The International Journal of Engineering
And Science (IJES)
||Volume|| 1 ||Issue|| 1 ||Pages|| 18-21 ||2012||
ISSN: 2319 – 1813 ISBN: 2319 – 1805

Multiplication another Hue And Cry

Barun Biswas[1], Samar Sen Sarma[2], Krishnendu Basuli[3]
[1]CVPR Unit, Indian Statistical Institute, 203 B.T Road Kolkata-700108, India
[2]Department of Computer Science and Engineering,
University of Calcutta, 92, A.P.C. Road, Kolkata –700 009, India.
[3]Dept. of Computer Science, West Bengal State University, India

----------**Abstract**----------

Multiplication process is not only bounded only in mathematics. It is also applied in many other fields where a sequence of summing or addition is to be applied. Multiplication is nothing but a sequence of addition performed repeatedly. Our basic approach is to consider as many bits of multiplier as possible so that the total number of partial product. In this paper we tried to implement such a technique where we can consider more than one two bits(compared to Booth's process of multiplication) of the multiplier to decrease the total number of partial product with the other operation like addition, shift and complement.

Key Words: Partial product, Shift operation, Complement operation, Booth's process of multiplication, Shift and add, repeated addition, Divide and conquer,

Date of Submission: 23, October, 2012 ⟺ Date of Publication: 10, November 2012

I. Introduction

Multiplication is not new in the field of mathematics. It was developed in ancient age. Around 18000BC it was developed [6]. Multiplication is the process of multiplying or extending some values to a desired value. For this process we need two numbers: multiplicand and multiplier. Multiplicand is the number which will be multiplied and multiplier is the number by which multiplicand will be multiplied. We know that multiplication is commutative and distributed over addition and subtraction [6].

In the field of multiplication we notice many new developed processes, many new techniques are being proposed. Many of them are widely accepted. There are many variations in the technique of multiplication. We will see shortly many of these techniques in brief.

Let see the original process of multiplication. In the multiplication process there are two main factors namely multiplicand and multiplier. The operation of multiplication is performed as a summation of multiple additions. The following process describes it in short. The result of multiplication is the total number (product) that would be obtained by combining several (multiplier) groups of similar size (multiplicand). The same result can be obtained by repeated addition. If we are combining 7 groups with 4 objects in each group, we could arrive at the same answer by addition.

For example, 4+4+4+4+4+4+4=28 is equivalent to the multiplication equation 7*4=28.

II. Previous Work

Before discussing the proposed process of multiplication let us take a look to those previous works that are available at present. To make our presentation clear it is necessary to discuss them.

A. Repeated addition multiplication: [4][5]

Among the available multiplication algorithms the most simple one is repeated addition multiplication. It has the complexity of O(n) all the time. The logic behind this process is very simple, simply decrease the multiplier by 1 and add the multiplicand to the accumulator and repeat the process until the multiplier become 0(zero). It is clear that the process will run number of times equals to the multiplier.

B. Shift and add [4][5]

The next process of multiplication is "shift and add". This process is performed on the binary number. Here the LSB of multiplier is checked, if it is 1 then multiplicand is added otherwise the previous partial product is shift one bit to the right. This process is continued until the MSB of the multiplier is checked. This is a sequential multiplication scheme with worst case time delay proportional to the numebr of bits O(n). So in this process we notice that

the pertial product of the process is less tahn the previous process. But in this process if all the multiplier bit is 1 jthen the total number of pertial product is equal to the number of bit present in the multiplier.

C. Array multiplier [4][5]

The next topic is to be discussed is Array multiplier. Checking the bits of the multiplier one at time and forming the pertial product is a shift operation that requirs a sequence of add and shift microperation. The multiplication of two binary numbers can be done with one micro operation by means of a combinational circuit that forms the product bits all at once.

