Low-Power Unsigned Divider and Square Root Circuit Designs using Adaptive Approximation

Honglan Jiang, *Member, IEEE,* Leibo Liu, *Member, IEEE,* Fabrizio Lombardi, *Fellow, IEEE,* and Jie Han, *Senior Member, IEEE*

Abstract—In this paper, an adaptive approximation approach is proposed for the design of a divider and a square root (SQR) circuit. In this design, the division/SQR is computed by using a reduced-width divider/SQR circuit and a shifter by adaptively pruning some insignificant input bits. Specifically, for a $2n/n$ division, $2k$ and k ($k<n$) consecutive bits are selected starting from the most significant '1' in the dividend and divisor, respectively. At the same time, redundant least significant bits (LSBs) are truncated or if the number of remaining bits after pruning is smaller than the number of bits to be kept, '0's are appended to the LSBs of the inputs. To avoid overflow, a $2(k+1)/(k+1)$ divider is used to compute the $2k/k$ division. Finally, an error correction circuit is proposed to recover the error caused by the shifter using OR gates. For a $2n$-bit approximate SQR circuit, similar pruning schemes are used to obtain a $2k$-bit radicand. A $2k$-bit SQR circuit and a shifter are then utilized to compute the SQR. This adaptive operation leads to very small maximum error distances of the approximate divider and SQR circuits, as shown by a theoretical error analysis. The proposed 16/8 approximate divider using an 8/4 exact array divider is $2.5\times$ as fast but only consumes 34.42% of the power of the accurate design. Compared to the accurate 16-bit array SQR circuit, the approximate design with a 6-bit radicand is $3.9\times$ as fast and consumes 20.66% of the power. The approximate SQR circuit using a 6-bit lookup table-based SQR circuit consumes 7.15% of the power of its corresponding accurate design. The proposed designs outperform other approximate designs in image processing applications including change detection (for the divider), envelope detection (for the SQR circuit) and image reconstruction (for both designs).

Index Terms—Adaptive approximation, divider, SQR circuit, overflow, low-power, image processing.

1 Introduction

Compared with multiplication and addition, division and square root (SQR) computation are less frequently used arithmetic operations [1]; however, their long latencies determine the speed of an application once they are utilized. Several schemes have been proposed to improve the performance of division/SQR, such as those using a high-radix design [2], [3], [4] or a carry/borrow lookahead circuit in an array divider/SQR circuit [5]. Nevertheless, the improvement in performance is usually obtained at the expense of a high power dissipation and a large area due to the complexity of its intrinsic structure.

On the other hand, precise computing is not required for many applications such as image processing, clustering and recognition due to their inherent error tolerance. As a result, there has been a wide interest in the investigation into the emerging paradigm of approximate or inexact computing [6]. A large number of approximate designs have been proposed for arithmetic circuits for pursuing improvements in speed, power dissipation and/or circuit complexity [7].

However, most designs are either hardware-efficient with a low accuracy or very accurate with a limited hardware saving, mostly due to the use of a static approximation. In this paper, an adaptive approximation strategy is proposed for unsigned divider and SQR circuit designs. The adaptive approximation leads to low-power and high-performance operations. In these designs, input pruning and error correction work synergistically to ensure a high accuracy with a very low maximum error distance (ED).

Specifically, by adaptively selecting some significant bits in the input operands, a reduced-width divider and SQR circuit are respectively utilized to approximately compute a division and SQR. The alterable circuit size and accuracy of the approximate designs are determined by the number of bits to be selected (i.e., $2k$ bits for the dividend and radicand, and k bits for the divisor). Compared with the exact $16/8$ array divider, the adaptively approximate divider (denoted as AAXD) achieves a speedup and power reduction by 27%-61% and 34%-66%, respectively, when k varies from 5 to 3. For the 16-bit array SQR circuit, it achieves improvements in power-delay product (PDP) and area-delay product (ADP) by more than 90% when $k=3$, while the PDP is reduced by more than 45% when $k=6$. A more significant power reduction is achieved when the adaptive approximation is applied to the lookup table (LUT)-based SQR circuit. It also achieves a high accuracy in an envelope detection application. When used in image reconstruction, the proposed approximate $32/16$ divider and 32-bit SQR circuit are significantly faster and more power-efficient than the other approximate designs to obtain a similar quality to an accurate design.

Some preliminary results have been reported in [8]. This paper presents the following novel contributions: Similar to the approximate divider, an approximate SQR circuit is designed using pruning and shift operation. The upper bounds of the ED for the divider and SQR circuit are analytically found. These analyses show that the proposed approximation strategy results in a very small maximum ED. Moreover, the proposed design approach

- *H. Jiang and L. Liu are with the Institute of Microelectronics, Tsinghua University, Beijing, China.*
  E-mail: honglan@ualberta.ca, liulb@tsinghua.edu.cn
- *F. Lombardi is with the Department of Electrical and Computer Engineering, Northeastern University, Boston, USA.*
  E-mail: lombardi@ece.neu.edu
- *J. Han is with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB T6G 1H9, Canada.*
  E-mail: jhan8@ualberta.ca

is further applied for sequential divider and SQR circuits. The quality of the approximate SQR circuit is assessed by using it in envelope detection. Finally, to evaluate the accuracy and hardware improvements, both the approximate divider and SQR circuit are applied to an image reconstruction application.

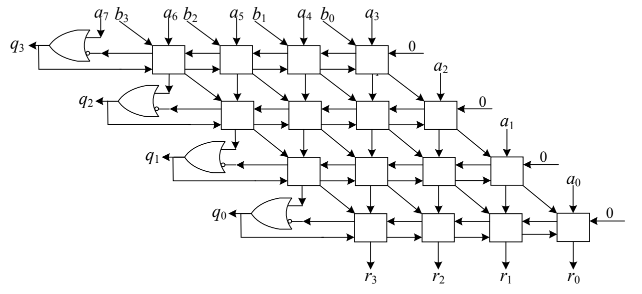
The rest of the paper is organized as follows. Section 2 introduces the exact designs of then divider and SQR circuit, and reviews current approximate divider designs. The approximate unsigned divider and SQR circuit are proposed in Section 3. The errors of the approximate designs are analyzed in Section IV. Section 5 shows the error and circuit characteristics of the approximate divider and SQR circuit. In Section 6, the approximate designs are applied to three image processing applications to assess their accuracy. Finally, the paper is concluded in Section 7.

2 Related Work

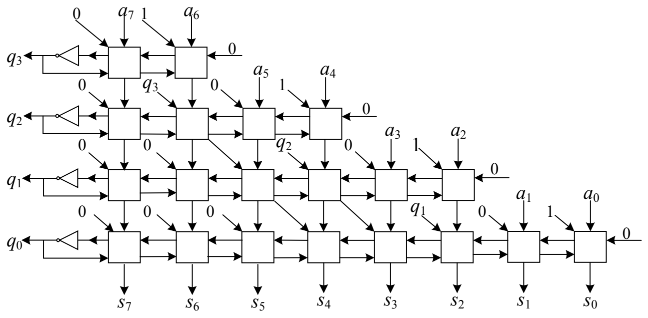
An unsigned integer division A/B can be formulated as $A = BQ + R$, where Q is the quotient, and R is the remainder. In the pencil-and-paper algorithm, the quotient of a $2n/n$ binary division is calculated by finding the final remainder, i.e., iteratively subtracting the product of a bit of the quotient and the n-bit divisor (B) from the $2n$-bit dividend (A). Starting with the most significant bit (MSB), each iteration generates one bit of the quotient. Thus, i bits of the quotient (referred to as the partial quotient) are generated after i iterations, denoted as $Q_i = \sum_{j=1}^{i} q_{n-j}2^{n-j}$, where n is the width of the quotient (due to the input range constraint [1]) and q_i is the i^{th} least significant bit (LSB) of Q, $i = 1, 2, \cdots, n$. The partial remainder of the i^{th} iteration is then given by $R_i = A - BQ_i$. In the $(i+1)^{th}$ iteration, the partial remainder $R_{i+1} = A - BQ_{i+1}$, where $Q_{i+1} = \sum_{j=1}^{i+1} q_{n-j}2^{n-j} = Q_i + q_{n-i-1}2^{n-i-1}$. Therefore, $R_{i+1} = A - BQ_i - Bq_{n-i-1}2^{n-i-1} = R_i - Bq_{n-i-1}2^{n-i-1}$. q_{n-i-1} is the bit of the quotient generated by the $(i+1)^{th}$ iteration. It is 1 (and $R_{i+1} = R_i - B2^{n-i-1}$) if $R_i - B2^{n-i-1} \geq 0$; otherwise q_{n-i-1} =0 (and $R_{i+1} = R_i$). In hardware, q_{n-i-1} is obtained by inverting the borrow output of $R_i - B2^{n-i-1}$ that is implemented by a shift subtration.

Similarly, an integer SQR $\sqrt{A}$ can be formulated as $A = Q^2 + R$, where Q is the SQR result, and R is the remainder. In the pencil-and-paper algorithm for binary SQR, the $2n$-bit radicand (A) is first divided into n groups of two, and two bits of the radicand are processed in each iteration to generate one bit in the SQR. The SQR is calculated starting with the MSB. Assume that the partial SQR after i iterations is $Q_i = \sum_{j=1}^{i} q_{n-j}2^{n-j}$, where n is the width of Q and q_i is the i^{th} LSB of Q, $i = 1, 2, \cdots, n$. The partial remainder of the i^{th} iteration is given by $R_i = A - Q_i^2$, while it is $R_{i+1} = A - Q_{i+1}^2 = A - (Q_i + q_{n-i-1}2^{n-i-1})^2 = R_i - (2Q_i + q_{n-i-1}2^{n-i-1})q_{n-i-1}2^{n-i-1}$ for the $(i+1)^{th}$ iteration. In the binary format, $2Q_i + q_{n-i-1}2^{n-i-1} = (\sum_{j=1}^{i} q_{n-j}2^{i-j+2} + q_{n-i-1})2^{n-i-1}$ is obtained by left shifting $(\sum_{j=1}^{i} q_{n-j}2^{i-j+2} + q_{n-i-1}) = (q_{n-1}q_{n-2}\cdots q_{n-i}0q_{n-i-1})_2$ for $(n-i-1)$ bits. Thus, q_{n-i-1} is determined by the sign of $R_i - (\sum_{j=1}^{i} q_{n-j}2^{i-j+2} + 1)2^{2(n-i-1)}$. That is, q_{n-i-1} =1 (and $R_{i+1} = R_i - (\sum_{j=1}^{i} q_{n-j}2^{i-j+2} + 1)2^{2(n-i-1)}$) if $R_i - (\sum_{j=1}^{i} q_{n-j}2^{i-j+2} + 1)2^{2(n-i-1)} \geq 0$; otherwise q_{n-i-1} =0 (and $R_{i+1} = R_i$).

