iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 9 **Issue:** VII **Month of publication:** July 2021

DOI: https://doi.org/10.22214/ijraset.2021.36474

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue VII July 2021- Available at www.ijraset.com

Efficient Memory-based Multiplier Technique for SWL DSP Systems

B Ajay Kumar[1], G Sai Vamshi Krishna[2], T Dheeraj Sai[3], N S Murti Sarma[4]
[1, 2, 3, 4]*Electronics and Communication Engineering Department, Sreenidhi Institute of Science and Technology, Ghatkesar, Hyderabad, Telangana, Ind*

Abstract: ***The DSP systems usually deal with a lot of multiplications as it is dealt with many discrete signals. The combinational circuits consume a lot of power as there are many intermediate blocks (i.e., usually full adders & and gates). The combinational circuits take more area and the delay is also more. Usually there is a tradeoff between area and delay. To make the multiplier more efficient we usually prefer memory-based multiplier. Different types of techniques are there in memory-based multipliers like the APC (anti-symmetric product coding), OMS (odd multiple storage) etc. In these techniques LUT based storage is used. The multiplied products are stored efficiently based on the technique used to store the data. To optimize the memory required we combine the APC and OMS technique for better storage and retrieval of data. In this project we show how combined technique increases the performance of multiplier. The suggested combined technique reduces the size of the LUT to one-fourth that of a standard LUT. It is demonstrated that the proposed LUT architecture for tiny input sizes can be used to execute high-precision multiplication with input operand decomposition in an efficient manner.***

Keywords: Antisymmetric product coding (APC); Odd multiple storage (OMS); Digital Signal Processing (DSP); lookup- table (LUT); memory-based computing.

I. INTRODUCTION

Memory-based computing is crucial in digital signal processing (DSP) techniques that use a fixed set of coefficients for multiplication. Memory-based computing is a subset of specialized systems in which look-up tables (LUT) perform the computational activities. In comparison to multiply accumulate structures, they are closer to human computing and more regular. Antisymmetric product coding (APC), odd-multiple-storage (OMS), and the combined approach of APC and OMS known as APC-OMS are three memory-based computation techniques that can be used to optimize LUTs for memory-based multiplication. Only the odd multiples of the fixed coefficients must be stored in LUT, resulting in odd-multiple-storage (OMS). The product words are encoded as antisymmetric pairs in the antisymmetric product coding (APC) technique. The LUT size is only reduced by a factor of two with each of these strategies. In the APC-OMS technique, the LUT size is reduced to one-fourth of the conventional LUT size. The complement operations might be greatly reduced in the combined technique of APC-OMS based LUT multiplier since the input address and LUT output are always mapped into odd integers. However, it displays a significant overhead in area in APC-OMS, so a modified APC-OMS technique based LUT multiplier is proposed.

The complement operations might be greatly reduced in the combined technique of APC-OMS based LUT multiplier since the input address and LUT output are always mapped into odd integers. However, it displays a significant overhead in area in APC-OMS, so a modified APC-OMS technique based LUT multiplier is proposed.

II. MEMORY BASED COMPUTATION TECHNIQUES

A. Antisymmetric Product Coding for LUT Optimization

Table 1 shows the result words for various input X values for word length L = 5. The input word X in the third column of each row is the two's complement of that in the first column of the same row, as seen in this table. Also, the total of product values in the same row corresponding to these two input values is 32A. We'll use positive integers for X (input) and A (fixed coefficient), and u and v for the product values in the second and fourth columns of the row. The sum of u and v is 32A, as shown in the table, and u and v may be calculated using the following equation.

u=[u+v]/2 +[v-u]/2 and v=[u+v]/2 and v=[u+v]/2-[v-u]/2 ……..(1)

Substituting (u + v) = 32A we get

u=16A+[v-u]/2 and v=16A-[v-u]/2……(2)

A negative symmetry can be shown on u and v in Equation 2. As a result of this behavior, we can cut the size of the lookup table in half by just storing [v-u]/2 for a pair of inputs on the same row. The APC word for X= (00000) is already stored as 16A. The product word is obtained by the following equation. The technique is called an antisymmetric product code because the representation of the product is derived from the antisymmetric behavior of the products. The four-bit address X'=(x3', x2', x1', x0') is accessible using the relation X'=XL for x4=1 and X=X'L for x4=0, where XL is the four-bit LSB of X and XL' is the two-s complement of XL.

