A NEW METHODOLOGY FOR THE DESIGN OF MULTIPLIERS FOR EFFICIENT AREA-POWER SAVINGS

S. Harsha Sree[1], S. Hanumantha Rao[2]

[1]M. Tech –VLSID Student, Department of Electronics and Communications Engineering, Shri Vishnu Engineering College for Women (Autonomous), Bhimavaram, India

[2]Professor, Department of Electronics and Communications Engineering, Shri Vishnu Engineering College for Women (Autonomous), Bhimavaram, India.

ABSTRACT:

Approximate circuits can reduce the hardware overhead saving power consumption and occupied area. Digital processing at nanoscale accepts this as an attractive strategy. For computer arithmetic designs approximate computing is particularly interesting scenario. A new design approach is presented for multiplier approximation. Approximate arithmetic blocks such as half-adder, full-adder and 4-2 compressor are not applied directly to the partial products of the multiplier; instead the original partial products are transformed to new partial products and are applied. The approximation is used in two forms of multipliers. Two approximate new16-bit multipliers are presentedand their results are compared with the different multiplier designs. The multipliers are implemented in Verilog, synthesis is performed using Cadence Encounter RTL compiler and layout is generated with Cadence Encounter.The newly designed multipliersalong with different concept oriented multipliers are synthesized and were compared using 180nm and 45nm technologies.

Keywords: *Approximate circuits, approximate multipliers, 4-2 compressor, Cadence Encounter.*

I. INTRODUCTION

For signal processing and multimedia based applications, exact arithmetic units are not always necessary. They can be replaced with the approximate circuits. Research based on inexact computing is on upswing for error tolerant applications. In computer arithmetic designs, addition and multiplication are the most frequently used operations. Inexact computing aims at designing simplified approximate circuits for low power consumption and high performance based applications [1-2]. In [3], for digital signal processing applications, approximate full-adders are proposed at transistor level. For accumulating the partial products in the multiplier, their respective full-adders proposed were utilized.

For faster multiplier circuit designs compressors have been used widely for lowering the dissipation of power. Exact and optimized 4-2 compressor designs were proposed in [4-6]. Two designs of approximate 4-2 compressor circuits are presented in [7] and are used in the reduction tree of partial products of Dadda

Multiplier. These compressors produce nonzero results for input value zeros. The design which was opted eliminates this drawback reducing error in most significant parts of the multipliers.

In [8] segment based multipliers are discussed. In Static Segment Multipliers (SSM), multiplication is done by selecting proper segments of length p from n bit multiplier based on the leading one bit positions of the operands. Instead of n x n multiplication, p x p multiplication is done and then the result is extended to 2n bits by properly appending zeros. In [9] Partial Product Perforation (PPP) multiplier is discussed which excludes k successive partial products starting from position of j where $j \in [0, n-1]$ and $k \in [1, \min(n-j, n-1)]$ for an n-bit multiplier. In [10], an inaccurate 2 x 2 building block based on altering an entry in the k-map is proposed. This block is used to develop 8 x 8, 16 x 16 and other higher approximate multipliers. In [11] 16-bit Dadda Multipliers are designed achieving low area and power compared with the exact designs.

II. PROPOSED ARCHITECTURE

2.1 APPROXIMATE ARITHMETIC UNITS

Approximate units or blocks include half adders, full adders and 4-2 compressors for reduction of partial products of the multipliers. Table 1 shows approximate half-adder truth table.Table 2 shows approximate full adder truth table and Table 3 shows approximate 4-2 compressor truth table. Approximate half -adder, full-adder and 4-2 compressor are employed to solve the partial products. The sum and carry for half adder are shown in (1, 2):

$$Sum = y1 + y2 \quad (1)$$

$$Carry = y1 \bullet y2 \quad (2)$$

Table.1. Approximate Half-Adder truth table

Inputs		Correct outputs		Approximate Outputs		Absolute difference
y1	y2	Carry	Sum	Carry	Sum	
0	0	0	0	0	0	0
0	1	0	1	0	1	0
1	0	0	1	0	1	0
1	1	1	0	1	1	1

Table.2. Approximate Full-Adder truth table

Inputs			Correct outputs		Approximate outputs		Absolute Difference
y1	y2	y3	Carry	Sum	Carry	Sum	
0	0	0	0	0	0	0	0
0	0	1	0	1	0	1	0
0	1	0	0	1	0	1	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	1	0
1	0	1	1	0	1	0	0
1	1	0	1	0	0	1	1
1	1	1	1	1	1	0	1

The XOR gate in adders and compressors tend to more area and delay. In half adder XOR gate is replaced with OR gate to obtain the Sum output. In full-adder one of the two XOR gates in the calculation of Sum are replaced by OR gate. This leads to error in the last two cases. Absolute difference is the difference between approximate and the exact result.

