Design And Development Of Fixed Width Radix 8 Booth Multipliers For High Accuracy And Low Power Operation

Bhumika Mandloi[1], Prof. Ashish Patidar[2]
[2] Professor
[1, 2] PCT Indore

Abstract- *Multiplier is one of the most essential and fundamental components in many multimedia and Digital Signal Processing (DSP) Systems. They play very important role since they can greatly influence the performance and the power dissipation of the system. Thus, for better performance of such systems, efficient realization of multiplier is very crucial. Many DSP applications employ fixed-point arithmetic where n bit signals are multiplied by n bit coefficients. To avoid infinite growth in the word size, 2n bit products obtained must be quantized to n bits. Also many multimedia systems maintain a fixed format and tolerate some loss in the accuracy where precision may be compromised for achieving improvisation in performance parameters, viz. speed, area and power dissipation. In such applications, fixed-width multipliers reverts to be one of the solutions towards enhancing the speed, reducing the area and power dissipation. Fixed-width multiplier is a n x n multiplier with n bits output. In fixed-width multipliers, half of the adder cells that add n least significant bits are removed and replaced by a suitable compensation bias to tradeoff accuracy and hardware cost. Various compensation techniques have been proposed in the literature for fixed-width radix-4 Booth multipliers to achieve high accuracy and low power in comparison with conventional multipliers. All these methods incur various performance metric and there exist a tradeoff between accuracy and hardware cost*

Keywords- Fixed-width Multiplier, Radix-8 Booth Encoding, Low-power Multiplier, High-Accuracy Multiplier.

I. INTRODUCTION

The dramatic increase in application of Very Large Scale Integrated (VLSI) circuits to numerous low power operated devices poses a bottleneck on power consumption, thereby demanding highly energy efficient and highly performing VLSI circuits. Energy consumption and suitable energy sources pose primary and significant critical design challenges. In order to evolve energy efficient VLSI circuits and systems, design of its individual functional blocks are to be characterized for energy efficiency. One of the significant circuit components in VLSI circuits that finds enormous applications in digital signal processing is the multiplier circuit, specifically, the fixed-width multiplier circuit. Owing to the importance of multiplication in numerous scientific and engineering computations, this research area has been given a lot of attention in the past decades. This has resulted in various design and implementation methodologies and techniques for multiplication process.

The diversity of application areas for multipliers demand different performance requirements such as speed, area, power consumption and other design parameters. Depending on these specifications and requirements, the priority on the design metrics of the particular multiplier do change. As per the priorities and requirements, the designer has to identify an appropriate multiplication algorithm. Traditionally, the top design priority had been given to the parameters such as speed and area. However, the rapid advancement of digital technology resulted in shift in the paradigm. Presently, other specifications such as low power, testability and reliability have been given importance. Power dissipation has proved to be one of the important design constraints in the design if digital systems. This tends to be more critical for battery-operated applications with extremely limited energy budget. In the present scenario, low power design has become one of the important area in the VLSI design and hence power aware design has become indispensable.

Rapid Progress in semiconductor technology has brought about System-on-Chip (SOC) era in the IC signal processing systems, including baseband, digital signal systems and DCT (He et al. 2015). With increase in transistor counts, clock frequencies, and desire for portability, the need for low power multipliers also increases. Further, consumer demand for increasingly portable and high performance product imposes stringent performance constraints on the individual internal components. Of these, multipliers are one of the most frequently encountered arithmetic operations in digital signal processors (DSPs). Multiplier circuits are one of the significant components in digital signal processing applications (Parhi 2007, Hsia and Wang 2007, Lee and Park 2007, Bougas et al. 2005) such as Discrete Cosine Transformation (DCT) (Hsia and Wang 2007), Fast Fourier Transform (FFT) (Lee and Park 2007) and Finite Impulse response (FIR) filter (Bougas et al. 2005). The performance of the multiplier circuits is crucial in affecting the overall performance with respect to overall speed, area, and power consumption of these applications. Further, multiplier circuits form a significant part in the data path of digital signal processing and multimedia systems. Owing to the importance of multipliers in many scientific and engineering applications, many researchers have suggested various techniques for efficient multiplication. Array multipliers, Dadda, Wallace Tree architectures, Baugh-Wooley, Booth multipliers are some of the standard methodologies adopted for multiplication. Out of these, Booth multipliers are highly sought owing to its high speed, less delay and low power consumption (Vaidya and Dandekar 2010).

