Design of Energy-Efficient Multiplier Circuits

[1]B Sangeetha, [2]Y Praveen Kumar Reddy

[1]PG student, [2]Assistant Professor
[1,2] Department of Electronics and Communication Engineering,
[1,2]Mahatma Gandhi Institute of Technology, Hyderabad, India.

ABSTRACT

High-speed digital circuits face significant energy challenges, particularly in array-based multiplier designs. This work proposes an approximate arithmetic computing (AAC) model incorporating an Error Compensation Unit (ECU) to optimize error compensation values and enhance error resilience. The design introduces don't-care conditions to reduce energy, delay, and area. Applied to a 16×16 fixed-width Booth multiplier using a 90nm CMOS library, the model achieves 44.85% and 28.33% reductions in energy and area, respectively. It also lowers average, maximum, and mean square errors by up to 28.11%, and improves the energy-delay-mean square error product by 19.10% over existing approximate designs.

KEYWORDS: Arithmetic circuits; High-speed and energy efficiency; Multipliers.

I. INTRODUCTION

As the CMOS technology and VLSI design complexity scale, delivering desired functionalities while managing chip power consumption has become a first-class design challenge. To remedy this grand energy-efficiency challenge, approximate arithmetic circuits, in particular array-based approximate arithmetic computing (AAAC) circuits, have been introduced as a promising solution to applications with inherent error resilience, including media processing, machine learning and neuromorphic systems. AAAC may allow one to trade off accuracy for a significant reduction of energy consumption for such error-tolerant applications. To this end, approximate multipliers and squarers have been a focus of a great deal of past and ongoing work. Two types of approximate multipliers exist: approximate AND-array multipliers, which utilize AND gates for partial product generation [1]-[2] and approximate Booth multipliers [3]-[8], which use the modified Booth algorithm to reduce the number of partial products. For squarer units, a series of approximate squarers have been proposed [9]-[11].

While a diverse set of array-based approximate arithmetic unit designs exists, what is currently lacking is systemic design guidance that allows one to optimally trade off between error, area and energy. While the area and energy of a given design can often be easily reasoned or estimated, getting insights on error and thereby providing a basis for optimally trading off between error, area and energy consumption appears to be challenging and not well understood.

II. PREVIOUS WORKS

The two types of approximate multipliers existing are: approximate AND-array multipliers, which utilize AND gates for partial products generation and approximate Booth multipliers, which use the

modified Booth algorithm to reduce the number of partial products. Constant correction [1] and variable correction [2] schemes are proposed for approximate AND-array multipliers. Constant correction scheme suggests adding one constant to compensate for the truncated error and variable correction multipliers add some signals in the partial product table to make compensation. However, since Booth multipliers are much more efficient than AND array multipliers, approximate Booth multipliers have been intensively investigated [3]-[8]. In particular, statistical linear regression analysis [3], estimation threshold calculation [4] and self-compensation approach [5] have been utilized to compensate for the truncation error. Accuracy is increased by using certain outputs from Booth encoders [6] [7]. To decrease energy consumption, a probabilistic estimation bias (PEB) scheme [8] is presented.

A series of approximate squarers have been proposed [9]-[11]. For instance, the designs of [9] and [10] compensate truncation error by utilizing constant and variable correction schemes, respectively. A LUT-based squarer [11] is proposed by employing a hybrid LUT-based structure.

Approximate Computing (AAAC) model that enables reasoning about approximation error control and optimal error compensation under ideal design conditions. The model is broadly applicable to array-based arithmetic units such as multipliers, squarers, dividers, adders, and logarithmic function units. It provides valuable design insights into generating error compensation signatures. Second, leveraging these insights, the study proposes new 16-bit fixed-width approximate Booth multiplier and squarer designs. The multiplier achieves up to 44.85% energy and 28.33% area savings while reducing average, maximum, and mean square errors by 11.11%, 28.11%, and 25.00%, respectively. The squarer design shows similar benefits with up to 31.25% error reduction and a 20%+ improvement in the energy-delay-mean square error product (EDEms). Additional gains in energy-delay-max error product (EDEmax) are achieved using extra signatures and don't care conditions. Accuracy further improves in full-width operation mode.

III.METHODOLOGY

In this work, we use nxn fixed-width Booth multipliers to refer to approximate Booth multipliers that operate on two n-bit inputs while outputting only an n-bit product [4]. For convenience of discussion, we assume the higher and lower n bits of the multiplicand and multiplier correspond to the integer and fractional parts of the inputs, respectively. In this regard, a fixed-width multiplier outputs, possibly in an approximate manner, the n-bit integer part of the exact product.

