INTEGRATION, the VLSI journal 81 (2021) 17–24

Contents lists available at ScienceDirect

Integration

journal homepage: www.elsevier.com/locate/vlsi

# Approximate multipliers based on a novel unbiased approximate 4-2 compressor

Bao Fang [a], Huaguo Liang [a], Dawen Xu [a], Maoxiang Yi [a], Yongxia Sheng [a], Cuiyun Jiang [b], Zhengfeng Huang [a], Yingchun Lu [a,*]

[a] School of Electronic Science & Applied Physics, Hefei University of Technology, Hefei, 230601, China
[b] School of Mathematics, Hefei University of Technology, Hefei, 230601, China

ARTICLE INFO

*Keywords:*
Approximate computing
Multiplier
Low power-delay product
Approximate compressor

ABSTRACT

Approximate computing, which relaxes full precision to achieve higher performance or lower power consumption, is widely used in error-resilient applications such as multimedia and machine learning. Multiplication is a fundamental operation for many of these applications. This study presents an unbiased approximate 4–2 compressor that generates positive and negative sign errors in balance. Based on the unbiased 4–2 compressor, two 8 × 8 unbiased approximate multipliers (UBAMs) are designed to meet various accuracy (or power) requirements. Experimental results indicate that one of the proposed designs has a smaller normalized mean error distance (NMED) than previous approximate multipliers, and the other offers a 39% smaller power-delay product (PDP) and almost 46% smaller energy-delay product (EDP). The proposed multipliers outperform other approximate designs in image filtering applications by achieving higher quality outputs with lower power consumption.

## 1. Introduction

With the rapid development of the Internet of Things (IoT) and edge computing, the demand for low power consumption by smart mobile devices (SMDs) is increasingly urgent, owing to their limited battery life [1]. Approximate computing, as an emerging trend in the field of low-power design, has shown great advantages in fields such as image processing, video surveillance, and robot vision [2]. Approximate computing relaxes full precision but still achieves meaningful and useful results in applications that can tolerate errors and imprecision [3]. This allows a hardware platform to run more simply and to perform at a higher level. Compared to precise logic circuits, approximate-computation hardware has the benefits of extremely high performance and low power consumption [4,5].

The multiplier is a fundamental arithmetic unit of a processor and is widely used in almost all applications, ranging from filtering to convolutional neural networks. Notable efforts have been made in recent years to use multipliers for approximate computation [6], and these can be divided into three categories: approximation of partial product (PP) generation, approximation of PP tree, and approximation of compressors [7].

From the aspect of PP generation, an underdesigned multiplier (UDM) changed the output of a 2 × 2 multiplier module corresponding to a Karnaugh map, and the 2 × 2 multiplier was used to generate the approximate partial product [8]. This technique has been extended to *N*-bit high-width wide multipliers. A UDM with carry prediction during PP accumulation was studied [9]. Architectural-space exploration of approximate multipliers was investigated using an approximate multiplier [10], compared to UDM [8], with half the maximum error value. But its maximum error occurred more frequently, and it showed a worse result from the delay aspect.

PP tree methods truncate the least significant bits in partial products and remove the corresponding adders for the least significant bits to reduce the circuit area of a chip and its power consumption. This introduces more obvious truncation errors. The truncated bits of the multiplier result are replaced by a correction constant [11], and then are calculated with an estimated value of the error generated by truncating the least significant bits and rounding the result to n bits [12]. Finally, appropriate correction functions, variable correction, and linear compensation are used to reduce the truncation error [13,14]. Many other designs use PP trees, such as the approximate Wallace tree multiplier (AWTM) [15], error-tolerant multiplier (ETM) [16], static

* Corresponding author.
*E-mail address:* luyingchun@hfut.edu.cn (Y. Lu).

https://doi.org/10.1016/j.vlsi.2021.05.003
Received 13 December 2020; Received in revised form 29 March 2021; Accepted 10 May 2021
Available online 25 May 2021

segment multiplier (SSM) [17], and significance-driven logic compression (SDLC) [18].

For approximation of compressors, researchers have proposed various circuit designs to optimize the compressor [19–21]. J. Han et al. proposed two approximate 4–2 compressors, both converting a multi-operand sum to two operand additions, and used many of these to build an 8 × 8 multiplier with the Dadda tree structure, significantly reducing the hardware complexity while incurring a high error rate [19]. A logic approximation method used a Karnaugh map to simplify the calculation logic to design three approximate compressors with a low error rate, but with significantly higher energy consumption [20]. Another energy-efficient approximate 4–2 compressor was proposed to better balance energy consumption and precision [21]. R. Marimuthu et al. designed an approximate 15–4 compressor using an approximate 5–3 compressor as a basic module [22]. Approximate half-adders, full-adders, and 4–2 compressors were proposed and used in two variants of 16-bit multipliers [23]. Several improved approximate compressors have been proposed in Refs. [24–26].

However, the 4–2 compressors mentioned above generate almost all positive errors. We design unbiased compressors that generate both positive and negative errors that neutralize each other. Benefiting from the error decrement by the compressors, we prune the most significant bits (MSBs) from n-1 to n-3 to reduce the timing latency of the critical path for high performance, since the timing latency of the accuracy-part circuit determinates the entire multiplier. Two approximate multipliers, UBAM-M1 and UBAM-M2, use unbiased compressors.

This paper makes the following contributions.

1. An approximate 6–2 compressor is built by combining the proposed approximate full adder and a compressor [28] to balance the signs of errors of each compressor.
2. The proposed unbiased approximate 4–2 compressor generates positive and negative sign errors in balance, increasing the chance that 4–2 compressors in the same column of the multiplier generate opposite sign errors, which decreases its error.
3. Two unbiased approximate multipliers, UBAM-M1 and UBAM-M2, are based on the proposed approximate compressors. UBAM-M1 improves the precision of approximate multipliers and is more accurate than previous designs. UBAM-M2 reduces the power consumption of approximate multipliers. Compared to previous designs, it reduces delay by 22%, power consumption by 28%, and area by 26%, and it improves the power-delay product (PDP) and energy-delay product (EDP) by almost 39% and 46%, respectively.

The remainder of this paper is organized as follows. Section 2 introduces the exact full adder and exact 4–2 compressor. The proposed approximate compressors and approximate multiplier architecture are described and compared to previous designs in section 3. Experimental results for the approximate multipliers with approximate compressors are provided in section 4, which reports on their hardware and error performance metrics. Section 5 concludes the article.