Product=16A+(sign value)(APC-word)

The most significant bit determines the sign value. When the most significant bit is 1, the sign value is 1, and when the most significant bit is 0, the sign value is -1. We use an adder circuit to do addition and subtraction operations, and the operations are performed between the 16A and look up table outputs.

B. APC-OMS for LUT Optimization

Only the odd multiples of the fixed coefficient (A) are stored in the LUT using the OMS technique, and all the even multiples of A are produced from the left-shift operations of one of those odd multiples. The product words for various input X values for an APC-OMS based LUT multiplier with L = 5 are displayed. For the input X =, just nine memory spaces are required to hold the product values (00000). In Table II, you'll see that the eight odd multiples are stored in only eight memory places. Even multiples 2A, 4A are obtained by left-shift operations on fixed coefficient A.

Input, X	product values	Input, X	product values	address $x_3'x_2'x_1'x_0'$	APC words
0 0 0 0 1	A	1 1 1 1 1	$31A$	1 1 1 1	$15A$
0 0 0 1 0	$2A$	1 1 1 1 0	$30A$	1 1 1 0	$14A$
0 0 0 1 1	$3A$	1 1 1 0 1	$29A$	1 1 0 1	$13A$
0 0 1 0 0	$4A$	1 1 1 0 0	$28A$	1 1 0 0	$12A$
0 0 1 0 1	$5A$	1 1 0 1 1	$27A$	1 0 1 1	$11A$
0 0 1 1 0	$6A$	1 1 0 1 0	$26A$	1 0 1 0	$10A$
0 0 1 1 1	$7A$	1 1 0 0 1	$25A$	1 0 0 1	$9A$
0 1 0 0 0	$8A$	1 1 0 0 0	$24A$	1 0 0 0	$8A$
0 1 0 0 1	$9A$	1 0 1 1 1	$23A$	0 1 1 1	$7A$
0 1 0 1 0	$10A$	1 0 1 1 0	$22A$	0 1 1 0	$6A$
0 1 0 1 1	$11A$	1 0 1 0 1	$21A$	0 1 0 1	$5A$
0 1 1 0 0	$12A$	1 0 1 0 0	$20A$	0 1 0 0	$4A$
0 1 1 0 1	$13A$	1 0 0 1 1	$19A$	0 0 1 1	$3A$
0 1 1 1 0	$14A$	1 0 0 1 0	$18A$	0 0 1 0	$2A$
0 1 1 1 1	$15A$	1 0 0 0 1	$17A$	0 0 0 1	A
1 0 0 0 0	$16A$	1 0 0 0 0	$16A$	0 0 0 0	0

For $X = (0\ 0\ 0\ 0\ 0)$, the encoded word to be stored is $16A$.

Table 1: APC Word Different Input Values for L=5

input X' $x_3'x_2'x_1'x_0'$	product value	# of shifts	shifted input, X''	stored APC word	address $d_3d_2d_1d_0$
0 0 0 1	A	0	0 0 0 1	$P0 = A$	0 0 0 0
0 0 1 0	$2 \times A$	1			
0 1 0 0	$4 \times A$	2			
1 0 0 0	$8 \times A$	3			
0 0 1 1	$3A$	0	0 0 1 1	$P1 = 3A$	0 0 0 1
0 1 1 0	$2 \times 3A$	1			
1 1 0 0	$4 \times 3A$	2			
0 1 0 1	$5A$	0	0 1 0 1	$P2 = 5A$	0 0 1 0
1 0 1 0	$2 \times 5A$	1			
0 1 1 1	$7A$	0	0 1 1 1	$P3 = 7A$	0 0 1 1
1 1 1 0	$2 \times 7A$	1			
1 0 0 1	$9A$	0	1 0 0 1	$P4 = 9A$	0 1 0 0
1 0 1 1	$11A$	0	1 0 1 1	$P5 = 11A$	0 1 0 1
1 1 0 1	$13A$	0	1 1 0 1	$P6 = 13A$	0 1 1 0
1 1 1 1	$15A$	0	1 1 1 1	$P7 = 15A$	0 1 1 1