Fig. 1 shows the schematics of the Radix-4 Booth encoding block applied in this research; the outputs of the encoding block are s_i, d_i, n_i, z_i, and c_i: (a) is the schematic of s_i and d_i generation; (b) is the schematic of n_i generation; (c) is the schematic of n_i generation; (d) is a Schematic of c_i, generation.

z_i signifies whether the partial product is zero or not, n_i specifies the sign of each partial product, $\{d_i, s_i\}$ determines magnitude of the value multiplied by A for the partial product (0, A or 2A), where, d_i is the more significant bit and s_i is the less significant bit. c_i is the correction constant required to generate the negative partial product. s_i, d_i, n_i, z_i, and c_i, are generated by three consecutive input bits of multiplier B, which are b_{2i-1}, b_{2i}, and b_{2i+1}. Fig. 2 presents the schematic of the selection block applied in this research, where $pp_{i,j}$ represents the j^{th} bit of i^{th} partial product.

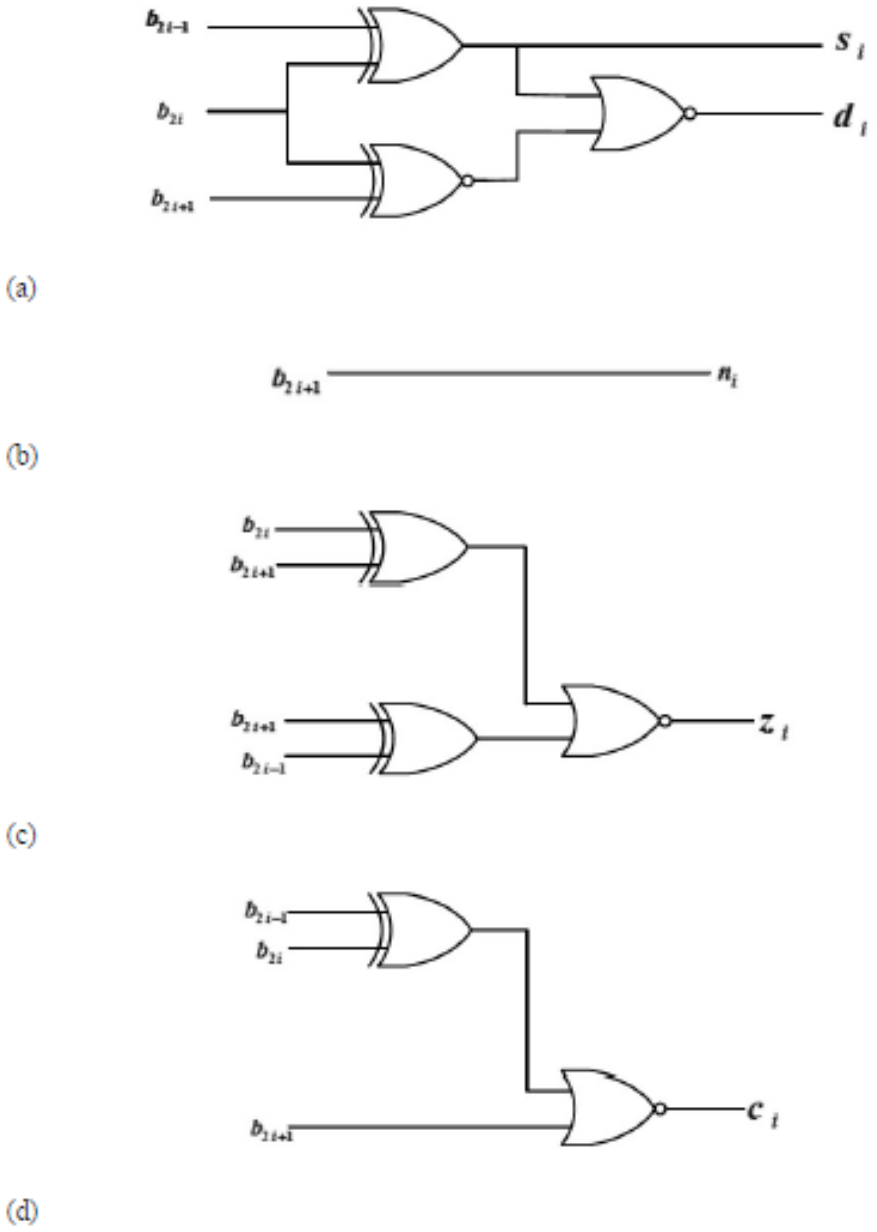

Fig. 1: Radix-4 Booth encoding block: (a) Schematic of s_i, d_i generation, (b) Schematic of n_i generation, (c) Schematic of z_i generation, (d) Schematic of c_i generation.