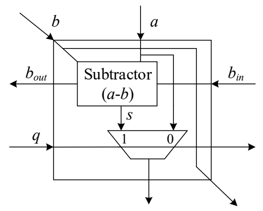
Following the pencil-and-paper algorithms, the division and SQR can be implemented in sequential and combinational circuits. Sequential division/SQR is usually implemented by using the digit recurrent algorithm [9] or the functional iterative algorithm [10]. Its latency is significantly longer than a combinational divider/SQR circuit due to the recurrent/iterative nature of the operation. A combinational division/SQR is implemented by shift and subtraction operations. An 8/4 unsigned restoring array divider and an 8-bit restoring array SQR circuit are shown in Fig. 1. In Fig. 1(a), an OR gate with an inverted input is used to generate one bit of the quotient that is the inverted borrow output from the left-most subtractor (when the currently most significant subtrahend bit is 0). In general, n^2 subtractor cells are required in a $2n/n$ array divider and $n^2 + n$ subtractor cells are needed for a $2n$-bit SQR circuit. Due to the borrow ripples in each row, the critical paths for the array divider and SQR circuit are in $O(n^2)$, while it is in $O(n)$ for an $n \times n$ array multiplier. Thus, an array divider/SQR circuit incurs a higher hardware consumption and a lower speed than an array multiplier.



(a) An 8/4 unsigned restoring array divider



(b) An 8-bit restoring array square root circuit



(c) Subtractor cell

Fig. 1. (a) An 8/4 unsigned restoring array divider and (b) 8-bit restoring array square root circuit with (c) the constituent subtractor cell [1].

Denoted as AXDr, the approximate dividers in [11] and [12] use inexact subtractors to replace the accurate ones at the less significant positions in an array divider. AXDr produces very small errors because only its less significant part is approximated. For the same reason, the critical path delay of this design is not significantly reduced. Furthermore, the improvements in power dissipation and area are relatively small. On the other hand, a dynamic approximate divider (DAXD) [13] shows a substantial improvement in speed, area and power consumption compared with AXDr. However, the accuracy of DAXD is much lower due to the overflow problem caused by truncation. In addition to array dividers, a rounding-based approximate division is implemented by a rounding block, a look-up table, a reduced-width multiplier, adders and a shifter, referred to as SEERAD [14]. Without using a traditional division structure, SEERAD is fast, but it incurs a substantial power dissipation and a large area due to the use of the look-up table.

Compared to the divider, design effort has been made on the recurrent addition/subtraction algorithm for an SQR operation [15]. However, the study of an approximate SQR circuit has not been found in the literature.

3 Proposed Approximate Divider/SQR Circuit

3.1 Motivation

For approximations in an adder or a multiplier, truncation is an efficient approach to reducing hardware and energy consumption [7], [16]. Improvements in power dissipation and critical path delay can also be obtained for an approximate divider/SQR circuit design; however, the approximation using static truncation on the LSBs of the input operands results in large relative errors, especially for small input operands. Thus, adaptive approximation is investigated in this paper by selectively pruning some insignificant bits of the input operands; then, a reduced-width divider/SQR circuit is used to process the remaining bits.

3.2 Approximate Divider

Different from multiplication and addition, the inputs for division have a strict range requirement. In a $2n/n$ divider, the n MSBs of the dividend A must be smaller than the divisor B to guarantee that no overflow occurs [1].

3.2.1 Design

The basic structure of the proposed approximate unsigned divider is shown in Fig. 2. In this design, $2k$ (or k) MSBs of the dividend (or divisor) are adaptively chosen from the $2n$ (or n)-bit input using leading one position detectors (LOPDs) and pruning circuits (here, $k < n$), according to the pruning schemes. An exact $2(k+1)/(k+1)$ divider is then used to compute the division of the selected bits. The $(k+1)$-bit quotient is shifted by a shifter for a number of bits calculated by a subtractor, which results in an $(n+1)$-bit intermediate result. Finally, the n-bit approximate quotient is obtained by correcting the $(n+1)$-bit intermediate result using an error correction circuit. The detailed structure of each circuit in Fig. 2 is discussed next.

3.2.2 Input Pruning

Fig. 3 shows a straightforward pruning scheme for a $2n$-bit unsigned dividend $A = \sum_{i=0}^{2n-1} a_i 2^i = (a_{2n-1}a_{2n-2}\cdots a_1 a_0)_2$. To obtain a $2k$-bit dividend, '0's at the bit positions higher than the most significant '1' are truncated; the redundant LSBs are pruned if the number of remaining LSBs is larger than $2k$. Similarly, a k-bit number is determined from the n-bit divisor $B = \sum_{i=0}^{n-1} b_i 2^i = (b_{n-1}b_{n-2}\cdots b_1 b_0)_2$.

Let the bit positions of the most significant '1', known as the leading '1' positions, for A and B be l_A and l_B, respectively. The input operands of a division can be determined as in Fig. 3 when l_A and l_B are larger than or equal to $2k-1$ and $k-1$, respectively. A different pruning scheme is required for the input operands

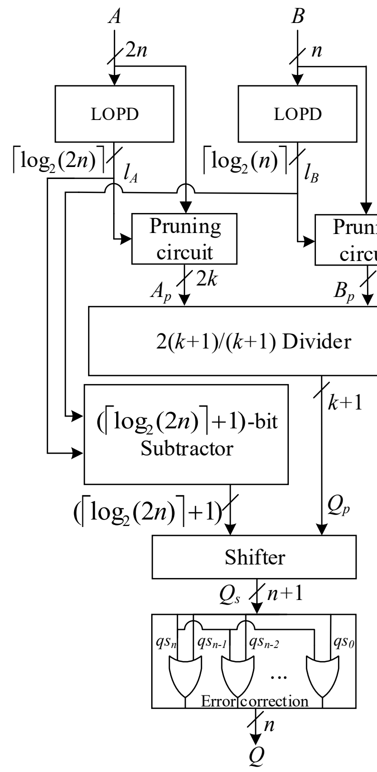


Fig. 2. The proposed adaptively approximate divider (AAXD). LOPD: leading one position detector.

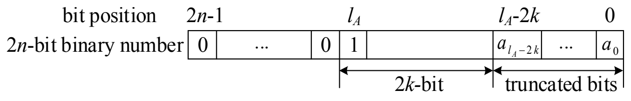


Fig. 3. Pruning scheme for a $2n$-bit unsigned number A when $l_A \geq 2k-1$ [13].

when $l_A < 2k-1$ or $l_B < k-1$, as shown in Fig. 4. The details of the pruning schemes are discussed in four scenarios for different values of l_A and l_B [8], as summarized as follows.

When $l_A \geq 2k-1$ and $l_B \geq k-1$, the pruning scheme in Fig. 3 is used. The pruned dividend is given by $A_p = (1a_{l_A-1}\cdots a_{l_A-2k+1})_2 = 2^{2k-1} + \sum_{i=l_A-2k+1}^{l_A-1} a_i 2^{i-(l_A-2k+1)}$, and the pruned divisor is $B_p = (1b_{l_B-1}\cdots b_{l_B-k+1})_2 = 2^{k-1} + \sum_{i=l_B-k+1}^{l_B-1} b_i 2^{i-(l_B-k+1)}$. To avoid overflow for the largest possible quotient of A_p/B_p, a $2(k+1)/(k+1)$ divider is used to compute the division. The $2k$-bit pruned dividend is expanded to $(2k+2)$-bit by adding two '0's at the most significant two bit positions; a '0' is added to the $(k+1)^{th}$ bit position of the pruned divisor. No overflow occurs because $(001a_{l_A-1}\cdots a_{l_A-k+2})_2$ is always smaller than $(01b_{l_B-1}\cdots b_{l_B-k+1})_2$.

The inputs A and B are approximated as $A_p 2^{l_A-2k+1}$ and $B_p 2^{l_B-k+1}$ after pruning, respectively. Thus, A/B is approximately given by

$$\frac{A}{B} \approx \frac{2^{l_A-2k+1}A_p}{2^{l_B-k+1}B_p} = \lfloor \frac{A_p}{B_p} \rfloor 2^{l_A-l_B-k}. \quad (1)$$

The result of (1) is the quotient obtained by the $2(k+1)/(k+1)$ divider multiplied by $2^{l_A-l_B-k}$. The multiplication is implemented by left shifting $\lfloor \frac{A_p}{B_p} \rfloor$ for $l_A - l_B - k$ bits.

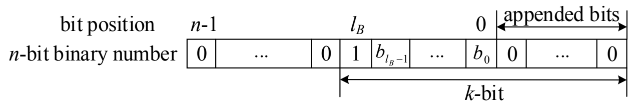


Fig. 4. Pruning scheme for an n-bit unsigned number B when $l_B < k-1$.

When $l_A \geq 2k-1$ and $l_B < k-1$, A_p is obtained using the pruning scheme in Fig. 3, and B_p is obtained using the scheme in Fig. 4, i.e., $A_p = (1a_{l_A-1}\cdots a_{l_A-2k+1})_2$, and $B_p = (1b_{k-1}\cdots b_0 0\cdots 0)_2$. Then, a $2(k+1)/(k+1)$ divider is used to compute A_p/B_p. The approximation of A/B is also given by (1).