Multiplication is one of the most common operations in many digital signal processing (DSP) applications, such as the Fast Fourier transform (FFT) and digital filtering, wavelet transforms etc. To avoid the bit-width growth, the multiplier outputs need to be truncated or rounded to a certain width. For n-bit inputs, conventional fixed-width multipliers perform the overall partial product summation before rounding or truncating the results to n-bit. In such post truncated multipliers, since all the adder cells are used to compute the 2n bit product, they produce more accurate outputs. However, this kind of multiplier incurs area overhead and high power dissipation. Thus, to overcome the above mentioned issues in post truncated multipliers, direct-truncated multipliers can be employed. In direct-truncation multipliers, half of the least significant partial products are simply eliminated by removing the adder cells which compute the least significant n bits of the 2n bit outputs. Area and power dissipation can be approximately reduced by 50% since half of the adder cells are removed. Since 50% of the adder cells are removed, it reduces critical path delay owing to reduced delay. However, huge truncation errors will be introduced. In order to realize a low-complexity and low-truncation error fixed-width multiplier, a compensation bias is estimated from the truncated part and is added to the retained adder cells i.e., the adder cells which add the most significant bits of the partial products.

II. LITERATURE REVIEW

The To achieve enhanced speed performance, parallel multipliers are preferred for high speed performance, though it leads to area complexity. In order to reduce chip area and to improvise the speed, many parallel multiplication algorithms (Wallace 1964, Baugh and Wooley 1973, Dadda 1965, K’andrea et al. 1995, Mahant-Shetti et al. 1999, Pezaris 1971) have been presented in the past. In the direction of reducing the chip area of the parallel multipliers without compromising on the performance, fixed-width multipliers can be adopted by exploiting the fact that the multiplication operations have fixed-width property in many DSP applications (Kuang et al. 1998, Terrell 1988). When an application demands a multiplication output which is n bits wide, it is superseded by a fixed-width multiplier. In the direct-truncated multiplier, partial product bits of LP are simply eliminated leading to an extremely huge truncation error. In the fixed width multiplier, a compensation value is estimated and added as a carry value to be propagated through LP to MP which in turn reduces the truncation errors considerably. Various compensation bias circuits have been proposed towards the implementation of fixed-width multipliers. For effective reduction of the truncation errors, the error compensation bias can be derived through constant scheme or by adopting adaptive scheme (variable correction scheme).

A radix-8 multiplier design for a specific purpose is presented in (Hidalgo et al. 1998). Multiplicand belongs to a previous known set of numbers which are stored in memory can be used to make modification in the radix-8 multiplication algorithm. Optimization is done based on prior knowledge of inputs.

A low power higher radix multiplication algorithm based on radix-16 32 x 32 Booth algorithm is presented (Kumar and Jayanthi 2015). Hard multiples are the major factors for power consumption in higher radix encoding. Technique uses radix-8 and radix-4 encoding along with radix-16 to avoid the hard multiples. Five bits group of radix-16 are separated and given to different encoders like radix-8 or radix-16 so that hard multiples will not be generated.

Performance comparison of radix-8 Booth multiplier has been presented in (Kishan Ramesh 2015). Multiplier used with carry save adder and kogge-stone adder has been proposed for speeding up the process of partial product addition.

Approximate adder has been employed to design low power approximate radix-8 Booth multiplier (Sindhuja and Thiruvenkatesan 2017). Coding is done in Verilog and synthesis, simulation is carried out using Xilinx ISE 14.5.

High performance modified radix-8 Booth multiplier has been presented by (G Manmadha Rao 2017) where in carry look ahead adder is used to improve the performance of the multiplier.

Power reduction has been achieved by avoiding the hard multiples in radix-16 encoding by employing radix-4 and radix-8 encoding (Jithin Kumar M V) Hybrid architecture is incorporated to achieve power and delay reduction. Synthesis has been carried out using Synopsis SDK 90nm, 1.3V standard cell library.

A modified 2D Discrete Wavelet Transform (DWT) architecture employing 16-bit radix-8 Booth multiplier has been proposed by Kumar et al. (2019). Replacing the existing Canonical Signed Digit (CSD) by 16-bit radix-8 Booth multiplier, better performance has been achieved in terms of area and power. n bits required by the specific coefficient is retained and the remaining n bits are truncated and the architecture occupies less number of clock cycles.

