Modeling $n \times n$ Bit Multiplication Blocks for DSP Applications Using VHDL

Siddika Berna Örs and Ahmet Dervisoglu

Istanbul Technical University, Electrical and Electronics Engineering Faculty,

Electronics and Communication Engineering Department 80626 Maslak, Istanbul, TURKEY

Tel: 90 212 285 36 12, 90 212 285 36 02, Fax: 90 212 285 36 79,

e-mail: berna@ehb.itu.edu.tr, dervis@ehb.itu.edu.tr

The Author to contact: Siddika Berna Örs

Topic Areas of the paper: Component modeling, DSP processors

Modeling $n \times n$ Bit Multiplication Blocks for DSP Applications Using VHDL

Abstract

In this paper, we propose two models of multiplication blocks by using VHDL. The algorithms that are used for writing the models are suitable for high speed multiplication and have regular cellular array structures. We have simplified some equations given in the references and then have written the VHDL model accordingly. Thus, a circuit synthesized by using the models proposed in this paper will have less area and gate count on the longest path than those given in the literature. We propose formulas for calculating area values and the gate count on the longest path of the circuits for both of multiplication blocks for any value of n for, $2 \leq n \leq 54$. Also we propose a method to determine the types of gates on the longest path of the circuit.

1 Introduction

Multiplication is one of the basic operations in digital signal processing systems providing filtering, correlation and range measurement. Various multiplication algorithms have been proposed and practically used [1, 2, 3, 4].

In this paper, we propose five models of multiplication blocks by using VHDL. The algorithms that are used for writing the models are suitable for high speed multiplication and have regular cellular array structure [1, 2, 3, 4] .

In literature all the equations for implementing some blocks of the multiplier circuits are given for general situation. In this paper, we have simplified some equations given in the references for some special cases and then have written the VHDL model accordingly. Therefore, a circuit synthesized by using the models proposed in this paper will have less area value and gate count on the longest path than those given in the literature.

The area values are given as proportional to n^2or n and the gate counts on the longest paths of the circuits are given as proportional to $log_2 n$ or n in references that the algorithms are proposed. In this paper, we propose formulas for calculating area values and the gate count on the longest path of the circuits depending on n for all of five multiplication blocks. Also we propose a method to determine the types of gates on the longest path of the circuits. By using the formulas about the area and the gate count on the longest path of the circuits, we have calculated these critical values of the circuits for different n values and have given the results in two tables. So a DSP chip designer can use these tables to decide about the multiplication block which meets the requirements. In literature, to our knowledge, VHDL models of the circuits are given for certain n values; we have written the VHDL models of the circuits as taking the n as a parameter: n can take any value for $2 \leq n \leq 54$.

2 VHDL Models of the Multiplier Blocks

2.1 Computation Model

We will be concerned with modeling of a multiplier as a combinational circuit. We assume that fan-in (in degree) of each computation element is restricted in a certain constant, but fan-out (out degree) is not restricted. Because all the multiplication blocks have regular cellular array suitable for VLSI implementation and all the connections between sub-blocks are short, we assume that wires have no delay to simplify the evaluation. According to these assumptions, the computation time on a combinational circuit is linearly proportional to the depth of it. The depth of a combinational circuit is the number of computation elements on the longest path, i.e., the path that has the most computation elements in it. We estimate the complexity of a combinational circuit by its depth and gate count. The gate count (the size) of a combinational circuit is the number of combination elements in it.

2.2 High-Speed Multiplier Using a Redundant Binary Adder Tree

To write the model of a high-speed multiplier using a redundant binary adder (RBA) tree the algorithm given in [1] is used. In the following, the algorithm and the VHDL models of blocks used for every step of the algorithm are given.

Step 1. Convert the multiplicand A and the multiplier B into equivalent redundant binary integers.