Similarly, the pruning schemes in Figs. 4 and 3 are respectively used to prune A and B when $l_A < 2k-1$ and $l_B \geq k-1$. The same approximate division is then given by (1). When the dividend A is zero, l_A is set to zero, which has the same leading one position as number $(00\cdots 01)_2$. Because $A_P = (a_0 0\cdots 0)_2$ (i.e., a_0 is kept) when $l_A = 0$, the quotient is always correct no matter a_0 is '0' or '1'. When $l_A < 2k-1$ and $l_B < k-1$, both the input operands are pruned using the scheme in Fig. 4, and an accurate $2n/n$ division is performed by using a $2(k+1)/(k+1)$ divider.

3.2.3 Circuit

As shown in Fig. 2, a leading one position detector (LOPD) is used to detect the bit position of the most significant '1' in each input. It is implemented by a priority encoder.

The leading one positions (l_A and l_B) are then used to determine the $2k$-bit A_p and the k-bit B_p from the $2n$-bit dividend and the n-bit divisor, respectively. The pruning circuits are used to implement the pruning schemes in Figs. 3 and 4; they select A_p and B_p starting from the most significant '1'. A_p and B_p are then processed by using an exact $2(k+1)/(k+1)$ divider. Note that the structure of the $2(k+1)/(k+1)$ divider can be different according to specific application requirements, e.g., an array divider, a sequential divider or a high-radix divider. Meanwhile, the shifting direction and number of bits are computed by subtracting the two leading one positions using a $(\lceil log_2(2n)\rceil+1)$-bit subtractor. Subsequently, a shifter is used to shift the $(k+1)$-bit output of the reduced-width divider for $|l_A-l_B-k|$ bits (left shifting if $l_A-l_B-k>0$, and right shifting if $l_A-l_B-k<0$), which results in $(n+1)$-bit intermediate result Q_s. Finally, the error correction circuit uses n OR gates to perform $q_i = qs_i \vee qs_n$, $i = 0, 1, \cdots, n-1$, where q_i and qs_i are the i^{th} LSBs of Q and Q_s, respectively. This circuit corrects the erroneous results that are larger than 2^n-1 (when $qs_n = 1$) to 2^n-1 ($qs_i = 1$ for $i = 0, 1, \cdots, n-1$), which ensures that an n-bit approximate quotient is obtained.

The most significant circuit of the proposed approximate divider is the $2(k+1)/(k+1)$ divider, whereas other components (LOPD, pruning circuit, subtractor and shifter) are relatively small. Moreover, the subtractor works in parallel with the $2(k+1)/(k+1)$ divider. Thus, the circuit complexity and critical path of the approximate divider are close to $O((k+1)^2)$ when a $2(k+1)/(k+1)$ array divider is used. This is significantly smaller compared with that of the exact array divider ($O(n^2)$), especially for a small k.

3.3 Approximate SQR Circuit

As shown in Fig. 1, an SQR circuit is implemented by shifts and subtractions. We propose an approximate SQR circuit (AXSR) by replacing the exact subtractors in the lower k bit positions with the approximate subtractor cells in [11]. The adaptive approximation strategy is also applicable to the design of SQR circuit (AASR), as shown in Fig. 5.

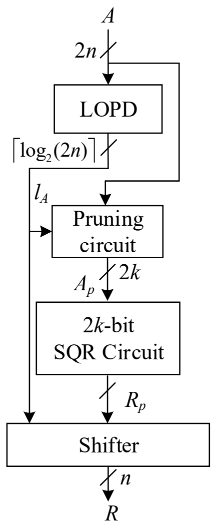


Fig. 5. The proposed adaptively approximate SQR circuit (AASR).

Using the same pruning schemes as in Figs. 3 and 4, the radicand A of a $2n$-bit SQR circuit can be approximated by a scaled $2k$-bit number A_p, i.e.,

$$A \approx A_p 2^{l_A-2k+1}, \quad (2)$$

where $k < n$, and l_A indicates the leading one position of A. Thus, the SQR of A is approximately generated by a $2k$-bit SQR circuit due to

$$\sqrt{A} \approx \lfloor\sqrt{A_p}\rfloor 2^{\frac{l_A-2k+1}{2}}. \quad (3)$$

As $\frac{l_A-2k+1}{2}$ is a fractional number when l_A-2k+1 is odd, $2^{\frac{l_A-2k+1}{2}}$ cannot be computed by a shift operation. To use shifting, l_A-2k+1 must be an even number. Therefore, l_A is limited to odd numbers in this design. This is ensured by setting an even l_A to l_A+1. In the circuit design, it is implemented by setting the LSB of l_A to '1' (not shown in Fig. 5). Note that the $2k$-bit SQR circuit can be any existing combinational or sequential SQR circuit.

As shown in Fig. 5, no adder/subtractor is required in the approximate SQR circuit. Hence, the hardware overhead of the auxiliary circuits is lower than that in the approximate divider. Moreover, (n^2+n-k^2-k) subtractor cells are saved in the approximate SQR circuit, a larger saving by $(n+k)$ cells compared to the approximate divider with a saving of $(n^2-(k+1)^2)$ cells. Therefore, the proposed adaptive approximation strategy is more efficient for an SQR circuit design.

4 Error Analysis

4.1 Approximate Divider

The error of the approximate divider is measured by comparing the approximate results with the accurate quotient in integer as the divider is designed for unsigned integers. As the quotient of the approximate unsigned divider is given by (1), the incurred error is

$$E = \lfloor\frac{A_p}{B_p}\rfloor 2^{l_A-l_B-k} - \lfloor\frac{A}{B}\rfloor, \quad (4)$$

where the dividend $A = A_p 2^{l_A-2k+1} + A_L$, and the divisor $B = B_p 2^{l_B-k+1} + B_L$. $A_L = \sum_{i=0}^{l_A-2k} a_i 2^i$ and $B_L = \sum_{i=0}^{l_B-k} b_i 2^i$ denote the truncated LSBs in A and B, respectively. $A_L = 0$ when $l_A < 2k$, and $B_L = 0$ when $l_B < k$. The truncated LSBs represented by A_L and B_L determine the error E to be positive or negative. To evaluate the proposed design, the upper bound of the ED (i.e., the absolute value for the difference between the approximate and accurate results) is analyzed for the worst case scenario. To this end, the positive and negative errors are respectively discussed in the following.

Expanding (4) and neglecting the floor operation, the error can be simplified to

$$E_{nr} = \frac{A_p B_L 2^{l_A-l_B-k} - A_L B_p}{(B_p 2^{l_B-k+1} + B_L) B_p}. \tag{5}$$

As A_p, B_p, A_L and B_L are positive, (5) indicates that one condition for generating the largest positive error is $A_L = (00\cdots0)_2 = 0$. Then, (5) becomes

$$E_{nr} = \frac{A_p 2^{l_A-l_B-k}}{(B_p/B_L 2^{l_B-k+1} + 1) B_p}. \tag{6}$$

Thus, the largest positive error is produced when B_L takes the largest possible value (i.e., $B_L = (11\cdots1)_2 = 2^{l_B-k+1} - 1$). The largest positive error is then given by

$$\begin{aligned} E_{pmax} &= \lfloor \frac{A_p}{B_p} \rfloor 2^{l_A-l_B-k} - \lfloor \frac{A_p 2^{l_A-2k+1}}{B_p 2^{l_B-k+1} + 2^{l_B-k+1} - 1} \rfloor \\ &= \lfloor \frac{A_p}{B_p} \rfloor 2^{l_A-l_B-k} - \lfloor \frac{\frac{A_p}{B_p} 2^{l_A-l_B-k}}{1 + \frac{1-2^{k-l_B-1}}{B_p}} \rfloor \end{aligned}. \tag{7}$$

As $\lfloor \frac{A_p}{B_p} \rfloor \le \frac{A_p}{B_p}$, $\lfloor \frac{A_p}{B_p} \rfloor = \frac{A_p}{B_p}$ ensures the positive error to be the largest. Let $\lfloor \frac{A_p}{B_p} \rfloor = \frac{A_p}{B_p} = C_q$ (C_q is a positive integer), substitute $\frac{A_p}{B_p}$ and $\lfloor \frac{A_p}{B_p} \rfloor$ by C_q, (7) becomes

$$\begin{aligned} E_{pmax} &= C_q 2^{l_A-l_B-k} - \lfloor \frac{C_q 2^{l_A-l_B-k}}{1 + \frac{1-2^{k-l_B-1}}{B_p}} \rfloor \\ &\le \lceil C_q 2^{l_A-l_B-k} - \frac{C_q 2^{l_A-l_B-k}}{1 + \frac{1-2^{k-l_B-1}}{B_p}} \rceil \\ &= \lceil \frac{C_q 2^{l_A-l_B-k}}{1 + \frac{B_p}{1-2^{k-l_B-1}}} \rceil. \end{aligned} \tag{8}$$

As per (1), $C_q 2^{l_A-l_B-k}$ is the output of the approximate divider; thus, $C_q 2^{l_A-l_B-k} \le 2^n - 1$ that is restricted by the error correction unit (discussed in Section 3.2). To reach the maximal positive error, $C_q 2^{l_A-l_B-k}$ and l_B should be their largest possible values, and B_p should be the smallest. Therefore, $C_q 2^{l_A-l_B-k} = 2^n - 1$, $l_B = n-1$ and $B_p = (10\cdots0)_2 = 2^{k-1}$. The upper bound of the positive error for the approximate divider is given by

$$E_{pmax} \le \lceil \frac{(2^n-1)(2^{n-k}-1)}{2^{n-1} + 2^{n-k} - 1} \rceil. \tag{9}$$

Also, the smallest negative error is determined to show the maximal negative deviation of the approximate result. As per (5), the smallest negative error occurs when $B_L = (00\cdots0)_2 = 0$ and $A_L = (11\cdots1)_2 = 2^{l_A-2k+1} - 1$. Then, (4) becomes

$$E_{nmin} = \lfloor \frac{A_p}{B_p} \rfloor 2^{l_A-l_B-k} - \lfloor \frac{A_p 2^{l_A-2k+1} + 2^{l_A-2k+1} - 1}{B_p 2^{l_B-k+1}} \rfloor. \tag{10}$$

