ZKG INTERNATIONAL
ISSN: 2366-1313

APPROXIMATE BOOTH SQUARER DESIGN FOR ERROR-TOLERANT COMPUTING

#1G. VAISHNAVI, *M.Tech(VLSID) Student,*
#2T.RAJU, Associate Professor,
Department of ECE,
JYOTHISHMATHI INSTITUTE OF TECHNOLOGY AND SCIENCE (AUTONOMUS), KARIMNAGAR

ABSTRACT: "Approximate computing" is an object construction technique that prioritizes speed and low power consumption over accuracy. The construction of approximate Booth multiplier systems is the main goal of this thesis. We can calculate the approximate multiplier for errors resulting from the approximate 4-2 compressor, the approximate regular partial product array, and the approximate radix-4 Booth encoding using a probabilistic error model. For both 8-bit and 16-bit approximation designs, NMEDs can be found using the suggested methodology. The simulation results show that the error model and its underlying structure are accurate. As such, the employed analytical techniques were sound.

Index Terms— Booth encoding and an approximation multiplier for probabilistic error models

1. INTRODUCTION

Digital signal processors, multimedia, embedded systems, and microprocessor arithmetic units need high-performance, low-power multipliers. Precision and exactness are less important in multimedia signal processing and machine learning, which include human perception. Reducing accuracy to speed up and power consumption will considerably reduce information processing time and energy. Accuracy requires more difficult multiplier performance and power usage enhancement. We call this "accurate calculation."

Arithmetic processor performance depends on addition and multiplication. For energy and time savings, approximation computing has studied adding. Approximaton adder design study uses unique measures such error distance, mean error distance, and normalized error distance. Many mathematical systems use approximate multiplication, despite its neglect. Multiplying causes product row gaps. Improved Booth encoding and high-performance multipliers reduce partially complete product rows.

There are several ways to create truncated and non-truncated multipliers. Truncated product lines might cause major errors. Truncation-based designs estimate the least significant partial products using a constant or omitting the lowest half, however partial product rows are shorter, resulting in considerable mistakes. Error compensation methods like the inexact array multiplier ignore some of the least significant columns and treat them as constants to improve truncated multiplier accuracy. The multiplier truncation correction constant comes from rounding and reduction error accounting.

Truncated multipliers with variable adjustment solve all-zero or all-one partial products in the least significant columns. Recently developed error correction methods increase fixed width Booth multipliers. The error compensation circuit replaces a reduced sorting network with Booth encoder outputs in [9]. We solved the quantization error of a fixed-width Booth multiplier with an adaptive conditional-probability estimator. Truncated Booth multipliers are more accurate with error correction. Approximation processing reduces this cost, but more compensatory circuits require more hardware.

Accurate calculation benefits low-power, high-speed, low-complexity systems. Adapt output precision for image and video processing. Fixed-width computation and simplified algorithms reduce time and power consumption but compromise approximation accuracy, according to the literature. Approximative arithmetic blocks can be used to build partial product values. This method has been used to study many approximators, compressors, and adders.

2. RELATED WORK

The proposed approximation squarers (PCS) use precomputed sums to generate Boolean equations by decreasing the most significant terms from the precise squaring equation. Systematically improve output accuracy. Square the equal halves of the input operand, x and y, using the quadratic formula (x + y)2. This squarer approximation (y2) misses the input operand's LSB bits. One can approximate squares by flattening. Fix-width squarers remove N LSBs from output or matching partial products. Dynamic error compensation divides the truncation component (TP) into minor (TPmi) and major (TPm) subparts, lowering TPmj. TPmj generates partial products, whereas TPmi biases the squared output. Left-to-right leading-digit high-radix dual recoding produces a squarer. After considering guard bits for the fixed-width squarer, input operands are encoded from top left to bottom right, ignoring the LSB columns. Per [14], an array-based approximation squarer corrects partial product matrix truncation mistakes with an updated error compensation unit. SquASH approximates vector-self inner product in self-healing squarer accumulators. Approximation squarer accumulator (SAC) reduces output errors by combining two errors.