## 2. Background

### 2.1. Exact full adder

The full adder calculates the number of "1"s in the input, and is expressed as

$$X1 + X2 + Cin = 2Carry + Sum, \tag{1}$$

where X1, X2, and Cin are inputs, and Carry and Sum are outputs. Sum and the inputs have the same weight, and Carry's weight is one bit higher. Sum and Carry are expressed as

$$Sum = X1 \oplus X2 \oplus Cin$$

$$Carry = X1 \cdot X2 + Cin \cdot (X1 \oplus X2). \tag{2}$$

### 2.2. Exact 4-2 compressor

The function of the exact 4–2 compressor is achieved by the proper connection of two exact full adders (see Fig. 1), which can be expressed as follows:

$$Sum = X1 \oplus X2 \oplus X3 \oplus X4 \oplus Cin$$

$$Cout = (X1 \oplus X2) \cdot X3 + \overline{(X1 \oplus X2)} \cdot X1$$

$$Carry = (X1 \oplus X2 \oplus X3 \otimes X4) \cdot Cin + \overline{(X1 \oplus X2 \oplus X3 \otimes X4)} \cdot X4. \tag{3}$$

The two stages of an exact 4–2 compressor chain are shown in Fig. 2 [27]. The output Sum has the same weight as the input signal, Cin is the carry from the lower compressor, Cout is the carry in for the next higher-order compressor (i.e., $Cin_i = Cout_{i-1}$), and the output Carry is accumulated like a PP bit in a one-bit-higher position. Note that Cout and Carry have the same weight. Its expression is

$$X1 + X2 + X3 + X4 + Cin = 2(Carry + Cout) + Sum. \tag{4}$$

## 3. Proposed approximate multiplier

We next propose an approximate full adder, which we use with an adder and 4–2 compressor to design a 6–2 compressor. We then propose an unbiased 4–2 compressor and design an approximate multiplier architecture.

### 3.1. Approximate full adder

A new approximate full adder is presented. The main goal of the compressor is to reduce delay and power. It mainly produces negative sign error, which may provide compensation when used with compressors that mainly produce positive sign error. The truth table of the proposed approximate full adder is shown in Table 1, where X1, X2, and Cin are the inputs of the full adder; Carry and Sum are the outputs of the exact full adder; and Carry' and Sum' are the outputs of the approximate adder, Value is the exact result, and Vapp is the result of the approximate adder. The proposed approximate full adder has no carry propagation, so Carry' and Sum' have the same weight. Vapp is defined as

$$Vapp = Carry' + Sum'. \tag{5}$$

The error between the exact and approximate compressor is

$$Err_i = Exact - Approximate. \tag{6}$$

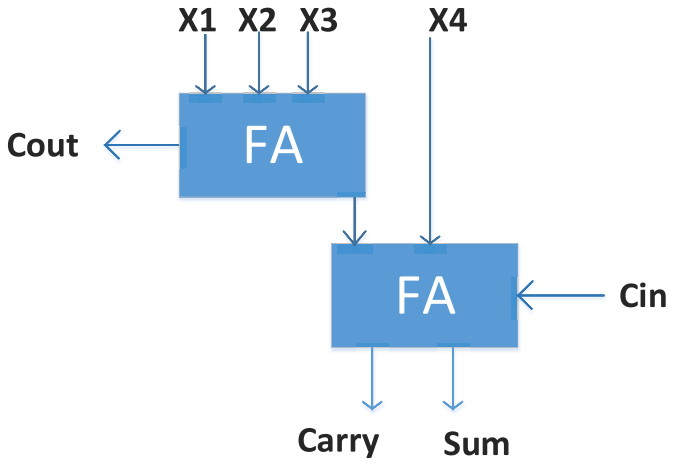


**Fig. 1.** Exact 4–2 compressor.

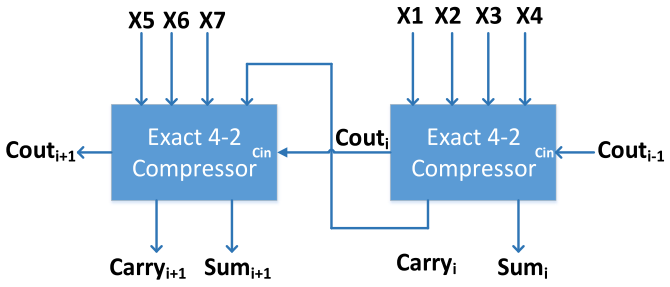


**Fig. 2.** Exact 4–2 compressor chain.

The mean error (ME) is used to estimate the error of the approximate compressor [28]:

$$\mathrm{ME} = \sum_{i}^{n} (Err_i \times \mathrm{Prob.}). \tag{7}$$

The probability that each partial product is high is 1/4 [28]. From the "Err$_i$" column in Table 1, we can deduce that the ME value of the proposed approximate full adder is −17/64. It mainly produces negative sign error, but the approximate full adder [28] only produces positive sign error, and these two adders probably compensate for each other.

Fig. 3(a) shows the gate-level circuit of the proposed approximate full adder. Fig. 3(b) shows the circuit of [28]. Assuming each gate has delay Δ, our design has critical path delay Δ, while [28] has delay 2Δ. Hence, we have reduced the timing delay by Δ. Furthermore, the proposed adder saves a certain chip area by using only a 2-input OR gate and 3-input OR gate.

### 3.2. Proposed approximate 6-2 compressor

An approximate 6–2 compressor is proposed to directly process six input values (X1, X2, …, X6) and generate two output values (Sum 1' and Sum 2'). Fig. 4 illustrates its implementation. It can be seen that the 6–2 compressor consists of the proposed approximate full adder, which mainly produces negative sign error, an approximate full adder [28] that mainly produces positive sign error, and a 4–2 compressor [28]. This structure increases the probability that the two full adders generate opposite sign errors. Since these three components are all carry-free, the proposed 6–2 compressor is also carry-free. The approximate 6–2 compressor outputs

$$\begin{aligned} Sum1' &= (X4 \cdot X5 + X6) \cdot (X4 + X5) + \\ &\quad (X1 + X2 + X3) + (X1 + X3) \\ Sum2' &= (X1 + X2 + X3) \cdot (X1 + X3) + \\ &\quad (X4 \cdot X5 + X6) + (X4 + X5). \end{aligned} \tag{8}$$

### 3.3. Unbiased approximate 4-2 compressor

A 4–2 compressor counts the number of "1″s in inputs. The circuit complexity will be effectively reduced with slight precision loss, when ignoring Cin and Cout [19–21]. In this case, expression (4) can be rewritten as

