A NEW VLSI ARCHITECTURE OF PARTIAL PRODUCT GENERATOR FOR REDUNDENT BINERY MULTIPLIER FOR DSP OR MULTIPLIER APPLICATIONS

BOJANAPELLI JYOTHIRMAI[1] **S.SRIKANTH**[2]

[1]PG Scholar,Dept of ECE ,Mtech in Embedded systems &VLSI, MALLAREDDY ENGINEERING COLLEGE FOR WOMEN (MRCW), MAISAMMAGUDA, DHULAPALLY, SECUNDERABAD.

[2]Assistant Professor, Dept of ECE, MALLAREDDY ENGINEERING COLLEGE FOR WOMEN (MRCW), MAISAMMAGUDA, DHULAPALLY, SECUNDERABAD.

Abstract: Adders are the key element of the arithmetic unit, especially fast parallel adder. Redundant Binary Signed Digit (RBSD) adders are designed to perform high-speed arithmetic operations. Generally, in a high radix modified Booth encoding algorithm the partial products are reduced in multiplication process. Due to its high modularity and carry-free addition, a redundant binary (RB) representation can be used when designing high performance multipliers. The conventional RB multiplier re -quires an additional RB partial product (RBPP) row, because an error-correcting word (ECW) is generated by both the radix-4 Modified Booth encoding (MBE) and the RB encoding. This incurs in an additional RBPP accumulation stage for the MBE multiplier. In this paper, a new RB modified partial product generator (RBMPPG) is proposed; it removes the extra ECW and hence, it saves one RBPP accumulation stage. Therefore, the proposed RBMPPG generates fewer partial product rows than a conventional RB MBE multiplier. Simulation results show that the proposed RBMPPG based designs significantly improve the area and power consumption when the word length of each operand in the multiplier is at least 32 bits.

Key Words: Redundant binary, modified Booth encoding, RB partial product generator, RB multiplier.

I. INTRODUCTION

The digital multiplier is a ubiquitous arithmetic unit in microprocessors, digital signal processors, and emerging media processors. It is also a kernel operator in application specific data path of video and audio codes, digital filters, computer graphics, and embedded systems. Compared with many other arithmetic operations, multiplication is time consuming and power hungry. The critical paths dominated by digital multipliers often impose a speed limit on the entire design. Hence, VLSI design of high-speed multipliers, with low energy dissipation, is still a popular research subject. Redundant binary (RB) representation is one of the signed digit representations first introduced by Avizienis [9] in 1961 for fast parallel arithmetic. Many algorithms and architectures have been proposed to design high-speed and low-power multipliers [1-13]. A normal binary (NB) multiplication by digital circuits includes three steps. In the first step, partial products are generated; in the second step, all partial products are added by a partial product reduction tree until two partial product rows remain. In the third step, the two partial product rows are added by a fast carry propagation adder. Two methods have been used to perform the second step for the partial product reduction. A first method uses 4-2 compressors, while a second method uses redundant binary (RB) numbers.

Both methods allow the partial product reduction tree to be reduced at a rate of 2:1. The RB addition is carry-free, making it a promising substitute for two's complement multi-operand addition in a tree-structured multiplier. Similar to a normal binary (NB) multiplier, an RB multiplier is anatomized into three stages and consists of four modules: the Booth encoder, RB partial product generator (also known as decoder), RB partial product accumulator, and RB-to-NB converter. A Radix-4 Booth encoding or a modified Booth encoding (MBE) is usually used in the partial product generator of parallel multipliers to reduce the number of partial product rows by half [5-6] [10-13]. A RBPP row can be obtained from two adjacent NB partial product rows by inverting one of the pair rows [5-6]; an N-bit convention-al RB MBE (CRBBE-2) multiplier requires N/4 RBPP rows. An additional error-correcting word (ECW) is also required by both the RB and the Booth encoding [5-6] [14]; therefore, the number of RBPP accumulation stages (NRBPPAS) required by a power-of-two word-length (i.e., 2 -bit) multiplier is given by: NRBPPAS = log (N /4 + 1)

= n − 1, if N = 2^n.