The absolute value of E_{nmin} is given by

$$|E_{nmin}| = \lfloor \frac{A_p}{B_p} 2^{l_A-l_B-k} + \frac{2^{l_A-l_B-k}}{B_p} - \frac{2^{k-l_B-1}}{B_p} \rfloor - \lfloor \frac{A_p}{B_p} \rfloor 2^{l_A-l_B-k}. \tag{11}$$

Assume $\lfloor \frac{A_p}{B_p} \rfloor = C_q$ (C_q is a positive integer), then A_p must be in the range of $[B_p C_q, B_p C_q + B_p - 1]$. Thus, the largest difference between $\frac{A_p}{B_p}$ and $\lfloor \frac{A_p}{B_p} \rfloor$ occurs when $Ap = B_p C_q + B_p - 1$. Then, (11) becomes

$$|E_{nmin}| = \lfloor 2^{l_A-l_B-k} - \frac{2^{k-l_B-1}}{B_p} \rfloor. \tag{12}$$

To reach the maximal value for $|E_{nmin}|$, $l_A - l_B$ must be n (because $l_A - l_B \le n$), and $\frac{2^{k-l_B-1}}{B_p}$ must be the smallest (i.e., $B_p = 2^k - 1$ and $l_B = n-1$ because $B_p \le 2^k - 1$ and $l_B \le n-1$). Thus, $\frac{2^{k-l_B-1}}{B_p} = \frac{1}{2^n - 2^{n-k}} < 1$ (for $n > k$), and we obtain

$$|E_{nmin}| = 2^{n-k} - 1. \tag{13}$$

As the right hand side of (9) is always larger than $2^{n-k} - 1$, the ED of the proposed approximate unsigned divider is s-maller than or equal to $\lceil \frac{(2^n-1)(2^{n-k}-1)}{2^{n-1}+2^{n-k}-1} \rceil$. As $\lceil \frac{(2^n-1)(2^{n-k}-1)}{2^{n-1}+2^{n-k}-1} \rceil = \lceil \frac{(2^{n-k}-1+2^{-n}-2^{-k})}{2^{-1}+2^{-k}-2^{-n}} \rceil \le 2^{n-k+1} - 2$, a looser upper bound of the ED for the approximate divider is given by

$$ED_{LUB} = 2^{n-k+1} - 2, \tag{14}$$

where the larger n and k are, the closer (14) to (9).

4.2 Approximate SQR Circuit

As the approximate SQR of the radicand A is computed by (3), the error of the approximate $2n$-bit SQR circuit using a $2k$-bit exact SQR circuit is given by

$$\begin{aligned} E &= \lfloor \sqrt{A_p} \rfloor 2^{\frac{l_A-2k+1}{2}} - \lfloor \sqrt{A} \rfloor \\ &= \lfloor \sqrt{A_p} \rfloor 2^{\frac{l_A-2k+1}{2}} - \lfloor \sqrt{A_p 2^{l_A-2k+1} + A_L} \rfloor \end{aligned}, \tag{15}$$

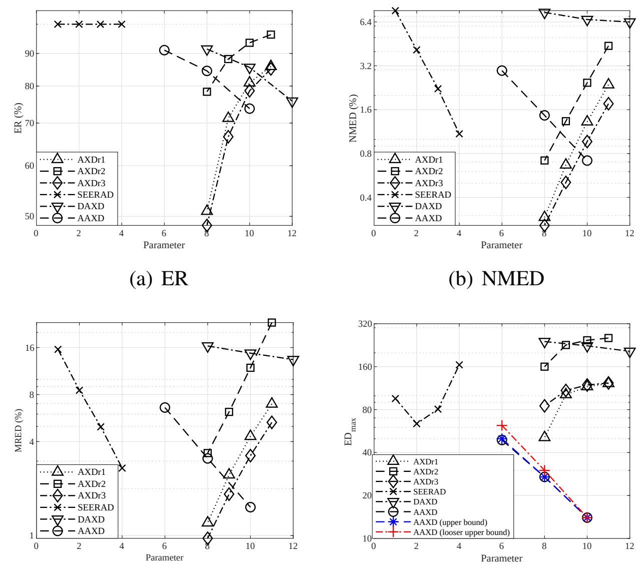
where A_p is the $2k$-bit pruned radicand, and $A_L = \sum_{i=0}^{l_A-2k} a_i 2^i$ denotes the truncated LSBs. (15) shows that the error of the SQR circuit is always smaller than or equal to zero because $A_L \ge 0$. The smallest negative error occurs when $A_L = (11\cdots1)_2 = 2^{l_A-2k+1} - 1$. Assuming a positive integer $R_p = \lfloor \sqrt{A_p} \rfloor$, then $R_p^2 \le A_p \le (R_p+1)^2 - 1$. Thus, the $2k$-bit SQR circuit generates the largest ED when $A_p = (R_p+1)^2 - 1$. Then, the largest ED of the proposed SQR circuit is given by

$$\begin{aligned} ED_{UB} &= \lfloor \sqrt{[(R_p+1)^2 - 1] 2^{l_A-2k+1} + 2^{l_A-2k+1} - 1} \rfloor \\ &\quad - R_p 2^{\frac{l_A-2k+1}{2}} \\ &= \lfloor \sqrt{(R_p+1)^2 2^{l_A-2k+1} - 1} \rfloor - R_p 2^{\frac{l_A-2k+1}{2}} \\ &= (R_p+1) 2^{\frac{l_A-2k+1}{2}} - 1 - R_p 2^{\frac{l_A-2k+1}{2}} \\ &= 2^{\frac{l_A-2k+1}{2}} - 1 \end{aligned}. \tag{16}$$

As the largest possible value of l_A is $2n-1$, the ED_{UB} is given by

$$ED_{UB} = 2^{n-k} - 1. \tag{17}$$

This result implies that the maximum ED occurs when the $n-k$ LSBs in the accurate SQR are all '1's but they are ignored due to the use of a $2k$-bit SQR circuit and resulting k-bit approximate result.



(a) ER (b) NMED

(c) MRED (d) ED_{max}

Fig. 6. Error characteristics of approximate 16/8 dividers. *Note*: The parameter value indicates the replacement depth and the accuracy level for AXDrs and SEERAD, respectively. It is the bit width of the pruned dividend in DAXD and AAXD.

5 Simulation Results

To assess the error and circuit characteristics, the proposed approximate divider and SQR circuits are implemented in MATLAB and VHDL. The other approximate dividers, AXDr, DAXD and SEERAD, are also considered for comparison. Note that the error and circuit characteristics of the considered dividers and SQR circuits are measured for a commonly used width ($2n = 16$) in the simulation. However, n can take a larger value as required by the application.

5.1 Error Characteristics

The error rate (ER), normalized mean error distance (NMED), mean relative error distance (MRED) and the maximum error distance (ED_{max}) are considered to evaluate the accuracy of 16/8 approximate dividers and 16-bit approximate SQR circuits. The NMED is defined as the mean value of the error distances normalized by the maximum possible accurate output. The MRED is the mean value of the relative error distance that is the ratio between the ED and the accurate output.

5.1.1 Approximate Divider

All valid combinations in the range of $[0, 65535]$ and $(0, 255]$ are considered as the input dividends and divisors. They are carefully selected to meet the no overflow condition of an accurate 16/8 divider. The simulation results are shown in Fig. 6, in which AXDr1, AXDr2, and AXDr3 are the approximate restoring array dividers with triangle replacement using approximate subtractor 1, 2, and 3, respectively [12].

Fig. 6 shows that the proposed AAXD has a similar ER as DAXD. The NMED and MRED of AAXD are in the medium domain among the considered designs. AAXD has the smallest ED_{max} that is very close to the upper bound analyzed in Section 4.1. The looser upper bound of ED for AAXD is closer to the ED_{max} for a larger parameter. For the SEERAD, only SEERAD-3 (for accuracy level 3) and SEERAD-4 (for accuracy level 4) have an accuracy that is comparable with the proposed design. AXDr1 and AXDr3 show very small values of ER, NMED and MRED; however, their circuit measurements are relatively high, as shown next. The accuracy of DAXD is lower than other designs due to the possible overflow.

TABLE 1. Error characteristics of the approximate 16-bit SQR circuits.

SQR	Parameter	ER (%)	NMED (%)	MRED (%)	ED_{max} (simulation)	ED_{UB} (analysis)
AXSR3	14	74.52	3.17	5.78	47	–
AXSR3	13	71.77	1.49	2.89	24	–
AXSR3	12	66.54	1.03	2.29	24	–
AXSR3	11	57.34	0.49	1.13	12	–
AASR	6	95.71	5.33	7.98	31	31
AASR	8	91.14	2.53	3.80	15	15
AASR	10	82.30	1.16	1.72	7	7
AASR	12	65.82	0.48	0.69	3	3

Note: The parameter value for AXSR3 indicates the replacement depth. It is the bit-width of the pruned radicand in AASR.

5.1.2 Approximate SQR Circuit

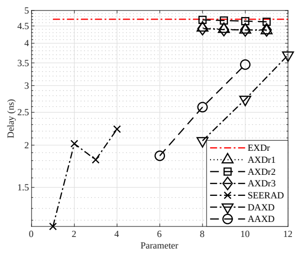
Among the approximate dividers using approximate subtractors, AXDr3 is the most accurate with the smallest ER, NMED and MRED (Fig. 6). Also, the circuit of AXDr3 is the smallest among AXDrs (shown later). This indicates that the approximate subtractor cell 3 in [12] is very efficient in the divider design. Thus, the approximate subtractor cell 3 is used in the approximate SQR circuit design, which is denoted as AXSR3.

Similarly, all unsigned numbers in $[0, 65535]$ are considered as inputs to measure the accuracy of the adaptively approximate 16-bit SQR circuit and AXSR3. The simulation results in Table 1 show that AXSR3 has a smaller ER than AASR, whereas its ED_{max} is much larger. The ED_{max} of AASR approximately decreases by half with a 2-bit increase in the bit width of the pruned radicand, which is consistent with the error analysis result in (17). For using a 12-bit SQR circuit, the small ED_{max} indicates that the computed results are very close to the accurate ones. AXSR3-11 has a similar NMED as AASR-12, but its MRED and ED_{max} are much larger. Compared to the proposed approximate divider, AASR has smaller values of ED_{max} but larger ERs and NMEDs, for the same value of k.