This complicates the encoder but accelerates partial product buildup. Booth multiplication is simplified by approximate radix-4 and radix-8 Booth multipliers. Redundant binary (RB) partial product rows multiply values quickly. The approximative radix-4 RB multiplier yields approximately RB partial products from RB 4-2 compressors like earlier multipliers. A low error bias approximative multiplier reduces stacking errors in consecutive approximation multiplications. This multiplier decreases errors with logarithmic and linear approximation. This concept approximates an internal self-healing multiplier accumulator. This MAC's two 2-size multiplier subblocks' average multiplier error is approximately nil due to canceling errors.

BOOTH MULTIPLIER

This signed number multiplication aligns positive and negative integers. In a typical addshift operation, multiplier bits multiply the multiplicand to be added to the partial output. Large multipliers require many multiplyicands. That many adds requires a longer multiplier delay. Reducing boosts performance. Multiplication circuits are slower and smaller with booth multiplication. In performance, "short multiplication" trumps "long multiplication".

MODIFIED BOOTH MULTIPLIER

Recent multiplier designs have exploded. Multiplier components are crucial to microprocessors and DSPs. Several multiplication operations will slow system processes. Recent multiplier designs minimize operational speed, size, and power consumption. Proper multipliers help digital signal processors filter and analyze spectrum more precisely. This splits the multiplier in half. PPGs start with Booth encoder and partial product generator.

Finally, first and second stage components make the product. Compressors can replace adders to reduce carry signal delay. Watching adders helps develop Carry Propagation and Carry Save circuits. Multiplication requires muliter and multiplicand. Standard Binary Multipliers use Multiplicand bits and their own to build Partial Products. Standard binary multipliers use gates for partial products. Zero multipliers add zeros to partial products, creating a row of zeros. Before the multiplier bit is one, the multiplicand travels left and adds to partial products.

Booth multiplier compressors and fast adders yield partial products. Quick adders finish products. The fundamental radix 4 Booth encoding setup is examined here. This Booth encoding development considers errors. Approximating normal partial product arrays reduces nothing. Booth multipliers use approximate Booth encoders and partial product arrays.

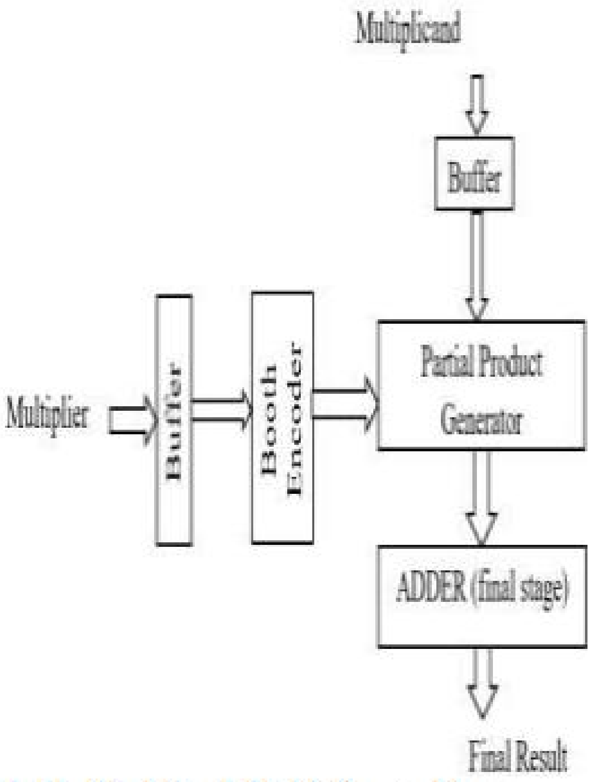


Fig.: Modified Booth Multiplier Architecture

3. DESIGN AND ANALYSIS

APPROXIMATE RADIX-4 BOOTH MULTIPLIERS

More efficient two-complement binary number multiplication using MBE and radix-4 Booth encodings was suggested. Without booth multipliers, partial product rows are impossible. In two-s complement, multiplier B is multiplied by A. Works like this:

REVIEW OF RADIX-4 BOOTH MULTIPLICATION

Booth encoders decode every subgroup using partial products -2A, -A, 0A, and 2A. Add '1' (designated Neg) to A's lower-left corner to negate.

$$A = -a_{N-1}2^{N-1} + \sum_{i=0}^{N-2} a_i 2^i,$$
$$B = -b_{N-1}2^{N-1} + \sum_{i=0}^{N-2} b_i 2^i.$$

