IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 16, Issue 4, Ver. VII (Jul – Aug. 2014), PP 27-30
www.iosrjournals.org

A Non-restoring Division Algorithm

Shovan Roy
Department of Computer Science, Kalinagar Mahavidyalaya, P. O.: Kalinagarhat, North 24 PGS, PIN: 743442, WB, India

Abstract: *Non-restoring division method originally defined by Robertson in 1958. Restoring and non-restoring division processes are the algorithms conventionally used to program division method on microprocessors to minimize the hardware cost. A new hardware algorithm is to be proposed for non-restoring division algorithm for nonnegative integers. Restoring division algorithm maximizes the hardware cost where as non-restoring division algorithm minimizes the hardware cost.*
Keywords: *Register, Accumulator, ALU, Shifting Operation, Non-restoring division algorithm, Hardware minimization.*

I. Introduction

Originally non-restoring division method defined by Robertson in 1958 [1, 4]. He described the non-restoring division method by the recursive relationship:

$P_{j+1} = RP_j - Q_{j+1} D$ with j = 0, l, …, m – l. Where,
P_j is the j^{th} partial remainder,
P_0 is the dividend,
P_m is the remainder,
Q_j is the j^{th} digit of the quotient to the right of the radix point,
m is the number of digits, radix R, used to represent the quotient,
D is the divisor.

Division is the reciprocal operation of multiplication. In the grammar school division algorithm using decimal number system we express a relationship for the division algorithm:

Dividend = Quiotent x Divisor + Remainder.

If both the dividend and divisor are positive then the quotient and remainder are also positive. We will ignore the sign through out my proposed algorithm & example.

Computer pioneers [2] accepted that at most half of the divisor register has functional information. So we can cut the divisor register and ALU in half. In this way we can minimize the hardware cost. In this concept division hardware block diagram is the following:

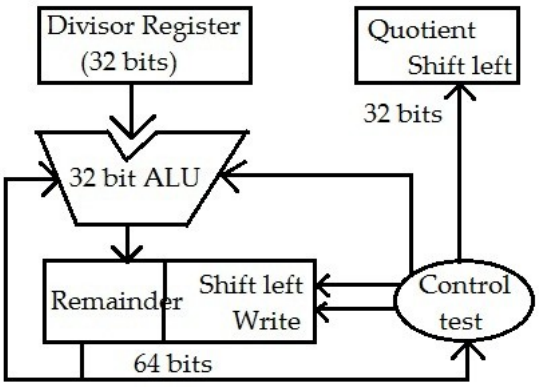

Fig. 1. Division Hardware: the ALU & divisor registers are halved & the remainder is shifted left.

After that historical observation computer pioneers axiom that the quotient register could be eliminated by shifting the bits of the quotient register into the remainder register. The block diagram of this concept is given below:

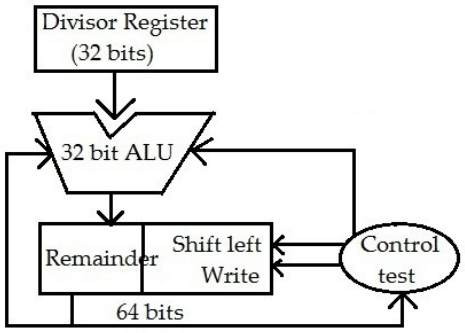

Fig. 2. Division Hardware: Combines the quotient register with the right half of the remainder register.

II. Existing non-restoring division algorithm [5]

Non-restoring division bases the selection of quotient register digits on the signs of the remainder register R_{i-1} and divisor register X. If the signs are equal, X is subtracted from $2R_{i-1}$ ($q_i = 1$), otherwise X is added to $2R_{i-1}$ ($q_i = -1$) as described by the following algorithm:

$$q_i = \text{sign}(R_{i-1})\ \text{sign}(X).$$

Algorithm: The **Binary Non-restoring Division Algorithm**

Step0:- Calculate $Y/X = Q + (2^{-N} R_N)/X$ radix 2
Step1:- Initial remainder, $R_0 = Y$
Step2:- Perform N_Q division iterations
For i = 1 to N_Q
Calculate quotient digit
$p_i = \text{sign}(R_{i-1})\ \text{sign}(X)$
Update remainder
$R_i = 2R_{i-1} - q_i X$

III. Preliminaries [6, 7]

3.1 **Register**

Register is a special storage device for words. A binary number (0/1) is a 1-bit memory. So, n flip-flops can store an n-bit word. This combination is referred to as a register.

3.2 **Accumulator**

Accumulator is a one kind of CPU register. It is a device that combined the functions of number storage and addition. It is used to store an input or an output operand that is result in the execution of most instructions. Accumulator automatically added any transferred quantity to its previous value.

3.3 **ALU**

Arithmetic-logic Unit (ALU) executes instructions. Instructions either involve in numerical operations (arithmetic) or nonnumerical operations.

3.4 **Shifting Operation**

Shifting operation performs in register. It is useful to be able to shift the contents of a register to the left or the right. A right shift operation changes the register state as follows:
$(0, r_0, r_1, \ldots, r_{n-2}) \leftarrow (r_0, r_1, \ldots, r_{n-1})$ while a left shift performs the transformation $(r_1, r_2 \ldots, r_{n-1}, 0) \leftarrow (r_0, r_1, \ldots, r_{n-1})$ where n-bit register is an ordered set of m flip-flops used to store an n-bit word $(r_0, r_1, \ldots, r_{n-1})$. A register organized to allow left or right shift operations of this kind is called a shift register. Right shift operation obey the ruleof multiplication and left shift operation obey the rule of division.