This paper focuses on the RBPP generator for designing a 2 - bit RB multiplier with fewer partial product rows by eliminating the extra ECW. A new RB modified partial product generator based on MBE (RBMPPG-2) is proposed. In the proposed RBMPPG-2, the ECW of each row is moved to its next neighbor row. Furthermore, the extra ECW generated by the last partial product row is combined with both the two most significant bits (MSBs) of the first partial product row and the two least significant bits (LSBs) of the last partial product row by logic simplification. Therefore, the proposed method reduces the number of RBPP rows from N /4 + 1to N /4, i.e., a RBPP accumulation stage is saved. The proposed method is applied to 8×8-bit, 16×16-bit, 32×32-bit, and 64×64-bit RB multiplier designs; the designs are synthesized using the NanGate 45nm Open Cell Library. The proposed designs achieve significant reductions in area and power consumption compared with existing multipliers when the word length of each of the operands is at least 32 bits.

II. Literature Survey

A high-radix Booth encoding technique can reduce the number of partial products. However, the number of expensive hard multiples (i.e., a multiple that is not a power of two and the operation cannot be per-formed by simple shifting and/or complementation) increases too. Besli et al. noticed that some hard multiples can be obtained by the differences of two simple power-of-two multiplies. A new radix-16 Booth en-coding (RBBE-4) technique without ECW has been pro-posed, it avoids the issue of hard multiples. A radix-16 RB Booth encoder can be used to overcome the hard multiple problem and avoid the extra ECW, but at the cost of doubling the number of RBPP rows. Therefore, the number of radix-16 RBPP rows is the same as in the radix-4 MBE. However, the RBPP generator based on a radix-16 Booth encoding has a complex circuit structure and a lower speed compared with the MBE partial product generator when requiring the same number of partial products.

III. Proposed System

The aim of this study is implementation of modified partial product generator for RB multipliers.

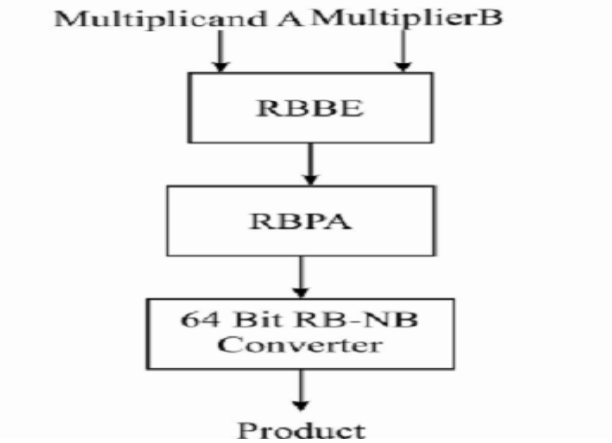


Fig.1.block diagram

A RB multiplier consists of a RB partial product (RBPP) generator, a RBPP reduction tree and a RB-NB converter. A Radix-4 Booth encoding or a modified Booth encoding (MBE) is usually used in the partial product generator of parallel multipliers to reduce the number of partial product rows by half. A RBPP row can be obtained from two adjacent NB partial product rows by inverting one of the pair rows.

Modified Booth Encoder

In order to achieve high-speed multiplication, multiplication algorithms using parallel counters, such as the modified Booth algorithm has been proposed, and some multipliers based on the algorithms have been implemented for practical use. This type of multiplier operates much faster than an array multiplier for longer operands because its computation time is proportional to the logarithm of the word length of operands.

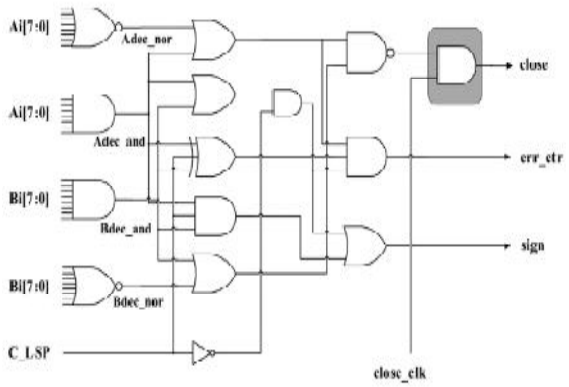


Fig 2. Modified Booth Encoder

Booth multiplication is a technique that allows for smaller, faster multiplication circuits, by recoding the numbers that are multiplied. It is possible to reduce the number of partial products by half, by using the technique of radix-4 Booth recoding. The basic idea is that, instead of shifting and adding for every column of the multiplier term and multiplying by 1 or 0, we only take every second column, and multiply by ±1, ±2, or 0, to obtain the same results. The advantage of this method is the halving of the number of partial products. To Booth recode the multiplier term, we consider the bits in blocks of three, such that each block overlaps the previous block by one bit. Grouping starts from the LSB, and the first block only uses two bits of the multiplier. Figure 3 shows the grouping of bits from the multiplier term for use in modified booth encoding.