Booth shows radix-4 encoder, decoder, and encoder schematics. Examples of Booth encoder output:

$$pp_{ij} = (b_{2i} \oplus b_{2i-1})(b_{2i+1} \oplus a_j) + \overline{(b_{2i} \oplus b_{2i-1})}(b_{2i+1} \oplus b_{2i})(b_{2i+1} \oplus a_{j-1}).$$

ALGORITHM 1

```
Algorithm 1 Probabilistic Error Model.
Require:
    Bits of the multiplier operand, $N$;
    Approximate factor, $p$; Operands, $b_i$, $a_j$;
    Modification of K-Map, $T$;
    Number of the Error in K-Map, $Q$;
Ensure:
    Probabilistic Error Model of ABMs, $E_{ABM}$
 1: Define: Probabilistic of the value "1" of logic expression
    [P(x)];
    %Based on circuit structure to analyze sub-model
 2: $E_{ABE} \triangle P(T)/Q$, $1 \triangle 0$ and $0 \triangle 1$ denoted as $T$;
    %The error model of ABE, i.e. $E_{ABE}$;
    %The logical expression of $T$, i.e. Eq. (9) and (10);
 3: $E_{APA} \triangle P(G)$,
    $G$: the last symbol compensation bit Neg;
    %The error model of approximate partial product array,
    i.e. $E_{APA}$;
    %The logical expression of $G$, i.e. Eq. (12);
 4: $E_{ACM} \triangle P(E)$, $E$: the difference between exact com-
    pressor and approximate compressor;
    %The error model is approximate compressor, i.e.
    $E_{ACM}$;
    %The logical expression of $E$, i.e. Eq. (14);
 5: Define: Function of error model [F(x)];
 6: for the sub-model do
 7:     $E_{ABM-S} \triangle F(N, p, Q, E_{ABE}, E_{APA})$;
 8:     $E_{ABM-C} \triangle F(N, p, Q, E_{ABE}, E_{APA}, E_{ACM})$
 9: end for
    %The probabilistic error function expression of $E_{ABM-S}$
    and $E_{ABM-C}$, i.e. Eq. (15) and(16);
10: return $E_{ABM}$;
```

APPROXIMATE BOOTH ENCODING

High-performance multipliers need booth encoding. Reducing rows and PPs speeds PP synthesis. Booth data accuracy can be assessed as follows:

$$pp_{ij} = (b_{2i} \circ b_{2i-1})(b_{2i+1} \circ a_j) + \overline{(b_{2i} \circ b_{2i-1})}(b_{2i+1} \circ b_{2i})(b_{2i+1} \circ a_{j-1})$$

ABE-1 and ABE-2 are crude Booth encoders. The figures show Booth encoding gate-level circuits. Eqs. for ABE-1 and ABE-2 are estimates.

$$app_{ij1} = (b_{2i} \circ b_{2i-1})(b_{2i-1} \circ a_j)$$

$$app_{ij2} = a_j\overline{b_{2i+1}} + \overline{a_j}b_{2i+1} = b_{2i+1} \circ a_j$$

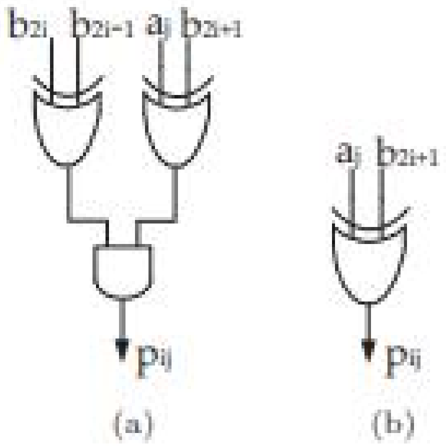


Figure: Gate level circuit of: (a) ABE-1, and (b) ABE-2 [9].

APPROXIMATE REGULAR PP ARRAY

Booth encoded PP array rows expand by N/2+1. Neg concluded with a compensation. Create the array PP by deleting the compensating bit. Figure depicts how the 8*8 modified Booth encoding (MBE) PP array now uses ppij instead of Neg. Also included is the sign extension phrase. Normalisation at the projected PP array's end loses a compensatory bit. The array contains fewer rows, thus PP accumulates faster.