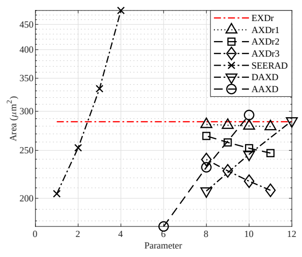
5.2 Circuit Characteristics

5.2.1 Approximate Divider

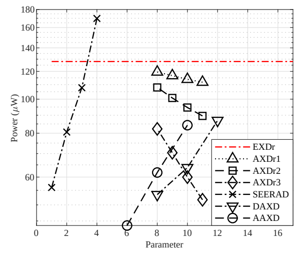
To obtain the circuit measurements, the approximate 16/8 dividers and the exact unsigned restoring array divider (EXDr) are implemented in VHDL and synthesized in ST's 28 nm CMOS process using the Synopsys Design Compiler with the same voltage, temperature and frequency. The supply voltage is 1 V, the simulation temperature is 25°C, and the frequency used for power estimation is 200 MHz. The critical path delay and area are reported by the Synopsys Design Compiler. The power dissipation is estimated by using the PrimeTime-PX tool for 5 million random input combinations. For ease of comparison, the same array structure and subtractor cells are used in the accurate part of AXDrs, DAXD and AAXD. To be consistent with the other designs, AXDr1, AXDr2 and AXDr3 are implemented at the gate level rather than



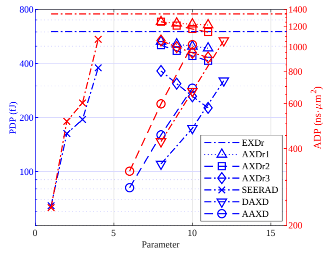
(a) Delay



(b) Area



(c) Power



(d) PDP (in blue) and ADP (in red)

Fig. 7. Circuit measurements of the 16/8 dividers. *Note*: EXDr refers to the exact array divider, and the X-axis is not applicable for it.

at the transistor level in [12]. As additional figures of merit, the power-delay product (PDP) and area-delay product (ADP) are calculated from the measured values. The results are shown in Fig. 7.

With a 6-bit pruned dividend, the proposed 16/8 AAXD achieves reductions in delay, area and power dissipation by 60.51%, 38.63% and 65.88% respectively, compared to EXDr. For the AAXD using a 12/6 exact divider (with a 10-bit pruned divided), it is 26.54% faster and consumes 34.13% less power than EXDr, albeit with a slightly larger area. As a result, the PDP and ADP of the proposed design are decreased by 51.61%-86.53%, and 24.18%-75.76%, respectively.

Among the approximate designs, SEERAD is the fastest; however, the power dissipation and area are relatively large for SEERAD-3 and SEERAD-4. With the lowest accuracy, DAXD have slightly higher values of all circuit measurements than AAXD, using a same sized accurate divider. Compared with AXDrs, AAXD outperforms AXDr1 and AXDr2 in delay, area and power dissipation. Also, it shows a shorter delay and a similar power dissipation and therefore, smaller values of PDP and ADP (except for AAXD-10) compared with AXDr3.

5.2.2 Approximate SQR Circuit

Other than the array structure shown in Fig. 1(b), a small SQR circuit can be efficiently implemented by LUTs. In a LUT-based circuit, the SQR of an input is obtained by reading a LUT that stores the precomputed SQR results. Thus, the critical path delay is very small. For a $2n$-bit integer SQR circuit, the size of the required LUT is 2^{2n}, where each input corresponds to a LUT cell. The size of the LUT will be very large for a large n; thus, interpolation is usually used for a large SQR circuit to reduce the size of the LUT. As the proposed approximation scheme uses a reduced-width exact SQR circuit ($2k$-bit) for a $2n$-bit SQR, the hardware of the LUT-based design is significantly reduced. In our simulation, both the array and LUT-based SQR circuits are

TABLE 2. Circuit measurements of the exact and approximate 16-bit SQR circuits.

SQR	Parameter	Delay (ns)	Area (μm^2)	Power (μW)	PDP (fJ)	ADP ($ns \cdot \mu m^2$)
ESRr_A	–	**4.32**	**222.4**	**93.82**	**405.30**	**961.0**
AXSR3	14	4.04	168.3	53.31	215.37	679.8
AXSR3	13	4.07	175.4	58.57	238.38	714.0
AXSR3	12	4.09	182.8	62.58	255.95	747.6
AXSR3	11	4.13	189.5	67.34	278.11	782.5
AASR_A	6	1.10	76.5	19.38	21.32	84.2
AASR_A	8	1.70	114.7	31.72	53.92	195.0
AASR_A	10	2.43	154.4	46.55	113.12	375.2
AASR_A	12	3.34	200.4	64.97	217.00	669.4
ESRr_T	–	**0.58**	**613.6**	**189.0**	**109.62**	**355.9**
AASR_T	6	0.67	60.38	13.52	9.06	40.46
AASR_T	8	0.81	87.31	20.79	16.84	70.72
AASR_T	10	1.01	123.2	32.01	32.33	124.4
AASR_T	12	1.31	217.4	69.59	91.16	284.8

considered for comparison. Using the same synthesis tool and technology library as for the dividers, the 16-bit exact restoring array and LUT-based SQR circuits (referred to as ESRr_A and ESRr_T, respectively), AXSR3, AASR_A (for which the $2k$-bit SQR circuit is implemented by an array structure) and AASR_T (for which the $2k$-bit SQR circuit is based on LUT without using interpolation) are synthesized at a frequency of 200 MHz.

Table 2 reports the circuit measurements of the considered designs. Among the array-based designs, AASR_A has a higher performance and consumes significantly smaller area and power than the accurate design. Specifically, AASR_A with a 6-bit pruned radicand is 74.54% faster and saves 65.60% in area and 79.34% in power compared with the accurate design. Accordingly, its improvements in PDP and ADP are 94.74% and 91.24%, respectively. For the design using a 12-bit exact array SQR circuit, it achieves 46.46% reduction in PDP and 30.34% reduction in ADP. Compared to the approximate divider, the approximate SQR circuit achieves more significant improvements in power dissipation and area because no additional subtractor is used. AXSR3 is slightly faster than ESRr_A. AXSR3-14 saves up to 24% in area and 47% in power dissipation compared to ESRr_A. AASR_A outperforms AXSR3 in all the circuit measurements except for $2k = 12$.

Compared with ESRr_A, ESRr_T is significantly faster, but it consumes 2.8× area and 2× power. Although AASR_T has a larger delay than ESRr_T due to the auxiliary circuits, AASR_T achieves 63% to 92% reductions in area and power dissipation when $2k$ varies from 12 to 6. Therefore, the savings in PDP and ADP are 17%-92% and 20%-89%, respectively. For the same k, AASR_T shows smaller values of critical path delay, area and power dissipation than AASR_A when $2k$ is less than 12. As a result, the AASR_T is more efficient than AASR_A at small values of k.

5.3 Discussion

5.3.1 Combinational Designs

For a further comparison of approximate dividers and SQR circuits, the error and circuit measures are jointly considered. The metrics MRED and PDP are selected as representatives to show the error and circuit characteristics. As shown in Fig. 8(a), the proposed AAXD has a much smaller value of MRED than the other approximate designs when a similar PDP is considered.

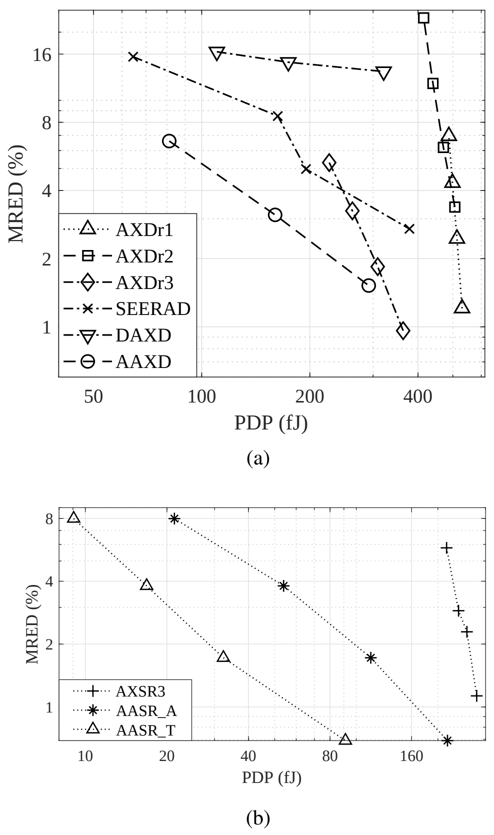


Fig. 8. A comparison of (a) approximate dividers and (b) SQR circuits in PDP and MRED. The replacement depths of AXDr1, AXDr2 and AXDr3 are from 8 to 11 from right to left. The accuracy levels of SEERAD are from 1 to 4 from left to right. The pruned dividend width is from 8 to 12 for DAXD, and it is from 6 to 10 for AAXD from left to right. The replacement depth for AXSR3 is from 11 to 14 from right to left. The pruned radicand widths for AASR_A and AASR_T are from 6 to 12 from left to right.

AXDr3 also shows a good tradeoff in MRED and PDP with a higher accuracy, however its delay is very long. Although some configurations of AXDr1 and AXDr2 show small MREDs, their PDPs are generally high. On the contrary, DAXD has a very low PDP but a significantly large MRED. The MRED and PDP are moderate for SEERAD, and they vary with the accuracy level. Overall, the proposed AAXD shows the best tradeoff among the considered approximate dividers.

As for the approximate SQR circuits (Fig. 8(b)), AASR_T is the most efficient design in terms of MRED and PDP. It requires a significantly lower PDP than AASR_A and AXSR3 for achieving a similar MRED. For the array-based designs, AASR_A is more efficient than AASR3.

5.3.2 *Sequential Designs*

As shown in Figs. 2 and 5, the main module of the proposed approximate divider/SQR circuit is the reduced-width divider/SQR circuit that can be implemented by an existing structure. Thus, an approximate design is built by adding the auxiliary circuits to a traditional divider/SQR circuit. In this section, the sequential divider and SQR circuits are further assessed; they are implemented by using a multibit subtractor, a shifter and a control unit based on the pencil-and-paper algorithms introduced in Section 2. The synthesis results are shown in Table 3, in which the same tools, technology and clock frequency (200 MHz) are used as for the dividers in Section 5.2.

TABLE 3. Circuit measurements of 16/8 sequential dividers and 16-bit sequential SQR circuits.

Circuit	Design	$2k$	Delay (ns)	Area (μm^2)	Power (μW)	N	EPO (pJ /op)	TPA (ops $/(s \cdot \mu m^2)$)
Divider	**EXDr_S**	**–**	**1.57**	**543.8**	**94.1**	**10**	**4.71**	**11,713**
	AAXD_S	6	1.57	521.4	95.66	6	2.87	33,932
	AAXD_S	8	1.57	574.5	104.5	7	3.66	22,628
	AAXD_S	10	1.57	625.9	114.2	8	4.57	15,901
SQR	**ESRr_S**	**–**	**1.57**	**577.4**	**101.7**	**10**	**5.09**	**11,031**
	AASR_S	6	1.57	424.5	63.79	5	1.59	60,021
	AASR_S	8	1.57	476.9	72.66	6	2.18	37,102
	AASR_S	10	1.57	529.6	80.55	7	2.82	24,545
	AASR_S	12	1.57	579.5	87.85	8	3.51	17,173

For an exact $2n/n$ divider (or a $2n$-bit SQR circuit), the number of clock cycles required to complete one division (or SQR) operation is given by $N = n+2$, i.e., one cycle for preparation, n cycles for shifted subtraction, and one cycle for outputting the result. As an exact $2(k+1)/(k+1)$ divider (or a $2k$-bit SQR circuit) is used in an approximate design, the values of N for the approximate designs are reduced linearly with k. This indicates that, in the sequential design, the main benefit for the approximate designs is that fewer iterations are required to complete a division (or SQR) than the exact designs.

Instead of PDP and ADP, the energy per operation (EPO) and throughput per area (TPA) are considered to evaluate the sequential designs. The EPO is defined as the energy consumed to complete one operation, and the TPA is the number of operations that are completed per unit time per unit area. They are respectively given by

$$\text{EPO} = N \times T \times P, \tag{18}$$

and

$$\text{TPA} = 1/(N \times D \times S), \tag{19}$$

where T and N are the clock period and the required number of clock cycles to complete one operation, P is the total power dissipation including the dynamic and leakage powers, D is the critical path delay, and S is the area of the circuit.

In Table 3, EXDr_S and ESRr_S denote the exact sequential 16/8 divider and 16-bit SQR circuit, respectively. AAXD_S and AASR_S are the proposed approximate 16/8 divider and 16-bit SQR circuit, in which the reduced-width divider and SQR circuit are sequentially implemented. This table shows that the approximation does not change the critical path delay of the sequential 16/8 divider or 16-bit SQR circuit. This occurs because the approximation strategy only decreases the sizes of the subtractor and shifter in the sequential divider and SQR circuit; they are relatively small compared to the control unit and the auxiliary circuits used for the approximation. For the same reason, AAXD_S incurs a larger area and power dissipation than EXDr_S; the improvements in area and power dissipation for AASR_S are not as significant as the combinational designs. As the values of N of AAXD_S and AASR_S are reduced linearly with k, AAXD_S-6 achieves a reduction in EPO by 39.07% with 1.9× increase in

TABLE 4. PSNRs of change detection results (*dB*).

Image	AXDr1 -10	AXDr2 -10	AXDr3 -9	SEERAD -4	DAXD -12	AAXD -10
tools	32.14	18.39	39.27	36.61	23.56	40.16
canoe	31.66	24.07	36.23	39.20	23.42	45.03
fountain	33.32	26.95	39.49	41.60	24.59	45.79
pedestrians	35.59	25.73	40.12	43.50	24.76	47.07
office	31.19	23.13	33.29	39.29	19.60	45.54
average	**32.78**	**23.65**	**37.68**	**40.00**	**23.18**	**44.71**

TPA compared with EXDr_S. The improvements in EPO and TPA for AASR_S-6 are 68.76% and 4.4× compared to ESRr_S.

6 Image Processing Application

As a common application of dividers, change detection is considered to further assess the accuracy of approximate dividers. Likewise, an envelope detector is implemented by the proposed approximate SQR circuits. Finally, image reconstruction using both dividers and SQR circuits are used to evaluate the efficiency of these approximate designs.

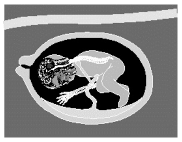
6.1 Change Detection

Change detection in image processing can be implemented by computing the ratio of two pixel values using a divider. The designs with similar values of PDP (about 300 fJ) are selected to implement change detection, including AXDr3-9, DAXD-12 and AAXD-10 (see Fig. 8). For the other designs, configurations with PDPs close to 300 fJ are selected, including AXDr1-10, AXDr2-10 and SEERAD-4. Table 4 shows the peak signal-to-noise ratios (PSNRs) of five output images and the average PSNRs. It shows that AAXD-10 achieves the highest PSNRs, followed by SEERAD-4 and AXDr3. The PSNRs for AXDr2-10 and DAXD-12 are significantly lower. AXDr1 has a moderate PSNR.

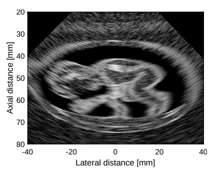
6.2 Envelope Detection in Ultrasound Imaging

As a medical tool, ultrasound imaging has widely been used in diagnostics and therapeutics; it uses high-frequency waves to produce visual images of the internal body tissue [17]. In an ultrasound imaging system, a transducer consisting of an array of small piezoelectric elements generates and transmits a pulse along a scan line. The echoes that are generated by the reflection or scattering of the pulses from tissues are received by the same transducer. As each echo carries the information about the relative position of a point at the tissue boundaries, a visual image of echo-producing features is obtained for a field of view when a large number of pulses are used. The B-mode (or brightness mode) is the most well-known imaging method to generate visual images, in which the amplitude of the echo envelope is mapped to the brightness of the image pixel.

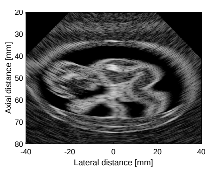
Fig. 9 shows a block diagram of a B-mode ultrasound imaging system [18]. Specifically, the received echo signals are amplified, digitized and combined by a beamformer, resulting in radiofrequency (RF) signals. As echoes from deeper targets attenuated more than those from the similar targets close to the transducer, time-gain compensation (TGC) is performed to compensate the attenuation of farther echoes. This ensures that the brightness of a B-mode image relates only to the reflection effect of each target regardless of its depth. The envelopes of the compensated



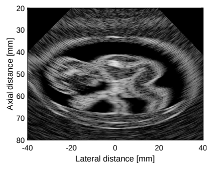
(a) Artificial fetus map



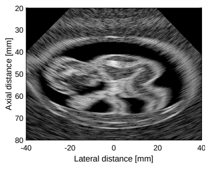
(b) Accurate output



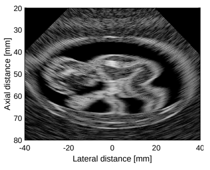
(c) AXSR3-16



(d) AXSR3-14



(e) AASR-2



(f) AASR-4

Fig. 10. Ultrasound imaging results by different SQR circuits.

TABLE 5. PSNRs of the ultrasound imaging results (*dB*).

Image	AXSR3 -16	AXSR3 -14	AXSR3 -12	AASR -2	AASR -4	AASR -6
fetus	34.27	34.26	34.26	27.91	37.36	39.60
kidney	32.69	32.69	32.71	28.04	35.49	37.88
liver	34.06	36.61	36.67	25.22	31.53	37.00
average	**33.89**	**35.19**	**35.23**	**27.14**	**34.95**	**38.37**

RF signals are then detected by the demodulation and SQR operations. The amplitude obtained by the envelope detection shows the strength of the reflection at each target. Finally, the scan conversion is performed to map the compressed amplitude (by the logarithmic compression) to its corresponding 2-D position, which results in a gray-scale image.

In this work, the approximate SQR circuits are used to implement the SQR operation in the envelope detection of ultrasound imaging, while the other arithmetic operations remain accurate. In the simulation, the Field II simulation toolbox [19], [20] is utilized to mimic an ultrasound imaging system (i.e., generating the phantom data, RF signals and demodulated signals for fetus, kidney and liver). The sum of squared values $I^2(t)+Q^2(t)$ are then quantized as 16-bit integers that are square rooted by approximate SQR circuits to calculate the envelope of the demodulated signals. 8-bit gray-scale images are finally obtained after the logarithmic compression and scan conversion. Fig. 10(b-f) show the imaging results for the artificial fetus imaged in Fig. 10(a) using the accurate and approximate 16-bit SQR circuits. The PSNRs of the output images and the average PSNRs over the three outputs are listed in Table 5. It shows that AASR-4 performs similarly as AXSR-14 in envelope detection; however, the PDP of AXSR-14 is more than 10× higher than that of AASR-4 (estimated from Table 2). The PSNRs for the images obtained by AASR-6 are generally higher than those processed by other approximate SQR circuits. Moreover, the PSNR increases more significantly for the image processed by the AASR than that by the AXSR3 when increasing the accuracy of the circuit.

6.3 QR Decomposition in Image Reconstruction

Matrix inversion is a useful operation in many applications, such as the multi-input multi-output receiver [21], computer graphics [22] and solving the linear least square problems [23]. To lower the computational complexity and latency, the inverse of a matrix

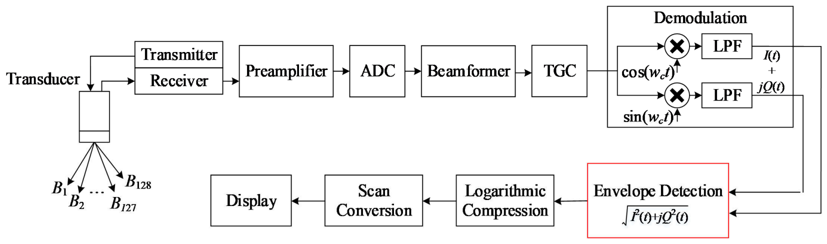


Fig. 9. The block diagram of a B-mode ultrasound imaging system. ADC: analog-to-digital converter; TGC: time-gain compensation; LPF: low-pass filter.

is usually obtained by using matrix decomposition, e.g., QR decomposition (QRD) [24], LU decomposition [25] and Cholesky decomposition [26]. Due to the ease of implementation and parallelization, QRD is considered here as an application to assess the accuracy of both the proposed approximate divider and SQR circuits. In QRD, a matrix $\mathbf{C}$ is decomposed into matrices $\mathbf{Q}$ and $\mathbf{R}$ (i.e., $\mathbf{C} = \mathbf{QR}$), where $\mathbf{Q}$ is an orthogonal matrix and $\mathbf{R}$ is an upper triangular matrix. Then, the inverse of $\mathbf{C}$ is given by $\mathbf{C}^{-1} = \mathbf{R}^{-1}\mathbf{Q}^T$, where $\mathbf{R}^{-1}$ can be easily obtained since it is an upper triangular matrix.

A popular algorithm to obtain a stable and accurate QRD result is the modified Gram-Schmidt algorithm [21]. Let the original $n \times n$ matrix $\mathbf{C}$, decomposed matrices $\mathbf{Q}$ and $\mathbf{R}$ be $\mathbf{C} = [\mathbf{c_1}, \cdots, \mathbf{c_n}]$, $\mathbf{Q} = [\mathbf{q_1}, \cdots, \mathbf{q_n}]$ and $\mathbf{R} = [\mathbf{r_1}, \cdots, \mathbf{r_n}]$, where $\mathbf{c_i}$, $\mathbf{q_i}$ and $\mathbf{r_i}$ ($i = 1, \cdots, n$) are the column vectors in $\mathbf{C}$, $\mathbf{Q}$ and $\mathbf{R}$, respectively. Then, $\mathbf{Q}$ and $\mathbf{R}$ are computed by using Algorithm 1 [21], where r_{ji} is the element in row j and column i in $\mathbf{R}$, $\langle \mathbf{q_j}, \mathbf{e_i} \rangle$ is the inner product of vectors $\mathbf{q_j}$ and $\mathbf{e_i}$, and $\|\mathbf{e_i}\|_2 = \sqrt{\sum_{j=1}^{n} e_{ji}^2}$ is the norm of the vector $\mathbf{e_i}$ (e_{ji} is the j^{th} element in $\mathbf{e_i}$). As division and SQR are more complex and time consuming than addition and multiplication, they are the performance bottlenecks for the algorithm.

Algorithm 1 Modified Gram-Schmidt Algorithm [21]

```
Input: c_i - the column vector in C.
Output: q_i - the column vector in Q.
        r_ji - the element in row j and column i in R.
1: for $i = 1$ to $n$ do
2:     $\mathbf{e_i} = \mathbf{c_i}$
3:     for $j = 1$ to $i - 1$ do
4:         $r_{ji} = \langle \mathbf{q_j}, \mathbf{e_i} \rangle$
5:         $\mathbf{e_i} = \mathbf{e_i} - r_{ji}\mathbf{q_j}$
6:     end for
7:     $\mathbf{q_i} = \frac{\mathbf{e_i}}{\|\mathbf{e_i}\|_2}$
8:     $r_{ii} = \|\mathbf{e_i}\|_2$
9: end for
```

Hence, approximate dividers and SQR circuits are applied in the QRD to lower the hardware consumption for image reconstruction. Specifically, images are compressed in the frequency domain using compressive sensing techniques; the Orthogonal Matching Pursuit (OMP) algorithm is then used to reconstruct the images [27]. To speed up the reconstruction, QRD is utilized for solving the least square problem in OMP [28]. Due to the wide range of the division input in Algorithm 1, 32/16 unsigned dividers and 32-bit SQR circuits are used for computing $\mathbf{q_i}$ and r_{ii}. For signed division, the absolute quotient is calculated by an unsigned divider, and an XOR gate is used to obtain the sign.

To compare the accuracy, the approximate dividers and SQR circuits with different configurations are tested. For the AXDrs, the two more efficient designs, AXDr1 and AXDr3, are considered (see Fig. 8). Three images (*lena*, *foreman* and *boats*), each with 256×256 pixels, are simulated for the image compression and reconstruction. The average PSNRs of the reconstructed images are shown in Table 6 for the combined use of approximate divider and SQR circuits.

The accuracy of approximate dividers and SQR circuits varies with the parameter used in the approximation schemes. In AXDr1, AXDr2, AXDr3 and ASR3, the parameter is the replacement depth of the approximate subtractors. The parameter for DAXD and AAXD is the width of the pruned dividend, whereas it is the width of the pruned radicand in AASR. For SEERAD, the parameter indicates the accuracy level. Therefore, a threshold parameter k_t is defined to indicate that the reconstructed images using the divider or SQR circuit with k_t have a similar quality as the accurate results and that the quality of the reconstructed image does not significantly change when the accuracy of the divider or SQR circuit is improved by increasing (or decreasing for AXDr1, AXDr3 and AXSR3) the parameter value. The simulation results show that these threshold values are 10, 10, 22, 14 and 6 for AXDr1, AXDr3, AAXD, AXSR3 and AASR, respectively. For SEERAD and DAXD, such a threshold value is not found due to their low accuracy.

Table 6 show that by using the approximate dividers and SQR circuits with their respective threshold parameter values the reconstructed images are as good as the accurate result, whereas the quality of the images reconstructed by using SEERAD and DAXD is very low. It is worth noting that the threshold parameter value of the proposed SQR circuit AASR is much lower than that of the proposed divider AAXD. It indicates that the SQR circuit in this application can tolerate more errors than the divider. However, the threshold parameter value of AXSR3 is not much higher than those of AXDr1 and AXDr3. It occurs because the relative error of AXSR3 is very large when the inputs are very small due to its approximation structure (i.e., k LSBs are approximated for AXSR3-k).

To compare the hardware overhead, the accurate and approximate 32/16 dividers and 32-bit SQR circuits with the identified threshold parameters are implemented in VHDL. Their circuit measurements are then obtained by using the same tools and technique as those used for evaluating the 16/8 dividers. As the critical path delay of the accurate array-based 32/16 divider is close to 20 *ns*, the clock frequency for power estimation is 50 *MHz*. The synthesis results for the combinational and sequential implementations are shown in Tables 7 and 8, respectively. Compared to the accurate implementations, the proposed

TABLE 6. Average PSNRs of three reconstructed images using different approximate dividers and SQR circuits (dB).

Design	Accurate	AXSR3-16	**AXSR3-14**	AASR-4	**AASR-6**
Accurate	27.29	27.29	27.29	27.29	27.29
AXDr1-12	23.35	11.50	21.83	25.02	27.29
AXDr1-10	27.29	11.30	27.29	25.05	27.29
AXDr3-12	25.36	11.29	24.15	25.73	26.96
AXDr3-10	27.29	11.20	27.29	25.39	27.29
SEERAD-3	15.26	15.26	15.26	15.26	15.26
SEERAD-4	12.49	12.49	12.49	12.49	12.49
DAXD-26	10.27	10.21	10.21	10.24	10.25
DAXD-28	10.21	10.26	10.19	10.26	10.25
AAXD-18	25.92	9.47	25.16	25.61	24.73
AAXD-20	27.29	9.53	27.29	25.66	27.29

TABLE 7. Circuit measurements of the combinational 32/16 dividers and 32-bit SQR circuits.

Circuit	Design	Delay (ns)	Area (μm^2)	Power (μW)	PDP (fJ)	ADP ($ns \cdot \mu m^2$)
Divider	**EXDr**	**18.49**	**1,218.0**	**136.80**	**2,529.43**	**22,520.1**
	AXDr1-10	18.03	1,212.1	132.80	2,394.38	21,853.9
	AXDr3-10	17.88	1,142.9	118.30	2,115.20	20,434.9
	AAXD-20	10.16	966.1	67.94	690.27	9,816.0
SQR	**ESRr_A**	**16.96**	**968.9**	**99.00**	**1,679.04**	**16,432.9**
	AXSR3-14	16.75	907.6	84.59	1,416.88	15,201.5
	AASR_A-6	1.30	137.7	6.17	8.02	179.1
	AASR_T-6	0.89	122.2	4.99	4.44	108.8

approximate divider and SQR circuit achieve significantly larger improvements in delay and power consumption than the other approximate designs. Specifically, AAXD-20 achieves a 45.05% speed up and 50.34% reduction in power consumption, as well as 72.71% and 56.41% decrease in PDP and ADP, respectively. AASR_A-6 is approximately 12× faster and consumes 7.29% of the power of the accurate 32-bit SQR circuit. AASR_T-6 is more efficient than AASR_A-6 in all measurements due to the small k. The hardware savings for AXDr1, AXDr3 and AXSR3 are relatively small because only a small number of subtractors are approximated in these designs. Table 8 shows that the AAXD_S-20 achieves 32.39% reduction in EPO and 2× TPA compared with EXDr_S. AASR_S-6 consumes 88.79% less power than ESRr_S with 88× TPA. Moreover, Tables 7 and 8 show that the proposed approximation strategy is more efficient for larger sizes of divider and SQR circuits, especially for sequential designs.

7 Conclusion

This paper proposes a design strategy using adaptive approximation for unsigned dividers and SQR circuits. A novel pruning scheme and error correction circuits are utilized for the divider to attain a high accuracy. The use of a reduced-width divider/SQR circuit and a shifter leads to a high-performance and low-power operation. As per the synthesis results in ST's 28 nm CMOS process, the proposed approximate divider achieves improvements by more than 60% in speed and power dissipation compared with an accurate design. The proposed divider is more accurate than the other approximate dividers when a similar PDP is considered. The change detection results further illustrate the accuracy and hardware efficiency of the proposed design.

Using a similar approximation strategy, the 16-bit approximate array-based SQR circuit is from 22.69% to 74.54% faster, and saves from 30.75% to 79.34% in power compared with the accurate design, depending on the size of the exact SQR circuit used. By using the LUT-based SQR circuit, the proposed approximate design obtains 60%-90% improvements in area and power consumption compared to its corresponding accurate design. In the application of envelope detection, the proposed SQR circuit generates results of a similar quality as the accurate design.

To assess the accuracy of the approximate divider and SQR circuit in a single application, they are both used to implement the QRD in an image reconstruction algorithm. The simulation results show that the proposed approximate divider achieves 45.05% and 50.34% reductions in delay and power, while reductions of more than 90% are achieved for the proposed approximate SQR circuit compared to the accurate array-based designs for obtaining a similar image reconstruction accuracy. Finally, the proposed design approach is applicable to sequential divider and SQR circuits. Its efficacy is supported by the simulation results.

TABLE 8. Circuit measurements of the sequential 32/16 dividers and 32-bit SQR circuits.

Circuit	Design	Delay (ns)	Area (μm^2)	Power (μW)	N	EPO (pJ/op)	TPA ($ops/(s \cdot \mu m^2)$)
Divider	**EXDr_S**	**2.36**	**844.1**	**38.06**	**18**	**13.71**	**1,549**
	AAXD_S-20	1.76	1,088.2	35.65	13	9.27	3,089
SQR	**ESRr_S**	**1.98**	**919.0**	**40.15**	**18**	**14.45**	**1,696**
	AASR_S-6	0.56	479.6	16.19	5	1.62	148,920

References

[1] B. Parhami, *Computer arithmetic: algorithms and hardware designs*. Oxford University Press, 2000.

[2] J. Fandrianto, "Algorithm for high speed shared radix 4 division and radix 4 square-root," in *IEEE Symposium on Computer Arithmetic*, 1987, pp. 73–79.

[3] L. Chen, F. Lombardi, P. Montuschi, J. Han, and W. Liu, "Design of approximate high-radix dividers by inexact binary signed-digit addition," in *Great Lakes Symposium on VLSI*, 2017, pp. 293–298.

[4] L. Chen, W. Liu, J. Han, P. A. Montuschi, and F. Lombardi, "Design, evaluation and application of approximate high-radix dividers," *IEEE Transactions on Multi-Scale Computing Systems*, 2018.

[5] M. Cappa and V. C. Hamacher, "An augmented iterative array for high-speed binary division," *IEEE Transactions on Computers*, vol. 100, no. 2, pp. 172–175, 1973.

[6] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," in *IEEE European Test Symposium (ETS)*, 2013.

[7] H. Jiang, C. Liu, L. Liu, F. Lombardi, and J. Han, "A review, classification, and comparative evaluation of approximate arithmetic circuits," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 13, no. 4, p. 60, 2017.

[8] H. Jiang, L. Liu, F. Lombardi, and J. Han, "Adaptive approximation in arithmetic circuits: A low-power unsigned divider design," in *Design, Automation & Test in Europe Conference*, 2018.

[9] W. Liu and A. Nannarelli, "Power efficient division and square root unit," *IEEE Transactions on Computers*, vol. 61, no. 8, pp. 1059–1070, 2012.

[10] M. J. Flynn, "On division by functional iteration," *IEEE Transactions on Computers*, vol. 100, no. 8, pp. 702–706, 1970.

[11] L. Chen, J. Han, W. Liu, and F. Lombardi, "Design of approximate unsigned integer non-restoring divider for inexact computing," in *Great Lakes Symposium on VLSI*, 2015, pp. 51–56.

[12] ——, "On the design of approximate restoring dividers for error-tolerant applications," *IEEE Transactions on Computers*, vol. 65, no. 8, pp. 2522–2533, 2016.

[13] S. Hashemi, R. Bahar, and S. Reda, "A low-power dynamic divider for approximate applications," in *Proceedings of the 53rd Annual Design Automation Conference*, 2016.

[14] R. Zendegani, M. Kamal, A. Fayyazi, A. Afzali-Kusha, S. Safari, and M. Pedram, "SEERAD: a high speed yet energy-efficient rounding-based approximate divider," in *Design, Automation & Test in Europe Conference & Exhibition*, 2016, pp. 1481–1484.

[15] M. D. Ercegovac, T. Lang, J.-M. Muller, and A. Tisserand, "Reciprocation, square root, inverse square root, and some elementary functions using small multipliers," *IEEE Transactions on computers*, vol. 49, no. 7, pp. 628–637, 2000.

[16] B. Barrois, O. Sentieys, and D. Menard, "The hidden cost of functional approximation against careful data sizing–A case study," in *Conference on Design, Automation & Test in Europe*, 2017, pp. 181–186.

[17] P. R. Hoskins, K. Martin, and A. Thrush, *Diagnostic ultrasound: physics and equipment*. Cambridge University Press, 2010.

[18] D. Dance, S. Christofides, A. Maidment, I. McLean, and K. Ng, *Diagnostic radiology physics: A handbook for teachers and students*. International Atomic Energy Agency, 2014.

[19] J. A. Jensen, "Field: A program for simulating ultrasound systems," in *10th Nordic-Baltic Conference on Biomedical Imaging, Vol. 4, Supplement 1, Part 1*, 1996, pp. 351–353.

[20] J. A. Jensen and N. B. Svendsen, "Calculation of pressure fields from arbitrarily shaped, apodized, and excited ultrasound transducers," *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, vol. 39, no. 2, pp. 262–267, 1992.

[21] C. K. Singh, S. H. Prasad, and P. T. Balsara, "VLSI architecture for matrix inversion using modified Gram-Schmidt based QR decomposition." in *VLSI Design*, 2007, pp. 836–841.

[22] X. Lei, X. Liao, T. Huang, H. Li, and C. Hu, "Outsourcing large matrix inversion computation to a public cloud," *IEEE Transactions on cloud computing*, vol. 1, no. 1, pp. 78–87, 2013.

[23] F. Ding, "Decomposition based fast least squares algorithm for output error systems," *Signal Processing*, vol. 93, no. 5, pp. 1235–1242, 2013.

[24] M. Karkooti, J. R. Cavallaro, and C. Dick, "FPGA implementation of matrix inversion using QRD-RLS algorithm," in *Asilomar Conference on Signals, Systems, and Computers*, 2005.

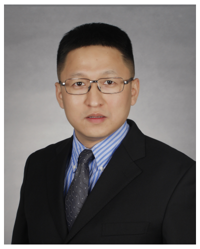
[25] M. K. Jaiswal and N. Chandrachoodan, "FPGA-based high-performance and scalable block LU decomposition architecture," *IEEE Transactions on Computers*, vol. 61, no. 1, pp. 60–72, 2012.

[26] A. Krishnamoorthy and D. Menon, "Matrix inversion using Cholesky decomposition," in *Signal Processing: Algorithms, Architectures, Arrangements, and Applications*, 2013, pp. 70–72.

[27] X. Yuan and R. Haimi-Cohen, "Image compression based on compressive sensing: End-to-end comparison with JPEG," *arXiv preprint arXiv:1706.01000*, 2017.

[28] J. L. Stanislaus and T. Mohsenin, "High performance compressive sensing reconstruction hardware with QRD process," in *IEEE International Symposium on Circuits and Systems*, 2012, pp. 29–32.

Honglan Jiang (S'14-M'18) received the B.Sc. and Master degrees in instrument science and technology from Harbin Institute of Technology, Harbin, Heilongjiang, China, in 2011 and 2013, respectively. In 2018, she received the Ph.D. degree in integrated circuits and systems from the University of Alberta, Edmonton, AB, Canada. She is currently a postdoctoral fellow in the Institute of Microelectronics, Tsinghua University, Beijing, China. Her research interests include approximate computing, reconfigurable computing and stochastic computing.



Leibo Liu (M'10) received the B.S. degree in electronic engineering and the Ph.D. degree with the Institute of Microelectronics, both from Tsinghua University, Beijing, China, in 1999 and 2004, respectively. He is currently a Full Professor with the Institute of Microelectronics, Tsinghua University. His current research interests include reconfigurable computing, mobile computing, and very large-scale integration digital signal processing.

Fabrizio Lombardi (M'81-SM'02-F'09) graduated in 1977 from the University of Essex (UK) with a B.Sc. (Hons.) in Electronic Engineering. In 1977 he joined the Microwave Research Unit at University College London, where he received the Master in Microwaves and Modern Optics (1978), the Diploma in Microwave Engineering (1978) and the Ph.D. from the University of London (1982). He is currently the holder of the International Test Conference (ITC) Endowed Chair Professorship at Northeastern University, Boston, USA. Dr. Lombardi was the Editor-In-Chief of the IEEE Transactions on Computers (2007-2010) and the inaugural Editor-in-Chief of the IEEE Transactions on Emerging Topics in Computing (2013-2017). Currently, he is the Editor-in-Chief of the IEEE Transactions on Nanotechnology. His research interests are bio-inspired and nano manufacturing/computing, VLSI design, testing, and fault/defect tolerance of digital systems. He has extensively published in these areas and coauthored/edited seven books. He is a Fellow of IEEE.

Jie Han (S'02-M'05-SM'16) received the B.Sc. degree in electronic engineering from Tsinghua University, Beijing, China, in 1999 and the Ph.D. degree from Delft University of Technology, The Netherlands, in 2004. He is currently an associate professor in the Department of Electrical and Computer Engineering at the University of Alberta, Edmonton, AB, Canada. His research interests include approximate computing, stochastic computation, reliability and fault tolerance, nanoelectronic circuits and systems, novel computational models for nanoscale and biological applications. Dr. Han and coauthors received the Best Paper Award at the International Symposium on Nanoscale Architectures (NanoArch 2015) and Best Paper Nominations at the 25th Great Lakes Symposium on VLSI (GLSVLSI 2015), NanoArch 2016 and the 19th International Symposium on Quality Electronic Design (ISQED 2018). He was nominated for the 2006 Christiaan Huygens Prize of Science by the Royal Dutch Academy of Science. His work was recognized by *Science*, for developing a theory of fault-tolerant nanocircuits (2005). He is currently an associate editor for IEEE Transactions on Emerging Topics in Computing (TETC) and IEEE Transactions on Nanotechnology. He served as a General Chair for GLSVLSI 2017 and the IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT 2013), and a Technical Program Committee Chair for GLSVLSI 2016 and DFT 2012.