International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-9 Issue-4, February 2020

# A Multiplication Algorithm

**S Subha**

***Abstract*: Paper Setup must be in A4 size with Margin: Top 0.7", Bottom 0.7", Left 0.65", 0.65", Gutter 0", and Gutter Position Top. Pap Abstract: Multiplication is common arithmetic operation in ALU. Many algorithm are proposed for multiplying two unsigned numbers in literature. This paper proposes algorithm to multiply two unsigned binary numbers of any size. The most significant two bits are used to determine the partial product by bit inspection. The rest of partial products are obtained by suitably shifting the previous partial products and adding the terms involving remainders. The remainder is obtained by taking one bit at a time from the MSB-2 position assuming numbers are indexed from zero in LSB to maximum-1 in MSB. The multiplication process is performed as series of additions, shifts in this method. The proposed method is simulated in Quartus2 Toolkit. It is compared to the in-built multiplication process of the tool. A timing improvement of 9.5% with comparable power consumption is obtained with same pin count.***

***Keywords* : ALU, Multiplication algorithm, performance, power consumption***

## I. INTRODUCTION

The ALU of the computer has multiplication circuits. The multiplication of two binary numbers is performed in these circuits. There are various algorithms for computing the product of two unsigned numbers. Many algorithms to perform the multiplication of two numbers in various notations are proposed in literature [2]. Some of them are modified Booth's algorithm for two's complement numbers [3, 4], Dadda multiplier [1], and Wallace Tree multiplier [6]. An algorithm for multiplication is proposed in [5] where the input is taken two bit slices for calculation. Multiplication is used in various applications like DSP, image processing. This paper proposes multiplication algorithm. The algorithm is based on the mathematical concept of modulus. The input numbers has bit positions MSB , MSB-1, MSB-2,.., and LSB. The algorithm computes the partial product of two MSB bits of the two numbers. The rest of partial products are obtained by shifting the previous partial product and adding the shifted remainders of two numbers based on the value of bits in two inputs from MSB-2 position. The algorithm is simulated with Quartos 2 Toolkit. The results are compared to the multiplication algorithm in the tool. An area increase with comparable power consumption and performance improvement of 9.5% is observed for the chosen parameters.

The rest of the paper is organized as follows. Section 2 gives the mathematical background, section 3 proposed model, section 4 simulations, and section 5 conclusion followed by references.

**Revised Manuscript Received on February 28, 2020.**
* Correspondence Author
**S Subha***, SITE, Vellore Institute of Technology, Vellore, India Email: ssubha@rocketmail.com



## II. MATHEMATICAL BACKGROUND

Consider two numbers A and B. Let both be n bits in length. From mathematics the following is observed

$A = 2\frac{A}{2} + X_1$ and $B = 2\frac{B}{2} + Y_1$ where . $X_1 and Y_1$ are bits.

$$AB = 4\frac{AB}{4} + X_1Y_1 + 2\left(\frac{AY_1}{2}_1 + \frac{BX_1}{2}\right) \quad (1)$$

The term $\frac{AB}{4}$ can again be expressed in equations as in (1). This process can be repeated till the product of two bits of A and two bits of B can be determined. The final answer is obtained by substitution in the expressions. The whole process of computing the product is thus a series of shifts, additions.

## III. PROPOSED ALGORITHM

Consider the two input numbers a [5:0] and b [5:0]. The following algorithm computes the product of the two numbers.

**Algorithm SS_Multiply:** Given two six bit numbers, this algorithm computes the product

1. Start
2. Let Ra = a[3], Rb = b[3], tempa = a[5:4] tempb = b[5:4]
3. Let prod1 = 4*(a [5:4]*b [5:4]). This is computed by bit inspection of the inputs.
4. Let prod1 = prod1 + Ra*Rb + 2(tempa * Rb + tempb * Ra)
5. Let Ra = a[2], R[b] = b[2], temp = prod1
6. Let prod2 = 4*prod1 + Ra*Rb + 2*(Ra*b[5:3] + Rb*a[5:3])
7. Let Ra = a[1], Rb = b[1], temp = prod2
8. Let prod3 = 4*prod2 + Ra * Rb + 2*( Ra*b[5:2] + Rb*a[5:2])
9. Let Ra = a[0], Rb = b[0], temp = prod3
10. Let prod4 = 4*prod3 + Ra * Rb + 2*( Ra*b[5:1] + Rb*a[5:1])
11. Print prod4
12. Stop

The above algorithm can be extended for any input length. The multiplication by four is obtained by left shifting twice the input. The time complexity of the algorithm depends on the input length. It is O (n) for two n bit inputs.