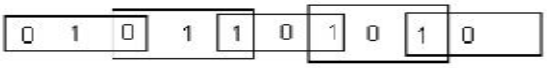


Fig.3 Grouping of bits from the multiplier term

Each block is decoded to generate the correct partial product. The encoding of the multiplier Y, using the modified booth algorithm, generates the following five signed digits, -2, -1, 0, +1, +2. Each encoded digit in the multiplier performs a certain operation on the multiplicand, X, as illustrated in Table 1

Block	Re - coded digit	Operation on X
000	0	0 X
001	+1	+1 X
010	+1	+1 X
011	+2	+2 X
100	-2	-2 X
101	-1	-1 X
110	-1	-1 X
111	0	0 X

For the partial product generation, we adopt Radix-4 Modified Booth algorithm to reduce the number of partial products for roughly one half. For multiplication of 2‟s complement numbers, the two-bit encoding using this algorithm scans a triplet of bits. When the multiplier B is divided into groups of two bits, the algorithm is applied to this group of divided bits. Figure 4, shows a computing example of Booth multiplying two numbers ”2AC9” and “006A”. The shadow denotes that the numbers in this part of Booth multiplication are all zero so that this part of the computations can be neglected. Saving those computations can significantly reduce the

power consumption caused by the transient signals. According to the analysis of the multiplication shown in figure 4, we propose the SPST-equipped modified-Booth encoder, which is controlled by a detection unit. The detection unit has one of the two operands as its input to decide whether the Booth encoder calculates redundant computations. As shown in figure 9. The latches can, respectively, freeze the inputs of MUX-4 to MUX-7 or only those of MUX-6 to MUX-7 when the PP4 to PP7 or the PP6 to PP7 are zero; to reduce the transition power dissipation. Figure 10, shows the booth partial product generation circuit. It includes AND/OR/EX-OR logic.

RB Partial Product Generator:

As two bits are used to represent one RB digit, then a RBPP is generated from two NB partial products [1-6]. The addition of two N-bit NB partial products X and Y using two's complement representation can be expressed as follows X +Y = X –Y -1 = (X ,Y-) -1.

Where Y- is the inverse of Y. The RBPP is generated by inverting one of the two NB partial products and adding -1 to the LSB.. Each RB digit X i belongs to the set{-1,0,1};this is coded by two bits (X i-, X i+) .RB numbers can be coded in several ways.

TABLE 2
RB Encoding Used in This Work [6]

X_i^+	X_i^-	RB digit (X_i)
0	0	0
0	1	$\bar{1}$
1	0	1
1	1	0

Both MBE and RB coding schemes introduce errors and two correction terms are required: 1) when the NB number is converted to a RB format, -1 must be added to the LSB of the RB number; 2) when the multiplicand is multiplied by -1 or - 2 during the Booth encoding, the number is inverted and +1 must be added to the LSB of the partial product. A single ECW can compensate errors from both the RB encoding and the radix-4 Booth recoding.

Proposed RB Partial Product Generator:

A new RB modified partial product generator based on MBE (RBMPPG-2) is presented in this section; in this design, ECW is eliminated by incorporating it into both the two MSBs of the first partial product row (PP 1+) and the two LSBs of the last partial product row (PP – (N/4)).