IV. Logical approach of proposed non-restoring division algorithm

1. It is the operation of repetitive left-shifting and subtraction.
2. At first initialize the divisor, dividend, accumulator (as 0), counter variable and carry out bit (as 1). If the carry out bit is 1 then left-shift the accumulator and dividend's content and modify carry out bit and accumulator subtracting divisor from it. Else left-shift an accumulator and dividend's content and modify carry out bit and accumulator adding divisor. Then initialize the last bit of dividend as carry out bit. Then decrease the counter variable. If counter content is equal to zero then store the quotient and divi-

dend and terminate. Else repeat from the step that is left-shift the accumulator and dividend's content and modify carry out bit and accumulator subtracting divisor from it.

3. If the divisor is 'B' and the dividend is 'A' then the result in non-restoring method is (2A-B).
4. The process continues until all the bits of the dividend are exhausted.

V. Proposed non-restoring division algorithm

```
Step0:- M ← Divisor, A ← 0;          /*A is the Accumulator*/
        Q ← Dividend, Count ← n;
        C ← 1; /*C is the Carry out bit of 2A + M or 2A - M */

Step1:- If (C = = 1) then
                Left-shift (A, Q) and (C, A) ←A - M;
        Else                                  /* When C = 0 */
                Left-shift (A, Q) and (C, A) ←A + M;
        End If

Step2:- Q0 ← C;

Step3:- Count = Count - 1;

Step4:- If (Count == 0) then
                Quotient in Q and Remainder in A and go to Step5;
        Else
                go to Step1;
Step5:- Stop.
```

VI. EXAMPLE

Allowing for 8-bits ALU, then according to division hardware shown above, Divisor and Quiotent register will be 8-bits and Remainder register will be 16-bits long.

Let us divide 28 by 9 using non-restoring division algorithm.
Enter no. of Binary Bits: 8
Enter Dividend in Decimal: 28
Enter Divisor in Decimal: 9
00011100/00001001

Steps	A	Q	Operation
1:	1 00000000(0)	00011100(28)	
2:	1 00000000	0011100*	Left Shift
3:	0 11110111(247)	0011100*	Sub
4:	0 11110111(247)	00111000(56)	
5:	0 11101110	0111000*	Left Shift
6:	0 11110111(247)	0111000*	ADD
7:	0 11110111(247)	01110000(112)	
8:	0 11101110	1110000*	Left Shift
9:	0 11110111(247)	1110000*	ADD
10:	0 11110111(247)	11100000(224)	
11:	0 11101111	1100000*	Left Shift
12:	0 11111000(248)	1100000*	ADD
13:	0 11111000(248)	11000000(192)	
14:	0 11110001	1000000*	Left Shift
15:	0 11111010(250)	1000000*	ADD
16:	0 11111010(250)	10000000(128)	
17:	0 11110101	0000000*	Left Shift
18:	0 11111110(254)	0000000*	ADD
19:	0 11111110(254)	00000000(0)	
20.	0 11111100	0000000*	Left Shift
21.	1 00000101(5)	0000000*	ADD

22.	1	00000101(5)	00000001(1)	
23.	1	00001010	0000001*	Left Shift
24.	1	00000001(1)	0000001*	Sub
25.	1	00000001(1)	00000011(3)	

Quotient=3
Remainder=1

VII. Limitations

1. The dividend must be expressed as a 2n-bit 2's complement number. Thus for example, the 4 bit 0111 becomes 00000111, and 1001 becomes 11111001.
2. The divisor and the dividend both must be positive.
3. It will not give the correct answer if the divisor is negative and the dividend is positive and vice versa.
4. It will not give the correct answer if the both dividend and divisor are negative.

Future scope of non-restoring division algorithm is to divide two signed binary numbers.

VIII. Conclusion

In this work, I am trying to improve the non-restoring algorithm to minimize the hardware cost. If dividend & divisor both are negative then proposed algorithm will not work. Though, in future I can develop this algorithm to divide two signed binary numbers.

Acknowledgement

I express my sincere gratitude to **Professor Samar Sen Sarma** (Department of Computer Science & Engineering, UCSTA, University of Calcutta). He helped me immensely by providing all sorts of reference books and articles and guided me with his valuable suggestions and timely advice. This paper would have remained incomplete without his valuable feedback.

References

[1] J. E. Robertson, "A new class of digital division methods," IRE Trans, of Elec. Comp., Vol. EC-7, No.3 (Sept. 1958), pp. 218-222.
[2] Patterson, D.A and Hennessy, J.L, "Computer Organization and De- sign: The Hardware/Software Interface", 1993 San Mateo, CA: Morgan Kaufmann Publishers, Second Edition, 1998.
[3] Stallings, W, "Computer Organization and Architecture (Designing for Performance)", Upper Saddle River, New Jersey: Prentice Hall Publishers, Fifth Edition, 1999.
[4] Daniel E. Atkins, "HIGHER RADIX, NON-RESTORING DIVISION: HISTORY AND RECENT DEVELOPMENTS", Computer Arithmetic (ARITH), IEEE 3rd Symposium, pp. 158-160, 1975.
[5] Jack Bukholdt Andersen, Anders Fzrgemand Nielsen, Ole Olsen, "A Systolic ON-LINE Non-restoring Division Scheme", Proceedings of the Twenty-Seventh Annual Hawaii International Conference on System Sciences, IEEE, 1994.
[6] John P. Hayes, "Computer Architecture and Organization", Tata McGraw-Hill Book Company, 2nd Ed., 1988.
[7] Jacob Millman, Christos C. Halkias, "Integrated Electronics: Analog and Digital Circuits and Systems", Tata McGraw-Hill Book Company, 14-th Reprint, 1991.