International Journal of Engineering Research and General Science Volume 4, Issue 2, March- April, 2016
ISSN 2091-2730

Partial Product Compression Methods: A Study and Performance Comparison Using a Tree Structured Multipliers

Parameshwara M. C[*] and Srinivasaiah H. C[+]
*Assistant Professor, Department of E & CE, Vemana Institute of Technology, VTU, Bangalore-34, Karnataka, India.
*Email:pmcvit@gmail.com, Phone: +919620902171
[+]Professor, Department of TCE, Dayananda Sagar College of Engineering, VTU, Bangalore-78, Karnataka, India.
[+]Email:hcsrinivas@gmail.com

Abstract— This paper presents the study and performance comparison of two traditional 'partial product compression methods' (PPCMs) viz. Wallace and Dadda in terms of the design metrics (DMs) such as power, delay, power-delay-product (PDP), and Area. For performance comparison an $N\times N$-bit unsigned 'tree structured multipliers' (TSMs) have been considered as a case study with 'N' being varied from 5 to 16 bits. To design TSMs a low PDP 3:2 counter (1-bit Full adder) along with half adder (HA) cells have been considered. Further all the TSMs are designed in the Cadence's virtuoso tool environment using 90 nm based 'generic process design kits' (GPDK) and simulated using Spectre simulator with BSIMv3v (Level 49) based transistor models. The power, delay, and PDP DMs of TSMs under consideration are extracted at nominal room temperature with supply voltage of V_{DD}=1.2 V and input signal frequency F_{in} = 200 MHz. This study and comparison of TSMs provides an insight to tradeoff among the Wallace and the Dadda PPCMs in terms of DMs.

Keywords— Tree structured multipliers, 1-bit adder, Arithmetic circuits, Low power adder, 3:2 counter, 28-T hybrid adder, Partial product compression, Wallace multipliers, Dadda Multipliers.

INTRODUCTION

The rapid advancements that are taking place in modern 'wireless communication technologies' (WCT) mirrored by the advancements in 'semiconductor technologies' (ST), demands the design of energy efficient wireless communication systems [1, 2, 3] to establish a seamless communication between the two end users. The transceiver being an important block of wireless communication system plays an important role in meeting the stringent requirements that are imposed by both WCT and ST. Further the transceivers are classified into two types viz. analog and digital. The digital transceivers are the natural choice for energy efficient wireless communication systems because of their important features like low phase noise, easily programmable, higher integration, very precise and controllable performance etc. [4, 5]. The digital transceiver is augmented with many sub-systems, among these the frequency synthesizer is one important sub-system that needs a greater attention in view of meeting the stringent requirements imposed by the modern WCT. The traditional frequency synthesizers are classified into viz. analog, digital, and analog and mixed mode. Among these the digital frequency synthesizers such as 'direct digital frequency synthesizer' (DDFS) is more popular because of its unique characteristics such as low power, low phase noise, high resolution, and good spectral purity. Traditionally the DDFS can be designed using the conventional architectures such as ROM or ROM less or sine computation techniques [6, 7, 8, 9, 10]. Among these DDFS architectures, most of the architectures employs the multiplier as a computational element to perform various 'digital signal processing' (DSP) operations. Thus the performance of multiplier is very important in meeting the performance of frequency synthesizer and there by the overall performance of the transceiver of a wireless communication system.

The block diagram of an $N\times N$-bit unsigned multiplier is as shown in Figure 1, where 'N' is the number of bits in the multiplier (X)(or multiplicand (Y)). In this paper the word sizes of both X and Y are chosen to be equal. Further the block diagram shown in the Figure 1 has three important sub-blocks viz. the 'partial product generation' (PPG), the 'partial product compression' (PPC), and the final 'carry propagation' (CP) sub-blocks. The inputs for the PPG block are an 'N' bit multiplier 'X' and multiplicand 'Y', these inputs are either in binary or recoding formats such as Booth radix-4, 'canonical signed digit' (CSD), 'minimal signed digit' (MSD) or any other recoding formats and using these inputs the PPG block generates the partial products. In this paper both 'X' and 'Y' have been considered as natural binary numbers thus a two input 'AND' gate array is used to generate the partial products. The total number of partial products that are generated corresponding to an 'N' bit X and Y are N^2 partial products. The output of PPG that is N^2 partial products are applied as an input to the PPC block. The PPC block sums up the entire partial products efficiently using either the Wallace tree or the Dadda tree [11, 12] approach and reduces to two rows. The most critical element of the PPC techniques is a 3:2 counter, which is a 1-bit full adder (FA). A counter compresses (k-1) rows of partial products into $log_2(k)$ partial products [13]. The final two rows of compressed partial products that have been derived from the output of the PPC blocks are the input for the CP block. The CP block adds final two rows of the partial products generated at the output of PPC and produces an 'W' bit product term 'Z' (Figure-1), where the size of W=2×N *bits.*