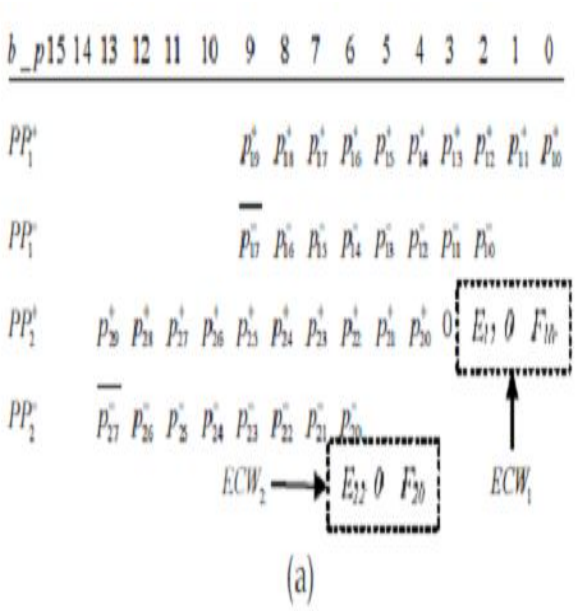


Fig.4(a)The first new RBMPPG-2 architecture for an 8-bit MB multiplier

It is differ from conventional type by its error correcting vector. In this type error correcting vectors ECW1 is generated by PP1 and ECW 2 is generated by PP2.

ECW1= 0 E12 0 F 10

ECW2= 0 E 22 0 F 20

To eliminate a RBPP accumulation, ECW 2 needs to be incorporated into PP1 and PP2. F20={-1 , b5 b4 b3=000,001,010,011,111 F20={0 , b 5b4b3=100,101,110

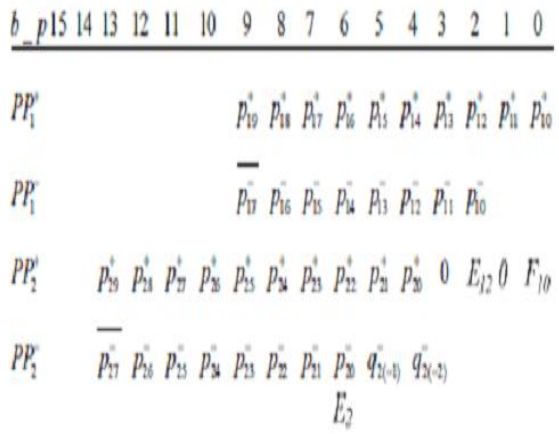


Fig 4(b) Revised RBMPPG by replacing E22 and F20

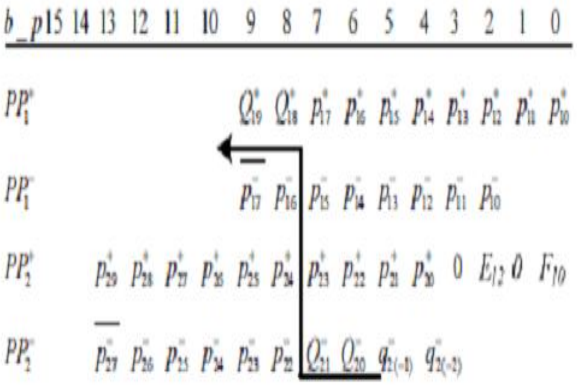


Fig4(c) final proposed rbmppg by totally eliminating ECW2.

Q 19+ ,Q 18+,Q 21-,Q20- are used to represent the modified partial products .By setting PP2+ to all ones and adding +1 to the LSB of the partial product ,F20 can then be determined only by b5

F20={-1, b5=0

F 20= {0, b5 =1

As -1 can be coded as 111 in RB format, E 22 and F 20 can be represented by E 2 ,q 2(-2-) ,q 2(-1)- as follows

$$E_2 = \begin{cases} E_{22}, & F_{20} = 0 \\ E_{22} - 1, & F_{20} = -1 \end{cases}$$

$$q_{2(-2)}^{-} = q_{2(-1)}^{-} = \begin{cases} 0, & F_{20} = 0 \\ 1, & F_{20} = -1 \end{cases}$$

This is further explained by the truth table ofE22, F20 and E2, q2-(-2),q2-(-1) .

Table III.

TRUTH TABLE OF E_2, $q_{2(-2)}^{-}$, $q_{2(-1)}^{-}$ AND p_{21}^{-},p_{20}^{-}.

$b_7b_6b_5$	$E_{22}F_{20}$	$E_2q_{2(-2)}^{-}q_{2(-1)}^{-}$	p_{21}^{-}	p_{20}^{-}
0 0 0	0 $\bar{1}$	$\bar{1}$ 1 1	0	0
0 0 1	0 0	0 0 0	a_1	a_0
0 1 0	0 $\bar{1}$	$\bar{1}$ 1 1	a_1	a_0
0 1 1	0 0	0 0 0	a_0	0
1 0 0	1 $\bar{1}$	0 1 1	$\overline{a_0}$	1
1 0 1	1 0	1 0 0	$\overline{a_1}$	$\overline{a_0}$
1 1 0	1 $\bar{1}$	0 1 1	$\overline{a_1}$	$\overline{a_0}$
1 1 1	0 0	0 0 0	0	0

The relationships between Q 19+ ,Q 18+,Q 21-,Q20- and P19+,P21- ,P20- are summarized in table.