Performance comparison of higher radix Booth multiplier using 45nm technology has been carried out by Jasbir Kaur (2016). Performance comparison has been carried out for 60-bit multiplier using radix-2, radix-4, radix-8 and radix-32 multipliers. Coding has been done in Verilog and Synthesis is carried out using Cadence RTL Compiler.

III. PROPOSED ALGORITHM

The This section details the multiplier and its inputs and outputs. The multiplier has two 8 bit inputs to load both the multiplicand and multiplier. The LOAD signal is a negative edge triggered signal, while the clock is a positive edge triggered signal. CLR is asserted high to clear all registers.

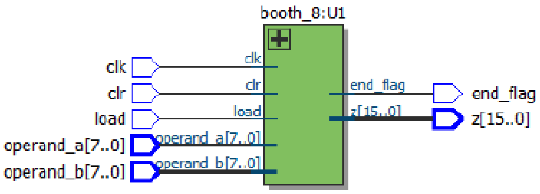


Figure 1: Booth8 Logic Device

Since the design requirements constitute a structural implementation, each unit must be carefully designed without the use of sequential VHDL features such as processes. Because of this, the Booth8 multiplier is designed from a bottom-up method. Each individual unit and gate is purposefully built and then verified. Once the units have been verified, they are used in other larger modules which are exposed to same process of verification. Additionally, the top level module being the multiplier will be subjected to multiplying its theoretical smallest and largest values along with zeros to ensure proper functionality. It should also be said that the design is focused to allow for code reuse with smaller functional blocks are re-used to build larger units.

The Booth multiplier has some substantial speed benefits as described previously due to the ability to substantially reduce the number of partial products needed. That being said, sign extending comes at a cost. Sign extensions are performed with additional fan-out, which ultimately delays performance and increases overall power consumption.

Also, while there are less additions needed, the additions are still large. Because of this, simple ripple carry adders will not suffice because of their inherent delay. Since speed is of utmost concern in this implementation, carry select adders are used. The use of these adders are purposefully selected so that redundant additions are not performed such as adding zeros where partial products are offset. This will be discussed in greater detail later on in the report.

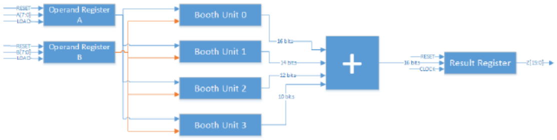


Figure 2: Booth8 Block Diagram

The pipelined variant operates in a similar manner as the non-pipelined variant. The difference lies in the implementation of additional registers which serve as pipeline stages. The implementation involves two pipeline stages:

1. Partial products generated by Booth Units and data valid ag stored in registers
2. Partial products added, producing the final result and stored into a register, along with the data valid flag

A simplified block diagram can be seen in Figure 3

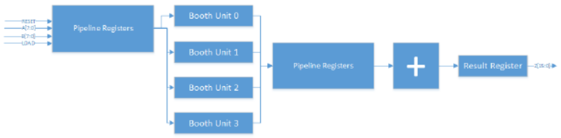


Figure 3: Booth8 Pipelined Implementation Block Diagram

IV. RESULTS

All the experiment analysis is done by Xilinx 14.1i in Vertex device family. The most important advantage of this tool is less memory with high speed analysis and complex logical circuit. The following are the main designing parameters for any reversible logic circuits:

- Digital circuits must use less number of constant inputs.
- These circuits must have less number of logic gates.
- Likewise, the intended circuits essential to be improved to produce less number of garbage outputs.

Booth4 Multiplier: At this point, the multiplier is finally implemented. An inversion of the LOAD signal is fed through a D Flip Flop to dictate if the result produced is valid. If LOAD is high, it is likely that the operands and the result are likely going to change very soon. Additionally, the D Flip Flop is used so that if the reset is active, any device that reads the output from the multiplier will know that the result obtained is not valid due to resetting. Two inverters are present to invert the load and clear signals rather than have the two inverters implemented on each register. The implementation is shown in Figure 4 and the result in Figure 5