Consider the following example. Let A = 101010 B = 110110. The following gives the stepwise algorithm implementation.

1. Start
2. Let Ra = a[3]=1, Rb = b[3]=0, tempa = a[5:4]=10 tempb = b[5:4]=11
3. Let prod1 = 4*(a [5:4]*b [5:4]. This is 4*0110 = 011000. This is computed by bit inspection of the inputs.
4. Let prod1 = prod1 + Ra*Rb + 2(tempa * Rb + tempb * Ra) = 011000 + 0 + 2(11). This is 011000 + 110 = 011110.

5. Let Ra = a[2] = 0, R[b] = b[2]=1, temp = prod1 = 011110
6. Let prod2 = 4*prod1 + Ra*Rb + 2*(Ra*b [5:3] + Rb*a [5:3]). This is 4*011110 + 0 + 2*101 = 01111000 + 1010 = 10000010
7. Let Ra = a [1] =1, Rb = b [1] =1, temp = prod2. Thus prod2 = 10000010
8. Let prod3 = 4*prod2 + Ra * Rb + 2*(Ra*b [5:2] + Rb*a [5:2]). Thus prod3 = 4*10000010 + 1 + 2*(1101 + 1010) = 1000001000 + 1 + 101110 = 1000110111
9. Let Ra = a[0] = 0, Rb = b[0] = 0 temp = prod3 = 1000110111
10. Let prod4 = 4*prod3 + Ra * Rb + 2*(Ra*b [5:1] + Rb*a [5:1]). Thus prod4 = 100011011100 + 0 + 0 = 100011011100.

The final answer is 100011011100.

## IV. SIMULATIONS

The proposed model is simulated using Quartus2 Toolkit. The proposed model is called prop in this discussion. The model is compared with the in-built multiplication algorithm in the Quartus2 Toolkit. This model is called trad in this discussion. The simulation parameters are shown in Table 1. Routines in Verilog were written for prop and trad models.

**Table I Simulation Parameters**

| Parameter | Value |
|---|---|
| Processor family | CycloneII |
| Auto device select | Fitter |
| Package | FBGA |
| Pin count | 484 |
| Speed grade | Fastest |

The prop model compared with trad model results are shown in Table2

**Table II Simulation Results**

| parameter | prop | trad | %improve |
|---|---|---|---|
| time | 8.179ns | 9.042ns | 9.544349 |
| logic elements | 141 | 1 | |
| pins | 25 | 25 | |
| power | 69.58mW | 69.51mW | -0.1007 |

As seen from Table 2 there is performance improvement with comparable power consumption.

## V. CONCLUSION

A multiplication algorithm involving shifts, additions is proposed in this paper. . The algorithm has time complexity of input length and is scalable. The proposed algorithm is simulated using Quartus2 Toolkit. Simulation results show improved performance of proposed model compared with trad model by 9.5% with comparable power consumption.

## REFERENCES


1. Dadda, Luigi, "Some schemes for parallel multipliers", *Alta Frequenza*, Vol. 34, No. 5 pp. 349-356, May, 1965
2. Israel Koren, "Computer Arithmetic Alogrithms", Printice Hall, NJ, 1993
3. Philip E Madrid, Brian Millar, Earl E Swartzlander Jr., "Modified Booth Algorithm for High Radix Floating point Multiplication", *IEEE Transactions on VLSI Systems*, Vol. 1 No. 2, June 1993, pp. 164-167
4. Rajendra Katti, "A Modified Booth Algorithm for High Radix Fixed Point Multiplication", *IEEE Transactions on VLSI Systems*, Vol. 2, No. 4, Dec. 1994, pp. 522-524
5. S.Subha, R.Sakthivel, "A Power Saving Multiplication Algorithm", IJAER, Vol. 11, No. 9, May 2016, pp. 6200-6203
6. Wallace, Christopher Stewart, *"A suggestion for a fast multiplier"*. *IEEE Transactions on Electronic Computers. EC-13 (1): 14–17*