The Sum and Carry signals are the outputs of these circuits. The weight of carry is more than the sum.Error in carry produces an error difference of two in the output. Approximation is done so that the absolute difference between the correct output and approximate output is set at one. The equations for Sum and Carry signals for the approximate full-adder are given in (4, 5):

$$W1 = y1 + y2 \quad (3)$$

$$Sum = W1 \oplus y3 \quad (4)$$

$$Carry = W1 \bullet y3 \quad (5)$$

Table.3. Approximate 4-2 compressor truth table

Inputs				Approximate outputs		Absolute Difference
y1	y2	y3	y4	Carry	Sum	
0	0	0	0	0	0	0
0	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	1	0	0
0	1	0	0	0	1	0
0	1	0	1	0	1	1
0	1	1	0	0	1	1
0	1	1	1	1	1	0
1	0	0	0	0	1	0
1	0	0	1	0	1	1
1	0	1	0	0	1	1
1	0	1	1	1	1	0
1	1	0	0	1	0	0
1	1	0	1	1	1	0
1	1	1	0	1	1	0
1	1	1	1	1	1	1

For the cases where all inputs are zero the approximate compressors in [6] produce nonzero output. This leads to high error in most significant parts of the partial product reduction tree. So an approximate 4-2 compressor isconsidered whichovercomes this drawback. In this three bits are required only if all the inputs are 1.If all the four inputs are one then the output “100” is replaced with “11” to maintain minimum error distance. One out of the three XOR gates is replaced with OR gate for sum computation.The expressions for Sum and Carry for approximate 4-2 compressor are shown in (8, 9):

$$W1 = y1 \bullet y2 \quad (6)$$

$$W2 = y3 \bullet y4 \quad (7)$$

$$Sum = (y1 \oplus y2) + (y3 \oplus y4) + W1 \bullet W2 \quad (8)$$

$$\text{Carry} = W1 + W2 \tag{9}$$

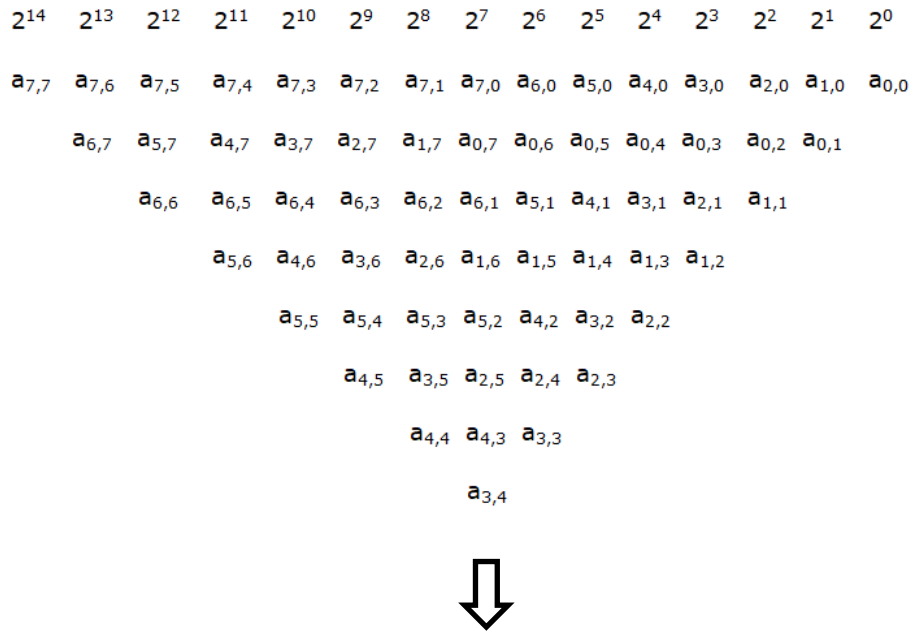
To make the Sum "1" corresponding to the inputs where all are ones, an additional term y1 • y2 • y3 • y4 is added to the expression of Sum.

2.2 8-BIT MULTIPLIER A

Multiplier implementation comprises three basic steps: partial products generation, reduction of partial products and the addition of sum and carry rows to obtain the final result. Of them partial product reduction stage utilizes more area and power. In this regard approximation is applied in this stage. To describe the proposed multiplier architecture, an unsigned 8-bit multiplier is employed.Consider two inputs $b = \sum_{l=0}^{7} b_l 2^l$ and $c = \sum_{m=0}^{7} c_m 2^m$ which are of 8-bit and are unsigned. The result of AND operation between the bits of b_l and c_m is given by the partial product $a_{l,m} = b_l \cdot c_m$ shown in Fig. 1.The partial products $a_{l,m}$ and $a_{m,l}$ are grouped to form modified partial products. The resulting signals form $p_{l,m}$(propagate) and $g_{l,m}$ (generate) signals shown in (10, 11):