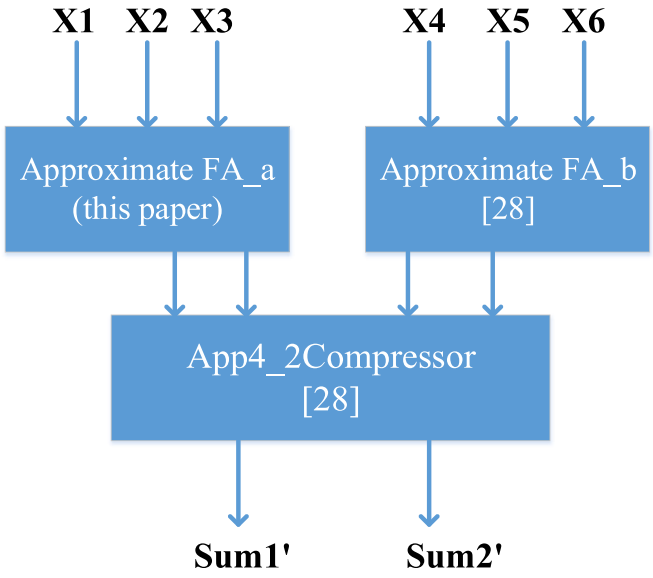


**Fig. 4.** Implementation of approximate 6–2 compressor.

**Table 1**
Truth table of approximate full adder.

| X1 | X2 | Cin | Carry | Sum | Value | Prob. | Carry' | Sum' | Vapp | Err$_i$ | Err$_i$ [28] |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 27/64 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 | 1 | 9/64 | 1 | 1 | 2 | −1 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 9/64 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0 | 2 | 3/64 | 1 | 1 | 2 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 1 | 9/64 | 1 | 1 | 2 | −1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 2 | 3/64 | 1 | 1 | 2 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 2 | 3/64 | 1 | 1 | 2 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 3 | 1/64 | 1 | 1 | 2 | 1 | 1 |

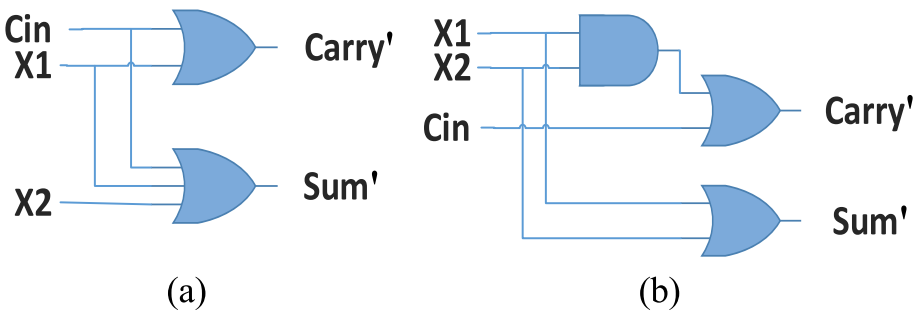


**Fig. 3.** Gate-level implementation of approximate full adder. (a) This paper. (b) Paper [28].

$$X1 + X2 + X3 + X4 = 2Carry + Sum. \quad (9)$$

Table 2 presents the Karnaugh map (K-MAP) of the exact 4–2 compressor without Cin and Cout. X1-X4 are the inputs, and Carry and Sum (CS) are the outputs.

The Karnaugh map is used to simplify the calculation logic. Tables 3 and 4 present the respective Karnaugh maps of the Carry' and Sum' of the proposed unbiased approximate 4–2 compressor App4-2Com. Ignoring Cin and Cout, X1-X4 are the inputs, and the outputs are Carry' and Sum'. In Tables 3 and 4, "0 → 1" indicates an entry in which "0" is replaced by "1," and "1 → 0" indicates the opposite. Hence the outputs of the approximate 4–2 compressor are

$$Carry' = X1 \cdot X2 + X3 \cdot X4 + X2 \cdot X4 + X1 \cdot X4$$

$$Sum' = X1 \oplus X2 + X3 \oplus X4 + X2 \cdot X4 + X1 \cdot X4. \quad (10)$$

Table 5 presents the truth table of the approximate 4–2 compressor, where X1 … X4 are the inputs, and $Err_i$ is the error of the App4-2Com. From (7), the ME of [21] is +37/256, while the ME of App4-2Com is 0. This is because App4-2Com produces both negative and positive sign error with the same probability. The proposed approximate 4–2 compressor has less loss of accuracy than [21] when an even number of approximate 4–2 compressors in the same column are used to accumulate partial products. Therefore, the errors of App4-2Com have a higher probability of compensating for each other to decrease the error of the entire multiplier, i.e., the approximate multiplier built with App4-2Com has less loss of precision.

Fig. 5 shows the gate-level circuit of the proposed approximate 4–2 compressor. The delay of App4-2Com is 2Δ, which is less than [21] by Δ. App4-2Com has four AND gates, two XOR gates, and two OR gates.

### 3.4. Approximate multiplier architecture

We design the approximate multiplier architecture using the proposed approximate compressors. A fast (exact) multiplier is commonly implemented in three parts [29]:

1. A set of partial products is generated with many AND gates.
2. These partial products are compressed (or reduced) to an addition of only two operands.
3. A carry propagation adder (CPA) is used to form the final binary product.

The second of these three parts plays a pivotal role in terms of area, delay, and power consumption. As a result, approximate compressors have been widely used to compress a partial product compression tree to achieve lower power consumption and better performance.

When using an approximate compressor to compress a partial product compression tree, the most significant bits (MSBs) of the partial product greatly influence the accuracy of the multiplier, and the least significant bits (LSBs) have little effect on accuracy. As shown in Fig. 6, to get an optimized tradeoff between accuracy and hardware overhead, our proposed 8-bit approximate multiplier (LSBs are from column 1 to column 8, and MSBs are from column 9 to column 15, where the rightmost column is the first column) divides the partial product compression tree into three parts: the exact part that is MSBs of the multiplier (column 11 to column 15), the approximate part whose significance is between MSBs and LSBs (column 6 to column 10), and the truncated part that is LSBs of the multiplier (column 1 to column 5).

