DESIGN OF HIGH SPEED MULTIPLIER ARCHITECTURE WITH REDUCED COMPLEXITY

[1]Dr. S. Aruna Masthani, [2]S. Kannappan

[1]Assistant Professor, JNTUA College of Engineering, Anantapuramu (India),

[2]Research Scholar, JNT University, Anantapuramu (India)

ABSTRACT

Multiplier is the crucial operation in many processors. For performing multiplication of higher word size operands, the design with less circuit complexity and improved speed performance is a challenging task for the designer. An 8-bit unsigned multiplier is proposed by using two 4-bit multipliers. The basic 4-bit multiplier is designed to produce 8-bit product without generating any partial products. To get 16-bit product of 8-bit multiplier, the products of two basic 4-bit multipliers are added with proper shift operations. By reusing the basic 4-bit multiplier the delay has reduced greatly instead of designing circuit for whole 8-bit multiplier.

Keywords: Multiplying circuits, Power dissipation, Adders

I. INTRODUCTION

In Computing especially digital signal processing, the multiply–accumulate operation is a common step that computes the product of two numbers and adds that product to an accumulator. Multiplication is basically a shift add operation. There are, however, many variations on how to do it. Some are more suitable for FPGA use than others. All the logic multipliers are complicated and also have more delay. Some of the multipliers are not suitable to implement using some FPGAs. To increase the speed of operation LUT based multipliers are used. It's very fast because is just a memory access.

1.1 Look-Up Table (LUT) Multiplier:

With definite outputs for every input, is called a look-up table, because the memory device simply" looks up" what the output should to be for any given combination of inputs states. Look-Up Table multipliers are simply a block of memory containing a complete multiplication table of all possible input combinations. The large table sizes needed for even modest input widths make these impractical for FPGAs. In all the above conventional LUT based multipliers X is an L bit input multiplicand that is multiplied with a constant coefficient A and the product is stored in the address which is pointed by input X itself. If the operand width increases then the memory size will increase exponentially. The first and efficient implementation for LUT optimization was presented by P. K. Meher. Based on this method [1], [2] the LUT size is reduced by a factor of two for a given input X of L bits. By using this approach, the LUT size can be reduce to half but complex overhead increases. In the APC-OMS [3] method by applying simple modifications to the APC, OMS and combining these two techniques the LUT size is reduced by a factor of four.

All these LUT size optimization techniques are for constant coefficient multiplication. It is only useful in particular applications to multiply the signal with constant coefficients. To perform multiplication for variable operands with increased speed a new combinational circuit is designed with less computational complexity and high speed performance. In LUT multipliers the odd products of second operand has to be stored in LUT. So, it is a constant coefficient multiplier. For each coefficient we need a LUT. To avoid usage of memory we simply calculating product by using the algorithm APC-OMS that was used in LUT multiplier, a new multiplier circuit is proposed which generate odd products adaptively and provide original product for the given two numbers without generating any partial products. It is also applicable for variable operands (any two operands).

II. A NEW HIGH SPEED 8- BIT MULTIPLIER

In this 8-bit multiplier, the multiplier operand B (7 down to 0) is divided into two groups' i.e. higher nibble and lower nibble. The multiplication of multiplicand B (7 down to 0) with these two 4 bit numbers are performed by using basic four bit multiplier design (design is independent of B)to get two separate products. To get final 16 bit product the higher nibble product is left shifted by four times and then added with lower nibble product. The block diagram for the proposed multiplier design is as shown in figure1.

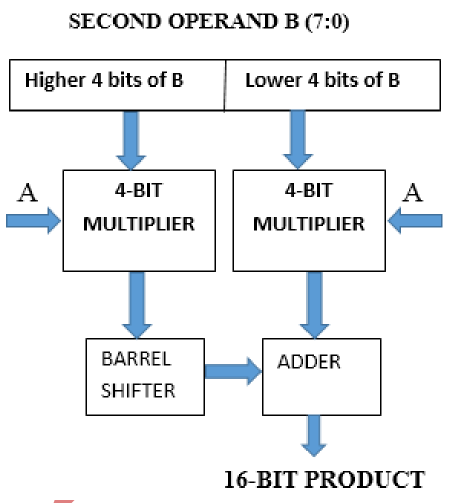

Fig.1: 8-bit unsigned multiplier using two 4-bit multiplier

The four bit multiplier is the basic multiplier which is shown in the figure.2. This 4-bit multiplier performs shift and add operations on second operand based on the control bits generated from first operand. This is performed at a time i.e. it calculates number of shifts and addition for the operand one at a time rather than the conventional multiplier which controls the shift and addition for individual bits in first operand. The proposed 8-bit multiplier consists of an 8 bit barrel shifter, two 16-bit add/sub and a control circuit. The control circuit generates required number of shifts, control signal to decide either addition or subtraction and to enable or disable the add/sub circuit as shown in figure 2. The control signals are derived by using the following equations. Number of shifts is given by

$$:s_1 = (b_3 \odot b_2) + (b_2\overline{b_1}\ \overline{b_0}) \quad (1)$$

$$:s_0 = (b_3 \odot b_1)\left(\overline{b_2} + b_0\right) + \overline{b_3}b_2b_1\overline{b_0} \quad (2)$$