Cin and Cout are not approximate compressor parameters. Approximaton compressors have input and output parameters, unlike 4-2. Use Carry and Sum to estimate properly. Figure depicts an approximator compressor with four inputs—P4, P3, P2, and P1. Logical considerations predict a 4-2 compressor:

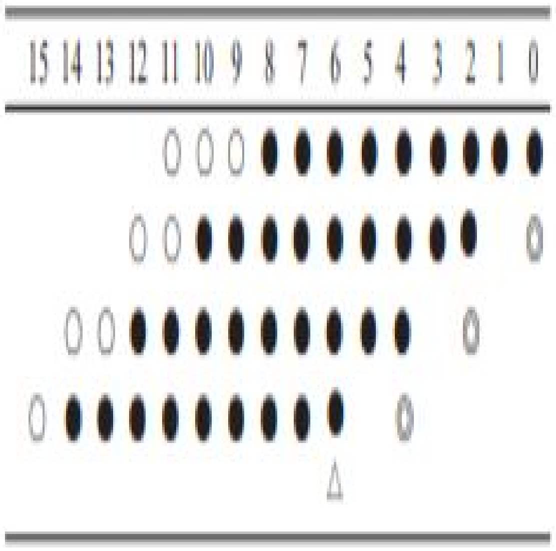


Figure: A conventional 8*8 MBE PP array

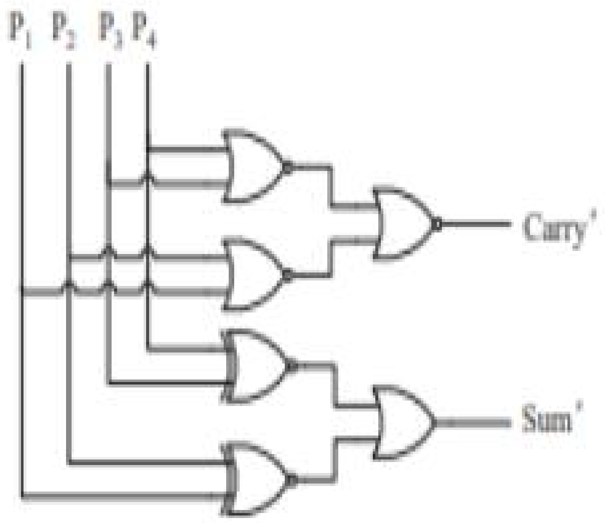


Figure: Gate level circuit of an inexact compressor.

APPROXIMATE COMPRESSOR

One must analyze all Booth multiplier error attributes. Approximative multipliers and adders can be assessed and normalized using error distance (ED).

Value (A), approximation (A'), maximum error value, n product outcomes, and results (MAX output).

$$Sum' = \overline{(P_1 \circ P_2)} + \overline{(P_3 \circ P_4)}$$

$$Carry' = \overline{\overline{P_1 + P_2} + \overline{P_3 + P_4}}$$

ERROR METRICS

ISSN: 2366-1313

Error models approximate booth multiplies. Error model submodels explain multiplication stages. Construction of the multiplier circuit creates sub-error models. Figure 1 shows the error model-circuit architecture relationship. ABM EABMs include Booth encoding error, consistent product array error, and 4-2 compressor error. Case in point: EACM.

$$ED = A \quad A'$$

$$MED = \sum_{i=0}^{n} \frac{ED}{n}$$

$$NMED = \frac{MED}{MAX_{output}}$$

Value (A), approximation (A'), number of alternative product outcomes n, and maximum error value are all included (MAX output).

PROBABILISTIC ERROR MODEL OF ABMS

Figure out Booth multipliers (ABMs) can be guessed with this mistake model. There are different parts to the mistake model, and each one explains a different step in the multiplication process. To make sub-error models, the multiplication circuit has to be built. There is a link between the error model and the circuit layout shown in Figure 1. There are three mistakes that make up the whole ABM's EABM: the exact Booth encoding error, the regular product array error, and the 4-2 compressor error, which is also called EACM.