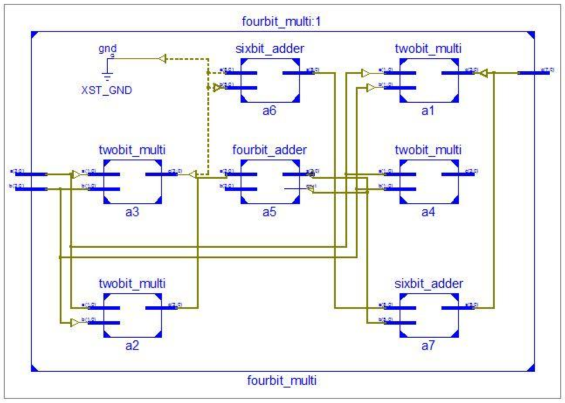


Figure 4: RTL view of Booth4 Multiplier

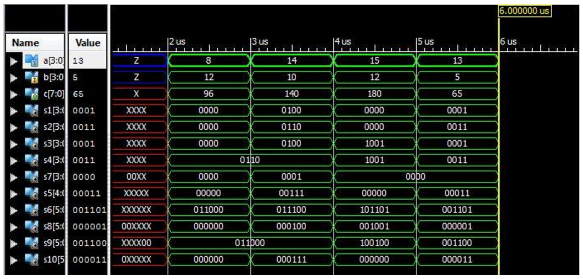
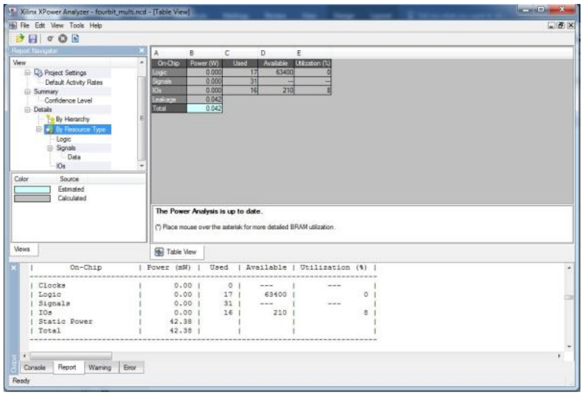

6.000000 us
Name
Value
2 us
3 us
4 us
5 us
6 us
a[3:0] 13 Z 8 14 15 13
b[3:0 5 Z 12 10 12 5
c[7:0] 65 X 96 140 180 65
s1[3:(0001 XXXX 0000 0100 0000 0001
s2[3:(0011 XXXX 0000 0110 0000 0011
s3[3:(0001 XXXX 0000 0100 1001 0001
s4[3:(0011 XXXX 0110 1001 0011
s7[3:(0000 00XX 0000 0001 0000
s5[4:(00011 XXXXX 00000 00111 00000 00011
s6[5:(001101 XXXXXX 011000 011100 101101 001101
s8[5:(000001 00XXXX 000000 000100 001001 000001
s9[5:(001100 XXXX00 011000 100100 001100
s10[5 000011 0XXXXX 000000 000111 000000 000011

Figure 5: Booth4 Multiplier Simulation



A	B	C	D	E
On-Chip	Power (W)	Used	Available	Utilization (%)
Logic	0.000	17	63400	0
Signals	0.000	31	---	---
IOs	0.000	16	210	8
Leakage	0.042			
Total	0.042			

The Power Analysis is up to date.

(*) Place mouse over the asterisk for more detailed BRAM utilization.

```
|        On-Chip        | Power (mW) |  Used  | Available | Utilization (%) |
----------------------------------------------------------------------------
| Clocks                |       0.00 |      0 |    ---    |       ---       |
| Logic                 |       0.00 |     17 |   63400   |               0 |
| Signals               |       0.00 |     31 |    ---    |       ---       |
| IOs                   |       0.00 |     16 |    210    |               8 |
| Static Power          |      42.38 |        |           |                 |
| Total                 |      42.38 |        |           |                 |
----------------------------------------------------------------------------
```

Figure 6: XPower Analyzer for Booth 4 Multiplier

Table 1: Device utilization summary for Booth4 Multiplier

S.No	Slice Logic Utilization	Used
1	Number of Slice LUTs	17
2	Number of occupied Slices	6
3	Number of bonded IOBs	16
4	Average Fanout of Non-	3.12
5	Delay (nSec)	3.15
6	Total memory usage (KB)	259512
7	Power Consumed (mW)	42.38

Booth8 Multiplier: At this point, the multiplier is finally implemented. Two inverters are present to invert the load and clear signals rather than have the two inverters implemented on each register. The implementation is shown in Figure 7 and the result in Figure 8.