THE TRUTH TABLE OF Q_{19}^{+}, Q_{18}^{+}, Q_{21}^{-}, Q_{20}^{-}

$p_{19}^{+}p_{18}^{+}p_{21}^{-}p_{20}^{-}$	$Q_{19}^{+}Q_{18}^{+}Q_{21}^{-}Q_{20}^{-}$ (E_2=0)	$Q_{19}^{+}Q_{18}^{+}Q_{21}^{-}Q_{20}^{-}$ (E_2=1)	$Q_{19}^{+}Q_{18}^{+}Q_{21}^{-}Q_{20}^{-}$ (E_2= −1)
0100	0100	0101	0011
0101	0101	0110	0100
0110	0110	0111	0101
0111	0111	1000	0110
1000	1000	1001	0111
1001	1001	1010	1000
1010	1010	1011	1001
1011	1011	1100	1010

Logic functions of Q19+ ,Q18+,Q21-and Q20- can be expressed as follows

$$Q_{19}^{+} = (b_7 \oplus b_5 + b_7b_6b_5) \cdot p_{19}^{+} + \overline{b_7}\,\overline{b_5} \cdot (p_{18}^{+} + p_{21}^{-} + p_{20}^{-}) + b_7\overline{b_6}b_5 \cdot (p_{18}^{+}p_{21}^{-}p_{20}^{-} \oplus p_{19}^{+})$$

$$Q_{18}^{+} = (b_7 \oplus b_5 + b_7b_6b_5) \cdot p_{18}^{+} + \overline{b_7}\,\overline{b_5} \cdot (\overline{p_{21}^{-}+p_{20}^{-}} \oplus p_{18}^{+}) + b_7\overline{b_6}b_5 \cdot (p_{21}^{-}p_{20}^{-} \oplus p_{18}^{+})$$

$$Q_{21}^{-} = (b_7 \oplus b_5 + b_7b_6b_5) \cdot p_{21}^{-} + \overline{b_7}\,\overline{b_5} \cdot \overline{p_{21}^{-} \oplus p_{20}^{-}} + b_7\overline{b_6}b_5 \cdot p_{21}^{-} \oplus p_{20}^{-}$$

$$Q_{20}^{-} = (b_7 \oplus b_5 + b_7b_6b_5) \cdot p_{20}^{-} + \overline{b_7}\,\overline{b_5} \cdot \overline{p_{20}^{-}} + b_7\overline{b_6}b_5 \cdot \overline{p_{20}^{-}}$$

Therefore, the extra ECW N/4 is removed by the transformation of 4 partial product variables and one partial product row is saved in

RB multipliers with any power-of-two word-length.

RBPA

In the second stage, a 4-stage RBA summing tree is used to sum 16 RB partial products. Each RBA block contains 64 RB full adder (RBFA) cells and a varying number of RB half adder (RBHA) cells depending on where it is located**.** The proposed RBMPPG-2 can be applied to any bit RB multipliers with a reduction of a RBPP accumulation stage compared with conventional designs. Although the delay of RMPPG-2 increases by 1-stage of TG delay, the delay of one RBPP accumulation stage is significantly larger than a 1- stage TG delay. Therefore, the delay of the entire multiplier is reduced. The improved complexity, delay and power consumption are very attractive for the proposed design. The multiplier consists of the proposed RBMPPG-2, three RBPP accumulation stages, and one RB-NB converter. Eight RBBE-2 blocks generate the RBPP they are summed up by the RBPP reduction tree that has three RBPP accumulation stages. Each RBPP accumulation block contains RB full adders (RBFAs) and half adders (RBHAs).