**Table 2**
K-MAP OF CS without CIN and Cout.

| X1X2 | X3X4 | | | |
|---|---|---|---|---|
| | 00 | 01 | 11 | 10 |
| 00 | 00 | 01 | 10 | 01 |
| 01 | 01 | 10 | 11 | 10 |
| 11 | 10 | 11 | 11 | 11 |
| 10 | 01 | 10 | 11 | 10 |

**Table 3**
K-MAP of carry'.

| X1X2 | X3X4 | | | |
|---|---|---|---|---|
| | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 1 | 0 |
| 01 | 0 | 1 | 1 | 1 → 0 |
| 11 | 1 | 1 | 1 | 1 |
| 10 | 0 | 1 | 1 | 1 → 0 |

**Table 4**
K-MAP of sum'.

| X1X2 | X3X4 | | | |
|---|---|---|---|---|
| | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | 0 | 1 |
| 01 | 1 | 0 → 1 | 1 | 0 → 1 |
| 11 | 0 | 1 | 1 | 1 |
| 10 | 1 | 0 → 1 | 1 | 0 → 1 |

**Table 5**
Truth table of approximate 4-2 compressor.

| X1 | X2 | X3 | X4 | Prob. | $Err_i$ | $Err_i$ [21] |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 81/256 | 0 | 0 |
| 0 | 0 | 0 | 1 | 27/256 | 0 | 0 |
| 0 | 0 | 1 | 0 | 27/256 | 0 | 0 |
| 0 | 0 | 1 | 1 | 9/256 | 0 | 0 |
| 0 | 1 | 0 | 0 | 27/256 | 0 | 0 |
| 0 | 1 | 0 | 1 | 9/256 | −1 | 1 |
| 0 | 1 | 1 | 0 | 9/256 | 1 | 1 |
| 0 | 1 | 1 | 1 | 3/256 | 0 | 0 |
| 1 | 0 | 0 | 0 | 27/256 | 0 | 0 |
| 1 | 0 | 0 | 1 | 9/256 | −1 | 1 |
| 1 | 0 | 1 | 0 | 9/256 | 1 | 1 |
| 1 | 0 | 1 | 1 | 3/256 | 0 | 0 |
| 1 | 1 | 0 | 0 | 9/256 | 0 | 0 |
| 1 | 1 | 0 | 1 | 3/256 | 0 | 0 |
| 1 | 1 | 1 | 0 | 3/256 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1/256 | 0 | 1 |

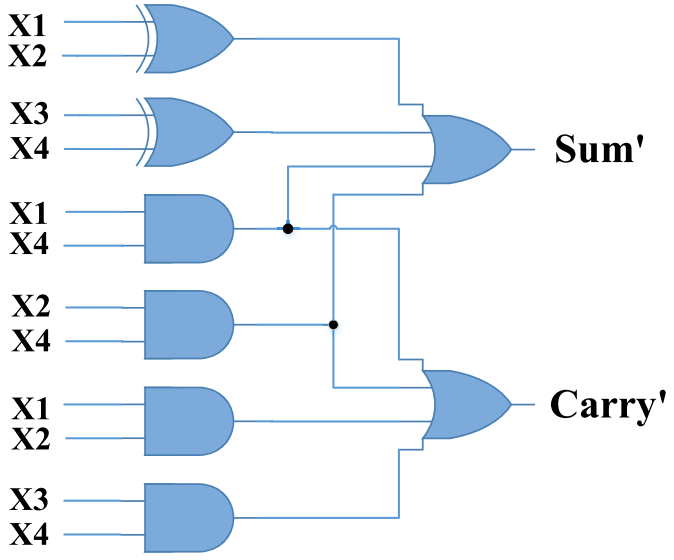


**Fig. 5.** Gate-level implementation of approximate 4–2 compressor.

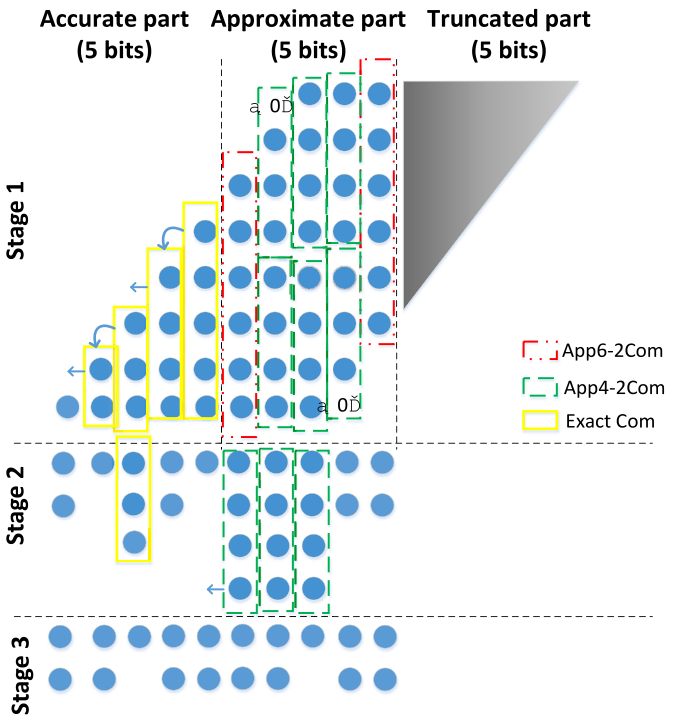


**Fig. 6.** Proposed approximate multiplier.

For the exact part, the exact compressors are used to calculate the partial product to ensure the accuracy of the MSBs. Since the critical path of the entire multiplier always comes from the accuracy-part circuit for calculating the first seven bits of the result in the multiplier, our main idea is to prune the accuracy bits from seven to five bits to decrease the timing delay of the critical path. The approximate unbiased compressor we proposed for the approximate part can improve the accuracy loss caused by pruning accuracy bits. The approximate part is from column 6 to column 10. For the first and last columns of the approximate part, six partial products are calculated with a 6–2 compressor and the two generated output values for the input of stage 2. Each of columns 7 to 9 is composed of two 4–2 compressors, and the column containing only seven partial products is padded with zeros. Each 4–2 compressor processes four input values and generates two output values for stage 2. In the truncated part, the least significant five bits are truncated, and the corresponding compressors are removed to reduce the chip area and power consumption.

## 4. Experimental results

Hardware performance, overhead, and accuracy are important criteria for the approximate multiplier. The proposed approximate multiplier was implemented in Verilog HDL and synthesized with Synopsys Design Compiler (DC) under the TSMC 90 nm library to evaluate its power, delay, area, power-delay product (PDP), and energy-delay product (EDP). For accuracy evaluations, a program was developed in C++ and simulated in Microsoft Visual Studio using the approximate multipliers. Image filtering was taken as the application. The multipliers with different strategies are shown in Table 6.

