new asynchronous pipeline scheme: application to the design of a self-timed ring divider", *IEEE J. Solid-State Circuits*, vol.31, no.7, pp.1001-1013, July 1996.

[10] Pedro A. Molina P. Y.K. Cheung and David S. Bormann, "Quasi delay-insensitive bus for fully asynchronous systems", *ISBN : 0-7803-3073-0*, pp.189-192, 1996.

[11] Kai Hwang, *Computer Arithmetic Principles, Architecture, and Design*, John Wiley & Sons Inc., 1979.

[12] Israel Koren, *Computer Arithmetic Algorithms*, Prentice-Hall Inc., 1993.

[13] N. Weste, K. Eshraghhian, *Principles of CMOS VLSI Design: A Systems Perspective, 2nd,* Addison-Wesley Publishing Company, 1993.

[14] N. Ohkubo, and M. Suzuki, " A 4.4ns CMOS 54x54-b multiplier using pass-transistor multiplexer", *IEEE J. Solid-State Circuits*, vol.30, no.3, pp. 251-256, March 1995.

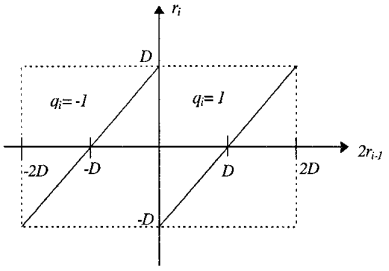

Figure 1. The Quotient Map of Non-Restoring Division

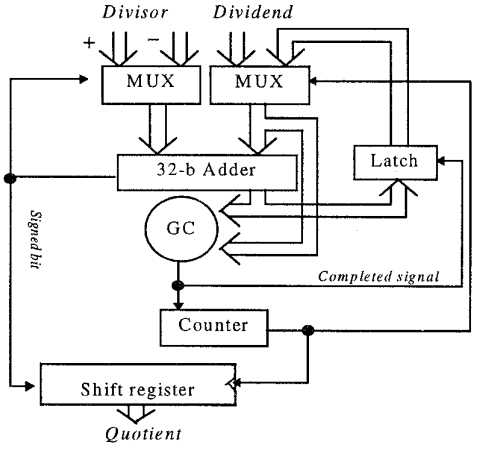

Figure 2. The Architecture of the Divider

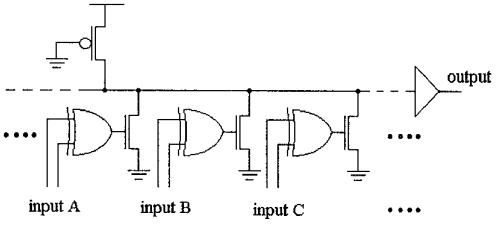

Figure 3. The Structure of GC Gate

Table 1. The Specification of the Divider

Process	The fastest division	The complete division	Power	Transistor count	Layout area
TSMC0 .6μm SPDM	3.7ns	160.2ns	157.5mW	4051	835μm × 840μm

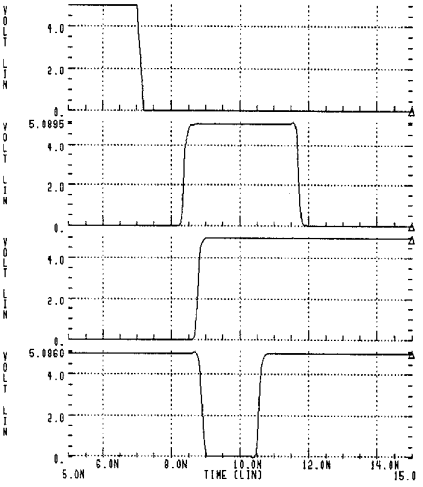

Figure 4. The Fastest Division

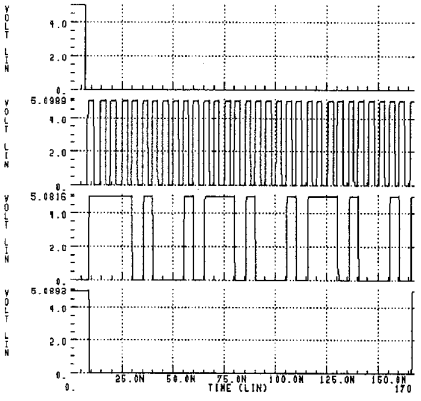

Figure 5. Full 32 Division Steps

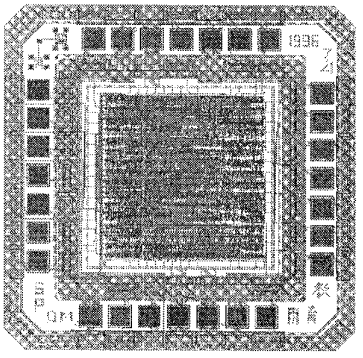

Figure 6. The VLSI Layout of Divider