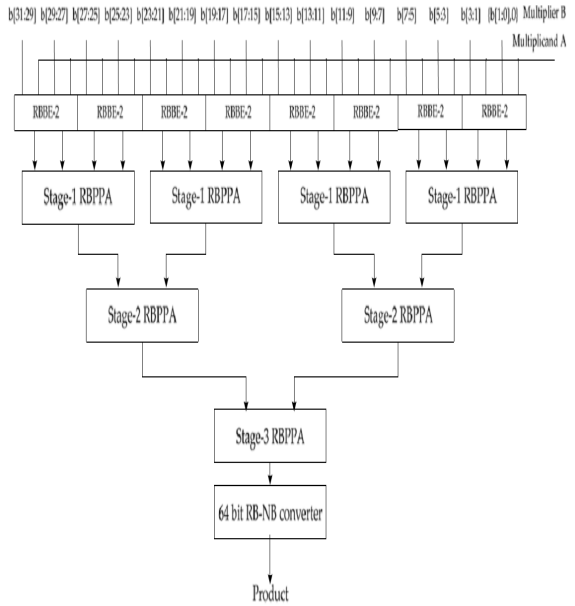


Fig. 5. The block diagram of a 32-bit RB multiplier using the proposed RBMPPG-2.

The proposed RBMPPG-2 can be applied to any 2n-bit RB multipliers with a reduction of a RBPP accumulation stage compared with conventional designs. Although the delay of RMPPG-2 increases by one-stage of TG delay, the delay of one RBPP accumulation stage is significantly larger than a one-stage TG delay. Therefore, the delay of the entire multiplier is reduced. The improved complexity, delay and power consumption are very attractive for the proposed design.

A 32-bit RB MBE multiplier using the proposed RBPP generator is shown in Fig. 5. The multiplier consists of the proposed RBMPPG-2, three RBPP accumulation stages, and one RB-NB converter. Eight RBBE-2 blocks generate the RBPP (pþ i , p i); they are summed up by the RBPP reduction tree that has three RBPP accumulation stages. Each RBPP accumulation block contains RB full adders (RBFAs) and half adders (RBHAs) [7]. The 64-bit RB-NB converter converts the final accumulation results into the NB representation, which uses a hybrid parallel-prefix/carry select adder [25] (as one of the most efficient fast parallel adder designs). There are four stages in a conventional 32-bit RB MBE multiplier architecture; however, by using the proposed RBMPPG-2, the number of RBPP accumulation stages is reduced from 4 to 3 (i.e., a 25 percent reduction). These are significant savings in delay, area as well as power consumption. The improvements in delay, area and power consumption are further demonstrated in the next section by simulation.

IV. CONCLUS ION

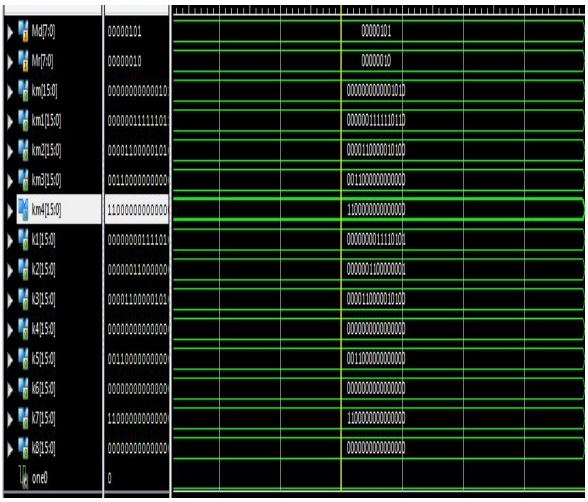
A new modified RBPP generator has been proposed in this paper; this design eliminates the additional ECW that is introduced by previous designs. Therefore, a RBPP accumulation stage is saved due to the elimination of ECW. The new RB partial product generation technique can be applied to any 2n-bit RB multipliers to reduce the number of RBPP rows from N/4 + 1 to N/4.

Simulation results have shown that the performance of RB MBE multipliers using the proposed RBMPPG-2 is improved significantly in terms of delay and area. The proposed designs achieve significant reductions in area and power consumption when the word length is at least 32 bits.