### 4.1. Circuit performance simulation

Table 6 summarizes the seven approximate multipliers and one exact multiplier. The exact 8-bit multiplier was designed using the Dadda tree structure. UBAM-M1 and UBAM-M2, as designed in this paper, were based on the proposed approximate compressors. The main goal of UBAM-M1 is to improve the precision of approximate multipliers, while UBAM-M2 is meant to reduce their power consumption.

**Table 6**
Approximate multipliers (AMs) and their features.

| Design | Features |
|---|---|
| Exact | Exact Multiplier |
| UBAM-M1 | Proposed Unbiased Approximate 4-2 Compressors |
| UBAM-M2 | Proposed Unbiased Approximate 4–2 and 6-2 Compressors & MSB Pruning & Truncation |
| AM [21]-T | [21] Compressors, MSB Pruning, & LSB Truncation |
| AM [21] | [21] Compressors |
| UDM [8] | 8-bit Underdesigned Multiplier (UDM) [8] |
| AM [10] | 8-bit Approximate Multiplier $ApproxMul_2$ [10] |
| mul8u_DM1 [30] | 8-bit Approximate Multiplier mul8u_DM1 [30] |

Approximate multiplier [21] (AM [21]) as proposed in Ref. [21], was built with approximate 4–2 compressors. AM [21]-T is designed to compare to AM [21], and it adopts the same MSB pruning and truncation technique as the proposed UBAM-M2.

Table 7 summarizes the area, power, delay, PDP, and EDP of the eight multipliers. Fig. 7 shows the improvement ratio of different approximate multipliers. Comparing AM [21]-T with AM [21], it can be seen that the MSB pruning technique reduces the circuit delay by 17%. This is because AM [21]-T prunes the accuracy bits from seven to five bits and reduces the timing delay of the critical path. For the power and area, comparing AM [21]-T to AM [21] shows that the MSB truncation technique reduces power consumption by about 26% and chip area by over 23%. The MSB pruning & LSB truncation technique reduces PDP by about 35%, and the MSB pruning & LSB truncation technique reduces EDP by about 41%.

As expected, the proposed UBAM-M2 is best among these multipliers except in area saving. Compared with AM [21], it improves delay by over 22%, power consumption by 28%, chip area by 26%, PDP by almost 39%, and EDP by almost 46%. The mul8u_DM1 design [30] performs well on power and area saving. However, it suffers in terms of delay, where a negative percentage connotes worsening. UBAM-M2 offers 36% delay saving and 7% power saving over mul8u_DM1 [30]. UBAM-M1 is proposed to improve the precision of approximate multipliers. Therefore, it plays a relatively minor role in the circuit performance, compared to UBAM-M2. But UBAM-M1 still achieves about 11% power saving and 16% PDP improvement over the exact multiplier.

### 4.2. Error performance analysis

An important metric for an approximate design is the output accuracy with respect to the exact result. Several metrics have been proposed to measure the error of approximate multipliers, including the error rate (ER), mean error (ME), normalized mean error (NME), normalized mean square error (NMSE) [21], mean error distance (MED), normalized mean error distance (NMED) [5], and maximum absolute error (MAE) [10,30]. ME, NME, NMSE, MED, and NMED are defined as

$$ME = \frac{\sum_{i=0}^{n}(\mathrm{P} - \mathrm{P_A})}{n}$$

$$NME = \frac{ME}{Output_{\max}}$$

$$MSE = \frac{\sum_{i=0}^{n}(\mathrm{P} - \mathrm{P_A})^2}{n}$$

$$NMSE = \frac{MSE}{Output_{\max}}$$

$$MED = \frac{\sum_{i=0}^{n}|\mathrm{P} - \mathrm{P_A}|}{n}$$

**Table 7**
Circuit performance of different multipliers.

| Multiplier | Delay (ns) | Delay Impro. | Power (μW) | Power Impro. | Area (μm$^2$) | Area Impro. | PDP (fJ) | PDP Impro. | EDP (fJ·ns) | EDP Impro. |
|---|---|---|---|---|---|---|---|---|---|---|
| Exact | 1.66 | 0% | 91.2 | 0% | 1438.0 | 0% | 151.4 | 0% | 251.3 | 0% |
| UBAM-M1 | 1.55 | 6.63% | 81.6 | 10.53% | 1320.2 | 8.19% | 126.5 | 16.45% | 196.1 | 21.97% |
| UBAM-M2 | **1.18** | **28.92%** | **53.7** | **41.12%** | 896.1 | 37.68% | **63.4** | **58.12%** | **74.8** | **70.23%** |
| AM [21] | 1.55 | 6.63% | 79.5 | 12.83% | 1282.1 | 10.84% | 123.2 | 18.63% | 191.0 | 24.00% |
| AM [21]-T | 1.26 | 24.10% | 55.5 | 39.14% | 951.2 | 33.85% | 69.9 | 53.83% | 88.1 | 64.94% |
| UDM [8] | 1.53 | 7.83% | 80.9 | 11.29% | 1071.7 | 25.47% | 123.8 | 10.23% | 189.4 | 24.63% |
| AM [10] | 1.69 | −1.81% | 84.4 | 7.46% | 1354 | 5.84% | 142.6 | 5.81% | 241.0 | 4.10% |
| mul8u_DM1 [30] | 1.77 | −6.63% | 59.7 | 34.54% | **800.2** | **44.35%** | 105.7 | 30.18% | 187.1 | 25.55% |