When A and B are converted into RB representation, we have written the model as to store A in two different vectors A_+ and A_- and B in two different vectors B_+ and B_-. When converting a number from two's complement representation to RB representation, if the most significant bit (MSB) is 1, it is converted to -1. No operation is done on the other bits of the number. 1, 0, -1 are represented by (1,0), (0,0) and (0,1) respectively. We have used the vectors A_+ and B_+ to store the first value of the pairs, A_- and B_- to store the second value of the pairs. So no gates are used in the circuit which converting the numbers from two's complement representation to RB representation. The area is "0" and the latency is "0".

Step 2. Generate n n-digit partial products PP_i $(i = 0, 1, \ldots, n-1)$ for n multiplier digits.

Because of the PP's are represented in RB representation, we have written the VHDL model as generating one "+" and one "$-$" vectors for each PP . The truth table for kth elements of "+" and "$-$" vectors of PP_i according to ith element of B and kth element of A is given in Table 1.

Table 1: The truth table for kth elements of "+" and "$-$" vectors of PP_i according to ith element of B and kth element of A

$A_+(k)$	$A_-(k)$	$B_+(i)$	$B_-(i)$	$PP_{i+}(k)$	$PP_{i-}(k)$
*	*	0	0	0	0
0	0	*	*	0	0
0	1	0	1	1	0
0	1	1	0	0	1
1	0	0	1	0	1
1	0	1	0	1	0

In the following, the equations for calculating the values of the pair for kth digit of PP_i are given.

$$PP_{i+}(k) = A_-(k) \cdot B_-(i) + A_+(k) \cdot B_+(i); \quad PP_{i-}(k) = A_+(k) \cdot B_-(i) + A_-(k) \cdot B_+(i) \tag{1}$$

Some digits of the vectors used to represent the multiplicand and the multiplier are always zero. By using this property we have simplified the circuit for producing the PP's. If $i = 0, 1, \ldots, n-2$ and $k = 0, 1, \ldots, n-2$, kth digit of the vectors of PP_i are as follows.

$$PP_{i+}(k) = A_+(k) \cdot B_+(i); \quad PP_{i-}(k) = 0 \tag{2}$$

If $i = 0, 1, \ldots, n-2$ and $k = n-1$, $(n-1)th$ digit of the vectors of PP_i are as follows.

$$PP_{i+}(n-1) = 0; \quad PP_{i-}(n-1) = A_-(n-1) \cdot B_+(i) \tag{3}$$

If $i = n-1$ and $k = 0, 1, \ldots, n-2$, kth digits of the vectors of PP_{n-1} are as follows.

$$PP_{(n-1)+}(k) = 0; \quad PP_{(n-1)-}(k) = A_+(k) \cdot B_-(n-1) \tag{4}$$

If $i = n-1$ and $k = n-1$, $(n-1)th$ digit of the vectors of PP_{n-1} are as follows.

$$PP_{(n-1)+}(n-1) = A_-(n-1) \cdot B_-(n-1); \quad PP_{(n-1)-}(n-1) = 0 \tag{5}$$

According to (2), (3), (4) and (5) there is one two input AND (AND2) gate on the longest path of the circuit designed for generating the RB partial products. So the latency of the circuit is a constant which is independent of the value of n. If the area of two input NAND (NAND2) gate is unit then the area of the circuit is $1.5n^2$ units.

Step 3. Add up the partial products by means of a binary tree of redundant binary adders and obtain the product S_+ and S_-. Perform all redundant binary additions at each level in the tree in parallel.

Additions are performed by a tree formed by RBAs, an example for addition tree used for multiplying 16-bit numbers is shown in Fig. 1.