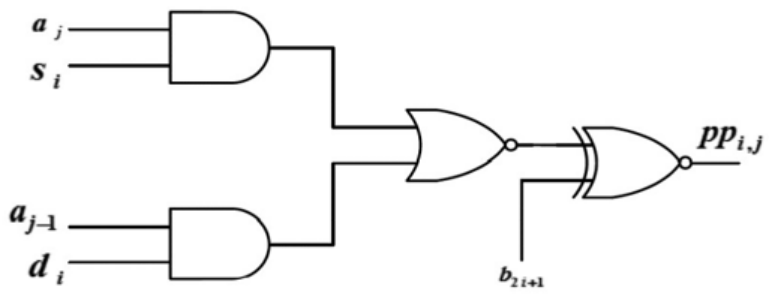

Fig. 2: Schematic of the Selection block.

Our goal in approximate fixed-width multiplier design is to approach the accuracy of a PTM without incurring its high overhead that is commensurate with that of a full-precision multiplier. Under the AAAC model, we associate the accurate part (AP) and the truncation part (TP) of the array in Fig. 8 with the LPCU and ECU, respectively. More specifically, the bits in AP are processed by the LPCU while the effects of the ones in TP are approximated by the ECU in the form of error compensation. The exact product (EFCU output) is

$$O_{EFCU} = O_{LPCU} + S_{TP} \tag{1}$$

where S_{TP} is the partial sum of TP, and O_{LPCU} is the LPCU output corresponding to AP. To reduce the amount of approximate error, we further divide TP into TP_H (i.e., the $n - 1^{th}$ column) and TP_L and have

$$S_{TP,H} = \frac{1}{2} SUM_{n-1} \tag{2}$$

$$S_{TP,L} = \frac{1}{4} SUM_{n-2} + \frac{1}{8} SUM_{n-3} + \cdots + \left(\frac{1}{2}\right)^{n} \ SUM_0 \tag{3}$$

where $S_{TP,H}$ and $S_{TP,L}$ correspond to the partial sums of TP_H and TP_L, and SUM_i represents the sum of all bits in the i^{th} column, respectively. Now it is clear that

$$S_{TP} = S_{TP,H} + S_{TP,L} \tag{4}$$

The main objective in the design of ECU is to well approximate

$$S_{TP} \approx O_{ECU} \tag{5}$$

such that a fixed-width n-bit output is produced, i.e.,

$$O_{EFCU} = O_{LPCU} + S_{TP} \approx O_{LPCU} + O_{ECU} \tag{6}$$

Approximate full-width multipliers, i. e., ones that approximate accurate nxn Booth multipliers by outputting a full-width 2n-bit approximate product, are also useful for many practical applications. The presented fixed-width design can be readily extended to facilitate full-width operation with the difference being that in this case we would like to approximate $S_{TP,L}$.

We use the method in [9] to implement squarers instead of using Booth algorithm as applied to multiplier design because squarers implemented by using the method in [9] are more energy-efficient and faster since most partial products bits are implemented by simple AND operation of two input bits instead of more complex Booth encoding and selection blocks. The squarer design process is similar to the one presented for the proposed multipliers (e.g., based on eqn. (9 - 13). Again, the key problem is to design an ECU to approximate well. By following the ECU design guidance above, we consider the signals on the $n-2^{th}$ column as signatures since they have the highest weight on TP_L and include all input bits which contribute to TP_L .

To simplify the design of the signature generator, we sum up the signals on the $n-2^{th}$ column to produce the first signature CA. We introduce one input bit as the second signature (CB) to further split the large input groups formed by CA. Accordingly, the input bit is a_6 chosen as the second signature CB for the proposed 16-bit squarer.