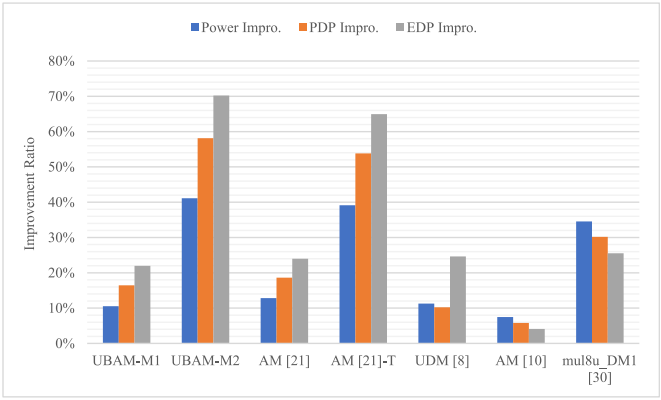


**Fig. 7.** Improvement ratio of different approximate multipliers.

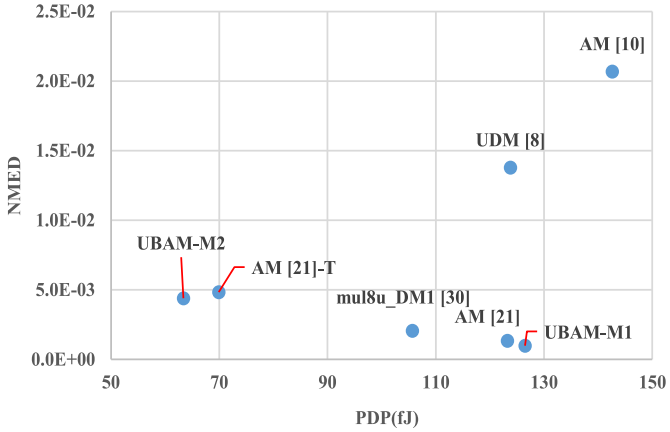


**Fig. 8.** PDP vs. NMED.

$$NMED = \frac{MED}{Output_{max}}. \tag{11}$$

In (11), P and $P_A$ denote the exact and approximate products, respectively; n is the number of all input cases (total $2^{16} = 65{,}536$); and $Output_{max}$ is the maximum value of the product generated by the exact multiplier (the maximum value is $256 \times 256 = 65{,}536$).

ER is the probability of an incorrect result. MAE is used to determine the approximate design error scale, and its value is the worst output result of the approximate design, i.e., the maximum error distance. Better designs will tend to have low values of these metrics, hence the approximate multiplier is shown to have higher precision.

The results in Table 8 show that the UBAM-M1 is more accurate than its competitors. UBAM-M1 has the smallest ME, NME, MED, NMED, and NMSE. UBAM-M1 and AM [21] show the best MAE, which indicates the upper limit of error is the lowest. Although UDM [8] has a low ER, it shows worse results on NMED, NMSE, and MAE, which indicates a high loss of accuracy. Compared to UBAM-M1, UBAM-M2 shows a small disadvantage in accuracy. However, the main goal of UBAM-M2 is to reduce power consumption and PDP, and it shows a significant improvement in terms of NME, NMSE, and MAE, compared with UDM [8] and AM [10].

Fig. 8 provides a comprehensive comparison of approximate multipliers to get an idea of the tradeoff between circuit performance and error metrics. Fig. 8 shows PDP vs. NMED for the considered unsigned 8 × 8 approximate multipliers. Note that designs at the lower-left corner are the best, with small PDPs and high accuracies. UBAM-M2 has the smallest PDP, and UBAM-M1 has the smallest NMED. Therefore, UBAM-M1 can be used for applications requiring high accuracy. For moderate power saving with better performance, UBAM-M2 is suggested.

### 4.3. Image filtering

We illustrate the accuracy in the application of the multipliers to image processing. The multipliers are used in Gaussian image filtering, which contains a large number of multiplication operations. Structural similarity (SSIM) and peak signal-to-noise ratio (PSNR) are two common criteria used to evaluate the differences of a distorted image from a reference image. Better designs will tend to have a high PSNR. SSIM was introduced to improve traditional image quality assessment methods such as PSNR [31]. The value range is [0,1], and a larger value is better.

Fig. 9 shows the four original test images for Gaussian filtering. The image processed by the exact multiplier is used as the reference image, and the SSIM and PSNR values for several approximate multipliers are measured in MATLAB. As shown in Table 9, UBAM-M1 is more accurate than the other designs. Note that with respect to the SSIM, our proposed design, UBAM-M1, is the best design with the highest SSIM value. UBAM-M2 offers a 58.12% PDP saving at the cost of 1.24% SSIM degradation (the Lena test image).

**Table 8**
Accuracy comparison for different approximate multipliers.

| Multiplier | ER | ME | NME | MED | NMED | NMSE | MAE |
|---|---|---|---|---|---|---|---|
| UBAM-M1 | 65.27% | **7.60** | **1.16E-04** | **6.46E + 01** | **9.85E-04** | **1.47E-01** | **520** |
| UBAM-M2 | 93.99% | 1.13E+02 | 1.72E-03 | 2.89E+02 | 4.41E-03 | 3.12 | 3777 |
| AM [21] | 66.45% | 8.78E+01 | 1.34E-03 | 8.78E+01 | 1.34E-03 | 2.53E-01 | **520** |
| AM [21]-T | 92.40% | 3.17E+02 | 4.84E-03 | 3.17E+02 | 4.84E-03 | 3.44 | 1994 |
| UDM [8] | **46.73%** | 9.03E+02 | 1.38E-02 | 9.03E+02 | 1.38E-02 | 9.86E+01 | 14,450 |
| AM [10] | 81.44% | 1.35E+03 | 2.07E-02 | 1.35E+03 | 2.07E-02 | 8.87E+01 | 7225 |
| mul8u_DM1 [30] | 98.16% | 2.03E+01 | 3.09E-04 | 1.34E+02 | 2.05E-03 | 4.27E-01 | 585 |