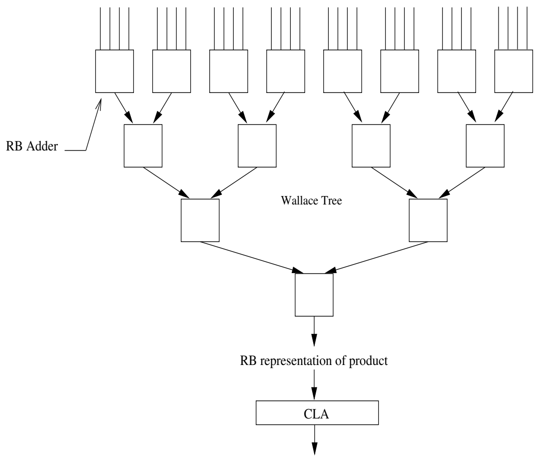

Figure 1: Adder tree for addition of partial products

Addition of two redundant binary numbers is performed in a constant time. Therefore, the computation at each level in the tree in Step 3 is performed in a constant time independent of n. A RBA cell is designed to perform the following equations [2].

$$d_i^+ = \overline{\alpha_i} \cdot \beta_{i-1}; \quad d_i^- = \alpha_i \cdot \overline{\beta_{i-1}}; \quad \alpha_i = \ell_i \oplus h_{i-1}; \quad \beta_i = \overline{\ell_i} \cdot k_i + \ell_i \cdot \overline{h_{i-1}}; \quad \ell_i = (a_i^+ + a_i^-) \oplus (b_i^+ + b_i^-);$$
$$h_i = a_i^- + b_i^-; \quad k_i = a_i^+ + b_i^+ \tag{6}$$

The cell designed to perform the above equations is used for the addition of two digits of two RB numbers that are not least significant bit (LSB). There are five two input OR (OR2) gates, four AND2 gates, four INVERTER (INV) gates and two XOR gates in the cell. The area of the cell is 19.5 units. There are two AND2 gates, one XOR gate, two INV gates and two OR2 gates on the longest path. The β and h used in the addition of LSB of the partial products are "0". Then we have simplified (6a-g) as follows.

$$d_0^+ = 0; \quad d_0^- = \ell_0; \quad \beta_0 = k_0 + \ell_0; \quad \ell_0 = (a_0^+ + a_0^-) \oplus (b_0^+ + b_0^-); \quad h_0 = a_0^- + b_0^-; \quad k_0 = a_0^+ + b_0^+ \tag{7}$$

There are five OR2 gates and one XOR gate in the circuit designed for addition of LSB of two partial products. The area of the circuit is 9.5 units instead of 19.5 units. After adding the MSBs of two partial products there is one more addition operation that is performed according to the following equations. Because of $a_n^- = b_n^- = a_n^+ = b_n^+ = 0$ we have simplified (6) for this situation also.

$$d_n^+ = \overline{h_{n-1}} \cdot \beta_{n-1}; \quad d_n^- = h_{n-1} \cdot \overline{\beta_{n-1}} \tag{8}$$

There are two INV gates and two AND2 gates in the circuit designed to perform above equations, so the area of the circuit is 4 units instead of 19.5 units. Because all addition operations are done in parallel the longest path is the same as the longest path on the circuit designed to perform the addition of the digits except LSB. There are $log_2 n$ levels in the addition tree. So there are (2 AND2+ XOR+2 INV+2 OR2)$\times log_2 n$ gates on the longest path of the addition tree. The area values of the RBA circuits on every level of the addition tree are given in Table 2.

Table 2: The area values of the RBA circuits on every level of the addition tree

Level Number	Bit number of the RBA	Area Of the RBA
1	$n+1$	$19.5n-6$
2	$n+3$	$19.5n+33$
3	$n+6$	$19.5n+91.5$
4	$n+11$	$19.5n+189$
5	$n+20$	$19.5n+364.5$
6	$n+37$	$19.5n+696$

As shown in Table 2 the area of RBA circuits are proportional to n. The area of the addition tree according to the value of n is given in Table 3.