$$p_{l,m} = a_{l,m} + a_{m,l} \tag{10}$$

$$g_{l,m} = a_{l,m} \bullet a_{m,l} \tag{11}$$



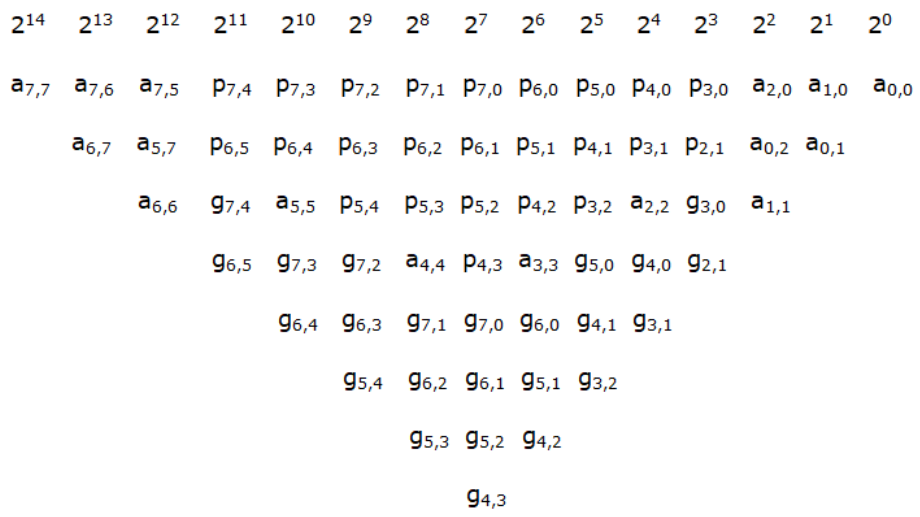


Fig. 1. Altering exact partial products to transformed partial products

The partial products from column 4 having weight 3 to column 12 having weight 11 are altered. The exact and altered partial product matrices are shown in Fig. 1.The arrangement of generate signals is done in a column manner. To group these, OR gates are used. As the number of generate signals increase in a column the error probability gets increased. So the number of OR gates used to combine the generate signals is kept at 4. If a column has m generate signals, [m/4] OR gates are used.

The reduction of transformed partial products of 8-bit inexact multiplier is shown in Fig. 2. Fig.2 shows the design of 8-bit Multiplier A where inexact computational units were used in all the columns of the multiplier. In the first stage, from column 4 to column 13, three approximate 4-2 compressors, one approximate full-adder, three approximate half-adders, four 2-input OR gates, four 3-input OR gates, one 4-input OR gate are used.

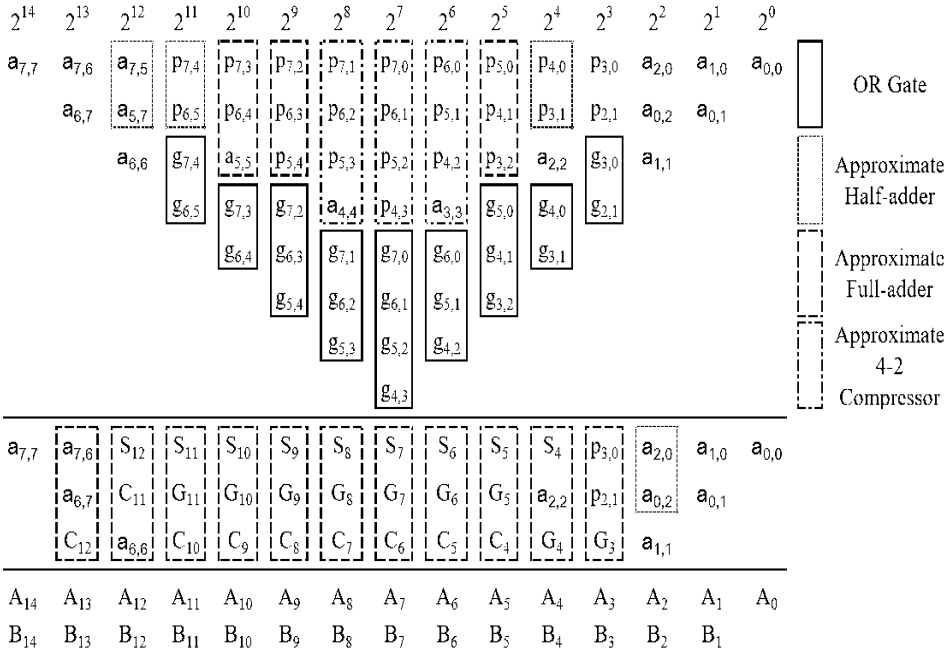


Fig.2. Design of 8-bit Multiplier A

In the second stage one approximate half adder, eleven full adders were used for reduction. The final level contains two rows A_i and B_i, are resolved by applying these to a ripple carry adder to obtain final sixteen bit result.Design of Multiplier B is same as Multiplier A whereas the approximate adders and compressors are applied to the least significant n-1 columns and exact units to the remaining most significant columns.

2.3. 16-BIT MULTIPLIER A

Fig. 3 shows the level 1 architecturefor 16-bit Multiplier A. From the level 1, the grouping of four indicates the approximate 4-2 compressor; three indicates the full-adder and two is the half adder. The generate signals designated as g are grouped using two, three and four input OR gates. The propagate signals designated as p are grouped using 4-2 compressors; three using full adders and two using half adders. Fig. 4 shows the levels 2, 3 and 4 designated as L1, L2 and L3. The s and c designations used in the levels 2,3 and 4 are the sum and carry signals. Sixteen rows in level 1 are reduced to six in level 2; four in level 3 and two in level 4.