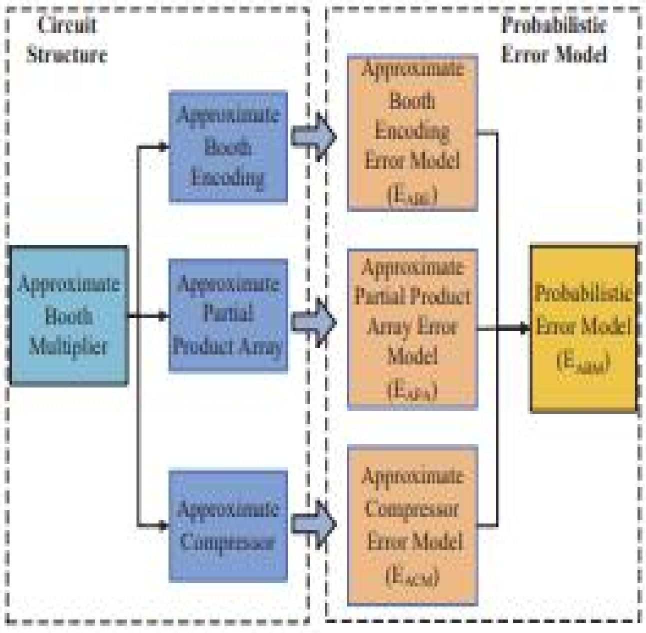

Circuit Structure
Probabilistic Error Model
Approximate Booth Encoding
Approximate Booth Multiplier
Approximate Partial Product Array
Approximate Compressor
Approximate Booth Encoding Error Model (E_{ABE})
Approximate Partial Product Array Error Model (E_{APA})
Approximate Compressor Error Model (E_{ACM})
Probabilistic Error Model (E_{ABM})

Figure: The probability error model based on the circuit.

APPROXIMATE BOOTH ENCODING ERROR MODEL

The ABEs show a mistake when one of the "0s" in the truth table is changed to a "1." It is possible for ABEs to make both good and negative mistakes. Then, if you change the number "1" to "0," T10 and T01 become 1 and 1, respectively. When ABE-1's approximation encoding is used, the mistake is taken into account when the correct PP is found.

$$pp_{ij1} = (b_{2i} \circ b_{2i-1})(b_{2i-1} \circ a_j) + T_{1-0}$$

Another way to write the probability error model is with the ABE-2 approximation.

$$pp_{ij2} = b_{2i+1} \circ a_j + T_{1-0} + T_{0-1}$$

The expected result is better than the exact match when zeros are replaced with zeros using the ABE's K-map. In other words, the real results and the predicted results are not the same. With a bigger difference between a "1" and a "0," the predicted result goes down. There is about the same amount of error introduced by each estimate encoding method.

Table: ED and MED of Approximate Booth Encoding

Encoding	Error Pattern	ED	MED
ABE-1	T_{1-0}	4	0.125
ABE-2	T_{1-0} , T_{0-1}	4	0.125

Table: Q and EABE of Approximate Encoding

Encoding	Q_{0-1}	Q_{1-0}	E_{ABE}
ABE-1	0	4	0.03125
ABE-2	6	2	0.01563

Table: Error of PP Array

ED	MED	E_{APA}
3	0.375	0.125

Here are the Booth encoder mistakes based on the new model for probabilistic errors. The updated number in the truth table is shown by the approximation Booth encoding as ED. With a T10 adjustment, ABE-1's K-map has been changed to a modified number of 4, which is better. The ABE-2 K-map will use either T10 or T01. In this case, the answer is -4 since T01 is negative 6 and T10 is positive 2. The ED, on the other hand, has an absolute number of four.

There are about 10 T10s and one T1 in Table 2. Q01 shows how many T10s there are, and Q10 shows how many there are in total. This choice is there to make the error model better. The prefix "Q" is used to connect Q01 through Q10. The sum of Equations Q01 and Q10 is shown by the letter Q.

The Booth encoding mistake is shown by the extra parameter Q, which was added to the NMED during EABE.

$$E_{ABE} = \frac{NMED}{Q} = \frac{MED}{Q \times MAX_{output}}$$

PROBABILISTIC ERROR MODEL OF ABMS

ABM1 and ABM2 both use a standard partial product array that has an expected Booth encoding. The phrase "single design" refers to everything. The amount of error goes up logarithmically when N is multiplied by an ABM. To find the mistake in the exponent, use log2 N-1. To show the PP array error of approximation Booth multipliers and encoders, the EABMS blends all the sub-error models into a single model of design flaws.