Table 3: The area of the addition tree according to the value of n

n	Area of Addition Tree
2	33
3	144
4	255
$5 \le n \le 8$	$(19.5n-6)\left\lfloor\frac{n}{2}\right\rfloor + (19.5n+33)\left\lfloor\frac{n}{4}\right\rfloor + 19.5n+91.5$
$9 \le n \le 16$	$(19.5n-6)\left\lfloor\frac{n}{2}\right\rfloor + (19.5n+33)\left\lfloor\frac{n}{4}\right\rfloor + (19.5n+91.5)\left\lfloor\frac{n}{8}\right\rfloor + 19.5n+189$
$17 \le n \le 32$	$(19.5n-6)\left\lfloor\frac{n}{2}\right\rfloor + (19.5n+33)\left\lfloor\frac{n}{4}\right\rfloor + (19.5n+91.5)\left\lfloor\frac{n}{8}\right\rfloor + (19.5n+189)\left\lfloor\frac{n}{16}\right\rfloor +$
	$19.5n+364.5$
$33 \le n \le 54$	$(19.5n-6)\left\lfloor\frac{n}{2}\right\rfloor + (19.5n+33)\left\lfloor\frac{n}{4}\right\rfloor + (19.5n+91.5)\left\lfloor\frac{n}{8}\right\rfloor + (19.5n+189)\left\lfloor\frac{n}{16}\right\rfloor +$ $(19.5n+364.5)\left\lfloor\frac{n}{32}\right\rfloor + 19.5n+696$

Step 4. Convert S_+ and S_- into equivalent binary number P. (P is product.)

The conversion is performed by subtracting S_- from S_+ by a carry look ahead adder (CLA). The computation time is proportional to $log_2 n$ and the gate count of CLA is proportional to n.

Total computation time for multiplication is proportional to $log_2 n$ and the gate count of multiplier is proportional to n^2.

2.3 Conventional Modified Booth Multiplier

The number of terms to be added is halved in the modified Booth recoding scheme by comparing every other bit of the multiplier operand with the next and previous bits and using this 3-bit code to determine which operation to perform [3].

In the circuit for Booth multiplier the control vectors ADD, SUB, ZERO, ADD$\times 2$, SUB$\times 2$ and NEG are generated from the multiplier operand using Booth encoding logic [3, 4].

The inputs of the encoder circuit are groups of the multiplier's three bits sequences. Multiplier is shown as $b_{n-1}b_{n-2}\cdots b_1 b_0$. If n is even the multiplier is grouped as shown in Fig. 2, if it is odd the multiplier is grouped as in Fig. 3. In the literature, only the case that n is even is considered, in this paper because of we have written the models for any values of n between 2 and 54, we have considered the case that n is odd. ith value of ADD, SUB, ZERO, ADD$\times 2$, SUB$\times 2$ and NEG vectors are calculated by using ith group of the multiplier ($i = 0, 1, \cdots, \frac{n}{2}$ if n is even, $i = 0, 1, \cdots, \frac{n-1}{2}$ if n is odd).

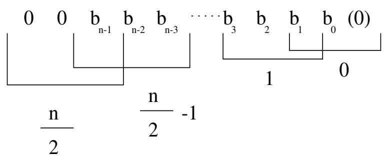

Figure 2: If n is even, the way that the multiplier is grouped

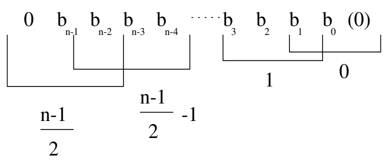

Figure 3: If n is odd, the way that the multiplier is grouped

ith values of the output vectors of encoder circuit are calculated according to the following equations ($i = 1, 2, \cdots, \frac{n}{2} - 1$ if n is even, $i = 1, 2, \cdots, \frac{n-1}{2} - 1$ if n is odd).

$$ZERO(i) = \overline{(b_{2i+1} + b_{2i} + b_{2i-1})}; \quad ADD(i) = \overline{b_{2i+1}}(b_{2i} \oplus b_{2i-1}); \quad ADD \times 2(i) = \overline{b_{2i+1}} b_{2i} b_{2i-1};$$

$$SUB(i) = b_{2i+1}(b_{2i} \oplus b_{2i-1}); \quad SUB \times 2(i) = b_{2i+1}\overline{(b_{2i} + b_{2i-1})}; \quad NEG(i) = SUB(i) + SUB \times 2(i) \qquad (9)$$

The name of the block we have designed to perform above equations is EN_1. The inputs of the block are b_{2i+1}, b_{2i} and b_{2i-1}, the outputs of the block are ith values of ADD, SUB, ZERO, ADD$\times 2$, SUB$\times 2$ and NEG vectors. There are three three input AND (AND3) gates, four INV gates, two OR2 gates, one three input NAND (NAND3) gates and one XOR gate in this block, so the area is 17.5 units. There are one AND2 gate and one XOR gate on the longest path of the block.