input X $x_4x_3x_2x_1x_0$	product values	encoded word	stored values	# of shifts	address $d_3d_2d_1d_0$
1 0 0 0 0	$16A$	0	— — —	— —	— — —
0 0 0 0 0	0	$16A$	$2A$	3	1 0 0 0

Table 2: OMS Words For Different Input Values for L=5

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue VII July 2021- Available at www.ijraset.com

III. IMPLEMENTATION OFTECHNIQUES

A. APC Based LUT Multiplier

The block diagram of an APC-based LUT multiplier for L=5 is shown in Figure. It comprises of a four-input LUT with 16 words for storing the APC values of product words, as shown in Table I's sixth column. It also includes an address mapping circuit as well as an add/subtract block. Accordingly, the address circuit will provide the desired address (x3'x2'x1'x0'). The LUT output is then calculated based on the address, and the LUT output is subsequently added to or removed from 16A based on the most significant bit. The LUT output is added to 16A when the most significant bit is 1, and it is removed from 16A when x4=0.

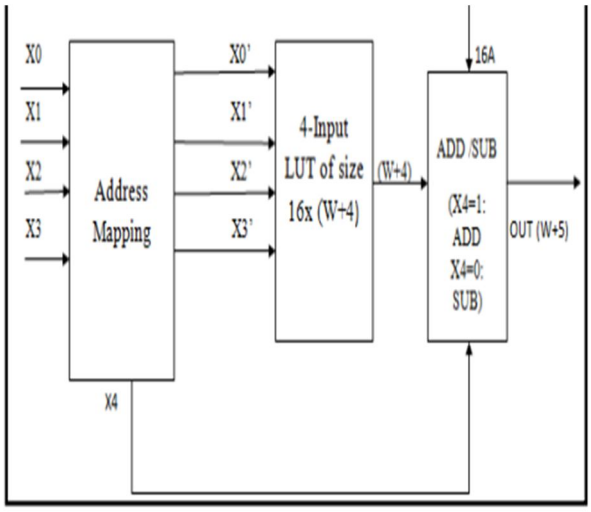


Figure 1: APC based LUT multiplier for L=5

B. APC-OMS based LUT multiplier

The block diagram of an APC-OMS based LUT multiplier for L=5 is shown in Figure. It comprises of a shifter, an address generating circuit, and a control circuit, as well as a LUT of nine words with a 4-bit width. The 5-bit input will be converted to a 4-bit address word by the address generating circuit (d3d2d1d0). The control circuit is utilized to generate the control bits s0, s1 that are used to produce the shifter's number of shifts. If the most significant bit is 0, the output is the same product word from the shifter; if the most significant bit is 1, the product is formed by adding addition16A and the shifted value. The following expression generates the control signals.

S0=(~(x0+(~(x1+(~x2)))) S1=~(x0+x1)