Fig. 3 shows the complete design of the proposed multiplier (a) and the proposed squarer (b). For proposed multiplier design, the encoding block applies the Radix-4 Booth Algorithm to encode the multiplier B, allowing the selection block to generate only half number of partial products needed for array multipliers with each product being one of the following: 0, A, 2A, -A, -2A (shown in Fig. 3). For the proposed squarer design, the similar partial product table is generated. After the dots which stand for partial products and contain AP, and (for fixed-width designs) or (for full-width designs), shown in Fig. 3(a) for multipliers and Fig. 3(b) for squarers, are generated, they are compressed to only two partial products by compressors in the compression block. Finally, the two partial products are fed into the final adding block, and the final result is generated using a Carry propagation adder (CPA).

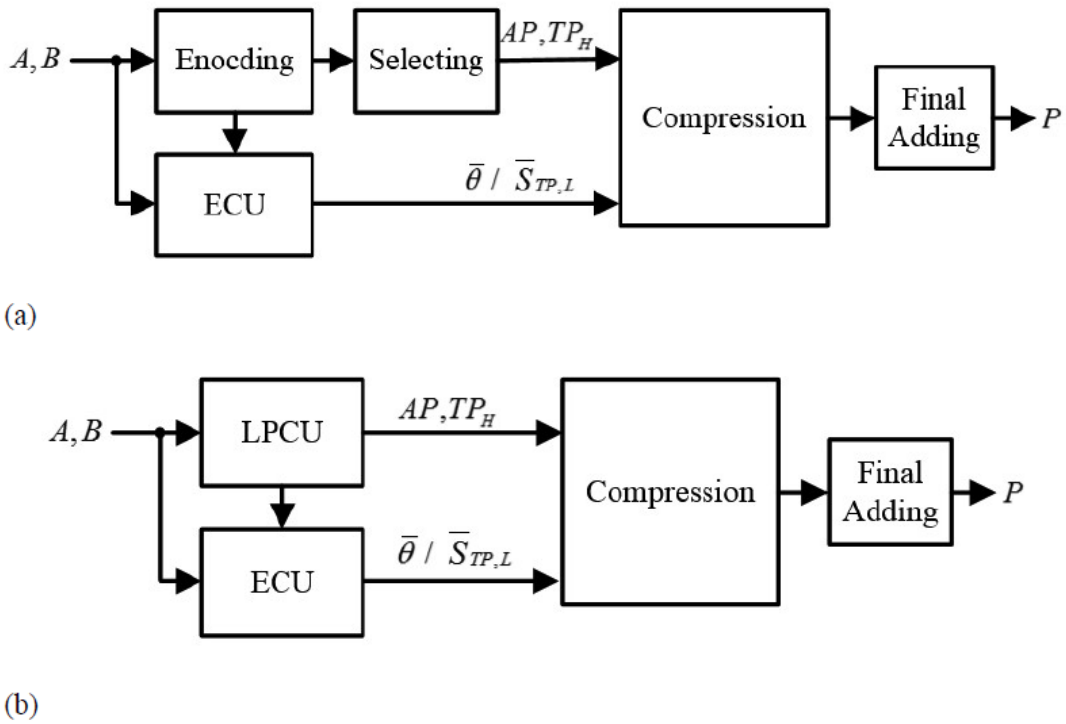

Fig. 3: Complete designs blocks: (a) the proposed multipliers, (b) the proposed squarers.

The purpose of using compressors in the compression block is that multiple compressors can run in parallel, thus speeding up the compression process. Now, we discuss three types of comparatively low-cost compressors (2:2 Compressors [20], 3:2 Compressors [20] and 4:2 Compressors [21]) that are used in the proposed multiplier and squarer design because they have comparatively less energy and delay overhead. We also discuss the processing steps involved in compression, in which multiple compressors run in parallel.

IV. RESULTS AND DISCUSSION

The proposed 16-bit fixed-width Booth multiplier and squarer are designed in Verilog HDL, synthesised using Synopsis Design Compiler [17] with a commercial 90 nm CMOS technology and standard cell library. From Synopsis Design Compiler synthesis (Design Vision) reports, we get the pre-layout delay, dynamic power, leakage power and area. We also implement four additional fixed-width Booth multipliers: DTM (Direct Truncated Booth Multiplier) [5], PEBM (with probabilistic estimation bias compensation) [8], ZSM [7] and PTM (Post Truncated Booth Multiplier, most accurate/expensive fixed-width multiplier) [3] for comparison purposes.