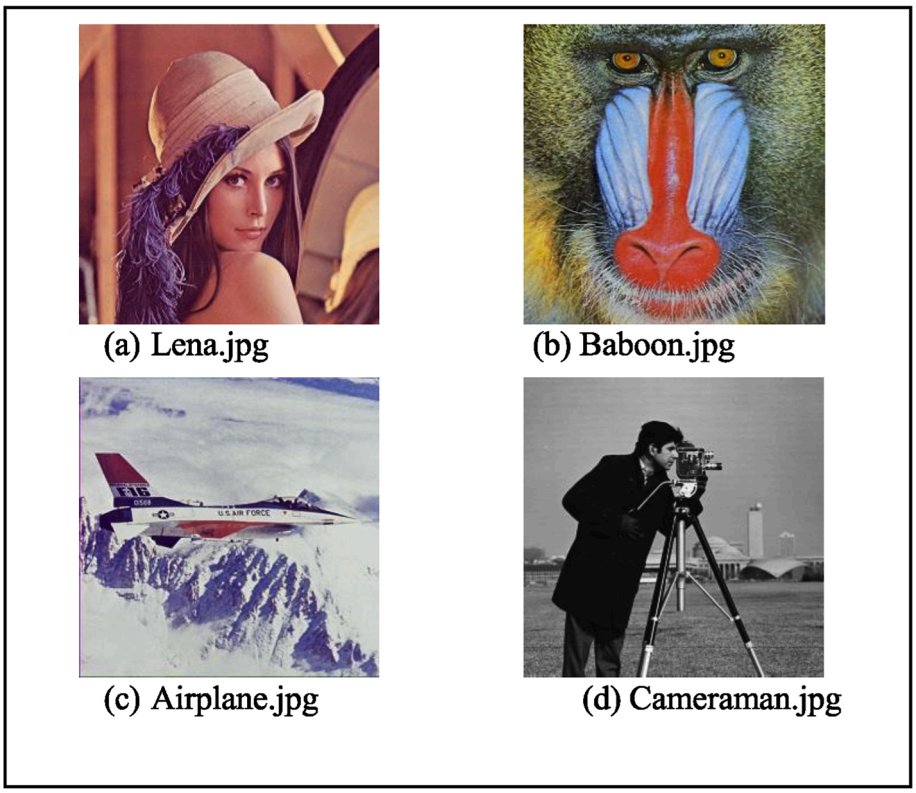


**Fig. 9.** The original test images for Gaussian filtering.

**Table 9**
Image quality comparison processed by Gaussian filtering.

| Multiplier | Lena.jpg | | Baboon.jpg | | Airplane.jpg | | Cameraman.jpg | |
|---|---|---|---|---|---|---|---|---|
| | PSNR (dB) | SSIM | PSNR (dB) | SSIM | PSNR (dB) | SSIM | PSNR (dB) | SSIM |
| UBAM-M1 | **34.91** | **0.9914** | **34.85** | **0.9916** | **34.61** | **0.9939** | **35.15** | **0.9502** |
| UBAM-M2 | 32.73 | 0.9876 | 33.1 | 0.9902 | 31.36 | 0.9866 | 34.02 | 0.9368 |
| AM [21]-T | 30.37 | 0.984 | 30.4 | 0.9846 | 30.73 | 0.9858 | 31.07 | 0.9348 |
| AM [21] | 33.01 | 0.9893 | 33.05 | 0.99 | 32.88 | 0.9931 | 33.54 | 0.9409 |
| UDM [8] | 31.02 | 0.9874 | 31.69 | 0.9876 | 28.59 | 0.9838 | 33.86 | 0.9485 |
| AM [10] | 27.27 | 0.9685 | 27.49 | 0.9699 | 28.07 | 0.9819 | 29.70 | 0.9203 |
| mul8u_DM1 [30] | 34.31 | 0.9898 | 34.05 | 0.9902 | 34.46 | 0.9929 | 33.92 | 0.9499 |

UBAM-M1 has the highest PSNR value, and shows significant improvement over AM [21]. This is because the proposed unbiased approximate 4–2 compressors decrease the error of the entire multiplier. UBAM-M2 is slightly lower than AM [21] because the MSB pruning technique reduces its accuracy. However, UBAM-M2 provides a higher PDP saving (58.12% vs. 18.63%). The image processed by UBAM-M2 also has high quality (a PSNR of 30 dB can often be considered good enough) [32]. The quality of images processed by UDM [8] and AM [10] is worse than that of our proposed designs, because they have a high loss of precision. The mul8u_DM1 [30] design also provides a good PSNR, but it performs worse in terms of delay.

## 5. Conclusion

We proposed an unbiased approximate 4–2 compressor, which generates positive and negative sign errors in balance. This can increase the opportunity for error compensation over the 4–2 compressors in the same column of the partial product compression tree, and decrease the error of the entire multiplier. A new approximate 6–2 compressor was proposed. Based on the proposed compressors, two novel approximate multipliers, UBAM-M1 and UBAM-M2, were designed.

UBAM-M1 can be used for applications where high accuracy is desired. UBAM-M1 has the smallest NMED among the approximate designs (Table 8), and its MAE is only about 1/28 that of UDM [8]. UBAM-M2 is suggested for better power saving and performance, as it offers 36% delay saving and 7% power saving over mul8u_DM1 [30]. Compared to AM [21], UBAM-M2 improves over 22% on delay, 28% on power consumption, 26% on chip area, almost 39% on PDP, and almost 46% on EDP, owing to its proposed multiplier architecture with MSB pruning and LSB truncation.

The proposed multipliers were evaluated in image filtering applications. It was shown that UBAM-M1 produces more accurate output than other approximate multipliers by achieving higher quality (in terms of SSIM or PSNR). UBAM-M2 offers 58.12% PDP saving at the cost of 1.24% SSIM degradation.

## Author statement

Conceptualization, methodology, Bao Fang, Huaguo Liang; software, Bao Fang, Yongxia Sheng; validation, formal analysis, investigation, resources, Dawen Xu, Maoxiang Yi, Cuiyun Jiang; Writing - Original Draft, Bao Fang, Zhengfeng Huang, Yingchun Lu; Writing - Review & Editing, Bao Fang, Maoxiang Yi; Funding acquisition, Huaguo Liang, Yingchun Lu. All authors have read and agreed to the published version

of the manuscript.

## Declaration of competing interest

There are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

## Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61834006; in part by the Major National Scientific Instrument and Equipment Development Project under Grant 62027815; and in part by the Science and Technology on Analog Integrated Circuit Laboratory of China under Grant 6142802200506.

We thank LetPub (www.letpub.com) for its linguistic assistance during the preparation of this manuscript.

## Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.vlsi.2021.05.003.