V. Simulation Results

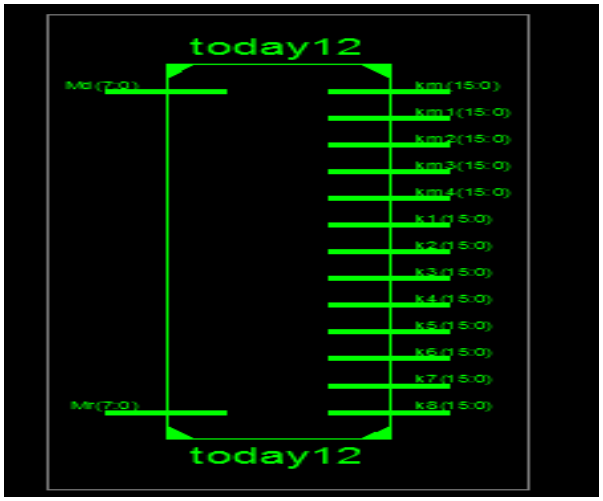
The written Verilog HDL Modules have successfully simulated and verified using Modelsim6.4b and synthesized using Xilinxise13.2.

Simulation result:

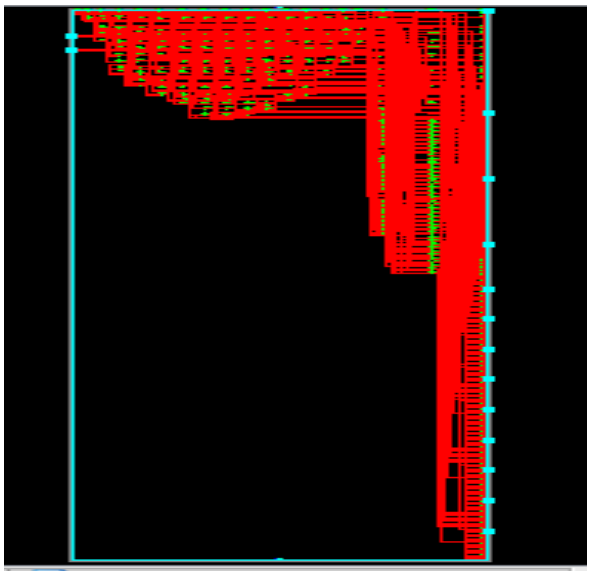


Synthesis Results:

RTL Schematic



Technology Schematic:



AREA:

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	78	4656	1%
Number of 4 input LUTs	136	9312	1%
Number of bonded IOBs	224	232	96%

Timing report:

```
Timing Summary:
---------------
Speed Grade: -4

   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 17.213ns
```

References

[1] I. Koren, Computer Arithmetic Algorithms, Prentice Hall, New York, 1993.

[2] A.D. Booth, A signed binary multiplication technique, Quarterly Journal of Mechanics and Applied Mathematics 4 (1951) 236–240.

[3] O.L. McSorley, High-speed arithmetic in binary computers, Proceedings of the Institute of Radio Engineers 49 (1961) 67–91.

[4] Y. Harata, Y. Nakamura, H. Nagase, M. Takigawa, N. Takagi, A high-speed multiplier

using a redundant binary adder tree, IEEE Journal of Solid-State Circuits 22 (1987) 28–34.
[5] H. Makino, Y. Nakase, H. Suzuki, H. Morinaka, H. Shinohara, K. Mashiko, An 8.8–11 s 54 54-bit multiplier with high-speed redundant binary architecture, IEEE Journal of Solid-State Circuits 31 (1996) 773–783.
[6] S.M. Yen, C.S. Laih, C.H. Chen, J.Y. Lee, An efficient redundant-binary number to binary number converter, IEEE Journal of Solid-State Circuits 27 (1992) 109–112.
[7] N. Besli, R.G. Deshmukh, A novel redundant binary signed-digit (RBSD) Booth's encoding, in: Proceedings of the IEEE Southeast Conference, Columbia, SC, 2002, pp. 426–431.
[8] N. Besli, R.G. Deshmukh, A 54 54-bit multiplier with a new redundant binary Booth's encoding, Proceedings of the Canadian Conference on Electrical and Computer Engineering 2 (2002) 597–602 (Winnipeg, Canada).
[9] S. Lee, S. Bae, H. Park, A compact radix-64 54 54 CMOS redundant binary parallel multiplier, IEICE Transactions on Electronics E 85-C (2002) 1342–1350.
[10] Y. Kim, B. Song, J. Grosspietsch, S. Gillig, A carry-free 54 54-bit multiplier using equivalent bit conversion algorithm, IEEE Journal of Solid-State Circuits 36 (2001) 1538–1545.