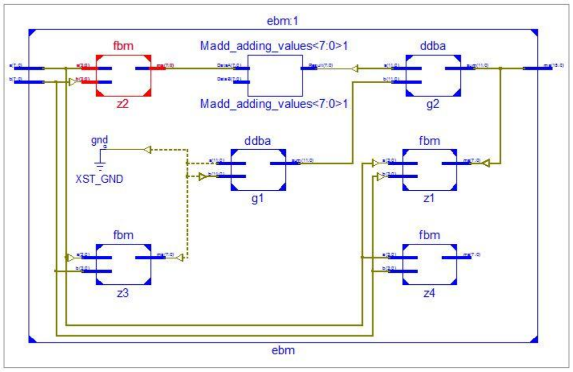


Figure 7: RTL view of Booth8 Multiplier

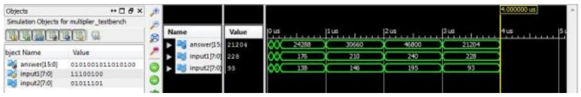
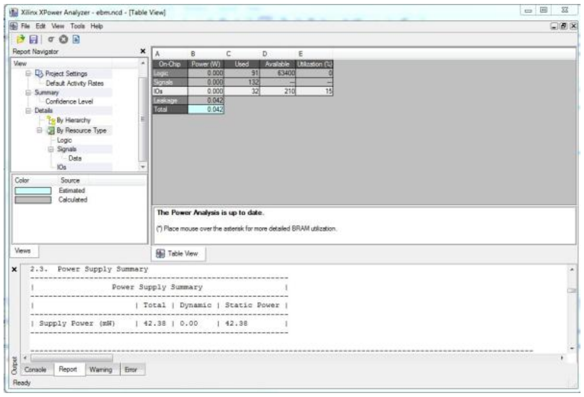

Objects
Simulation Objects for multiplier_testbench
bject Name
Value
answer[15:0] 0101001011010100
input1[7:0] 11100100
input2[7:0] 01011101
4.000000 us
Name
Value
0 us
1 us
2 us
3 us
4 us
5 u
answer[15: 21204 24288 30660 46800 21204
input1[7:0] 228 176 210 240 228
input2[7:0] 93 138 146 195 93

Figure 8: Booth8 Multiplier Simulation



A	B	C	D	E
On-Chip	Power (W)	Used	Available	Utilization (%)
Logic	0.000	91	63400	0
Signals	0.000	132	--	--
IOs	0.000	32	210	15
Leakage	0.042			
Total	0.042			

The Power Analysis is up to date.

(*) Place mouse over the asterisk for more detailed BRAM utilization.

```
2.3.  Power Supply Summary
---------------------------------------------------------
|                  Power Supply Summary                 |
---------------------------------------------------------
|                      | Total | Dynamic | Static Power |
---------------------------------------------------------
| Supply Power (mW)    | 42.38 | 0.00    | 42.38        |
---------------------------------------------------------
```

Figure 9: XPower Analyzer for Booth8 Multiplier

Table 2: Device utilization summary for Booth8 Multiplier

S.No	**Slice Logic Utilization**	**Used**
1	Number of Slice LUTs	92
2	Number of occupied Slices	50
3	Number of bonded IOBs	32
4	Average Fanout of Non-Clock Nets	4.09
5	Delay (nSec)	6.929
6	Total memory usage (KB)	445044
7	Power Consumed (mW)	42.38

The following table is showing the comparison between proposed architecture and architecture suggested in base paper.

Table 3: Comparison between power consumed

Component	**Power consumed in Base Paper architecture (W)**	**Power consumed in Proposed architecture (W)**
Booth4	0.57	**0.042**
Booth8	0.608	**0.042**
Booth16	0.628	**0.042**

Table 4.4: Comparison between Area

Component	**LUT's used in**	**LUT's used in**
Booth4	22	**17**
Booth8	84	**80**
Booth16	94	**92**

V. CONCLUSION

The proposed Booth8 multiplier is synthesized for implementation on a field programmable grid array (FPGA). By synthesizing, the source code in VHDL is mapped to physical constructs on the FPGA. For this project, the FPGA used is a Xilinx Virtex-7. A simplified FPGA has numerous "slices", which are discrete units that contain a look-up table (LUT) and some number of flip-flops. A look-up table has n inputs and returns one output. When synthesizing HDL code, the synthesizer will use these look-up tables to implement combinational logic and possibly infer the flip-flops in the LUT. Once laid out, the synthesis tool will wire and link the slices to produce the circuits described in the HDL. A timing analysis can also be performed to determine if a circuit can operate at a given speed. The features of FPGA design toolkits is beyond the scope of this report, for more information please refer to the ISE Manual. The maximum operating frequency for the pipelined version was found to be 144 MHz. Also, the power consumed is less than the base paper.