Let us illustrate the process with an example, let there are two bit multiplicand b_0, b_1 and two bits multiplier $a_0 a_1$. Now the product is $c_3 c_2 c_1 c_0$ The operation is as follows

	b_1		b_0	
	a_1		a_0	
	$a_0 b_1$		$a_0 b_0$	
$a_1 b_1$	$a_1 b_0$			
c_3	c_2	c_1	c_0	

A combinational circuit binary multiplier with more bits can be constructed in a similar fashion. A bit of the multiplier is ANDed with each bit of the multiplicand in as many levels as there are bits in multiplier. Binary out put in each level of AND gate is added in parallel with partial product of the product. For j multiplier bits and k multiplicand bits wee need j*k AND gates and (j-1)*k bits adders to produce a product of j+k bits.

As our process is based on the concept of the Booth's multiplication we shall discuss the Booth's process next .

D. *Divide and conquer [2]*

In traditional process of multiplication we see that the process of multiplication of two n bit binary number require O(n^2) digit operation . but in divide-n-conquer technique the multiplication requires O(n^{log3}) or O($n^{1.59}$) (approximately) bit operation.

In this process binary number is divided into two parts: say the number is X, so the number can be represented as X= a + b. So in this process we see that there may be four n/2 bit operation but the product (say X*Y) can be represented as three n/2 bit multiplication. So in this process the complexity can be represented as O(n^{log3}) = O($n^{1.59}$).

E. Booth's Multiplication[1][3]

Booth's multiplication algorithm is a multiplication algorithm that multiplies two signed binary numbers in two's complement notation. The algorithm was invented by Andrew Donald Booth in 1951 while doing research on crystallography at Birkbeck College in Bloomsbury, London. Booth used desk calculators that were faster at shifting than adding and created the algorithm to increase their speed. Booth's algorithm is of interest in the study of computer architecture

Booth's algorithm involves repeatedly adding one of two predetermined values Multiplicand and Multiplier to a product, then performing a rightward arithmetic shift on product. Let **M** and **Q** be the multiplicand and multiplier, respectively; and let x and y represent the number of bits in **m** and **r**. Then the number of bit in the product is equal to the (x+y+1).

The Booth's multiplication process is based on the four basic steps. In this process the two multiplier bit is checked and depending on their combination (2^2=4) four different steps is performed. Now let us discuss the Booth's multiplication by taking an example.

Let M(multiplicand) = 000000000001101

Q(multiplier) =0000000001010101

M =0000000000001101

Q =0000000001010101
=1111111111110011 // multiplier bit 10
=000000000001101* // multiplier bit 01
=11111111110011** // multiplier bit 10
=0000000001101*** // multiplier bit 01
=111111110011**** // multiplier bit 10
=00000001101***** // multiplier bit 01
=1111110011****** // multiplier bit 10
=000001101******* // multiplier bit 01
=0000010001010001 // product

In the above example we notice that there are three operations performed, namely shift, addition, complement.
The number of operation is as follows
Shift = 7
Addition= 7
Complement= 4.

I. MOTIVATION

One of the techniques of multiplication is Shift and Add multiplication. We observed the modified Shift and Add process by Feynman. This technique is illustrated in below to clear the topic.

Let Multiplicand(M) is 22(10110) and multiplier(Q) is 5(101). So the multiplication process is as follows

M	=10110
Q	=101
	10110
	10110**
	1101110

In the above process we notice that the process of multiplication is a Shift and Add multiplication process, but when a 0(zero) occur the shift as twice. So here is the logic that when there are one or more than one 0s we can take them as one sequence of 0s.We can apply this logic to some other algorithm to make that algorithm more efficient.

II. PROPOSED PROCESS

In the previously discussed process we notice one thing that when there is a sequence of 0's we consider each 0(zero) for one single step. Here for this reason the number of steps increases each time. So our observation is that if we can consider a sequence of zeros in a single step then the whole number of steps decreases. One more thing when there a sequence of 1's we can take the sequence as one step. We notice this process can be applied to Booth's algorithm to make it more efficient.

The proposed algorithm is as follows

Algorithm:

While starting this algorithm we will take one thing in consideration that one extra 0 is added as the LSB. Three bits will be considered at a time in general and the last bit from previous bit pattern will be considered.