Because b_{-1} is always "0", we have simplified (9a-f) to calculate $0th$ value of the vectors as follows.

$$ZERO(0) = \overline{(b_1 + b_0)}; \quad ADD(0) = \overline{b_1} b_0; \quad ADD \times 2(0) = 0; \quad SUB(0) = b_1 b_0;$$

$$SUB \times 2(0) = b_1\overline{b_0}; \quad NEG(0) = b_1 \tag{10}$$

The name of the block we have designed to perform above equations is EN_0. The inputs of the block are b_1 and b_0, the outputs of the block are the $0th$ values of ADD, SUB, ZERO, ADD×2, SUB×2 and NEG vectors. There are three AND2 gates, two INV gates and one NOR gate in this block, so the area is 6.5 units instead of 17.5 units.

If n is even, because of b_n and b_{n+1} are always "0" we have simplified (9) to calculate $\frac{n}{2}th$ values of the output vectors of encoder circuit as follows.

$$ZERO(\tfrac{n}{2}) = \overline{b_{n-1}}; \quad ADD(\tfrac{n}{2}) = b_{n-1}; \quad ADD \times 2(\tfrac{n}{2}) = SUB(\tfrac{n}{2}) = SUB \times 2(\tfrac{n}{2}) = NEG(\tfrac{n}{2}) = 0 \tag{11}$$

The name of the block we have designed to perform above equations is EN_2. The input of the block is b_{n-1}, the outputs are $\frac{n}{2}th$ values of ADD, SUB, ZERO, ADD×2, SUB×2 and NEG vectors. There is one INV gate in this block, so the area is 0.5 unit.

If n is odd, because of b_n is always "0" we have simplified (9a-f) to calculate $\frac{n-1}{2}th$ values of the output vectors of encoder circuit as follows.

$$ZERO(\tfrac{n-1}{2}) = \overline{(b_{n-1} + b_{n-2})}; \quad ADD(\tfrac{n-1}{2}) = (b_{n-1} \oplus b_{n-2}); \quad ADD \times 2(\tfrac{n-1}{2}) = b_{n-1}b_{n-2};$$
$$SUB(\tfrac{n-1}{2}) = SUB \times 2(\tfrac{n-1}{2}) = NEG(\tfrac{n-1}{2}) = 0 \tag{12}$$

The name of the block we have designed to perform above equations is EN_3. The input of the block are b_{n-1} and b_{n-2}, the outputs of the block are $\frac{n-1}{2}th$ values of ADD, ZERO and ADDx2 ADD, SUB, ZERO, ADD×2, SUB×2 and NEG vectors. There are one AND2 gates, one two input NOR (NOR2) gate and one XOR gate in this block, so the area is 4.5 units.

If n is even one EN_0, $\left(\frac{n}{2} - 1\right)$ EN_1 and one EN_2 blocks are used while writing the VHDL model of the encoder circuit. So the area of the encoder circuit is calculated according to the following equation.

$$6.5 + 17.5 \cdot \left(\tfrac{n}{2} - 1\right) + 0.5 = 8.75n - 10.5 \tag{13}$$

If n is odd, one EN_0, $\left(\frac{n-1}{2} - 1\right)$ EN_1 and one EN_3 blocks are used while writing the VHDL model of the encoder circuit. So the area of the encoder circuit is calculated according to the following equation.

$$6.5 + 17.5 \cdot \left(\tfrac{n-1}{2} - 1\right) + 4.5 = 8.75n - 15.25 \tag{14}$$

As all the values of the output vectors are calculated in parallel, the longest path is the same as the longest path of EN_1 for both of the cases.