Fig.3. Architecture for level 1 of 16-bit Multiplier A

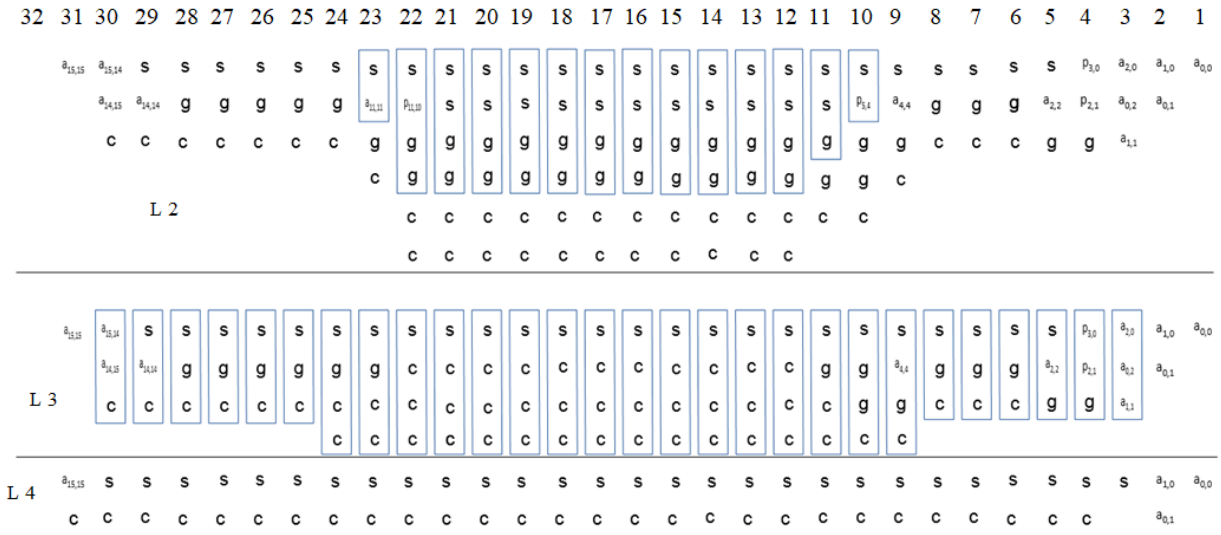


Fig.4. Architecture for level 2, 3 and 4 of 16-bit Multiplier A

2.4. 16-BIT MULTIPLIER B

Fig. 5 shows the level 1 architecture for 16-bit Multiplier B. From the level 1, the grouping of four indicates the approximate 4-2 compressor; three indicates the approximate full-adder and two is the approximate half adder for the least significant fifteen columns of the multiplier. The columns from sixteen to thirty two use exact half-adder, full-adder and 4-2 compressor units. The generate signals designated as g are grouped using two, three and four input OR gates. The result of OR gates in the level 1 are designated as g in levels 2 and 3. The c_{in} signal is the fifth input to the exact compressor from column sixteen.The arrow mark output from the compressor is the c_{out} signal given as an input to the compressors in the next higher columns. Fig. 6 shows the levels 2, 3 and 4 designated as L1, L2 and L3. The s and c designations used in the levels 2, 3 and 4 are the sum and carry signals. Sixteen rows in level 1 are reduced to six in level 2; four in level 3 and two in level 4. The rows in level 4 are given to the ripple carry adder to generate the 32-bit result.

Fig.5. Architecture for level 1 of 16-bit Multiplier B

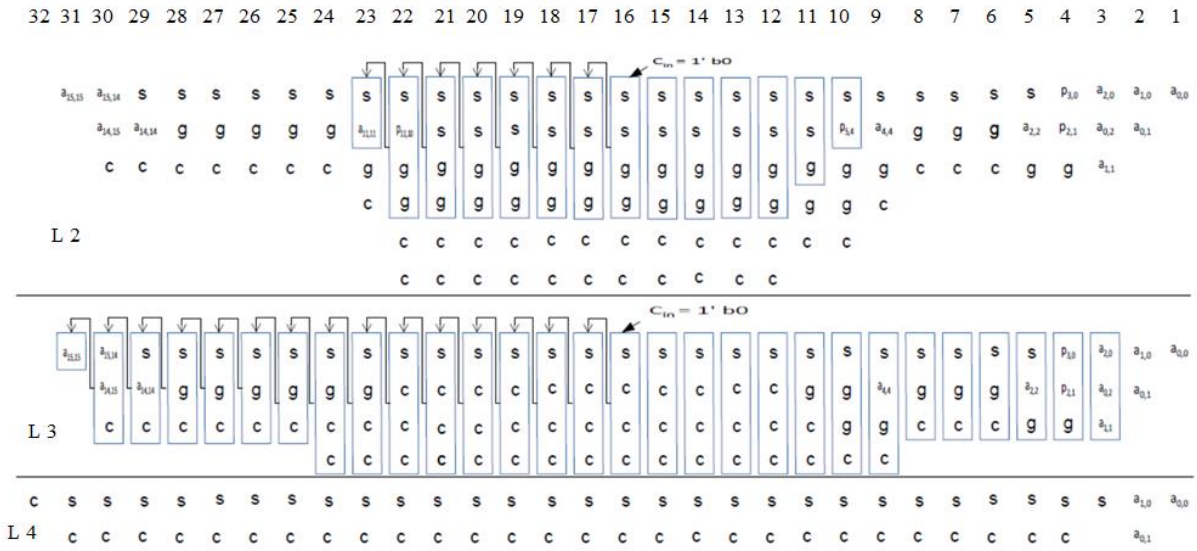


Fig.6. Architecture for level 2,3 and 4 of 16-bit Multiplier B

III. RESULTS AND DISCUSSIONS

Consider Table.4, where exact 16-bit Daddamultiplier is designed using tree structure. In MultiplierA, approximate blocks such as half-adder, full-adder and 4-2 compressor which are discussed in section II are applied to each column of the multiplier while to the least significant n-1 columns in case of MultiplierB. Multipliers based on Compressor (MC1 and MC2) architectures are implemented for 16-bit. Design2 from [7] is used to implement MC1 and MC2. For MC1 approximate compressor design is applied to all the columns of the multiplier whereas for n-1 least significant columns in case of MC2. Multipliers based on the segment length criteria are implemented from [8] designated as Static Segment Multiplier(SSM). The length of the segment used is 12-bit segment for 16-bit SSM design. Length of the result is 24-bit which is extended to 32 bit by properly appending zeros at appropriate positions. Multiplier based on Partial Product Perforation (PPP) is designed [9] for j=2,k=2 for a 16-bit multiplier. An Under Designed Multiplier (UDM) of 16-bit is implemented using the 2x2 approximate building blocks in [10].