Additional control signals to control the adder are

$$: m = \overline{b_0}\,\overline{b_1}\left(\overline{b_3} + b_2\right) + b_3\overline{b_2}(b_1 \oplus b_0) + \overline{b_3}b_2b_1 \tag{3}$$

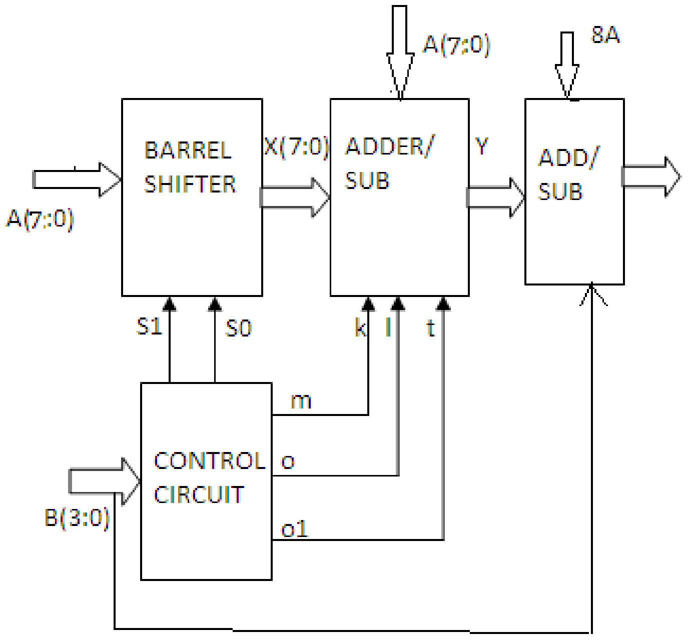

Fig. 2: Block diagram of proposed 4x4 unsigned multiplier

If $m = 0$ then adder will add the output of barrel shifter to operand A. If $m = 1$ then adder will be disabled and it will pass the input of adder to its output without any addition or subtraction. i.e output of barrel shifter is send to final add/sub unit to get final product. To get 6A, barrel shifter adder is added with 2A. This is only for B=0010 and B= 1110,to enable this addition $o = 1$.

$$: o = \overline{b_3}\,\overline{b_2}b_1\overline{b_0} + b_3b_2b_1\overline{b_0} \tag{4}$$

Subtraction is performed for

$$: o1 = \overline{b_2}\,\overline{b_1}(b_3 \oplus b_0) + b_3b_2b_1b_0 \tag{5}$$

The outputs of the add/sub block are from 0A to 7A for corresponding pairs of 2's complemented numbers which is given to final add/sub unit. Final add/sub unit add its input to 8A or subtract from 8A based on MSB bit of operand B. i.e.

: If $b_3 = 1$ then $product = 8A + Y$

: If $b_3 = 0$ then $product = 8A - Y$

The complete behavior of the design is represented by the following truth table.

TABLE 1: Truth table for proposed 4x4 unsigned multiplier

B(3:0)	**B(3:0)**	**SHIFTS**		**Y**	**PRODUCT**	
		S1	**S0**		$b_3 = 0$ **(8A-Y)**	$b_3 = 1$ **(8A+Y)**
0000	0000	1	1	8A	0	--
0001	1111	1	1	7A	A	15A
0010	1110	1	0	6A	2A	14A
0011	1101	1	0	5A	3A	13A
0100	1100	1	0	4A	4A	12A
0101	1011	0	1	3A	5A	11A
0110	1010	0	1	2A	6A	10A
0111	1001	0	0	A	7A	9A
1000	1000	0	0	0	--	8A

Example: 11011010(218) * 10101101(173) = 1001001101010010(37714)

One of the operands is divided into two 4 bit group. Here 110111010 is divided into 1101 and 1010. By using basic block, the intermediate products are 1101*10101101=100011001001 and 1010*10101101=11011000010. First product is left shifted by four times i.e. 1000110010010000. And it is added with the second product. i.e., 1000110010010000+11011000010= 1001001101010010. This is the required final product.

III. RESULTS

The proposed design is simulated using Xilinx ISE 12.1 and synthesized using XST synthesis tool. The behavioral simulation of 8 bit multiplier is as shown in figure 3. And Top module of the design and RTL schematic are shown in figures 4.