REFERENCES

[1] Stan, M. R. and Burleson, W. P. (1997), 'Low-power encodings for global communication in cmos vlsi', IEEE Transactions on Very Large Scale Integration (VLSI) systems 5(4), 444–455.

[2] Bellaouar, A. and Elmasry, M. (2012), Low-power digital VLSI design: circuits and systems, Springer Science and Business Media.

[3] Kao, J. T. and Chandrakasan, A. P. (2000), 'Dual-threshold voltage techniques for low power digital circuits', IEEE Journal of Solid-state circuits 35(7), 1009–1018.

[4] Parhi, K. K. (2007), VLSI digital signal processing systems: design and implementation, John Wiley and Sons.

[5] Lee, H.-Y. and Park, I.-C. (2007), 'Balanced binary-tree decomposition for area efficient pipelined fft processing', IEEE Transactions on Circuits and Systems I: Regular Papers 54(4), 889–900

[6] Bougas, P., Kalivas, P., Tsirikos, A. and Pekmestzi, K. Z. (2005), 'Pipelined array based fir filter folding', IEEE Transactions on Circuits and Systems I: Regular Papers 52(1), 108–118.

[7] Vaidya, S. and Dandekar, D. (2010), 'Delay-power performance comparison of multipliers in vlsi circuit design', International Journal of Computer Networks and Communications (IJCNC) 2(4), 47–56.

[8] Baugh, C. R. and Wooley, B. A. (1973), 'A two's complement parallel array multiplication algorithm', IEEE Transactions on Computers 100(12), 1045–1047.

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

[10] Wallace, C. S. (1964), 'A suggestion for a fast multiplier', IEEE Transactions on electronic Computers (1), 14–17.

[11] Stine, J. E. (2012), Digital computer arithmetic datapath design using Verilog HDL, Springer Science and Business Media.

[12] Mahant-Shetti, S. S., Balsara, P. T. and Lemonds, C. (1999), ‘High performance low power array multiplier using temporal tiling’, IEEE Transactions on Very Large Scale Integration (VLSI) systems 7(1), 121–124.

[13] Kuang, S.-R., Jou, J.-M. and Chen, Y.-L. (1998), ‘The design of an adaptive on-line binary arithmetic-coding chip’, IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications 45(7), 693–706.

[14] Van, L.-D., Wang, S.-S. and Feng, W.-S. (2000), ‘Design of the lower error fixed-width multiplier and its application’, IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing 47(10), 1112–1118.

[15] Joshi, P., Deshmukh, R., Pandey, K. and Admane, S. (2015), A novel combined approach of constant correction and variable correction method to minimize the mean square error and compensation hardware for fixed width multiplier design, in ‘2015 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)’, IEEE, pp. 1–6.

[16] Vahdat, S., Kamal, M., Afzali-Kusha, A. and Pedram, M. (2019), ‘Tosam: An energy efficient truncation-and rounding-based scalable approximate multiplier’, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 27(5), 1161–1173.

[17] Kumar, M. J. and Jayanthi, K. (2015), ‘A low-power hybrid multiplication technique for higher radix hard multiples suppression’, Indian Journal of Science and Technology 8(13), 1.

[18] Kishan Ramesh, N. B. P. (2015), ‘Delay comparison of radix-8 multiplier using mac unit’, International Journal of Software and Hardware Research in Engineering 3(12), 33–38.

[19] Sindhuja, K. and Thiruvenkatesan, C. (2017), ‘Design of low power approximate radix- 8 booth multiplier’, International Journal of Engineering Research and Technology 5(17), 1–5.

[20] G Manmadha Rao, T. S. K. (2017), ‘Design of high performance modified radix-8 booth multiplier’, International Journal of Mechanical Engineering and Technology 8(8), 1376–1379.