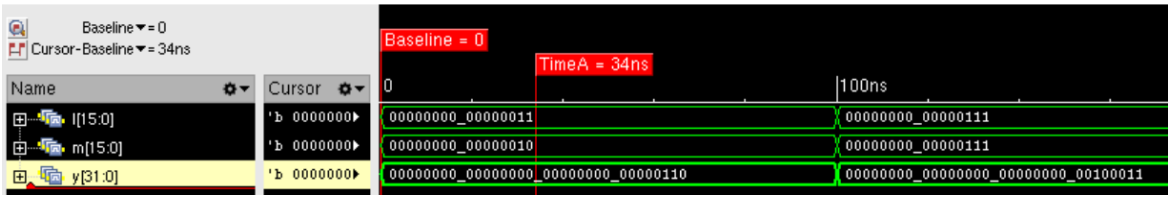


Fig.7. Simulation results of 16-bit MultiplierA

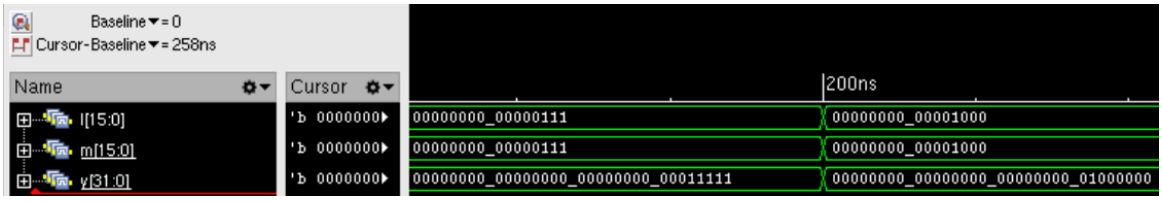
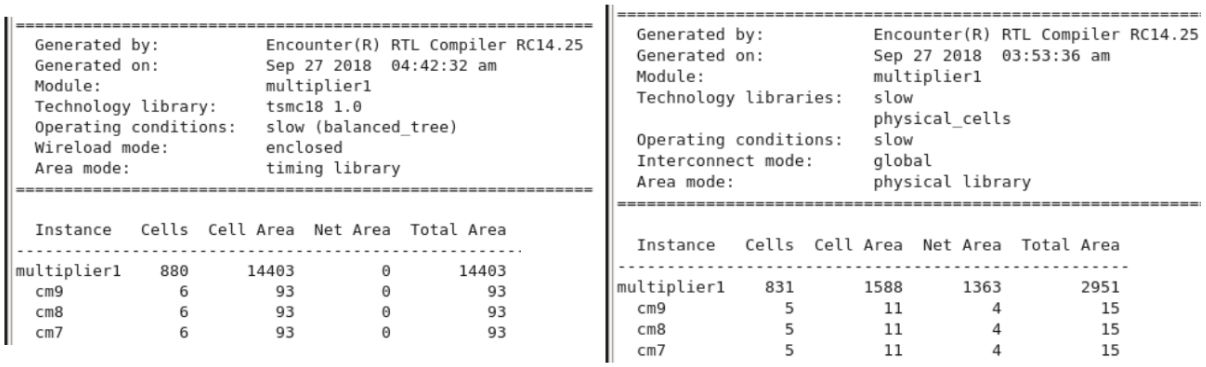


Fig.8. Simulation Results of 16-bit MultiplierB

Multipliers in Table 4 are designed for n=16. Table 4 provides the area, power and area-power product results of different multipliers along with the newly designed MultiplierA and B and their comparison for 180nm and 45nm technologies.



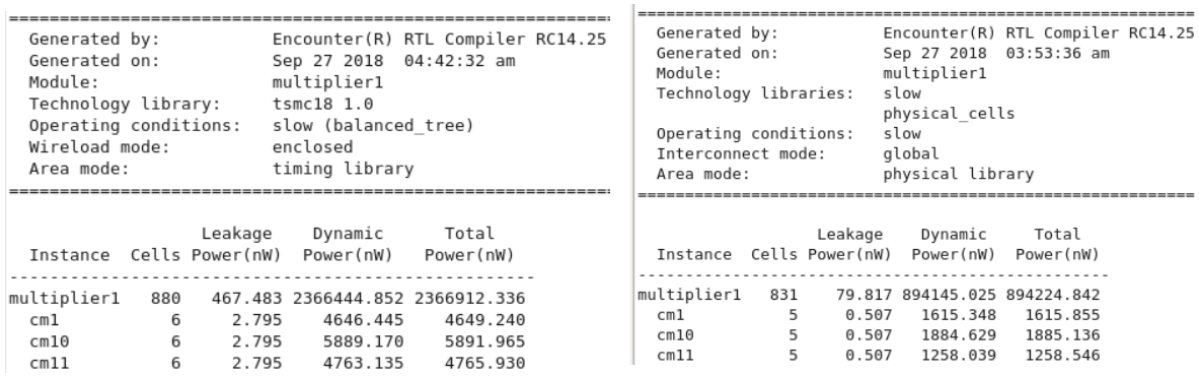
```
  Generated by:           Encounter(R) RTL Compiler RC14.25
  Generated on:           Sep 27 2018  04:42:32 am
  Module:                 multiplier1
  Technology library:     tsmc18 1.0
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library

  Instance   Cells  Cell Area  Net Area  Total Area
multiplier1    880      14403         0       14403
  cm9            6         93         0          93
  cm8            6         93         0          93
  cm7            6         93         0          93
```

```
  Generated by:           Encounter(R) RTL Compiler RC14.25
  Generated on:           Sep 27 2018  03:53:36 am
  Module:                 multiplier1
  Technology libraries:   slow
                          physical_cells
  Operating conditions:   slow
  Interconnect mode:      global
  Area mode:              physical library

  Instance   Cells  Cell Area  Net Area  Total Area
multiplier1    831       1588      1363        2951
  cm9            5         11         4          15
  cm8            5         11         4          15
  cm7            5         11         4          15
```