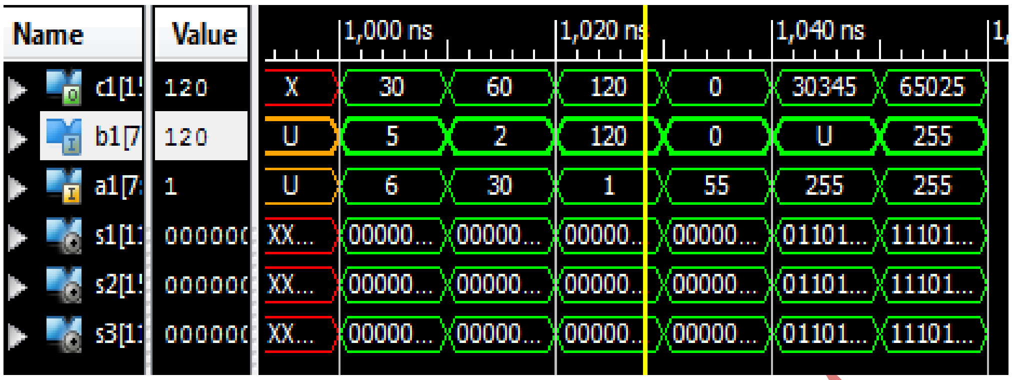
Name
Value
1,000 ns
1,020 ns
1,040 ns
c1[15
120
X 30 60 120 0 30345 65025
b1[7
120
U 5 2 120 0 U 255
a1[7
1
U 6 30 1 55 255 255
s1[11
000000
XX... 00000... 00000... 00000... 00000... 01101... 11101...
s2[15
000000
XX... 00000... 00000... 00000... 00000... 01101... 11101...
s3[11
000000
XX... 00000... 00000... 00000... 00000... 01101... 11101...

Figure.3: Behavioral Simulation of 8 bit multiplier

3.1. SYNTHESIS REPORT:

All values displayed in nanoseconds (ns)

===

Timing constraint: Default path analysis

Total number of paths / destination ports: 232520 / 16

Delay: **17.681ns (Levels of Logic = 22)**

Source: b1<3> (PAD)

Destination: c1<15> (PAD)

Data Path: b1<3> to c1<15>\

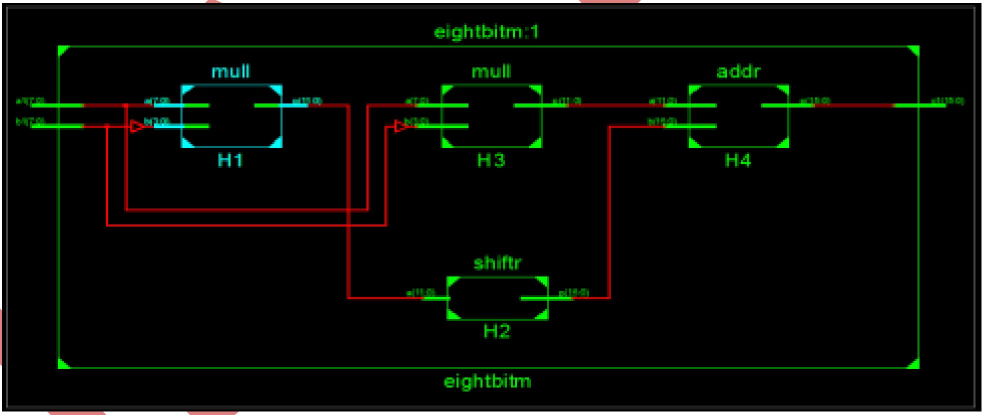

Figure.4: RTL Top module schematic

IV. CONCLUSION

A new design of combinational shift/add multiplier is designed with improved speed performance than conventional binary array multiplier using a simple algorithm. Using Xilinx ISE 12.1 tool, it was observed that this new 8x8 multiplier has less delay and less complexity in algorithm than conventional binary multiplier. And also it has less delay than the well-known Urdhva Trikbhyam (Vedic multiplier [4]). It can be possible to achieve less delay for large operand sizes by reusing the same design for required number of times with little circuit modifications.

REFERENCES:

[1] P. K. Meher, "New approach to LUT implementation and accumulation for memory-based multiplication," in *Proc. IEEE International Symposium on Circuits and Systems, ISCAS '09*, May 2009, pp. 453–456.

[2] P. K. Meher, "New look-up-table optimizations for memory-based multiplication," in *Proc. 12th International Symposium on Integrated Circuits, ISIC '09*, Dec. 2009, pp. 663–666.

[3] P. K. Meher, "New approach to look-up-table design and memory-based realization of FIR digital filter," *IEEE Transactions on Circuits & Systems-I:Regular Papers*, vol. 57, no. 3, pp. 135–139, Mar. 2010.

[4] Swami Sri Bharati Krisna Jagadguru Tirthaji Maharaja, "Vedic Mathematics or Sixteen Simple Mathematical Formulae from the Veda, Delhi (1965), Motilal Banarsidas, Varanasi, India, 1986.

[5] Shaik Nasar, K. Subbarao, "Design & Implementation of MAC Unit Using Reversible Logic," International Journal of Engineering Research and Applications Vol. 2, Issue5, pp. 1848-1855, 2012.

[6] R Dhanabal,V Bharathi,Anand N, George Joseph, Suwin Sam Oommen, Dr Sarat Kumar "Comparison of Existing Multipliers and Proposal of a New Design for Optimized Performance" R Dhanabal et al. / International Journal of Engineering and Technology (IJET), ISSN : 0975-4024, Vol 5 No 2 Apr-May 2013

[7] A. Anand Kumar, "Fundamentals of Digital Circuits," pp. 242-245 PHI Learning Private Ltd. New Delhi, 2008.