Four additional squarers are implemented: DTS (Direct Truncated Squarer), CCS (with a constant compensation) [9], VCS (the signals on the column as the compensation) [10] and PTS (Post Truncated Squarer most accurate/expensive fixed-width squarer). For all Booth multiplier and squarer designs implemented in this research, partial products are generated and then compressed to two partial products

using 2:2, 3:2 [20] and 4:2 [21] compressors. As discussed, 2:2, 3:2 and 4:2 compressors provide an efficient method for compressing the number of partial products to two because they enable the compression process to run in parallel. Finally, the two compressed partial products are added up by a carry propagation adder (CPA) to produce the final results. Table 1 shows the comparison of various metrics for 16x16 bits fixed-width Booth multipliers.

Table 1. Implementation results of different 16x16 bits fixed-width Booth multipliers.

Multiplier	Area (um^2)	Delay (ns)	Power (mW)	Energy (pJ)	E_{ms}
DTM	2,645	2.61	0.86	2.24	9.85
PTM	5,239	3.72	1.75	6.51	0.08
PEBM	2,937	2.79	0.98	2.73	0.35
ZSM	3,256	2.99	1.11	3.32	0.20
Proposed	3,755	2.99	1.20	3.59	0.15

A more detailed accuracy comparison among different approximate multipliers. Error reduction or accuracy improvement of the proposed design over the existing designs is defined as $\frac{|E_{existing}-E_{Proposed}|}{E_{existing}} \times 100$, where $E_{existing}$ is one of the error metrics (E_{ave}, E_{max}, and E_{ms}) of the compared existing design and is defined as one of the error metrics (E_{ave}, E_{max}, and E_{ms}) of the proposed design. We evaluate the accuracy of the five different designs in terms of E_{ave}, E_{max}, and E_{ms}, for n = 8 (bits) in Table 2.

Table 2. Error Metrics of 8x8 Fixed-Width Booth Multipliers.

Multiplier	E_{ave}	E_{max}	E_{ms}
DTM	1.50	4.00	2.69
PTM	0.25	0.50	0.08
PEBM	0.35	1.50	0.18
ZSM	0.30	1.17	0.14
Proposed	0.29	1.00	0.13

Then we evaluate the accuracy of the five different designs in terms of E_{ave}, E_{max}, and E_{ms}, for n = 12 (bits) in Table 3.

Table 3. Error Metrics of 8x8 Fixed-Width Booth Multipliers.

Multiplier	E_{ave}	E_{max}	E_{ms}
DTM	3.00	8.00	9.85
PTM	0.25	0.50	0.08
PEBM	0.48	2.50	0.35
ZSM	0.36	2.17	0.20
Proposed	0.32	1.56	0.15

In Table 4, five fixed-width squarers are compared in terms of area, delay, power (sum of dynamic power and leakage power), energy, and E_{ms}. The energy consumption and area of the proposed multiplier are slightly larger than CCS and VCS, but are much smaller than PTS, with a 42.43% and 30.70% reduction, respectively. On the other hand, the proposed design has a significantly reduced cost compared with DTS, CCS and VCS. This indicates that our design delivers a much-improved accuracy with a very small amount of additional overhead.

Table 4. Implementation results of different 16x16 bits fixed-width squarers.

Multiplier	Area (um^2)	Delay (ns)	Power (mW)	Energy (pJ)	E_{ms}
DTS	1,566	2.21	0.36	0.80	4.34
PTS	3,016	2.93	0.70	2.05	0.08
CCS	1,891	2.22	0.44	0.98	0.28
VCS	1,997	2.34	0.46	1.08	0.16
Proposed	2,090	2.45	0.48	1.18	0.11

V. CONCLUSION

In conclusion, this work presents a comprehensive model for array-based approximate arithmetic computing, guiding the efficient design of Booth multipliers and squarers. By formulating four key theorems, we address fundamental challenges in Error Compensation Unit (ECU) design, specifically determining the optimal error compensation value and selecting the most suitable scheme. The introduction of don't care conditions further simplifies ECU logic, leading to reduced energy and area costs. Experimental results demonstrate that the proposed 16×16 fixed-width approximate Booth multiplier achieves substantial savings—up to 44.85% in energy and 28.33% in area—alongside significant error reductions. Similarly, the approximate squarer achieves notable improvements in accuracy and efficiency, with EDE metrics outperforming existing designs. The incorporation of extra signatures and don't cares offers further optimization, particularly in reducing EDE_{max}. When operated in full-width mode, both the multiplier and squarer exhibit enhanced accuracy, reaffirming the effectiveness of the proposed model. Overall, this research offers a robust and scalable framework for designing energy-efficient and error-resilient approximate arithmetic units.