Fig. 4 shows a multiplexer for a single bit position of the multiplicand. It consists of five three-state buffers (TSB), which use the five control signals generated by a Booth encoder, to select the appropriate bit of the multiplicand or its inverse. Subtraction is performed using two's complement addition. This involves adding the inverse of the multiplicand and adding a further "1" at the least significant bit (LSB) position for SUB and at the next most significant bit for SUB×2. The extra "1", the subtract flag, is generated by the Booth encoding logic in NEG vector. Because all the bits of partial products are generated parallel the longest path of the multiplexer circuit is only one TSB long. For one partial product the area of the multiplexer is $5(n + 1)$ units. If n is even there is $\frac{n}{2} + 1$ partial products, so the total area of multiplexer circuit is $5(n + 1)(\frac{n}{2} + 1)$ units. If n is odd there are $\frac{n-1}{2} + 1$ partial products, so the total area of multiplexer circuit is

$5(n+1)(\frac{n-1}{2}+1)$ units.

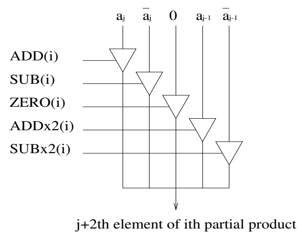

Figure 4: Multiplexer for $j+2th$ element of ith partial product

Fig. 5 shows, in schematic form, the standard 16-bit Booth multiplier block diagram, where the product is formed by summing the outputs from nine multiplexers in the sequence. All the addition rows are formed with $n+4$ bit full adders. There are three AND2 gates, one three input OR (OR3) gate and one XOR gate in a full adder, the area of the circuit is 10.5 units. There are two XOR gates on the longest path of the full adder circuit. Because addition in one row is performed in parallel the longest path of a addition row is the same as the longest path of one full adder, the area of one addition row is $10.5(n+4)$ units. The number of addition rows used to add all partial products is two less than the number of partial products, so if n is even there are $(\frac{n}{2}-1)$ rows and the area of the circuit is $10.5(n+4)(\frac{n}{2}-1)$ units, there are $2(\frac{n}{2}-1)$ XOR gate on the longest path, if n is odd there are $\frac{n-1}{2}-1$ rows and the area of the circuit is $10.5(n+4)(\frac{n-1}{2}-1)$ units, there are $2(\frac{n-1}{2}-1)$ XOR gates on the longest path.

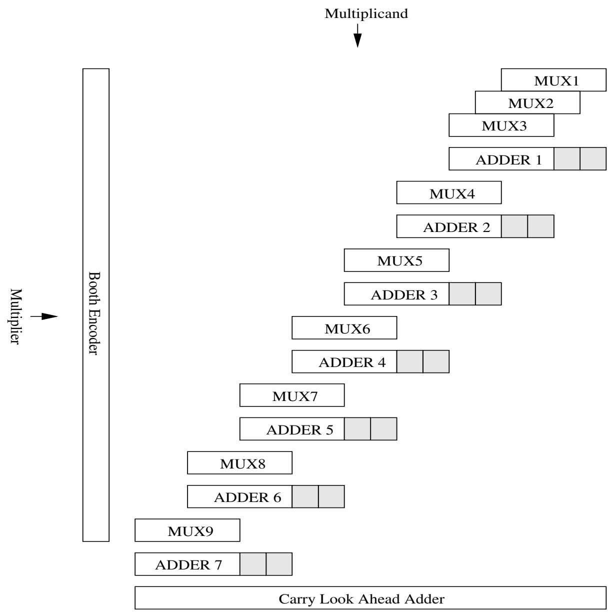

Figure 5: Schematic of standard 16-bit Booth multiplier

The sums and carries from the final row of the adders in the carry save array are combined in a CLA to form the result. The latency of CLA is proportional to $log_2 n$ and the area of CLA is proportional to n. The area of multiplier circuit is proportional to n^2 and the latency of the circuit is proportional to n.