## References

[1] L. Cui, et al., Joint optimization of energy consumption and latency in mobile edge computing for Internet of Things, IEEE Internet of Things J. 6 (3) (June 2019) 4791–4803.
[2] W. Liu, L. Qian, C. Wang, H. Jiang, J. Han, F. Lombardi, Design of approximate Radix-4 Booth multipliers for error-tolerant computing, IEEE Trans. Comput. 66 (8) (Aug. 2017) 1435–1441.
[3] J. Han, M. Orshansky, Approximate computing: an emerging paradigm for energy-efficient design, in: Proc. 18th IEEE Eur.Test Symp., 2013, pp. 1–6.
[4] S. Venkataramani, S.T. Chakradhar, K. Roy, A. Raghunathan, Computing Approximately, and Efficiently, Design, Automation & Test in Europe Conference & Exhibition (DATE), 2015, pp. 748–751.
[5] J. Liang, J. Han, F. Lombardi, New metrics for the reliability of approximate and Probabilistic Adders, IEEE Trans. Comput. 63 (9) (Sep. 2013) 1760–1771.
[6] Weiqiang Liu, Cao Tian, Peipei Yin, Yuying Zhu, Chenghua Wang, Earl E. Swartzlander, Fabrizio Lombardi, Design and analysis of approximate redundant binary multipliers, IEEE Trans. Comput. 68 (6) (Jun. 2019) 804–819.
[7] H. Jiang, C. Liu, N. Maheshwari, F. Lombardi, J. Han, A comparative evaluation of approximate multipliers, in: Proc. IEEE/ACM Int.Symp. NANOARCH, Jul.2016, pp. 191–196.
[8] P. Kulkarni, P. Gupta, M. Ercegovac, Trading accuracy for power with an underdesigned multiplier architecture, in: Proc.24th Int. Conf. VLSI, Des.,2011, pp. 346–351.
[9] K. Bhardwaj, P. Mane, J. Henkel, Power-and area-efficient approximate Wallace tree multiplier for error-resilient systems, in: Proc. 15th IEEE Int.Symp. Quality Electron., Des.,2014, pp. 263–269.
[10] S. Rehman, W. El-Harouni, M. Shafique, A. Kumar, J. Henkel, J. Henkel, Architectural-space exploration of approximate multipliers, in: 2016 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), Austin, TX, 2016, pp. 1–8.
[11] J. Schulte, E.E. Swartzlander Jr., Truncated multiplication with correction constant, in: Proc. Workshop VLSI Signal Process. VI, 1993, pp. 388–396.
[12] K.-J. Cho, K.-C. Lee, J.-G. Chung, K.K. Parhi, Design of low error fixed-width modified Booth multiplier, IEEE Trans. Very Large Scale Integr. Syst. 12 (5) (May 2004) 522–531.
[13] D. De Caro, N. Petra, A.G.M. Strollo, F. Tessitore, E. Napoli, Fixed-width multipliers and multipliers-accumulators with min-max approximation error, IEEE Trans. Circ. Syst. I, Reg. Pap. 60 (9) (Sep.2013) 2375–2388.
[14] S. Balamurugan, P.S. Mallick, "Error compensation techniques for fixed-width array multiplier design—a technical survey, J. Circuits,Syst. Comput. 26 (3) (Mar. 2017) 1730003.
[15] K. Bhardwaj, P.S. Mane, J. Henkel, Power- and area-efficient approximate Wallace tree multiplier for error-resilient systems, in: Proc. Int. Symp. Quality Electron. Design, 2014, pp. 263–269.
[16] K.Y. Kyaw, W.L. Goh, K.S. Yeo, Low-power high-speed multiplier for error-tolerant application, in: Proc. IEEE Int. Conf. Electron Devices Solid-State Circuits, Dec. 2010, pp. 1–4.
[17] S. Narayanamoorthy, H.A. Moghaddam, Z. Liu, T. Park, N.S. Kim, Energy-efficient approximate multiplication for digital signal processing and classification applications, IEEE Trans. Very Large Scale Integr. Syst. 23 (6) (Jun. 2015) 1180–1184.
[18] I. Qiqieh, R. Shafik, G. Tarawneh, D. Sokolov, A. Yakovlev, Energy-efficient approximate multiplier design using bit significance-driven logic compression, in: Proc. Des., Automat. Test Eur. Conf. Exhib. (DATE), 2017, pp. 7–12.
[19] A. Momeni, J. Han, P. Montuschi, F. Lombardi, Design and analysis of approximate compressors for multiplication, IEEE Trans. Comput. 64 (4) (Apr. 2015) 984–994.
[20] Z. Yang, J. Han, F. Lombardi, Approximate compressor for error-resilient multiplier design, in: IEEE Int. Symp. On Defect Fault Tolerance in VLSI and Nano. Syst., MA, USA, Oct. 2015.
[21] Xilin Yi, Haoran Pei, Ziji Zhang, Hang Zhou, Yajuan He, Design of an energy-efficient approximate compressor for error-resilient multiplications, in: IEEE International Symposium on Circuits and Systems (ISCAS), 2019.
[22] R. Marimuthu, Y.E. Rezinold, P. Mallick, Design and analysis of multiplier using approximate 15-4 compressor, IEEE Access 5 (2017) 1027–1036.
[23] S. Venkatachalam, S.-B. Ko, Design of power and area efficient approximate multipliers, IEEE Trans. Very Large Scale Integr. Syst. 25 (5) (May 2017) 1782–1786.
[24] M.M. Dominic Savio, T. Deepa, Design of higher order multiplier with approximate compressor, in: 2020 IEEE International Conference on Electronics, Computing and Communication Technologies, CONECCT, Bangalore, India, 2020, pp. 1–6.
[25] H. Pei, X. Yi, H. Zhou, Y. He, Design of ultra-low power consumption approximate 4-2 compressors based on the compensation characteristic, in: IEEE Transactions on Circuits and Systems II: Express Briefs, 2020 (Early Access).
[26] A.G.M. Strollo, D. De Caro, E. Napoli, N. Petra, G. Di Meo, Low-power approximate multiplier with error recovery using a new approximate 4-2 compressor, in: 2020 IEEE International Symposium on Circuits and Systems, ISCAS, Sevilla, 2020, pp. 1–4.
[27] M.S. Ansari, H. Jiang, B.F. Cockburn, J. Han, Low-power approximate multipliers using encoded partial products and approximate compressors, IEEE J. Emerg. Select. Top. Circ. Syst. 8 (3) (Sept. 2018) 404–416.
[28] Darjn Esposito, Antonio Giuseppe Maria Strollo Ettore Napoli, Davide De Caro, Nicola petra." approximate multipliers based on new approximate compressors, IEEE Trans. Circ. Syst. I: Reg. Pap. 65 (12) (Dec.2018) 4169–4182.
[29] Chip-Hong Chang, Jiangmin Gu, Mingyan Zhang, Ultra low-voltage low-power CMOS 4-2 and 5-2 compressors for fast arithmetic circuits, IEEE Trans. Circ. Syst. I: Reg. Pap. 51 (10) (Oct. 2004) 1985–1997.
[30] https://ehw.fit.vutbr.cz/evoapproxlib/?folder=multiplers/8x8_unsigned/pareto_pwr_mse.
[31] Z. Wang, A.C. Bovik, H.R. Sheikh, E.P. Simoncelli, Image quality assessment: from error visibility to structural similarity, IEEE Trans. Image Process. 13 (4) (Apr. 2004) 600–612.
[32] M.S. Ansari, H. Jiang, B.F. Cockburn, J. Han, Low-power approximate multipliers using encoded partial products and approximate compressors, IEEE J. Emerg. Select. Top. Circ. Syst. 8 (3) (Sept. 2018) 404–416.