Fig.9. Area results for 16-bit Multiplier A (a)180nm technology (b)45nm technology



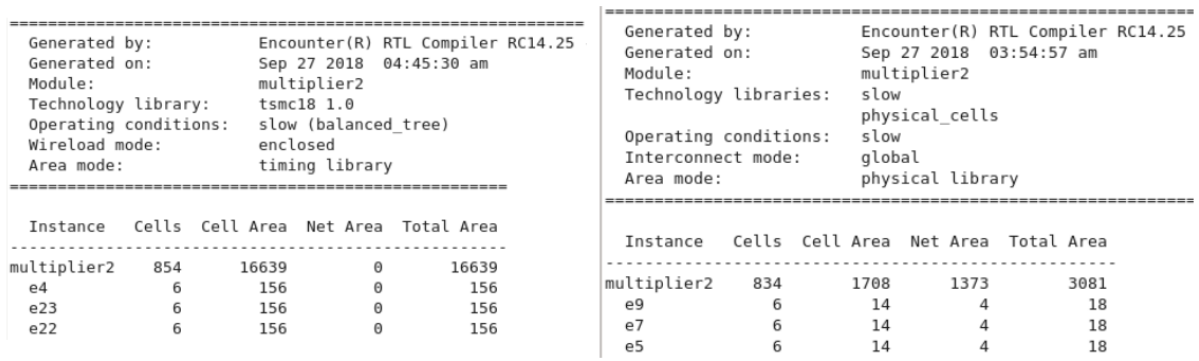
```
  Generated by:           Encounter(R) RTL Compiler RC14.25
  Generated on:           Sep 27 2018  04:42:32 am
  Module:                 multiplier1
  Technology library:     tsmc18 1.0
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library

                    Leakage    Dynamic      Total
  Instance  Cells Power(nW)  Power(nW)   Power(nW)
multiplier1   880   467.483 2366444.852 2366912.336
  cm1           6     2.795    4646.445    4649.240
  cm10          6     2.795    5889.170    5891.965
  cm11          6     2.795    4763.135    4765.930
```

```
  Generated by:           Encounter(R) RTL Compiler RC14.25
  Generated on:           Sep 27 2018  03:53:36 am
  Module:                 multiplier1
  Technology libraries:   slow
                          physical_cells
  Operating conditions:   slow
  Interconnect mode:      global
  Area mode:              physical library

                    Leakage    Dynamic     Total
  Instance  Cells Power(nW)  Power(nW)  Power(nW)
multiplier1   831    79.817 894145.025 894224.842
  cm1           5     0.507   1615.348   1615.855
  cm10          5     0.507   1884.629   1885.136
  cm11          5     0.507   1258.039   1258.546
```

Fig.10. Power results for 16-bit Multiplier A (a) 180nm technology (b) 45nm technology



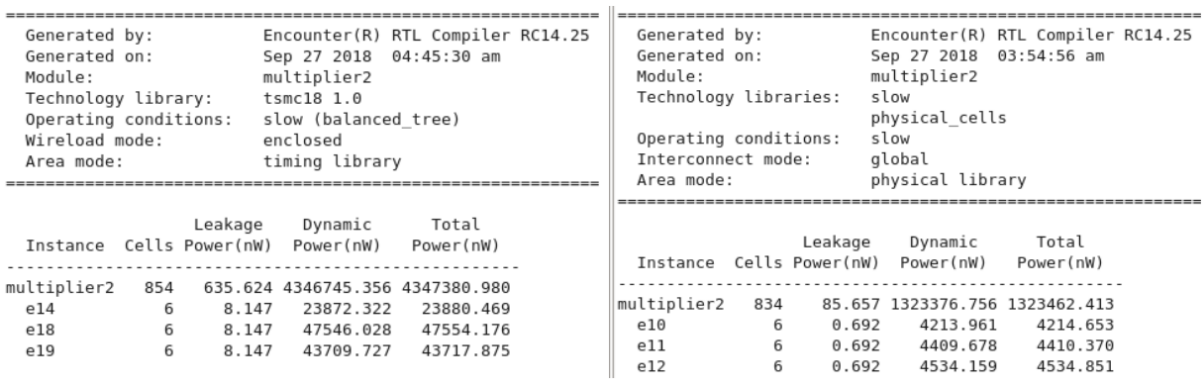
```
  Generated by:           Encounter(R) RTL Compiler RC14.25
  Generated on:           Sep 27 2018  04:45:30 am
  Module:                 multiplier2
  Technology library:     tsmc18 1.0
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library

  Instance   Cells  Cell Area  Net Area  Total Area
multiplier2    854      16639         0       16639
  e4             6        156         0         156
  e23            6        156         0         156
  e22            6        156         0         156
```

```
  Generated by:           Encounter(R) RTL Compiler RC14.25
  Generated on:           Sep 27 2018  03:54:57 am
  Module:                 multiplier2
  Technology libraries:   slow
                          physical_cells
  Operating conditions:   slow
  Interconnect mode:      global
  Area mode:              physical library

  Instance   Cells  Cell Area  Net Area  Total Area
multiplier2    834       1708      1373        3081
  e9             6         14         4          18
  e7             6         14         4          18
  e5             6         14         4          18
```

Fig.11. Area results for 16-bit Multiplier B for (a) 180nm technology (b) 45nm technology



```
============================================================
  Generated by:           Encounter(R) RTL Compiler RC14.25
  Generated on:           Sep 27 2018  04:45:30 am
  Module:                 multiplier2
  Technology library:     tsmc18 1.0
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

                    Leakage    Dynamic      Total
  Instance  Cells Power(nW)  Power(nW)   Power(nW)
------------------------------------------------------------
multiplier2   854   635.624 4346745.356 4347380.980
  e14           6     8.147   23872.322   23880.469
  e18           6     8.147   47546.028   47554.176
  e19           6     8.147   43709.727   43717.875
```

```
============================================================
  Generated by:           Encounter(R) RTL Compiler RC14.25
  Generated on:           Sep 27 2018  03:54:56 am
  Module:                 multiplier2
  Technology libraries:   slow
                          physical_cells
  Operating conditions:   slow
  Interconnect mode:      global
  Area mode:              physical library
============================================================

                    Leakage    Dynamic      Total
  Instance  Cells Power(nW)  Power(nW)   Power(nW)
------------------------------------------------------------
multiplier2   834    85.657 1323376.756 1323462.413
  e10           6     0.692    4213.961    4214.653
  e11           6     0.692    4409.678    4410.370
  e12           6     0.692    4534.159    4534.851
```

Fig.12. Power results for 16-bit Multiplier B for (a) 180nm technology (b) 45nm technology

Fig.9, 10 displays the synthesis results of area and power regarding 16-bit Multiplier A whereas Fig. 11,12 displays the results of area and power of 16-bit Multiplier B. The technology schematics of 16-bit Multiplier A and B are produced in Fig. 13 and layouts were displayed in Fig. 14. Multiplier A and B provides huge power and area savings when compared with the different multiplier designs presented.

Table.4. Synthesis results of Exact, MC1 and MC2,SSM, PPP, UDM and newly designed Multipliers A and B

Multiplier type	Technology (nm)	Power (pW)	Area (μm^2)	APP ($\mu m^2 \cdot pW)(10^7$)
Exact	180	6258	23371	14.62
	45	1660	3604	0.59
Compressor based Multiplier 1 (MC 1)	180	3227	16096	5.19
	45	1036	2781	0.28
Compressor based multiplier 2 (MC 2)	180	5161	19972	10.3
	45	1431	3262	0.40
Static Segment Multiplier (SSM)	180	4888	13435	6.56
	45	1612	3890	0.62
Partial Product Perforation Multiplier (PPP)	180	5553	20411	11.33
	45	1528	3162	0.48
Under Designed Multiplier (UDM)	180	5806	17101	9.92
	45	1548	2797	0.43
Multiplier A	180	2366	14403	3.4
	45	894	2951	0.26
Multiplier B	180	4347	16639	7.23
	45	1323	3081	0.47

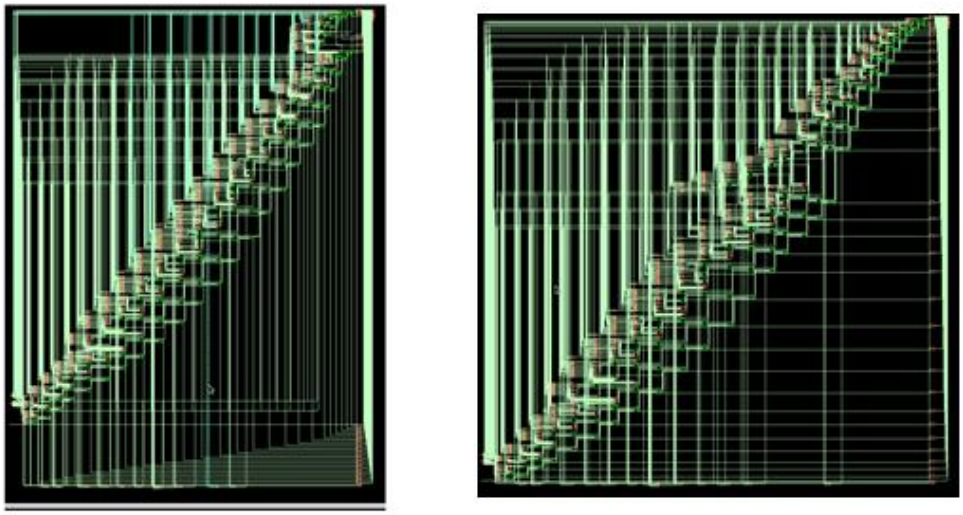


Fig.13. Technology Schematic of 16-bit (a) MultiplierA (b) Multiplier B

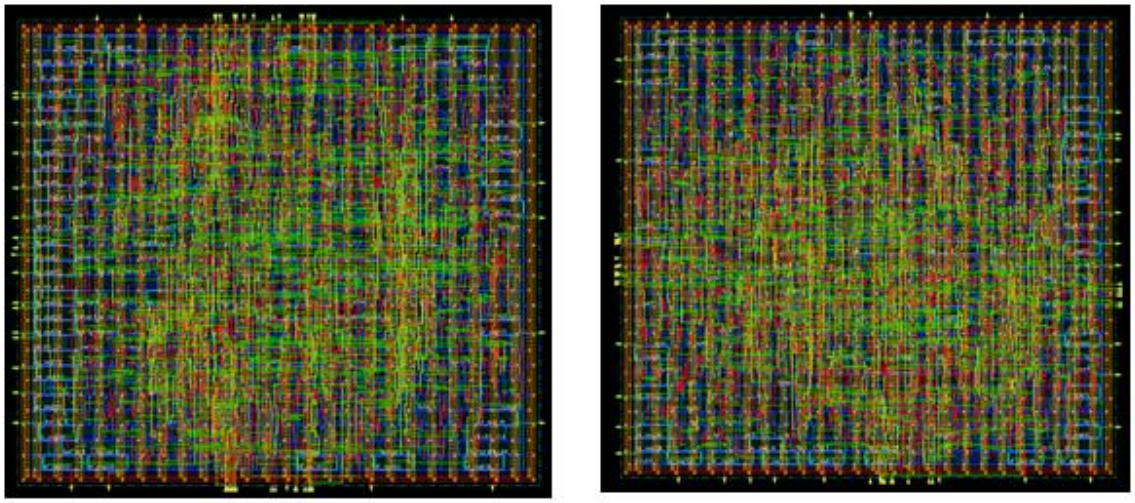


Fig.14. Layout of 16-bit (a) MultiplierA (b) Multiplier B

IV.CONCLUSION

In this scenario, for effective approximate multipliers to be proposed, the existing partial products are modified using propagate and generate signals. Generate signals were resolved using OR gates. Approximate half-adders, full-adders and 4-2 compressors are used to solve the left-over partial products. Two forms of approximate multipliers were proposed where approximation is applied to all columns of the multiplier in the former case and to n-1 least significant columns in the latter one. These effective proposed multiplier designs can be used in applications where power and area can be saved in an efficient manner.

REFERENCES

[1]. H. R. Mahdiani, A. Ahmadi, S.M. Fakhraie, and C. Lucas, "Bio-inspired imprecise computational blocks for efficient VLSI implementation of soft computing applications," *IEEE Trans. Circuits Syst. I, Reg. Papers, vol*. 57, *no. 4, pp. 850-862, Apr. 2010.*

[2]. J. Liang, J. Han, F. Lombardi, "New Metrics for the Reliability of Approximate and Probabilistic Adders," *IEEE Transactions on Computers, vol.63, no. 9, pp.1760-1771, 2013.*

[3]. V. Gupta, D. Mohapatra, A. Raghunathan and K. Roy, "Low-power digital signal processing using approximate adders," *IEEE Trans. Comput. Aided Design Integr. Circuits Syst., vol 32, no. 1, pp. 124-137, Jan 2013.*

[4]. C. Chang, J. Gu, M. Zhang, " Ultra Low-Voltage Low- Power CMOS 4-2 and 5-2 Compressors for Fast Arithmetic Circuits," *IEEE Transactions on Circuits and systems, Vol. 51, No. 10,pp. 1985 – 1997, Oct. 2004.*

[5]. M. Margala and N. G. Durdle, "Low-power low-voltage 4-2 compressors for VLSI applications," *in Proc. IEEE Alessandro Volta Memorial Workshop Low-Power Design, 1999, pp. 84-90.*

[6]. J. Gu, C. H. Chang, "Ultra Low-voltage, low-power 4-2 compressor for high speed multiplications," *in Proc.36th IEEE Int. Symp. Circuits Systems, Bangkok, Thailand, May 2003.*

[7]. A. Momeni, J. Han, P. Montuschi, and F. Lombardi, "Design and analysis of approximate compressors for multiplication." *IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 12, no. 5, pp. 522-531, May 2004.*

[8]. S. Narayanamoorthy, H. A. Moghaddam, Z. Liu, T. Park, and N.S Kim, "Energy-efficient approximate multiplication for digital signal processing and classification applications*," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 23, no. 6, pp. 1180-1184, Jun 2015.*

[9]. G. Zervakis, K. Tsoumanis, S. Xydis, D. Soudris, and K. Pekmestzi, "Design-efficient approximate multiplication circuits through partial product perforation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 24, no. 10, pp.3105-3117, Oct. 2016.*

[10]. P. Kulkarni, P. Gupta, and M. D. Ercegovac, "Trading accuracy for power in a multiplier architecture," *J. Low Power Electron., vol. 7, no. 4, pp. 33-38.*

[11]. S. Venkatachalam, Seok-Bum Ko, "Design of Power and Area Efficient Approximate Multipliers" *IEEE Transactions on VLSI systems, vol 25, no. 5, pp. 1782-1786.*