Step 1. If the bits are 000 then continue checking the bits until the bit is 1. Then shift the partial product to the right same as the number of the 0s.

Step 2. If the bits are 001 then continue the checking the next bit until it is 1. Then add multiplicand and shift the partial product to the right as the number of 0s.

Step 3. If the bits are 010 then subtract multiplicand and shift the partial product three positions right.

Step 4. If the bits are 101 then add multiplicand and shift the partial product three positions right.

Step 5. If the bits are 110 then check the next bits until it is 0 and subtract the multiplicand and shift the partial product to the right as the number of 1s.

In the proposed process we have seen a new process of multiplication. Now let us take an example to illustrate the process and show how efficient this process is. The example is given below.

Let M=10 and Q=10011

So according to the proposed process the multiplication of the numbers is as follows

M	=00000010	
Q	=00010011	
	=11111110	//multiplier bits 110
	=000010**	// multiplier bits 001
	=1110****	//multiplier bits 010
	=010*****	//multiplier bits 001
	=00100110	//final product

So in the illustrated process we notice that a big advantage is achieved. If the booth process is applied in the same multiplier (Q) the number of partial product, shift operation and complement operation are as follows

Partial product =4
Shift operation =5
Complement operation=2

Whereas the same operations in the proposed is as follows

Partial product =3
Shift operation =5
Complement operation=2

In the Booth's process when there is a long sequence of 0's or 1's the whole sequence is treated individually, whereas in our proposed the whole sequence is treated at a single step. As a result the number of steps is decreased. It is easily understandable that if the numbers of steps decrease the space complexity also decreased. And also it is noticeable that as the number of bits decreased the partial products also decrease. Here is the main goal of the proposed process.

If we consider chips to perform these operations then we can see that the time required for shift operation (4 bit bidirectional Universal shift register LS 194) is 1/36 ns [7] [8], add operation (4 bit carry look ahead adder LS 7483) is 45 ns [7] [8] and for complement operation (LS hex inverter 7404) is 12 ns [7] [8]. So here we can make an assumption that how much time is saved if we apply this process of multiplication instead of others.

III. Future Scope

In the proposed process of multiplication we notice that if we consider the consecutive 0's or 1's in a single step then the number of partial product decreases also the other operations like shift operation and complement operation. So here the total number of other operations also decreases and hence the space and time complexity of the whole operation also decreases. We notice that the divide and conquer process can also decreases the time and space required for an operation. So we will try to merge the proposed process of multiplication with the divide and conquer technique so that the total number of operation with the time and space complexity can be decreased. We notice in different case where divide and conquer process technique were applied and the result were as expected. We think in our case it would not be an exception.

IV. Conclusion

In our proposed process we notice that some time it seems that considering the sequence of 0's and 1's at a single step may increase the maintenance complexity and over burden the whole process. But when the proposed process is in real the process is a big advantage over the existing process of multiplication.

REFERENCES

[1] **A.** Booth, **"A** signed binary multiplication technique," *Q.* J. ***Me& Appl.*** *March.*, vol. 4, pp. **236-240,** 19.51.

[2] Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullmant "The design and analysis of computer algorithm" Pearson, 1974.

[3] Booth Multiplication Algorithm Abenet Getahun Fall 2003 CSCI 401

[4] John P. Hayes "Computer architecture and organization", Second Edition, McGraw-Hill, 1988.

[5] M. Morris Mano "Computer System Architecture", Third Addition, Prentice Hall, 1993.

[6] http://en.wikipedia.org/wiki/Multiplication, visited at 12 nov. 2012

[7] William D.Anderson, Roberts Loyd Morris, John Miller "Designing with TTL integrated circuit", McGraw-Hill, 1971

[8] "TTL data book for design Engineer", 2nd edition, Texas instrument Inc, 1981

Barun Biswas
CVPR Unit, Indian Statistical Institute,
203 B.T Road Kolkata-700108, India

Krishnendu Basuli
Assistant Professor, Department of Computer Science, West Bengal State University

Dr. Samar Sen Sarma
Professor, Department of Computer Science and Engineering, University Of Calcutta, India