Four Ideas and Approximations for Booth Multiplier Designs If an object has already been used by, it is marked as useless by() and used otherwise.

Multiplier	ABE-1	ABE-2	Array	Compressor
ABM1	♦	⊕	♦	⊕
ABM2	⊕	♦	♦	⊕
ABM3	♦	⊕	♦	♦
ABM4	⊕	♦	♦	♦

$$E_{ABM-S} = k_i \}[\frac{1}{N}(E_{ABE}\frac{[(N/2+1)\times 2 \quad Q|Q}{(N/2+1)\times 2})]^{\log_2 N-1} p + (E_{APA}/N)^{\log_2 N-1} \times p/2|$$

In this case, ki = pi/pi1. An guess factor of pi/pi1 = 1.5 is used in a multiplier design with a radix-4 Booth encoding unit count of pi.

There needs to be two to four compressors between single versions (EABMS) and composite systems (ABM3 and ABM4). A combined design error model is also available from the EABMC.

$$E_{ABM-C} = k_i(E_{ABM-S} + \frac{1}{N} E_{ACM}{}^{\log_2 N-1})$$

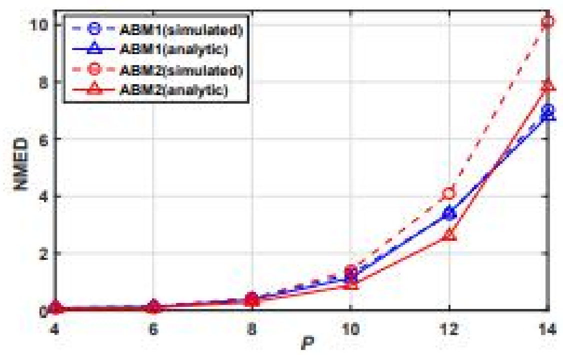
4. RESULTS

We look into why the simulation results and the error model for the four approximate Booth encoded factors at different p values don't match up. At the gate level, the Synopsys VCS checks all Verilog HDL simulation outputs. The model analysis must be used to figure out the error for each of the four approximation factors.

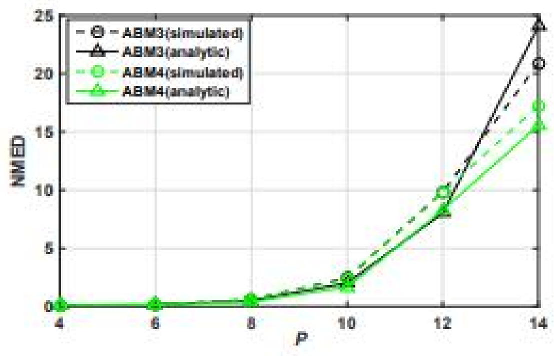
ISSN: 2366-1313

There are four 8-bit ABM models in the NMED findings table. The number of the variable p can be anywhere from 4 to 14 different ones. As p goes up, the error numbers go up almost exponentially. We can see that the single design (ABM1, ABM2) has less error than the combined design (ABM1, ABM2) because it only uses the approximate Booth encoding module and the approximate partial product array module (ABM3, ABM4).

We can see NMED (both analytical and modeling values) vs. p in Figures (a) and (b). These show the 8-bit ABMs EABMS and EABMC. The ABM1 model made the most accurate prediction, and the results of the analysis were a lot like the results of the exercise. Though not always, analytical errors are less than the simulated numbers. The results of the 8-Bit ABM NMED



(a)

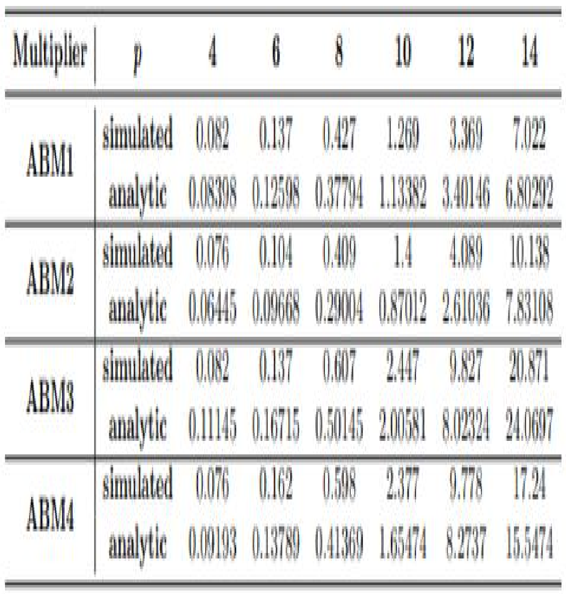


(b)