Area values of the multiplication blocks depending on different values of n are given in Table 4. The gate count on the longest path of the multiplication blocks are given in Table 5. A DSP chip designer can decide to use which of the five multiplication blocks according to Table 4 and Table 5.

Table 4: Area values of multiplication blocks depending on different n values

n	1^{st}Circuit	2^{nd}Circuit
4	279	183
8	1321	662
12	3133	1389
16	5520	2364
20	8851	3587
24	12582	5058
28	15189	6777
32	22156	8744
36	28176	10959
40	34594	13422
44	39786	16133
48	49545	19092
52	54663	22299

Table 5: Gate counts on the longest paths of multiplication blocks depending on the value of n

n	1^{st} Circuit	2^{nd} Circuit
4	5 AND2+2 XOR+4 INV+4 OR2	AND2+3 XOR+TSB
8	7 AND2+3 XOR+6 INV+6 OR2	AND2+7 XOR+TSB
12	9 AND2+4 XOR+8 INV+8 OR2	AND2+11 XOR+TSB
16	9 AND2+4 XOR+8 INV+8 OR2	AND2+15 XOR+TSB
20	11 AND2+5 XOR+10 INV+10 OR2	AND2+19 XOR+TSB
24	11 AND2+5 XOR+10 INV+10 OR2	AND2+23 XOR+TSB
28	11 AND2+5 XOR+10 INV+10 OR2	AND2+27 XOR+TSB
32	11 AND2+5 XOR+10 INV+10 OR2	AND2+31 XOR+TSB
36	13 AND2+6 XOR+12 INV+12 OR2	AND2+35 XOR+TSB
40	13 AND2+6 XOR+12 INV+12 OR2	AND2+39 XOR+TSB
44	13 AND2+6 XOR+12 INV+12 OR2	AND2+43 XOR+TSB
48	13 AND2+6 XOR+12 INV+12 OR2	AND2+47 XOR+TSB
52	13 AND2+6 XOR+12 INV+12 OR2	AND2+51 XOR+TSB

3 Conclusions

In this paper, we propose two different multiplication blocks for DSP applications by using VHDL. We have written the models by taking the number of multiplier's and multiplicand's digits n as a parameter. The models are suitable for all of the multiplications from 2×2 bits to 54×54 bits. We have simplified some equations given in literature, which decreased the area of the corresponding circuit. We propose formulas to calculate the area values and the gate count of the longest path of all the five blocks of the circuits. Also we have indicated the types of gates on the longest paths of the circuit. By using the formulas we have calculated the critical values for both of the circuits for different n values and we have given the results in Table 4 and Table 5. The DSP chip designers can choose the multiplication blocks which meet the requirements by these tables.

The VHDL models are available on the URL address: http://www.ehb.itu.edu.tr/˜berna.

References

[1] N., Takagi, H., Yasuura, S., Yajima, "High-Speed VLSI Multiplication Algorithm with a Redundant Binary Addition Tree", IEEE Trans. on Comp., vol. C-34, no. 9, pp. 789-796., September 1985.

[2] H., Makino, Y., Nakase, H., Suzuki, H., Morinaka, H., Shinohara, K., Mashiko, "An 8.8-ns 54×54-Bit Multiplier with High Speed Redundant Binary Architecture", IEEE Jour. of Solid-State Cir., vol. 31, no. 6, pp. 773-783, June 1996.

[3] A. R., Cooper, "Parallel architecture modified Booth multiplier", IEE Proceedings, vol. 135, Pt. G, no. 3, pp. 125-128, June 1988.

[4] N. R., Shanbhag, P., Juneja, "Parallel Implementation of a 4x4-bit Multiplier Using Modified Booth's Algorithm", IEEE J. Solid-State Circuits, vol. 23, no. 4, August 1988.