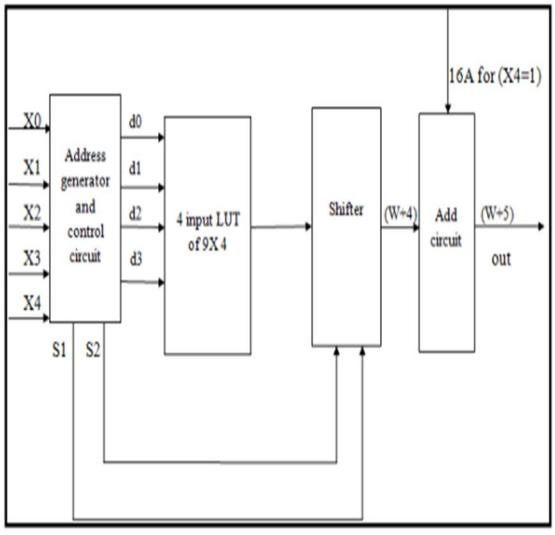


Figure 2 : APC-OMS based LUT multiplier for L=5

IV. SIMULATION RESULTS

The LUT-based multipliers based on APC,and APC-OMS are coded in Verilog HDL and generated in the Xilinx ISE design suite. The simulator used is I-sim tool. The family used is Automotive Artix 7,device is XA7A100T,Package is CSG324,and speed is -2.The delay may vary based on the family, device and package used.

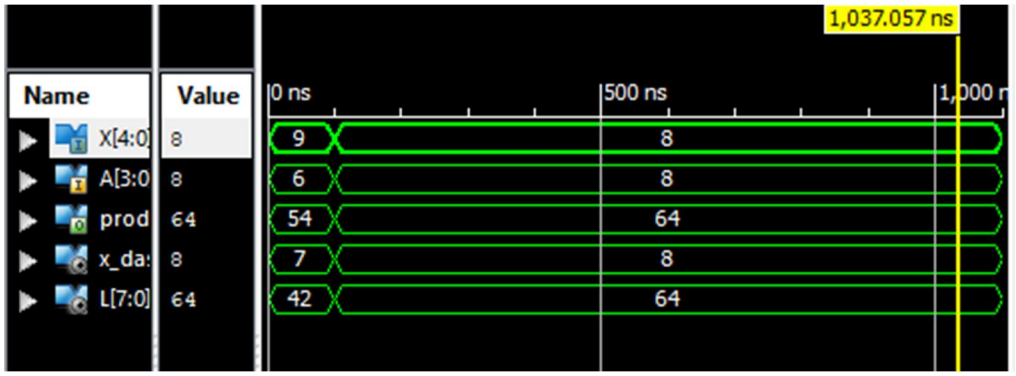


Figure 3: Simulation result of APC based LUT multiplier for L=5

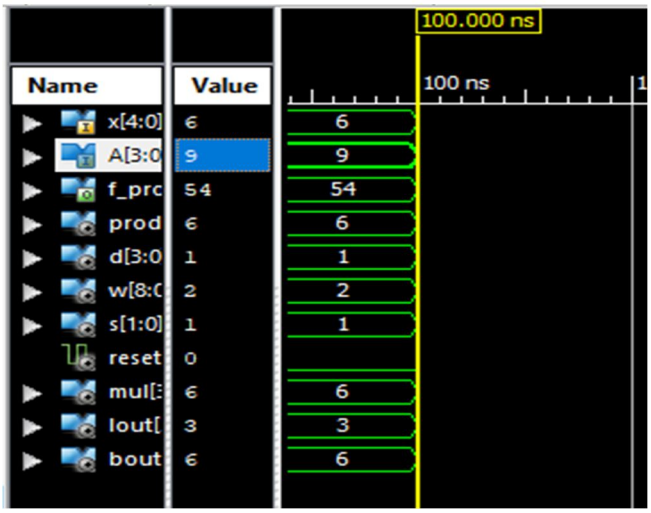


Figure 4: Simulation result of APC- OMS based LUT multiplier for L=5

```
Timing constraint: Default path analysis
  Total number of paths / destination ports: 2587 / 9
-------------------------------------------------------------------------
Delay:               6.148ns (Levels of Logic = 11)
  Source:            X<1> (PAD)
  Destination:       product<8> (PAD)

  Data Path: X<1> to product<8>
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             8    0.001   0.610  X_1_IBUF (X_1_IBUF)
     LUT3:I0->O            3    0.097   0.628  m1/Mxor_x_dash<1>_xo<0>1 (x_da
     LUT6:I2->O            5    0.097   0.594  ADDERTREE_INTERNAL_Madd1_lut<3
     LUT6:I3->O            2    0.097   0.758  ADDERTREE_INTERNAL_Madd1_xor<4
     LUT6:I0->O            2    0.097   0.748  ADDERTREE_INTERNAL_Madd2_lut<0
     LUT6:I1->O            3    0.097   0.628  ADDERTREE_INTERNAL_Madd2_cy<0>
     LUT6:I2->O            1    0.097   0.439  ADDERTREE_INTERNAL_Madd2_xor<0
     LUT3:I1->O            1    0.097   0.000  m3/Maddsub_out_lut<7> (m3/Madd
     MUXCY:S->O            0    0.353   0.000  m3/Maddsub_out_cy<7> (m3/Madds
     XORCY:CI->O           1    0.370   0.339  m3/Maddsub_out_xor<8> (product
     OBUF:I->O                  0.000          product_8_OBUF (product<8>)
    ----------------------------------------
    Total                      6.148ns (1.403ns logic, 4.745ns route)
                                       (22.8% logic, 77.2% route)
```