VI. REFERENCES

1. Schulte, Michael J., and Earl E. Swartzlander. "*Truncated multiplication with correction constant [for DSP]*." VLSI Signal Processing, VI, 1993.,[Workshop on]. IEEE, 1993.

2. King, Eric J., and E. E. Swartzlander. "*Data-dependent truncation scheme for parallel multipliers*." Signals, Systems & Computers, 1997. Conference Record of the Thirty-First Asilomar Conference on. Vol. 2. IEEE, 1997.

3. Jou, Shyh-Jye, Meng-Hung Tsai, and Ya-Lan Tsao. "*Low-error reduced-width Booth multipliers for DSP applications*." Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on 50.11 (2003): 1470-1474.

4. Min-An, S. O. N. G., V. A. N. Lan-Da, and K. U. O. Sy-Yen. "*Adaptive low-error fixed-width Booth multipliers*." IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences 90.6 (2007): 1180-1187.

5. Huang, Hong-An, Yen-Chin Liao, and Hsie-Chia Chang. "*A self-compensation fixed-width booth multiplier and its 128-point FFT applications*." Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on. IEEE, 2006.

6. Cho, Kyung-Ju, et al. "*Design of low-error fixed-width modified Booth multiplier*." Very Large Scale Integration (VLSI) Systems, IEEE Transactions on 12.5 (2004): 522-531.

7. Wang, Jiun-Ping, Shiann-Rong Kuang, and Shish-Chang Liang. "*High-accuracy fixed-width modified Booth multipliers for lossy applications*." Very Large-Scale Integration (VLSI) Systems, IEEE Transactions on 19.1 (2011): 52-60.

8. Li, Chung-Yi, et al. "*A probabilistic estimation bias circuit for fixed-width Booth multiplier and its DCT applications*." Circuits and Systems II: Express Briefs, IEEE Transactions on 58.4 (2011): 215-219.

9. Walters III, E. George, Michael J. Schulte, and Mark G. Arnold. "*Truncated squarers with constant and variable correction*." Optical Science and Technology, SPIE 49th Annual Meeting. International Society for Optics and Photonics, 2004.

10. Walters III, E. George, and Michael J. Schulte. "*Efficient function approximation using truncated multipliers and squarers*." Computer Arithmetic, 2005. ARITH- 17 2005. 17th IEEE Symposium on. IEEE, 2005.

11. Hoang, Van-Phuc, and Cong-Kha Pham. "*Low-error and efficient fixed-width squarer for digital signal processing applications*." Communications and Electronics (ICCE), 2012 Fourth International Conference on. IEEE, 2012.

12. Du, Kai, Peter Varman, and Kartik Mohanram. "*High performance, reliable variable latency carry select addition*." Design, Automation & Test in Europe Conference & Exhibition (DATE), 2012. IEEE, 2012.

13. Zhu, Ning, Wang Ling Goh, and Kiat Seng Yeo. "*An enhanced low-power high-speed adder for error-tolerant application*." Integrated Circuits, ISIC'09. Proceedings of the 2009 12th International Symposium on. IEEE, 2009.

14. Kim, Yongtae, Yong Zhang, and Peng Li. "*An energy efficient approximate adder with carry skip for error resilient neuromorphic VLSI systems*." Proceedings of the International Conference on Computer-Aided Design. IEEE Press, 2013.

15. Hachtel, Gary D., and Fabio Somenzi. *Logic synthesis and verification algorithms.* Springer, 2006.

16. Bergamaschi, Reinaldo A., et al. "*Efficient use of large don't cares in high-level and logic synthesis*." Proceedings of the 1995 IEEE/ACM international conference on Computer-aided design. IEEE Computer Society, 1995.

17. Synopsys. Inc., "*Design compiler user guide*" http://www. synopsys. com, 2010.

18. Weste, Neil, and David Harris. *CMOS VLSI design: a circuits and systems perspective*. Addison-Wesley Publishing Company, 2010.

19. de Angel, Edwin, and E. E. Swartzlander. "*Low power parallel multipliers*." VLSI Signal Processing, IX, 1996.,[Workshop on]. IEEE, 1996.

20. Wallace, Christopher S. "*A suggestion for a fast multiplier*." Electronic Computers, IEEE Transactions on 1 (1964): 14-17.

21. Oklobdzija, Vojin G., David Villeger, and Simon S. Liu. "*A method for speed optimized partial product reduction and generation of fast parallel multipliers using an algorithmic approach*." Computers, IEEE Transactions on 45.3 (1996): 294-306.