The measured and simulated NMED numbers for ABM are shown in this graph. There are two 8-bit ABMs that can be used: EABMS and EABMC.

5. CONCLUSION

This study approximates radix-4 Booth multipliers. Two approximative Booth encoders with incorrect error model (p goes from 4 to 14 and the order of magnitude is 102) are shown in the table below.



Multiplier	p	4	6	8	10	12	14
ABM1	simulated	0.082	0.137	0.427	1.269	3.369	7.022
	analytic	0.08398	0.12598	0.37794	1.13382	3.40146	6.80292
ABM2	simulated	0.076	0.104	0.409	1.4	4.089	10.138
	analytic	0.06445	0.09668	0.29004	0.87012	2.61036	7.83108
ABM3	simulated	0.082	0.137	0.607	2.447	9.827	20.871
	analytic	0.11145	0.16715	0.50145	2.00581	8.02324	24.0697
ABM4	simulated	0.076	0.162	0.598	2.377	9.778	17.24
	analytic	0.09193	0.13789	0.41369	1.65474	8.2737	15.5474

truth table terms reduce PDP by 59%. Analytical models for the ABM error model considered all multiplier structure elements. Every approximate circuitry unit has the right multiplier circuit topology for compression, PP array, and Booth encoding. This approach calculated NMED, a crucial error statistic. ABMs can be evaluated quickly and accurately using this method. The ABM1 probabilistic error model excels on 8-bit computers.

REFERENCES

1. Wang, Z., & Chen, Y. (2017). "Design of low-power and high-speed approximate Booth multipliers." IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 25(5), 1737-1746.
2. Liu, J., & Han, J. (2018). "Approximate computing for energy-efficient Booth multipliers." IEEE Transactions on Emerging Topics in Computing, 6(3), 370-383.
3. Kumar, P., & Singh, V. (2018). "Error-tolerant Booth multipliers for image processing applications." Microelectronics Journal, 77, 11-19.
4. Tan, C., & Lim, S. (2019). "Design of high-performance approximate Booth multipliers

ISSN: 2366-1313

for digital signal processing." IEEE Transactions on Circuits and Systems I: Regular Papers, 66(7), 2658-2668.
5. Zhang, H., & Wang, Y. (2019). "Approximate Booth squarers for error-tolerant computing systems." ACM Journal on Emerging Technologies in Computing Systems, 15(2), 23-33.
6. Shen, W., & Li, P. (2020). "Energy-efficient approximate Booth squarers for machine learning applications." IEEE Transactions on Circuits and Systems II: Express Briefs, 67(5), 823-827.
7. Kang, J., & Kim, H. (2020). "Design and analysis of error-tolerant Booth squarers for approximate computing." Microprocessors and Microsystems, 77, 103246.
8. Liu, X., & Xu, W. (2021). "Approximate Booth squarers for low-power and high-speed computing." Journal of Low Power Electronics and Applications, 11(1), 23-36.
9. Chen, L., & Zhou, Q. (2021). "Design exploration of approximate Booth squarers with configurable accuracy." Integration, the VLSI Journal, 80, 44-53.
10. Rao, N., & Patel, M. (2022). "Energy-efficient approximate Booth squarers for real-time error-tolerant applications." IEEE Transactions on Biomedical Circuits and Systems, 16(2), 198-206.
11. Mehta, A., & Gupta, S. (2022). "Reliability-aware design of approximate Booth squarers for embedded systems." IEEE Transactions on Device and Materials Reliability, 22(3), 412-420.
12. Singh, R., & Sharma, A. (2023). "Adaptive voltage scaling for approximate Booth squarers in IoT devices." IEEE Internet of Things Journal, 10(1), 67-76.