Figure 5:The timing report of APC based circuit

```
Timing constraint: Default path analysis
  Total number of paths / destination ports: 1855 / 9
-------------------------------------------------------------------------
Delay:               5.380ns (Levels of Logic = 10)
  Source:            x<1> (PAD)
  Destination:       f_prod<8> (PAD)

  Data Path: x<1> to f_prod<8>
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             14   0.001   0.815  x_1_IBUF (x_1_IBUF)
     LUT5:I0->O             1   0.097   0.439  b1/m1/Mmux_y_d[0]_MUX_8_o131 (b
     LUT3:I1->O             1   0.097   0.753  b1/m1/Mmux_y_d[0]_MUX_8_o132 (b
     LUT6:I0->O            15   0.097   0.517  s1/Mmux_s41 (prod<3>)
     LUT6:I4->O             5   0.097   0.594  Mmult_f_prod_Madd1_cy<4>1 (Mmul
     LUT5:I2->O             1   0.097   0.616  Mmult_f_prod_Madd1_cy<5>11 (Mmu
     LUT6:I2->O             1   0.097   0.000  Mmult_f_prod_Madd2_lut<7> (Mmul
     MUXCY:S->O             0   0.353   0.000  Mmult_f_prod_Madd2_cy<7> (Mmult
     XORCY:CI->O            1   0.370   0.339  Mmult_f_prod_Madd2_xor<8> (f_pr
     OBUF:I->O                  0.000          f_prod_8_OBUF (f_prod<8>)
    ----------------------------------------
    Total                      5.380ns (1.306ns logic, 4.074ns route)
                                       (24.3% logic, 75.7% route)
```