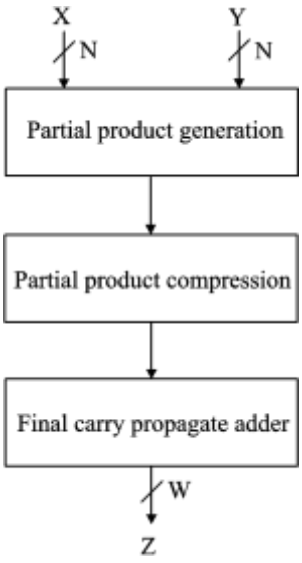


Figure 1: Block diagram of unsigned N×N-bit multiplier

PARTIAL PRODUCT COMPRESSION METHODS (PPCMs)

The most critical blocks in a conventional high speed multipliers are PPC and CPA blocks, these two blocks are very important in determining the overall speed of a multiplier. In general, the method of achieving the PPC using a 2:2 counter and a 3:2 counter is as shown in the Figure-2. Traditionally there are two partial product compression techniques viz. Wallace tree [11] and Dadda [12] tree based approaches and these are more frequently used to compress the partial products. The method of adding the partial products using Wallace and Dadda methods are illustrated using the '*dot*' diagrams and is shown in Figure 3 for a N=12- bits. In this figure, each '*dot*' in a particular 'Column-Row' represents an individual partial product, generated from the logical '*AND*' operation of the respective bits of '*X*' and '*Y*'. To explain the PPCM an unsigned 12 × 12 multiplier is taken as a case study.

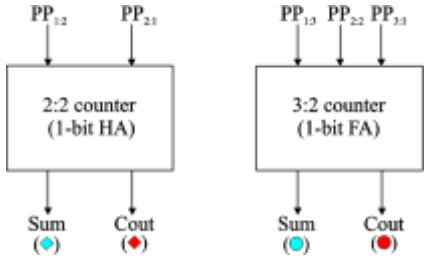
PPCM based on Wallace approach:

The Figure-3a is a PPCM based on Wallace tree approach, here the method of achieving the PPC between the PPG and final CPA is shown in five different stages viz. Stage-1 (S2), Stage-2 (S2), Stage-3 (S3), Stage-4 (S4), and Stage-5 (S5). The very first step in a Wallace tree based TSM is the generation and reorganization (shifting) of PPs. The main functionality of individual stages is explained as follows: In the S1 all the individual PPs that are generated out of PPG block in the form of a '12-rows' × '12-columns' matrix (herein referred as 'PP matrix (PPM)') is reorganized into the form of '12-rows' × '23-columns' (herein referred as RPPM). To reorganize the 12 × 12 PPM into the 12 × 23 PPM, the PPs in each row starting from the row-1 (R1) to the final row-12 (R12) are arranged such that each row under consideration is shifted towards left by 1-bit in position from its respective previous row. For N=12-bit, a total of 144 PPs are generated in PPG block and are reorganized into 12-rows and 23-columns. Each row of PPM contains about 12 individual PPs, after reorganization the PPs in each row starting from R1 to R12 are spread over 23-columns that is from column-1 (C1) to column-23 (C23). The 12 PP rows that are spread in 23-columns are grouped as group-1 (G1), group-2 (G2), group-3 (G-3), group-4 (G4) (Figure: 2a), where each group contains 3-rows. The 3-PP rows of each group are simultaneously compressed into 2-PP rows by using a 3:2 counter and 2:2 counters. The method of PP rows compression is explained as below:

Let 'R_i' be the i^{th} row of the group under consideration, where 'i' varies from 1, 2, 3, ... N,
Let 'C_j' be the j^{th} column of the group under consideration, where 'j' varies from 1, 2, 3, ... 2N, and
Let '$PP_{i:j}$' be the individual partial product of i^{th} row and j^{th} column in the group under consideration

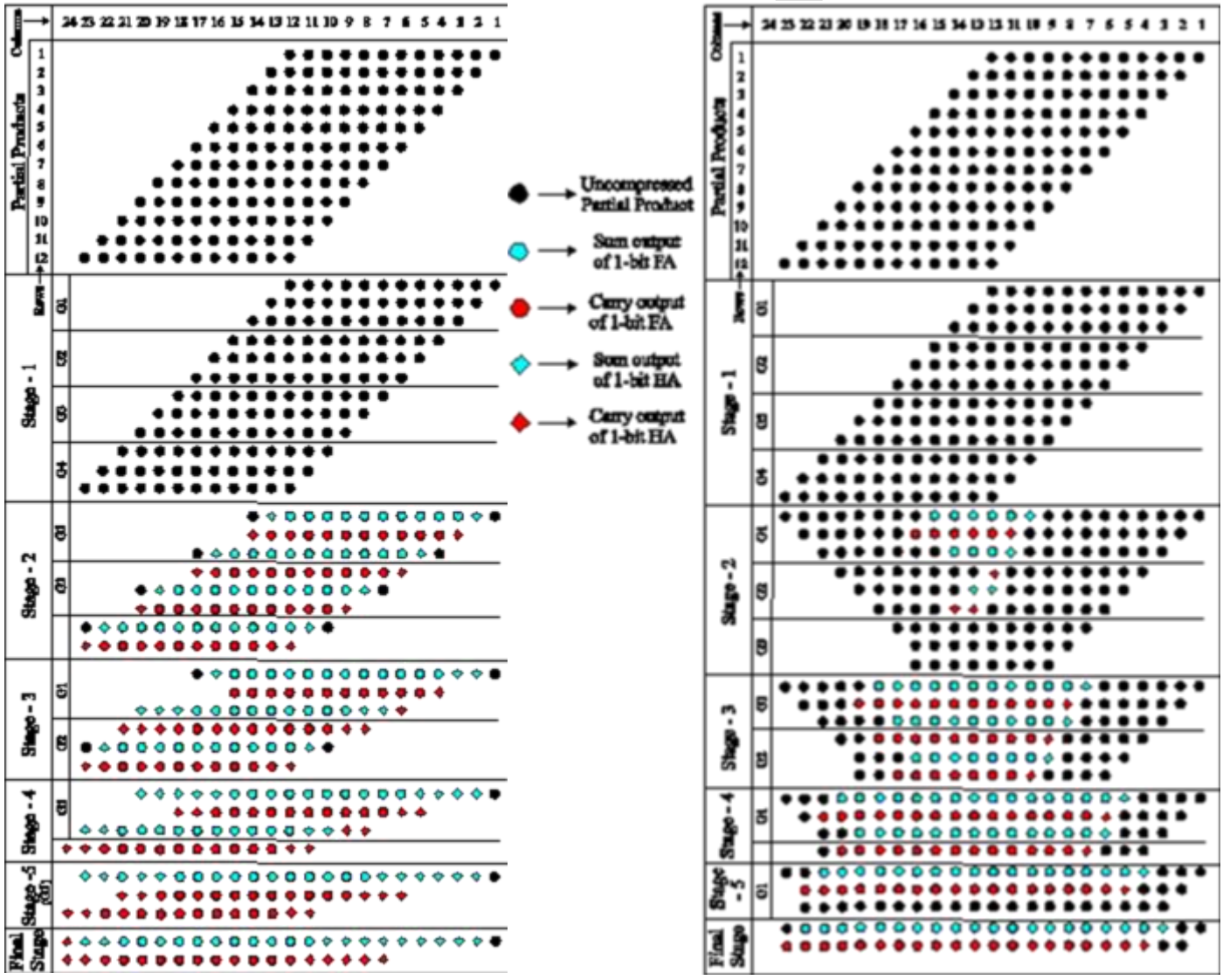
Considering the R1-R3 of G1 in S1, the $PP_{1:1}$ and $PP_{3:14}$ are the PPs that are not subjecting to compression (represented by dark solid circle '●') and are carried forward to the next stage (S1). The "$PP_{1:2}$ and $PP_{2:1}$" and "$PP_{2:13}$ and $PP_{3:13}$" are compressed by using a 2:2 counters. For example the PPs in the C2: $PP_{1:2}$ and $PP_{2:1}$ that are subjecting to compression are shown in the Figure- 2a. The PPs in the remaining respective rows and respective columns are compressed by using a 3:2 counters. For example the PPs in the C3 that are $PP_{1:3}$, $PP_{2:2,}$ and $PP_{3:1}$, subjecting to compression are shown the Figure: 2b below. The Sum and Cout outputs of a 3:2 counter (1-bit FA) are represented by solid circles with cyan and red colors respectively. Whereas, the Sum and Cout outputs of a 2:2 (1-bit HA) counter are represented by solid diamonds with cyan and red colors respectively. Thus the total 12-rows of PPs are reduced to 8-rows of PPs using a set of 3:2 counters and 2:2 counters in each group.

The second stage of compression that is S2 has total 8 rows of PPs, out of these 8-rows the first 6-rows of PPs are organized into two groups with each group being 3-rows of PPs. The last 2-rows of PPs remain ungrouped and are carried forward to the



(a) (b)

Figure 2: An example of PPC using a) 2:2 counter b) 3:2 counter



(a) (b)

Figure 3: Partial Product Compression Methods a) Wallace Tree based b) Dadda Tree based

next stage for further PPs compression. The grouped PPs are subjecting to compressions using a set of counters, this result into the reduction of 6-rows of PPs into 4-rows of PPs. The next level of PPs compression will be carried in the third stage (S3). The stage three (S3) has a total

6-rows of PPs, which are organized into 2-groups and these groups are subjecting to PPs compression using appropriate number of counters (3:2 and 2:2 counters).After PPs compression the 6-rows of PPs in S3 will reduce to 4-rows. These 4-rows are further subjecting to compression till the number of rows will reduce to 2-rows. The 4-rows of PPs that are the result of PPs compression of S3 will be further grouped and compressed in the next stage (S4). The S4 has total 4-rows of PPs, the first 3-rows of PPs are organized into one group and the remaining 1-row of PPs ungrouped and is carried forward to the next level for PPs compression. After compression of PPs the 3-rows of PPs will reduces to 2-rows of PPs, these 2-rows of PPs along with the 1-row of PPs (uncompressed in S4) are subjecting to compression in the next stage S5. The stage S5 has total 3-rows of PPs; these 3-rows of PPs are organized into single group and compressed using a set of counters. After compression the 3-rows of PPs will reduce to 2-rows, thus the PPs compression will be stopped at this level. Finally the 2-rwos of the compressed PPs will be added by using CPA adder to get the final product.

PPCM based on Dadda approach:

The PPCM using Dadda based approach is shown in Figure 3b, here also the generated PPs in the form of PPM is converted into the form of RPPM. The RPPM has total 12-rows of PPs and these rows are divided into 4-groups with each group being 3-rows. The divided groups are compressed and reduced to two rows and these two rows of PPs are further added by a CP adder to generate the final product of a multiplier. The compression of PPs into two rows is achieved through different stages viz. S1-S5 using 3:2 and 2:2 counters as shown in the Figure 3b. The number of stages required to compress the PPs can be determined by using Dadda algorithm. The important steps of Dadda algorithm for determining the number of stages required to compress the PPs in 12 × 12 multiplier are explained as follows:

Let $S_F = 2$ be the height (number of PPs rows) of the final stage or height of the CP stage, then the height of its preceding stages is calculated as given below.

$$S_5 = \lfloor 1.5 \times S_F \rfloor = 3$$

$$S_4 = \lfloor 1.5 \times S_5 \rfloor = 4$$

$$S_3 = \lfloor 1.5 \times S_4 \rfloor = 6$$

$$S_2 = \lfloor 1.5 \times S_3 \rfloor = 9$$

$$S_1 = \lfloor 1.5 \times S_2 \rfloor = 13$$

Since the multiplier under consideration is 12 × 12, therefore the maximum height of 13 is unnecessary and hence the Dadda algorithm converges. The Figure 3b is the dot diagram of 12 × 12 bit multiplier using Dadda based PPCM and has total 5-stages viz. S1, S2, S3, S4, and S5. The S1 is the RPPM and has a total of 12 PPs rows, the 12 rows are further divided into 4-groups viz. G1, G2, G3, and G4 with each group being 3-rows. Since the height of the S1 is 12, to achieve the S2 height = 9, the G1, G2, and G3 of S1 are subjecting to PP compression using appropriate number of 3:2 and 2:2 counters. This results in stage S2 with the height = 9. Thus the stage S2 has a total 9 rows of PPs in the form of RPPM and these rows are further divided into 3 groups G1, G2, and G3 with each group being of 3-rows of PPs. Further the G1, G2, and G3 of S2 are subjecting to compression to achieve the height of S3 = 6. The S3 has total 6 rows of PPs and grouped into G1 and G2. These groups that is G1 and G2 of S3 are subjecting to PPs compression to achieve a height of S4 = 4. Further in the 4 rows of S4, the first 3-rows are grouped as G1 and last row is remain ungrouped and is carried forward to the next stage. The G1 of S4 is subjecting to the PPs compression to achieve a height of S5 = 3. Thus S5 has a total 3 PPs rows and these 3 PPs rows are grouped as G1, the G1 is subjecting to PPs compression to achieve a height of final CP stage =2.

MICRO ARCHITECTURES OF TSM

The critical units of PPCM and final CP adder of the TSM is the 2:2 and 3:2 counters as shown in Figure-2. The 2:2 counter is a half adder that takes two inputs signals viz. A and B and produces the two output signals viz. the Sum and the Cout. The 3:2 counter is a pseudo 1-bit adder that takes 3-input signals viz. A, B, and Cin and produces two output signals viz. the Sum and the Cout. The micro-architectures of these counters are as shown in Figure 4. The Figure 4a is the micro-architecture of 2:2 counter (1-bit HA), this architecture has been designed by using static CMOS logic and it conceives a total 16-transistors (16Ts). The Figure 4b is the micro-architecture of 3:2 counter (1-bit FA), this architecture has been derived using a 'mixed logic style' (MLS) and conceives a total 28Ts [14]. The MLS based 3:2 counter (herein referred as 'MLSFA') combines the best advantages of the static CMOS logic and CMOS transmission gate logic. The MLSFA has the low power and low PDP advantages as compared to any other 1-bit FAs as reported in the literature [15].

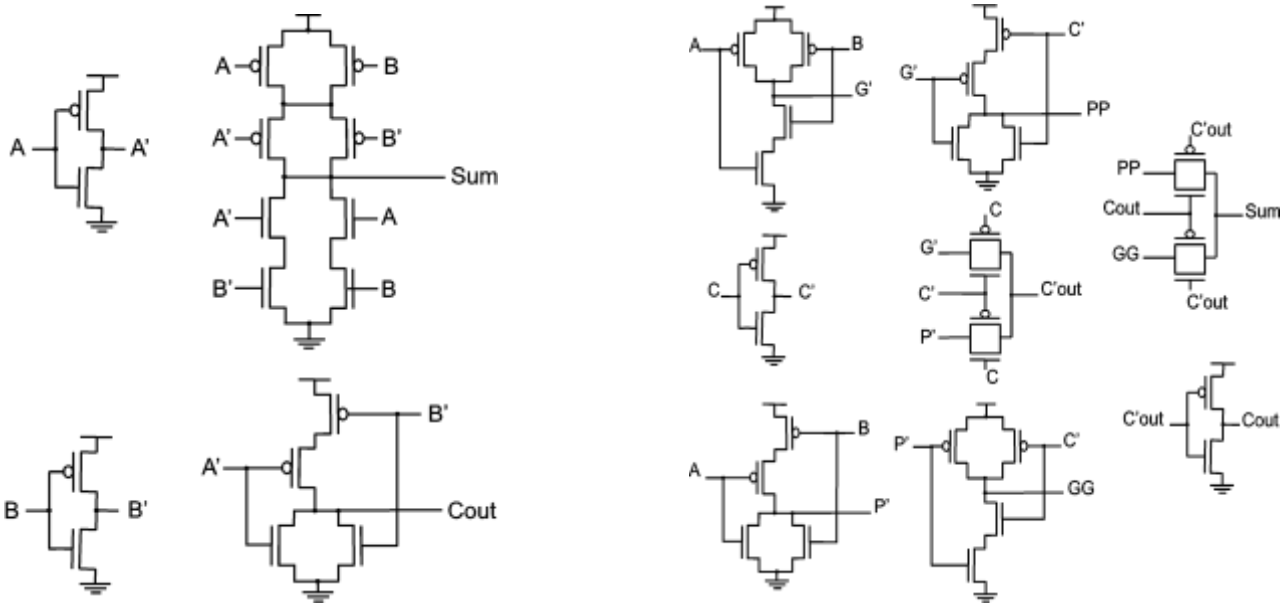


Figure 4: Schematic of a) 2:2 counter (1-bit HA) b) 3:2 counter (1-bit FA)

SIMULATION ENVIRONMENT TO EXTRACT DESIGN METRICS

The test bench that is used to extract DMs such as power, delay, and PDP of a TSM under consideration is as shown in the Figure 5. The 'circuit under test' (CUT) is the Wallace based or the Dadda based N × N bit TSM. All the TSMs under consideration are designed by using 1-bit FA and 1-bit HA as shown in Figure 4. To extract the DMs of a CUT under consideration we have used standard input test patterns as suggested in [16] and these patterns are listed in the Table-1. In the Table-1, the N in the first column represents the multiplier size, the 'PL' in the second column represents the 'propagation length' of the final CP adder, the 'X' and 'Y' represents the critical vectors that are used to derive the DMs of the respective N bit multiplier.

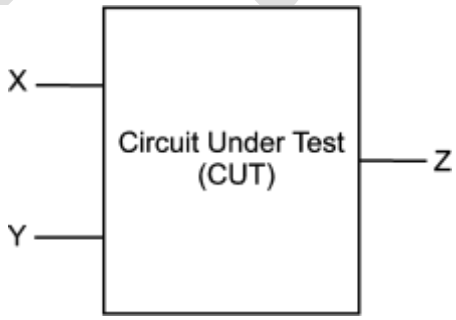


Figure 5: Test bench used to extract the power, propagation delay, and PDP of TSMs under consideration

All the TSM simulations have been carried out under common process-voltage-temperature (PVT) conditions with input signal clock frequency Fin = 200MHz. From the Table-1, it is observed that the set of test vectors (X and Y) for each N value are different. Also the propagation length (PL) varies corresponding to each set of test vectors and is linearly increasing with respect to N.

TABLE I
INPUT TEST VECTORS USED TO EXTRACT WORST CASE POWER AND DELAY OF TSMS

N	PL	**Multiplier (X)**	**Multiplicand (Y)**	**Product (Z)**
5	8	11011	10011	1000000001
6	8	011011	010011	1000000001
7	9	1101011	1000011	1110000000001
8	13	1111011	11000111	1100000000000001
9	14	1110101	111100111	111000000000000001
10	14	0111010	0111100111	111000000000000001
11	18	1111110	1110000111	11100000000000000001
12	18	0111111	0111000011	11100000000000000001
13	21	1011011	1111011100	101100000000000000000000001
14	24	1000000	0011111101	100000000000000000000000001
15	25	0100110	1001111100	1100000000000000000000000001
16	28	1011010	1110000110	10100000000000000000000000000000

PERFORMANCE COMPARISON OF TREE STRUCTURED MULTIPLIERS

The performance comparison of TSMs under consideration is discussed in terms of the DMs such as power, delay, and PDP. The DMs of the $N \times N$ TSMs that are extracted using the test bench of Figure-5 are listed in the Table- 2. From the Table-2, the important points that are observed for both 'Wallace tree based TSM' (WTSM) and 'Dadda based TSM' (DTSM) are discussed as below.

- Considering the first row of Table-2, with N=5 and PL=8; the power dissipation of the WTSM (14.27μW) is low as compared to DTSM (14.82μW), whereas the delay of WTSM (0.311 ns) and DTSM (0.312 ns) are comparable. Further the PDP of WTSM (4.45 fJ) is less than that of DTSM (4.62 fJ). The hardware requirement is same for both the TSMs. Thus for N=5 the WTSM is power and energy efficient than that of DTSM.

- From the second row of the Table-2, with N=6 and PL=8, again the WTSM is power and energy efficient as compared to DTSM. Whereas the hardware requirement in terms of adders is less for DTSM as compared to WTSM.

- Observing the third row of the Table-2, with N=7 and PL=9, the power and delay of WTSM and DTSM are comparable. Whereas the DTSM is more area efficient as compared to WTSM.

- For N=8 to N=16, it is observed that the DTSM outperforms the WTSM in terms of DMs viz., power, delay, PDP, and area.

- Considering the last row of Table-2, with N=16 and PL=28; the power dissipation of the WTSM (181.5 μW) is higher than that of a DTSM (174.5 μW), whereas the delay of WTSM (1.1 ns) is low as compared to DTSM (0.95 ns). Further the PDP of WTSM (199.6 fJ) is also higher than that of DTSM (166 fJ). Also the hardware requirement in terms of adders is more for WTSM as compared to DTSM.

- Thus from the comparison table, it is observed that the Wallace tree based PPCM is a good a choice for N=5 to N=7, whereas for N > 7, the choice is the Dadda based PPCM.

TABLE II

COMPARISON OF WALLACE AND DADDA BASED PPCM USING TSMS IN 90NM TECHNOLOGY NODE

		Wallace tree based TSM						Dadda tree based TSM					
N	PL	Average Power (uW)	Delay (ns)	PDP (fJ)	No. of HAs	No. of FAs	Total adders	Average Power (uW)	Delay (ns)	PDP (fJ)	No. of HAs	No. of FAs	Total adders
5	8	14.27	0.31145	4.45	5	15	20	14.82	0.3122	4.62	5	15	20
6	8	16.99	0.50025	8.50	10	23	33	17.87	0.52885	9.45	6	24	30
7	9	23.42	0.53255	12.47	13	34	47	23.39	0.5564	13.01	7	35	42
8	13	45.07	0.62105	27.99	16	47	63	43.24	0.40675	17.58	8	48	56
9	14	59.7	0.66815	39.88	22	61	83	57.5	0.58045	33.37	9	61	70
10	14	67.9	0.71735	48.70	26	79	105	66.02	0.635	41,92	10	80	90
11	18	99.98	0.7935	79.33	29	98	127	88.43	0.6848	60.55	11	99	110
12	18	99.76	0.82835	82.63	35	119	154	97.99	0.58235	57.06	12	120	132
13	21	112.1	0.8858	99.29	39	141	180	95.51	0.9084	86.76	13	141	154
14	24	102.4	0.9047	92.64	46	167	213	98.67	0.9284	91.60	14	168	182
15	25	138.4	1.06125	146.8	50	194	244	132.2	1.04225	137.78	15	195	210
16	28	181.5	1.1004	199.6	54	223	277	174.5	0.9513	166	16	224	240

CONCLUSION

In this paper, the two traditional PPCMs have been compared in terms of power, delay, PDP, and area (in terms of adders) by using two conventional TSMs. To compare the performance of PPCMs, the WTSM and DTSM (in which the PPCMs were embedded) are designed for the sizes of N=5 to N=16 and simulated using Cadence's EDA tools based on 90nm GPDK. The TSMs have implemented using a 2:2 counter and a 3:2 counters in Cadence's Virtuoso and simulations were carried out using Cadence's Spectre simulator based on 90nm BSIMv3v (level 49) transistor models. All the simulations of the circuits under considerations were carried out at common PVT conditions with Fin =200 MHz. From the simulation results it is observed that, for N=5 to 7, the WTSM (based on Wallace tree PPCM) has better power and energy efficiency as compared to DTSM (based on Dadda PPCM). For N>7 the DTSM has better performance in terms of power, delay, PDP, and area than the WTSM. Thus depending on the size of 'N' and an application, one can have an option to choose the required PPCM to trade-off between the DMs.

REFERENCES:

[1] Cheng-Xiang Wang et al.: "Cellular architectures and key technologies for 5G wireless communication networks" IEEE Communications Magazine, pp. 122-130, Feb. 2014.

[2] F. Akyildiz, David M. Gutierrez-Estevez, and Elias Chavarria Reyes: 'The evolution to 4G cellular systems: LTE-Advanced'. Elsevier Journal of Physical Communication, 2010, pp. 217-244.

[3] '2013PIDS_Summary.pdf', http://www.itrs.net/ITRS.

[4] Bennet C. Wong, and Henry Samueli, "A 200MHz All-Digital QAM Modulator and Demodulator in 1.2μm CMOS for Digital

Radio Applications," IEEE Journal of Solid State Circuits, vol. 26, no 12, pp. 1970-1979, Dec. 1991.
[5] F. P. Chan, M. P. Quirk, and R. F. Jurgens, "High-Speed Digital Baseband Mixer," TDA Progress Report 42-81, Communication System Research Section, January-March, pp. 63-80, 1985.
[6] Byung-Do Yang, Jang-Hong Choi, Seon-Ho Han, Lee-Sup Kim, and Hyun-Kyu Yu, "An 800 MHz Low-Power Direct Digital Frequency Synthesizer With an On-Chip D/A Converter," IEEE Journal of solid-state circuits, Vol. 39, no. 5, pp. 761-774, May 2005.
[7] Loke Kun Tan, and Henry Samueli, "A 200MHz Quadrature Digital Synthesizer/Mixer in 0.8μm CMOS," IEEE Journal of Solid State Circuits, Vol. 30, No. 3, pp. 193-200, March 1995.
[8] A. Yamagishi, M. Ishikawa, T. Tsukahara, and S. Date, "A 2-V, 2-GHz low-power direct digital frequency synthesizer chip-set for wireless communication," IEEE J. Solid-State Circuits, vol. 33, pp. 210–217, Feb. 1998
[9] A. M. Sodagar and G. R. Lahiji, "Mapping from phase to sine-amplitude in direct digital frequency synthesizers using parabolic approximation," IEEE Trans. Circuits Syst. II, vol. 47, pp. 1452–1457, Dec. 2000.
[10] J. M. P. Langlois and D. Al-Khalili, "ROM size reduction with low processing cost for direct digital frequency synthesis," in Proc. IEEE Pacific Rim Conf. Communications, Computers and Signal Processing, pp. 287–290, Aug. 2001.
[11] C. S. Wallace, "A suggestion for parallel multipliers," IEEE Trans. Electron Comput., vol. EC-13, no. 1, pp. 14–17, Feb. 1964.
[12] L. Dadda, "Some schemes for parallel multipliers", Alta Frequenza, vol. 34, March 1965.
[13] Wen-Chang Yeh, and Chein-Wei Jen, "High-speed booth encoded parallel multiplier design," IEEE Trans. Computers, Vol. 49, No. 7, pp. 692–701, July 2000.
[14] Parameshwara M. C. and Srinivasaiah H. C. "Study of Power-Delay Characteristics of a Mixed-Logic-Style Novel Adder Circuit at 90nm Gate Length", IJCA, vol. 119, no 4, pp 27-33, June 2015.
[15] Parameshwara M. C. and Srinivasaiah H. C. "Choice of Adders for Multimedia Processing Applications: Comparison of Various Existing and a Novel 1-Bit Full Adder", IOSR, vol. 10, no 3, pp 69-73, May 2015.
[16] Henrik Eriksson, Per Larsson-Edefors, and Daniel Eckerbert, " Toward Architecture-Based Test-Vector Generation for Timing Verification of Fast Parallel Multipliers," , IEEE Trans. VLSI systems, vol. 14, no. 4, April 2006.