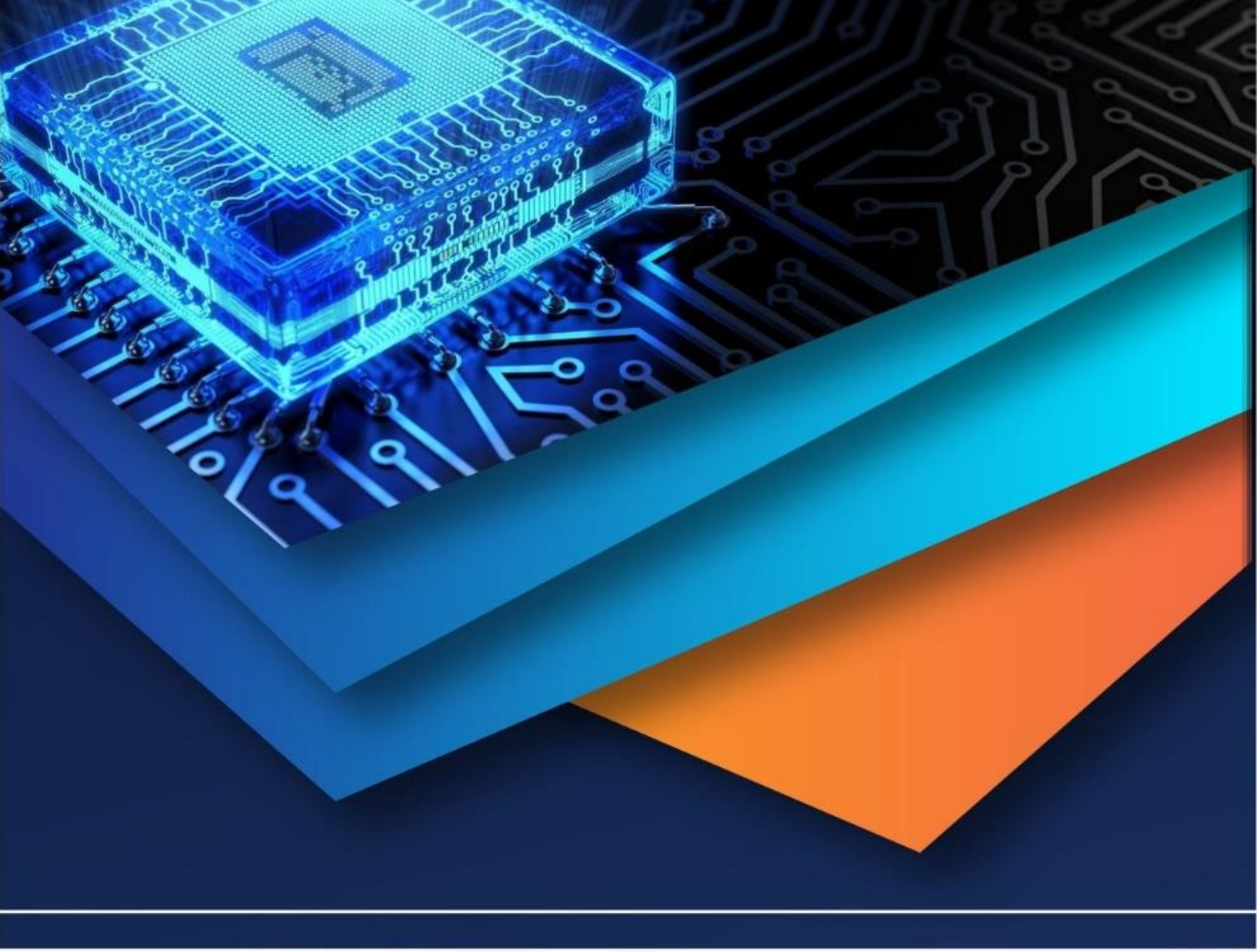
Figure 6: The timing report of APC-OMS based circuit

V. CONCLUSION

In this article, we demonstrated how LUT-based multipliers can be used to provide constant multiplication in DSP applications. However, if the LUTs are implemented as NAND or NOR read-only memories and the arithmetic shifts are done by an array barrel shifter using metal–oxide–semiconductor transistors, the full benefits of the proposed LUT-based system can be realized**.** Point to the future work is to observe this multiplier by incorporating it in FIR Filter and Adaptive Filter design. For reasonable area–delay tradeoffs, more work might be done to generate OMS–APC-based LUTs for larger input sizes using alternative types of decompositions and parallel and pipelined addition algorithms. As a result, the LUT multiplier based on APC-OMS can be employed in DSP applications such as interpolator designs. Furthermore, replacing the multipliers in the interpolator with the modified APC-OMS LUT based multiplier can improve operating speed while also reducing hardware complexity.

REFERENCES

[1] R R.Schaller," Technological innovation in the semiconductor industry: a case study of the international technology roadmap for semiconductors (itrs)," Ph.D. dissertation, George Mason University, 2004.
[2] P. K. Meher, "Memory-based hardware for resource-constraint digital signal processing system," in Information, Communications & Signal Processing, 2007 6th International Conference on .IEEE, 2007, pp.1–4.
[3] P. K. Meher, "New approach to LUT implementation and accumulation for memory-based multiplication," in Proc. IEEE ISCAS, May 2009, pp. 453–456.
[4] P. K. Meher, "New approach to look-up-table design and memory-based realization of fir digital filter," Circuits and Systems I: Regular Papers, IEEE Transactions on, vol. 57, no. 3, pp. 592–603, 2010.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY