Scientific Research Publishing

Wireless Sensor Network, 2022, 14, 1-22
https://www.scirp.org/journal/wsn
ISSN Online: 1945-3086
ISSN Print: 1945-3078

Hardware-Efficient Approximate Dividers for Image Processing in WSN Edge Devices

Duhwan Kim, Sunggu Lee

Department of Electric Engineering, Pohang University of Science and Technology, Pohang, Korea
Email: kdh2016@postech.ac.kr, slee@postech.ac.kr

How to cite this paper: Kim, D. and Lee, S. (2022) Hardware-Efficient Approximate Dividers for Image Processing in WSN Edge Devices. *Wireless Sensor Network*, **14**, 1-22.
https://doi.org/10.4236/wsn.2022.141001

Received: December 16, 2021
Accepted: January 27, 2022
Published: January 30, 2022

Abstract

This paper proposes a hardware-efficient implementation of division, which is useful for image processing in WSN edge devices. For error-resilient applications such as image processing, accurate calculations can be unnecessary overhead, and approximate computing that obtains circuit benefits from inaccurate calculations is effective. Since there are studies showing sufficient performance with few bit operations, this paper proposes a combinational arithmetic circuit design of 16 bits or less. The proposed design is an approximate restoring division circuit implemented with a 2-dimensional array of 1-bit subtractor cells. The main drawback of such a design is the long "borrow-chain" that traverses all of the rows of the 2-dimensional subtractor array before a final stable quotient result can be produced, thereby resulting in a long delay and excessive power dissipation. This paper proposes two approximate subtractor cell designs, named ABSC and ADSC, that break this borrow chain: the first in the vertical direction and the second in the horizontal direction, respectively. The proposed approximate divider designs are compared with an accurate design and previous state-of-the-art designs based on accuracy and hardware overhead. The proposed designs have accuracy levels that are close to the best accuracy levels achieved by previous state-of-the-art approximate divider designs. In addition, the proposed ADSC design had the lowest delay, area, and power characteristics. Finally, the implementation of both proposed designs for two practical applications showed that both designs provide sufficient division accuracy.

Keywords

Approximate Computing, Division, Energy Efficiency, Low Latency Arithmetic

1. Introduction

Data collected using a Wireless Sensor Network (WSN) needs to be collected

and processed before it can be utilized effectively. For efficient utilization of large WSNs, a lot of this processing can be done in edge devices, before they are sent to a central server for further processing. This paper considers the use of large WSNs for intelligent tasks involving image processing, wherein the image processing tasks involve various types of computer arithmetic operations including division.

In general applications, a division is less heavily utilized than addition or multiplication. However, division circuits tend to have an unusually long latency compared to other arithmetic circuits, so when they are used, the division operation has an inordinate effect on the overall processing delay. In addition, the division requires a large silicon area and a lot of power due to the highly iterative nature of its typical computation method.

The division is primarily used in scientific applications, image processing, and deep learning training and inference. However, highly accurate calculations are typically not required due to the error tolerance of applications such as image processing and recognition [1]. As a result, the promising paradigm of approximate or inaccurate computing has been investigated with great interest. Approximate circuits are typically designed using highly efficient hardware while providing an acceptable level of accuracy [2].

Since division is an infrequently used operation, it is typically implemented using minimal digital logic hardware. Division can be implemented as a sequential or combinational logic circuit. Since pixels in images are typically 8-bit values, a divider in image processing applications does not need to be too large, and thus is typically implemented in a straightforward manner using subtractor cells arranged in a regular 2-dimensional (2-D) array structure, referred to as a restoring divider array, shown in **Figure 1**. The main drawback of this design is the

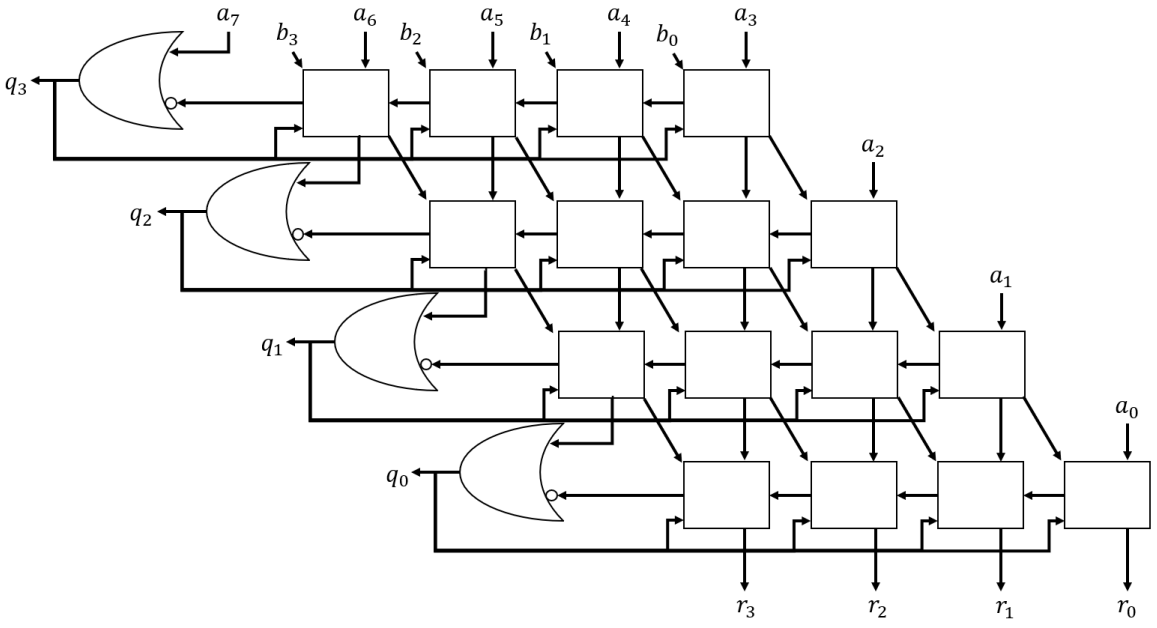


Figure 1. An 8/4 unsigned restoring array divider.

excessively long propagation delay, which is due to the need to traverse almost all of the subtractor cells in the 2-D array, as shown using a red line in Figure 4(a). The need for this long propagation delay is primarily due to the fact that division requires the computation of quotient bits one at a time, followed by conditional subtraction of the divisor.

This paper proposes a new approximate divider design approach based on breaking the "borrow-chain" in the 2-D subtractor array used in a restoring division method. This paper analyzes the horizontal and vertical outputs of subtractor cells, which are the borrow out and difference outputs, respectively, and proposes a divider circuit based on a 2-D subtractor array that uses two novel approximate subtractor cells designs. The quotient bits of a division operation cannot be processed in parallel, unlike the partial product bits of a multiplication operation. So, when the subtractor cell of a division circuit is approximated, it is necessary to consider how such a change will affect the overall quotient. In this paper, two novel approximate subtractor cell designs are proposed, with the aim of breaking the long propagation chain used in division. The characteristics of divider designs based on the proposed approximate subtractor cells are analyzed for various bit lengths, replacement methods, and approximation depths. Since the presented cells are applied to an array-based divider, they can be combined into various arrays and designed to suit the application. The evaluations presented in this paper can be used as a reference for designing for specific criteria.

The rest of the paper is organized as follows. Section 2 introduces the exact design of the typical array divider circuit and reviews the existing approximate divider designs. The two proposed subtractor cells and divider designs using these cells are introduced in Section 3. Section 4 presents a complete analysis based on accuracy, power, latency, and area overhead. Section 5 demonstrates that the proposed designs provide sufficient accuracy by the implementation of two practical applications. Section 6 provides concluding remarks.

2. Background and Related Work

The division $A \div B$ of a dividend A and a divisor B is defined as $A = B * Q + R$, where Q is the quotient and R is the remainder. Multiplication of two n-bit values results in a $2n$-bit product. Analogously, a $2n$-bit dividend is divided by an n-bit divisor to produce an n-bit quotient and n-bit remainder. In the case of an unsigned restoring divider, the basic subtractor cell shown in Figure 2, consisting of a full subtractor and a multiplexer to select between the current subtraction result and the previous partial result, can be used [3].

In computer arithmetic, the basic algorithm used for division has an inherent delay propagation problem due to the nature of the division problem. Consider the division of two numbers in unsigned binary, such as 00101110 (dividend bits $a_7a_6a_5a_4a_3a_2a_1a_0$) divided by 1011 (divider bits $b_3b_2b_1b_0$). The quotient bits are determined one at a time, starting from the most significant bit position, by subtracting the divider—if the result is negative, the quotient bit is 0; otherwise, it is

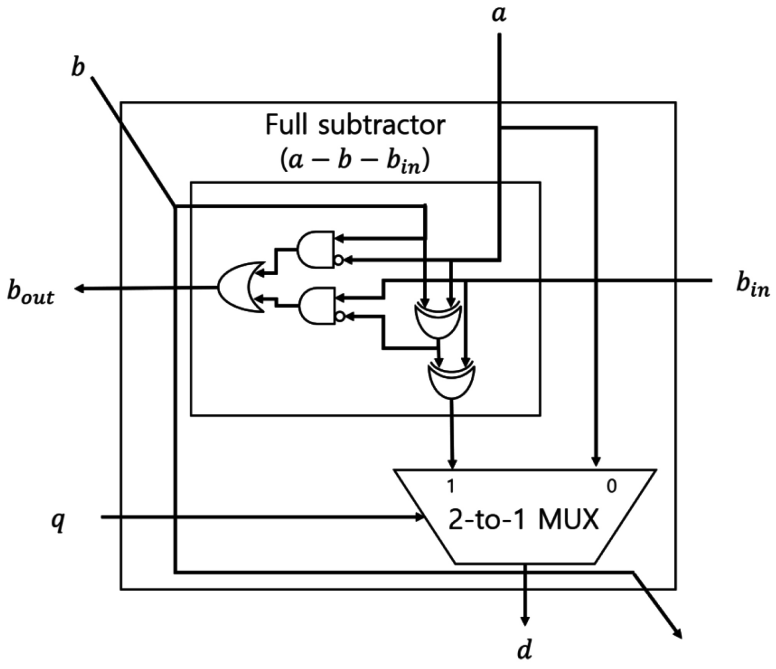


Figure 2. A subtractor cell.

1. In the next step, if the quotient bit was a 1, then the divider is subtracted from the previous subtraction result. However, if the quotient bit was a 0, then the divider is subtracted from the result before the previous subtraction. Referring to Figure 1 and using the example shown above, in Step 1 (Row 1 of Figure 1), 00101 – 1011 produces 11,000, which is a negative number. Thus, the quotient bit q_3 is 0. Then, in Step 2 (Row 2), 01011 (bits $a_6a_5a_4a_3a_2$) – 1011 = 00000, which is non-negative. Thus, q_2 is 1. Then, in Step 3 (Row 3), the previous subtraction result with the "next" a_1 bit appended 00001 – 1011 = 10100, a negative number. Thus, q_1 is 0. Then, in Step 4 (Row 4), the previous subtraction result with the "next" a_0 bit appended 00011 – 1011 = 00010, which is non-negative. Thus, q_0 is 0. The final result is the quotient $q_3q_2q_1q_0$ = 0100 with remainder 0010 (last subtraction result). The "borrow" propagation problem illustrated in this example stems from the fact that when going from Row i to Row i+1, the choice of the number to use to subtract the divisor 0010 depends on the previous subtraction result.

Previous works on approximate divider circuits have focused on different aspects of problems with the basic divider circuit [4]-[11]. Chen *et al.* approximated a fixed error with inexact subtractor cells [4] [5]. Each cell was approximated in units of transistors, and because only a small portion was approximated, a small error occurred, but the hardware gain was also small. Jiang et al. proposed a method of truncating operands based on leading zeros so that only the most relevant bits are used for calculation [6] [7]. The truncated dividend

and divisor are calculated by a narrow width divider, and the error is fixedly determined by this width. Behroozi *et al.* [8], Melchert *et al.* [9], and Jeong *et al.* [10] also used truncated operands, but the accuracy was dynamically scaled during runtime as a multiplicative divider [8] [9] [10]. As the Taylor series approximation method was used, the accuracy could be increased with the number of iterations used. Adams et al. proposed two approximation methods, one involving an approximate subtractor cell that breaks the "borrow chain" and approximates the difference output and the second involving the use of truncation to compute the least significant quotient bits in an approximate manner [11].

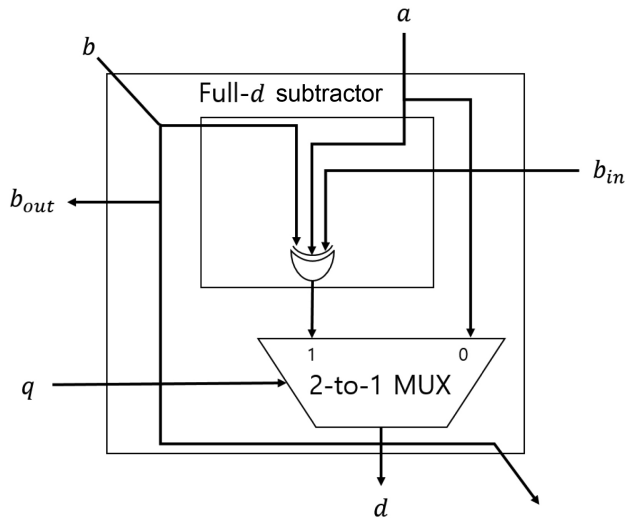
3. Proposed Approximate Divider

3.1. Approximate Subtractor Cells

In this paper, two types of approximate subtractor cells, ABSC (approximated borrow subtractor cell) and ADSC (approximated difference subtractor cell) are proposed. The logic diagrams for these two designs are shown in **Figure 3**, while the truth tables are shown in **Table 1**. The colored entries in **Table 1** show where the accurate output bits and approximate output bits differ for the two designs. ABSC results in approximate borrow and accurate difference outputs, while ADSC results in accurate borrow and approximate difference outputs. In ABSC and ADSC, the number of incorrect output bits is four of the sixteen output entries for the borrow output and difference output, respectively.

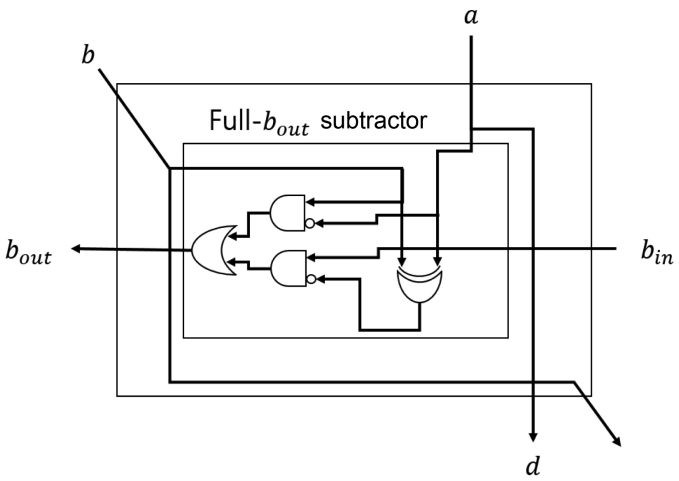
Table 1. Truth table for the two approximate subtractor cell designs.

q	a	b	b_{in}	*Exact*		*ABSC*		*ADSC*	
				b_{out}	d	b_{out}	d	b_{out}	d
0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	1	0
0	0	1	0	1	0	1	0	1	0
0	0	1	1	1	0	1	0	1	0
0	1	0	0	0	1	0	1	0	1
0	1	0	1	0	1	0	1	0	1
0	1	1	0	0	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	0
1	0	0	1	1	1	0	1	1	0
1	0	1	0	1	1	1	1	1	0
1	0	1	1	1	0	1	0	1	0
1	1	0	0	0	1	0	1	0	1
1	1	0	1	0	0	0	0	0	1
1	1	1	0	0	0	1	0	0	1
1	1	1	1	1	1	1	1	1	1



$$b_{out} = b$$
$$d = q \cdot a \oplus b \oplus b_{in} + \bar{q} \cdot a$$

(a)

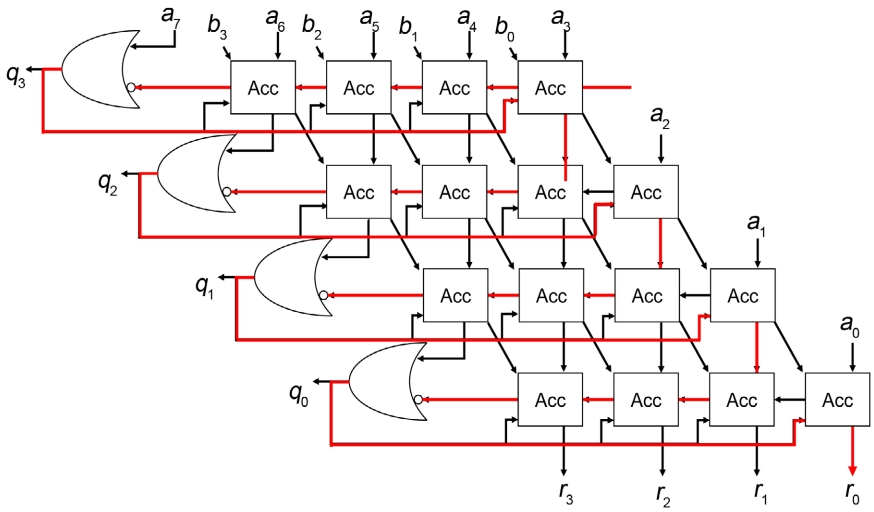


$$b_{out} = \bar{a} \cdot b + \bar{a} \cdot b_{in} + b \cdot b_{in}$$
$$d = \mathrm{a}$$

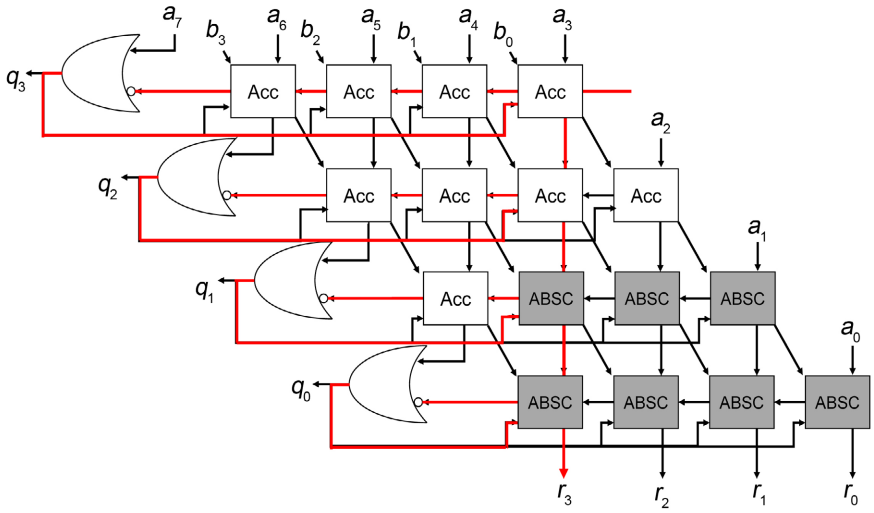
(b)

Figure 3. The two proposed approximate subtractor designs: (a) ABSC and (b) ADSC.

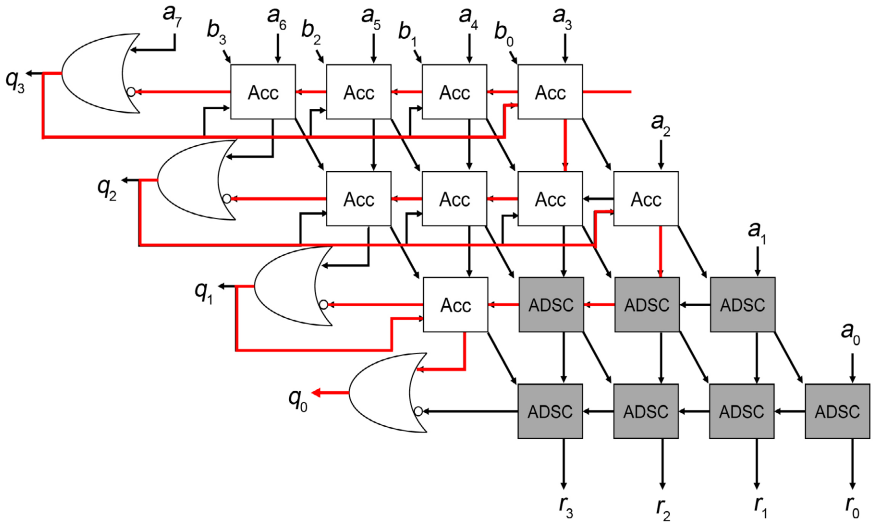
The ABSC subtractor cell is designed with the purpose of approximating the borrow while maintaining the difference output. This has the effect of breaking the horizontal right-to-left propagation chain. The logic design for this cell is shown in **Figure 3(a)** and its effect on the propagation delay of an array divider is shown in **Figure 4(b)**. As can be seen, once the first ABSC subtractor cell is encountered, the horizontal right-to-left borrow chain is no longer part of the critical delay path. This should have the effect of breaking the borrow chain while maintaining, to a certain extent, the accuracy of the different outputs in the vertical direction.



(a)



(b)



(c)

Figure 4. The 2-D restoring divider array with subtractor cells that are (a) fully accurate, (b) ABSC cells, and (c) ADSC cells.

Conversely, the ADSC subtractor cell is designed with the purpose of approximating the difference output while maintaining the borrow output. This has the effect of breaking the vertical downward-direction propagation chain. The logic design for this cell is shown in **Figure 3(b)** and its effect on the propagation delay of an array divider is shown in **Figure 4(c)**. As can be seen, once the first ADSC subtractor cell is encountered, the vertical downward-direction chain is no longer part of the critical delay path. This should have the effect of approximating the difference output while maintaining, to a certain extent, the accuracy of the borrow chain that leads to the quotient bit outputs.

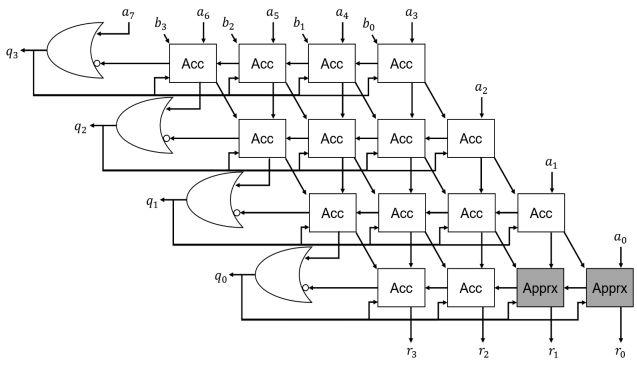
3.2. Cell Replacement

The approximate divider can be implemented by replacing each cell with one of the approximate subtractor cells proposed in Section III.1, and the number of replaced cells can be expressed as the depth. However, since the cell replacement has different effects depending on the characteristics of the cell used and the location where it is replaced, three cell replacement methods, which reflect the most logical progression of cell replacements, have been utilized as in [4] [5]. The three types of replacement methods used are referred to as *horizontal*, *vertical*, and *stepwise* replacement methods.

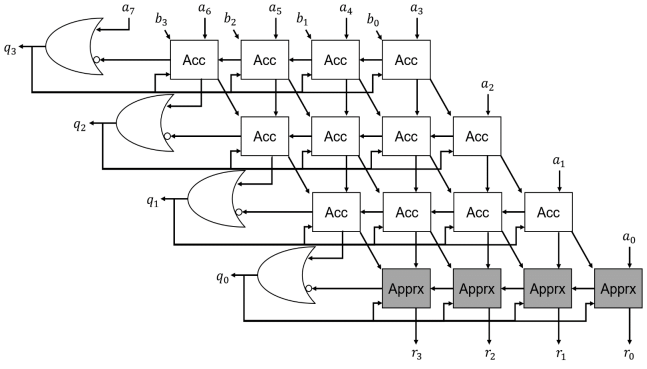
- Horizontal Replacement (HR): In this replacement method, cells closer to the LSB (Least Significant Bit) of the Quotient are replaced first. Starting with the lowermost row (the LSB quotient bit), cells are replaced from the right to the left (LSB to more significant bit positions of the difference result in each row). Once the lowermost bit positions have all been replaced, the procedure repeats with the next row above (the next most significant quotient bit). This replacement method is depicted in **Figure 5(a)**. Note that, with this method, accurate quotient bit calculation is possible, using the restoring division procedure, until the first approximate cell is encountered.
- Vertical Replacement (VR): This is a method in which the cells of the array divider are replaced in a vertical manner from right to left. This has the effect of approximating the rightmost part (least significant bit portion) of the 2-D subtractor array. Thus, if the cell at position (i, j) cell is replaced, the cell at position $(i+1, j)$ is replaced next. Then, after the j^{th} row cell of the MSB column is replaced, the cell at position $(0, j+1)$ will be replaced next. This replacement method is shown in **Figure 5(b)**.
- Stepwise Replacement (SR): This replacement method is a combination of HR and VR. When (i, j) is replaced, the next cell to be replaced is $(i+1, j-1)$, which is the directly connected upper column cell. Then, if the replaced (i, j) cell uses the dividend bit as its input instead of the difference bit from its cell, the next replacement occurs at position $(0, j+1)$. This method is illustrated in **Figure 5(c)**.

4. Results

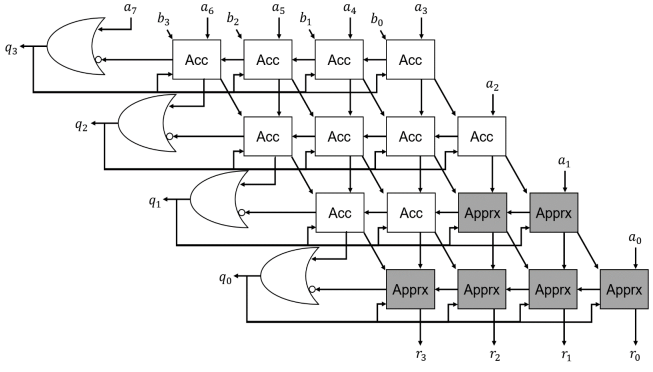
To evaluate the accuracy and circuit characteristics of dividers designed using



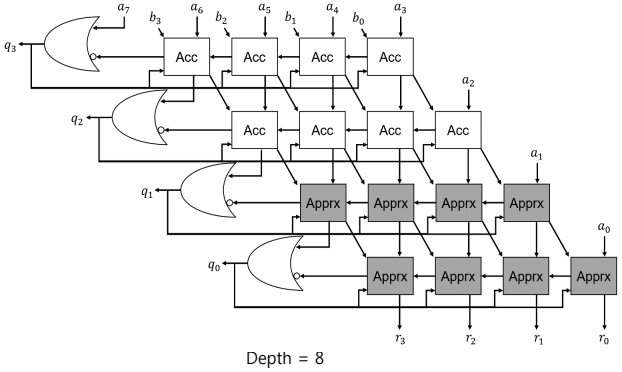
Depth = 2



Depth = 4

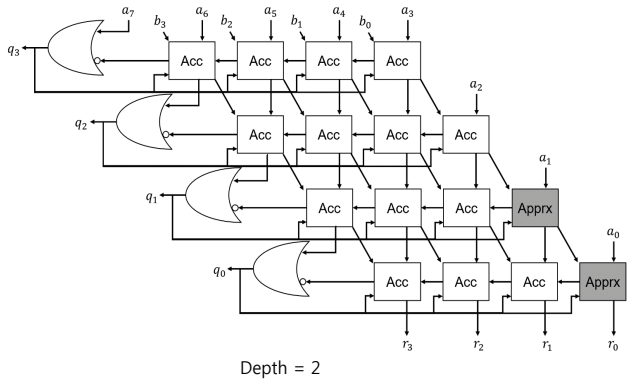


Depth = 6

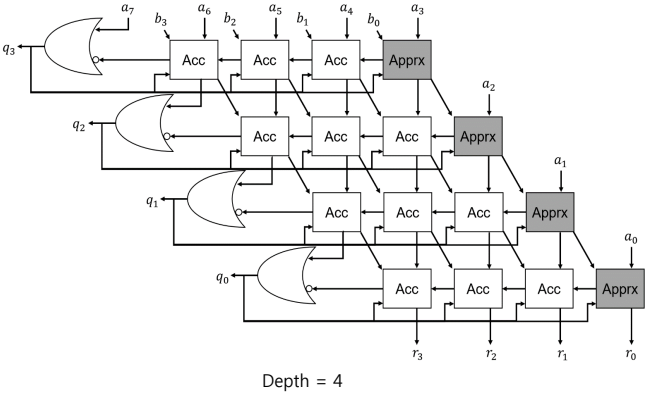


Depth = 8

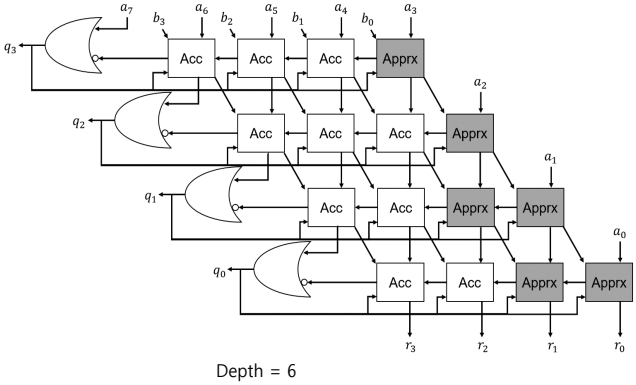
(a)



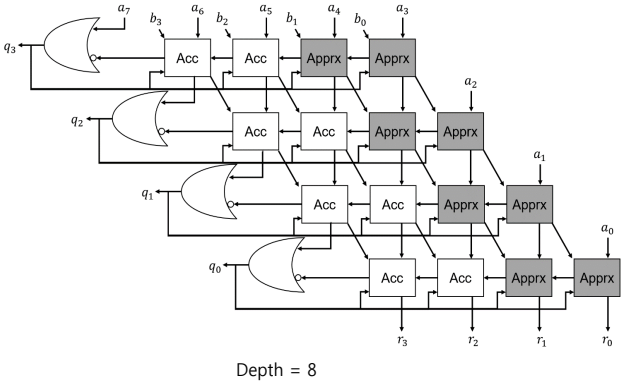
Depth = 2



Depth = 4

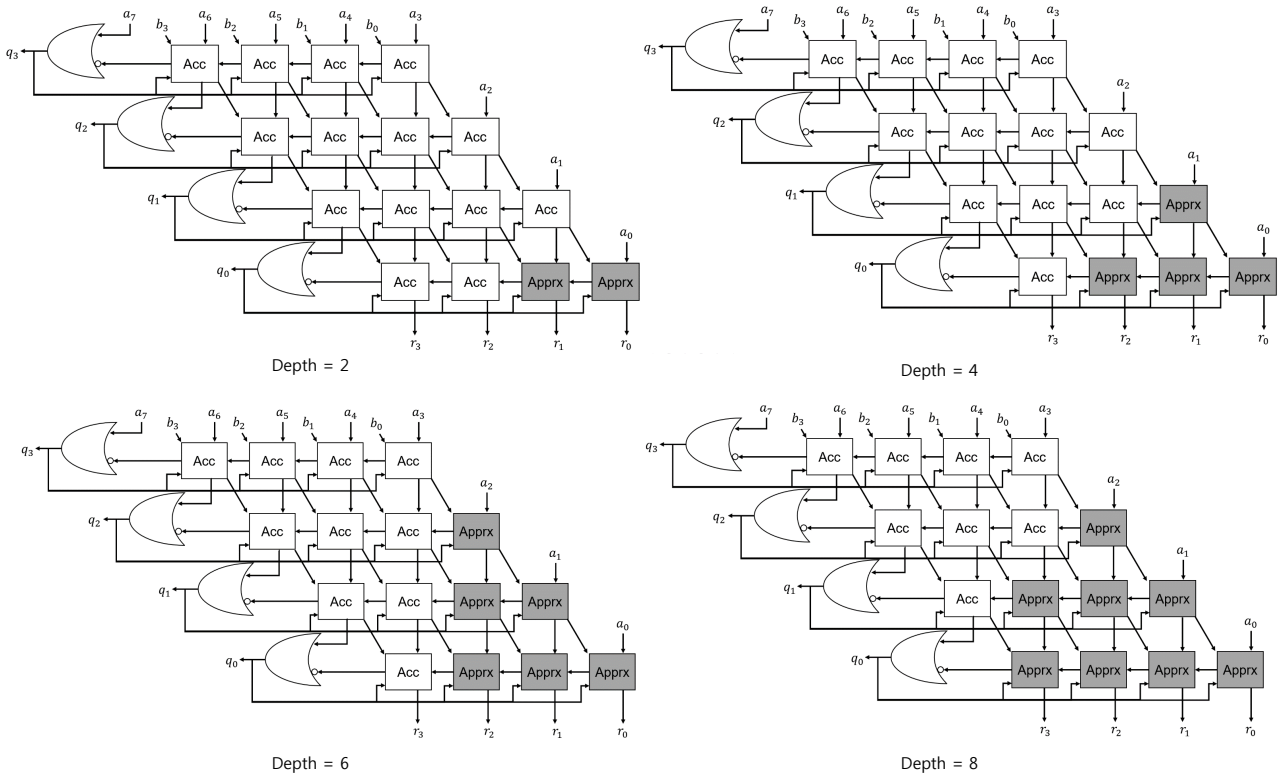


Depth = 6



Depth = 8

(b)



(c)

Figure 5. Cell replacement example for depth = 2, 4, 6, and 8 using the (a) horizontal (HR), (b) vertical (VR), and (c) stepwise replacement (SR) method.

the proposed approximate divider circuits, a sequence of successively more approximate dividers using the designs have been implemented in C and VHDL. Successive $2n/n$ divider designs were implemented with $2n = 4, 8, 12, 16$ for each combination of approximate cells and replacement methods. Comparisons were made with an exact divider design and several of the state-of-the-art (SOTA) approximate array divider designs.

4.1. Accuracy Comparison with Accurate Design

Accuracy can be measured using one of several measures that indicate how close the approximate division results are to accurate division results. However, in order to facilitate comparisons with previous SOTA methods, the most common accuracy metrics used in those methods are also used here. The error distance is the absolute difference between the integer values of the outputs that produced by an accurate design and an approximate design. Then, the *Mean Relative Error Distance* (MRED) is the mean of the error distance values divided by their respective accurate values. Alternatively, the *Normalized Error Distance* (NED) is the mean of the error distance values divided by the maximum possible error (typically $2n$ for a $2n/n$ divider) [12]. The *Root Mean Squared Error* (RMSE) is the square root of the mean squared error between approximate and exact values.

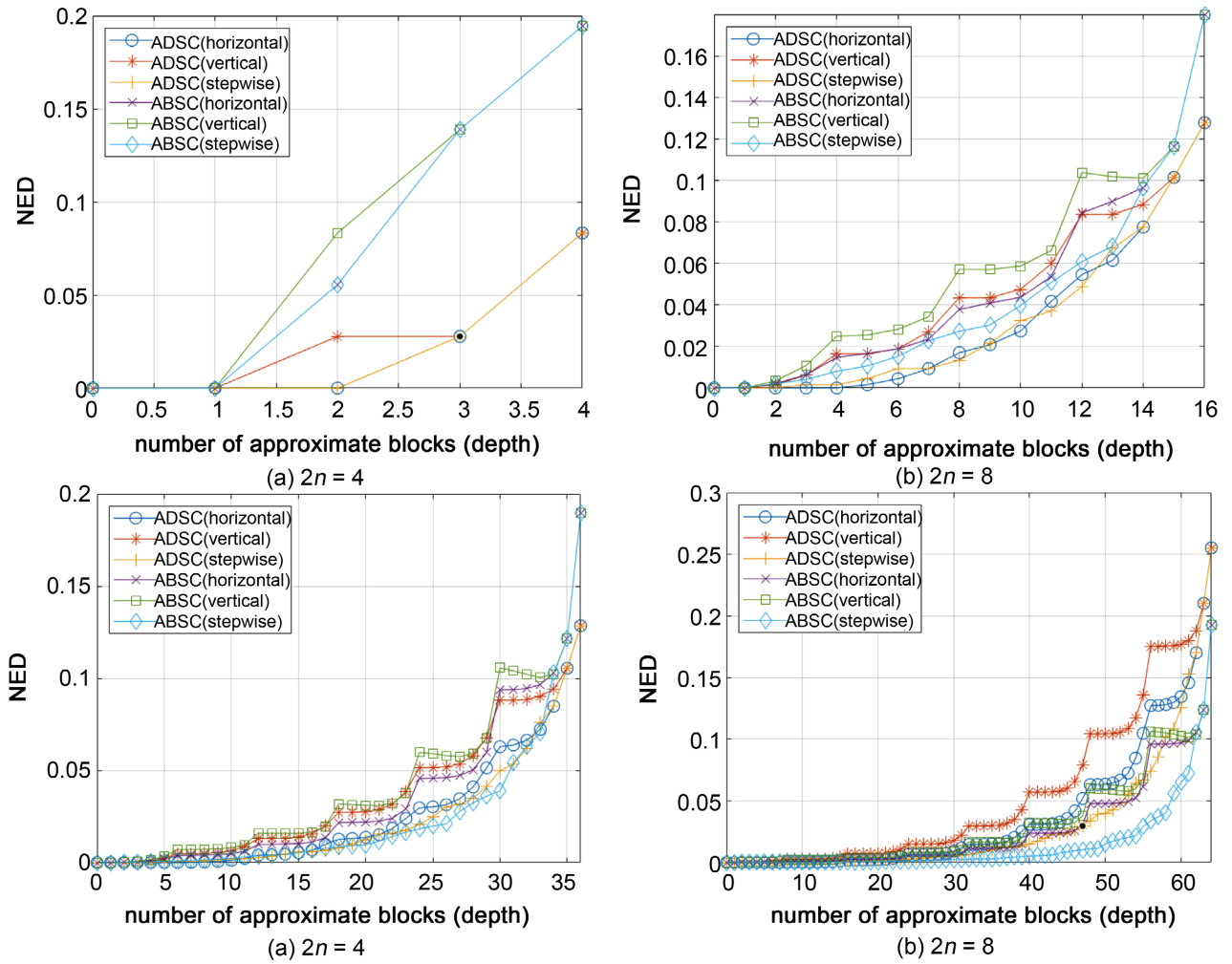


Figure 6. Normalized Error Distance (NED) of the proposed approximate divider circuits.

The results for accuracy levels are illustrated in Figure 6 & Figure 7, and the representative 4/2 and 16/8 dividers are shown in Table 2. Factor r_{apprx} shows the approximation rate, where r_{apprx} = depth/number of cells in the divider. With a small bit width, such as $2n = 4$, ADSC circuits resulted in lower NEDs than ABSCs. However, for large bit widths $2n = 16$, the two lowest NEDs were achieved by ABSC with HR and SR. In general, the NED levels are lowest when using the SR replacement method and highest when using the VR replacement method. The ABSC divider resulted in low MRED values. As in NED, the lowest MRED was observed in ABSC with SR for large bit widths $2n = 16$. When nearly 90% of the cells were replaced, the ADSC circuit showed a lower MRED than the ABSC circuit. In terms of MRED, SR is still the best replacement method, but the rest of the methods were affected by the characteristics of the approximate cell used.

4.2. Circuit Characteristic Comparisons with Accurate Design

To measure the circuit characteristics of the proposed designs, silicon area, power dissipation, and propagation delay were used as the primary metrics. In addition,

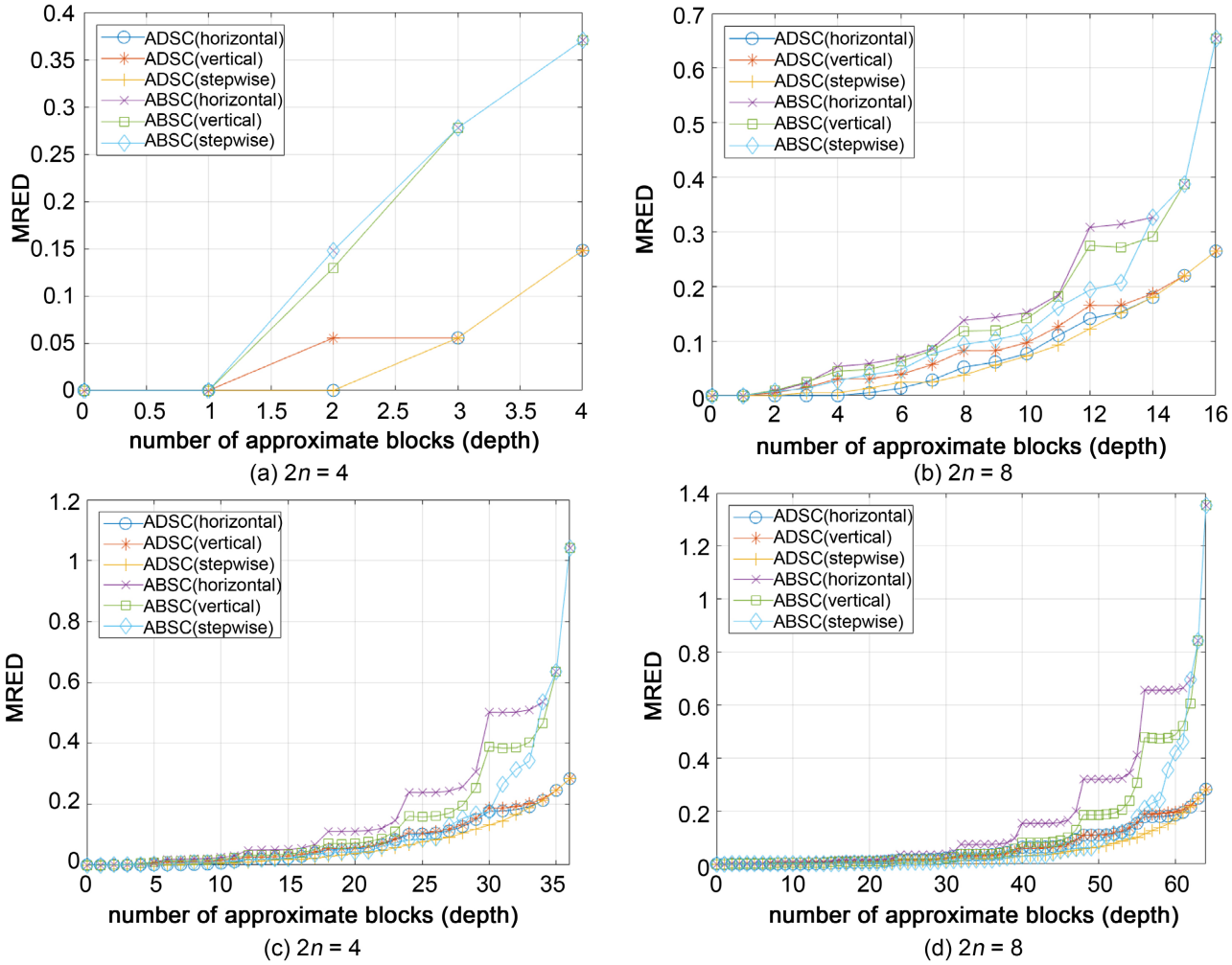


Figure 7. Mean Relative Error Distance (MRED) of the proposed approximate divider circuits.

combinations of these metrics were computed to assess the combined effects of the primary metrics. The divider circuits were implemented in Verilog and synthesized in a Samsung 28 nm CMOS cell library. Synopsys Design Compiler was used for synthesis. The supply voltage was 1.1 V, the temperature was 25˚C, and the clock frequency was 200 MHz.

4.2.1. Effect of Subtractor Cell Approximation Method

The overall effect and difference, between the ABSC and ADSC subtractor cell approximation methods were evaluated by examining the results for the extreme case of a $2n = 16$ divider with a depth of 64, which corresponds to using all approximate subtractor cell designs. The results are shown in part (g) of **Figures 8-10**. Although the circuit characteristics were slightly affected by the replacement method used, the results tended to depend on the characteristics of the approximate cell. Although several gates were omitted in ABSC, the approximation and circuit overhead reduction effects of omitting the 2-to-1 multiplexer in ADSC were greater. For this extreme case, the ABSC divider resulted in 98.1% of the area, 70.5% of the power, and 32.0% of the delay when compared to the

Table 2. Accuracy values of quotient for proposed approximate 4/2 and 16/8 dividers.

Design	r_{apprx}	*ER* (%)		*MRED* (10^{-2})		*NED* (10^{-2})		*RMSE*		ED_{max}	
		$n = 2$	8	2	8	2	8	2	8	2	8
ABSC-HR	0.25	0	42.93	0	0.194	0	0.250	0	1.085	0	3
	0.5	22.2	67.1	14.8	0.29	5.56	1.16	0.47	4.89	1	15
	0.75	55.6	83.2	27.8	0.51	13.9	4.82	0.75	20.2	1	63
ABSC-VR	0.25	0	44.4	0	0.11	0	0.43	0	2.84	0	255
	0.5	33.3	69.9	13.0	0.28	8.33	1.68	0.58	9.57	1	255
	0.75	55.6	85.4	27.8	0.41	13.9	6.05	0.75	29.8	1	255
ABSC-SR	0.25	0.0	10.9	0.0	0.03	0.0	0.04	0.0	0.35	0.0	16
	0.5	22.2	50.6	14.8	0.13	5.56	0.25	0.47	1.10	1	64
	0.75	55.6	84.4	27.8	0.32	13.9	1.10	0.75	4.88	1	255
ADSC-HR	0.25	0.0	25.1	0.0	0.57	0.0	0.20	0.0	0.50	0	1
	0.5	0.0	67.6	0.0	3.39	0.0	1.45	0.0	2.73	0	7
	0.75	11.1	88.6	5.56	10.9	2.78	6.30	0.33	11.4	1	31
ADSC-VR	0.25	0.0	49.1	0.0	0.69	0.0	0.69	0.0	1.96	0	109
	0.5	11.1	79.8	5.56	3.01	2.78	2.95	0.33	7.04	1	127
	0.75	11.1	92.8	5.56	10.9	2.78	10.4	0.33	21.1	1	127
ADSC-SR	0.25	0.0	9.54	0.0	0.17	0.0	0.08	0.0	0.33	0	13
	0.5	0.0	53.5	0.0	1.37	0.0	0.63	0.0	1.39	0	53
	0.75	11.1	89.1	5.56	5.69	2.78	3.38	0.33	6.46	1	127

accurate divider. The ADSC divider resulted in 41.0% of the area, 20.9% of the power, and 10.1% of the delay when compared to the accurate divider. The accuracy, as reflected by the RMSE, of the ABSC and ADSC dividers were 80.72 and 45.89, respectively.

4.2.2. Effect of Subtractor Cell Replacement Method

Next, to examine the effect of the subtractor cell replacement used with the ABSC and ADSC subtractor cells, the circuit metrics were evaluated for all combinations of bit widths, replacement methods, and depths. As stated in the above subsection, ADSC has a greater circuit overhead reduction effect than ABSC. Accordingly, in Figure 8 & Figure 9, it can be seen that ADSC with HR and SR have the best area and power characteristics for small bit widths while ADSC with HR, VR, and SR perform similarly, in terms of area and power, for the largest bit widths. Figure 10 shows the ADSC method with all 3 replacement methods showing similar levels of delay reduction.

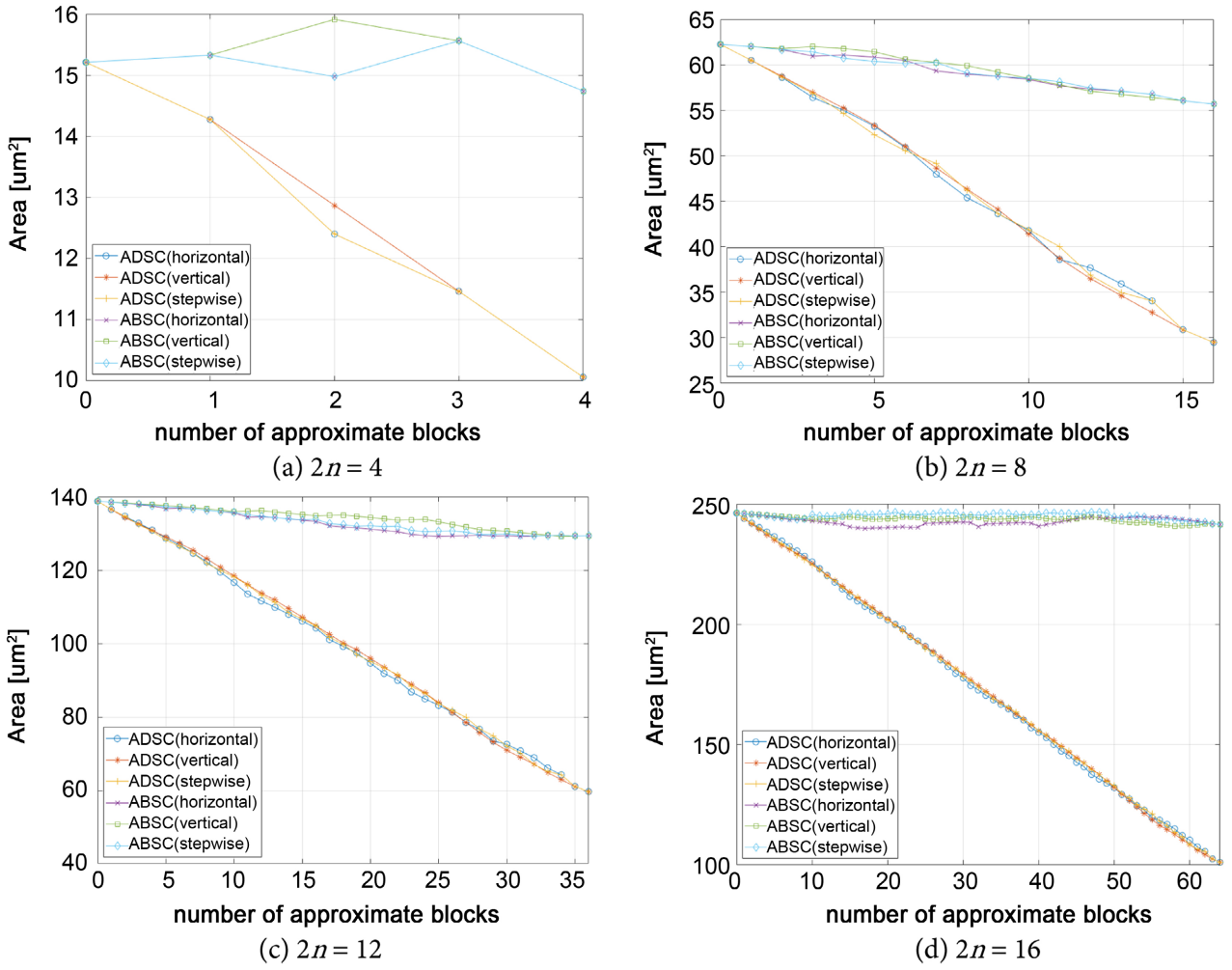


Figure 8. Area of the approximate divider circuits.

4.2.3. Combined Metric Results for Comparison with Accurate Design

In order to evaluate the combined effects of the primary metrics, the Power Delay Product (PDP) is evaluated for the various designs considered. **Figure 11** shows the results.

With respect to the power and propagation delay effects, the PDP results of **Figure 11** show that ADSC is superior to ABSC. This is in line with the delay and power measurements of **Figure 9** & **Figure 10**. Circuit delay and overhead values for the representative 4/2 and 16/8 dividers are shown in **Table 3**.

4.2.4. Comparisons with Previous State-of-the-Art Designs

Comparisons were also made with previous state-of-the-art (SOTA) designs. Although the work by Jeong *et al.* [10] is the most recent SOTA design, it was an iterative array divider design and thus could not be directly compared with the proposed designs. Thus, the most recent, comparable designs were the AXRD-M1 design proposed by Adams *et al.* [11], and the AXDr1-3 designs proposed by Chen *et al.* [5].

Table 4 & **Table 5** show a detailed comparison of the proposed ABSC-SR and

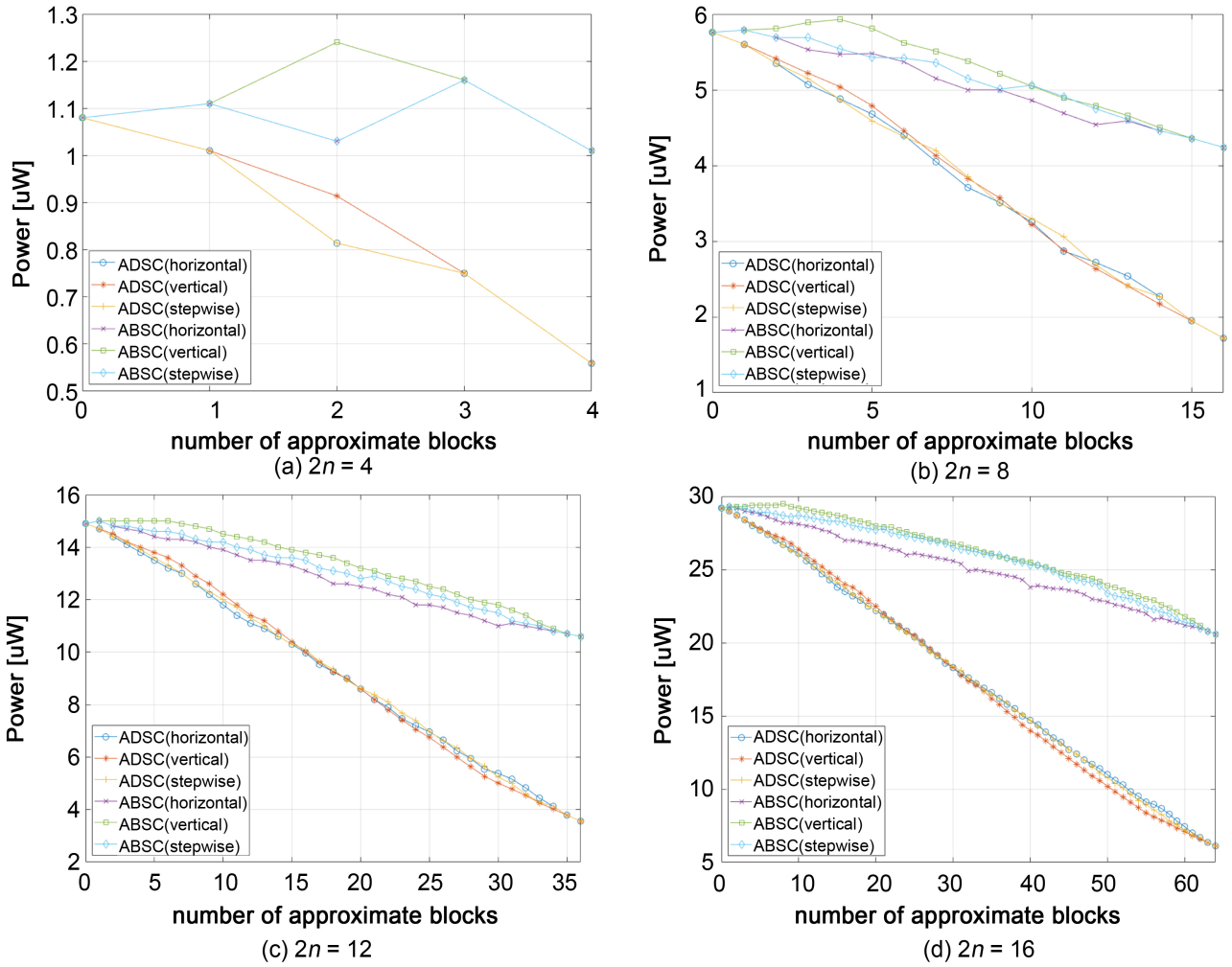


Figure 9. Power of the approximate divider circuits.

ADSC-SR designs with the AXDR-M1 and AXDr1-3 designs. The approximation factor, denoted by p, indicates the number of columns of subtractor cells that use approximate subtractor cells. The p value is shown in red in Table 4 & Table 5.

In Table 4, the accuracies of all methods compared are measured in terms of ER, MRED, NED, and RMSE. The best accuracy results, for each p value, are marked in red. Although the proposed ABSC and ADSC methods do not have the best accuracy results, their values can be seen to be fairly close to the best, with ABSC slightly outperforming ADSC in all cases. The results of all methods showed similar trends for each of the ER, MRED, NED, and RMSE metrics.

Table 5 compares the circuit characteristics for all methods. As can be seen, and marked in red, the proposed ADSC method has the best delay, area, and power characteristics. It thus follows that it also has the best PDP characteristics.

5. Verification Using Practical Applications

The proposed dividers were tested with two kinds of pixel division applications,

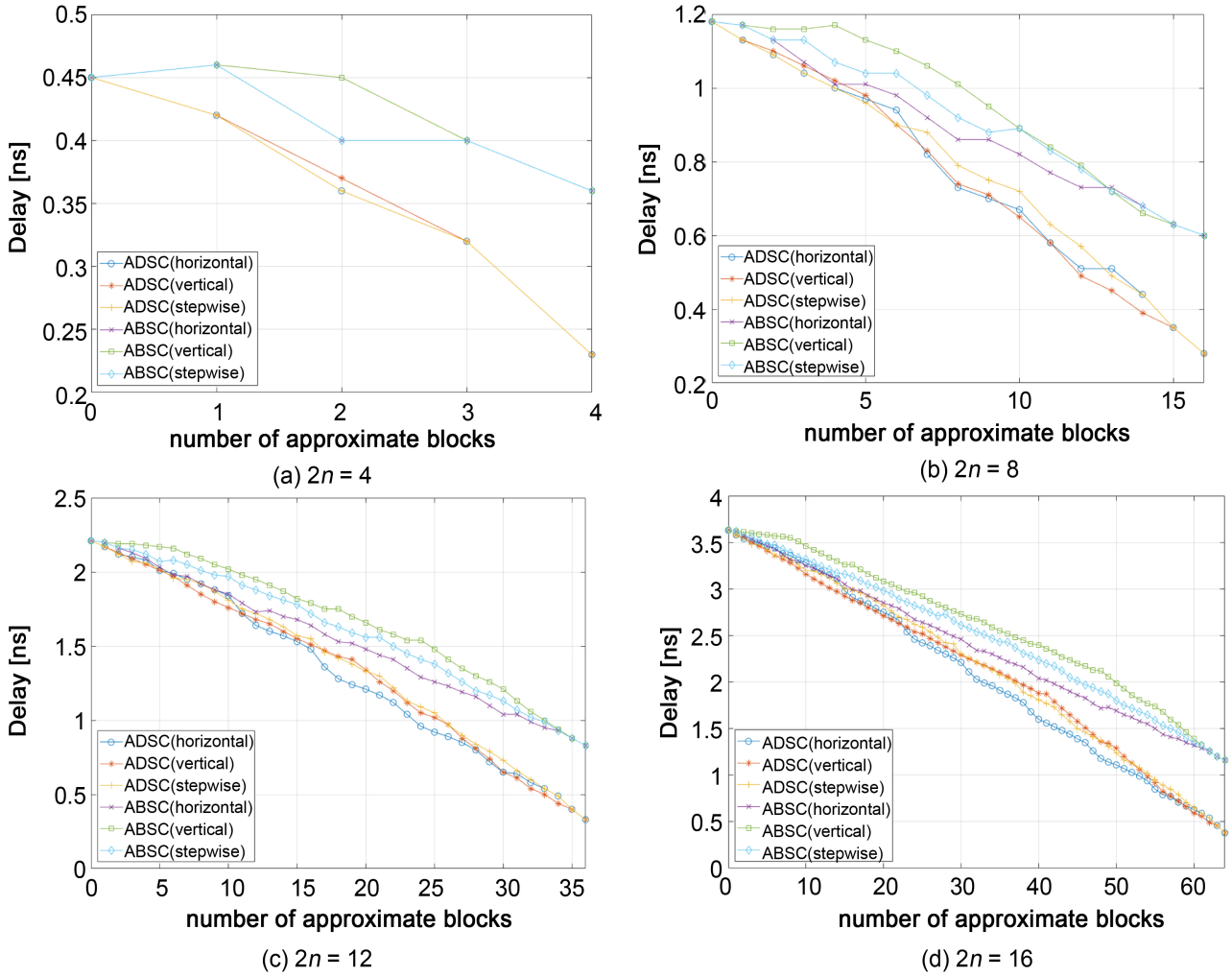


Figure 10. Delay of the approximate divider circuits.

change detection and foreground extraction.

5.1. Change Detection

Consider two almost identical images in which a few objects have been moved or have slightly different shading or color characteristics. Such differences can be caught by an image processing operation referred to as change detection, which can be implemented by simply dividing the pixel values in the first image by the pixel values in the second image. This is demonstrated using two 8-bit grayscale images as illustrated in **Figure 12**.

The level of inaccuracy that results from using approximate division is quantified using two metrics, peak signal to noise ratio (PSNR) and structural similarity index measure (SSIM), which are the two most commonly used image quality comparison metrics [13]. PSNR is used to evaluate the effect of noise on the image and calculated by a log of the ratio between the possible maximum pixel value and the root mean squared error distance. SSIM is developed to consider the quality perception of the human visual system. When comparing exact and approximate

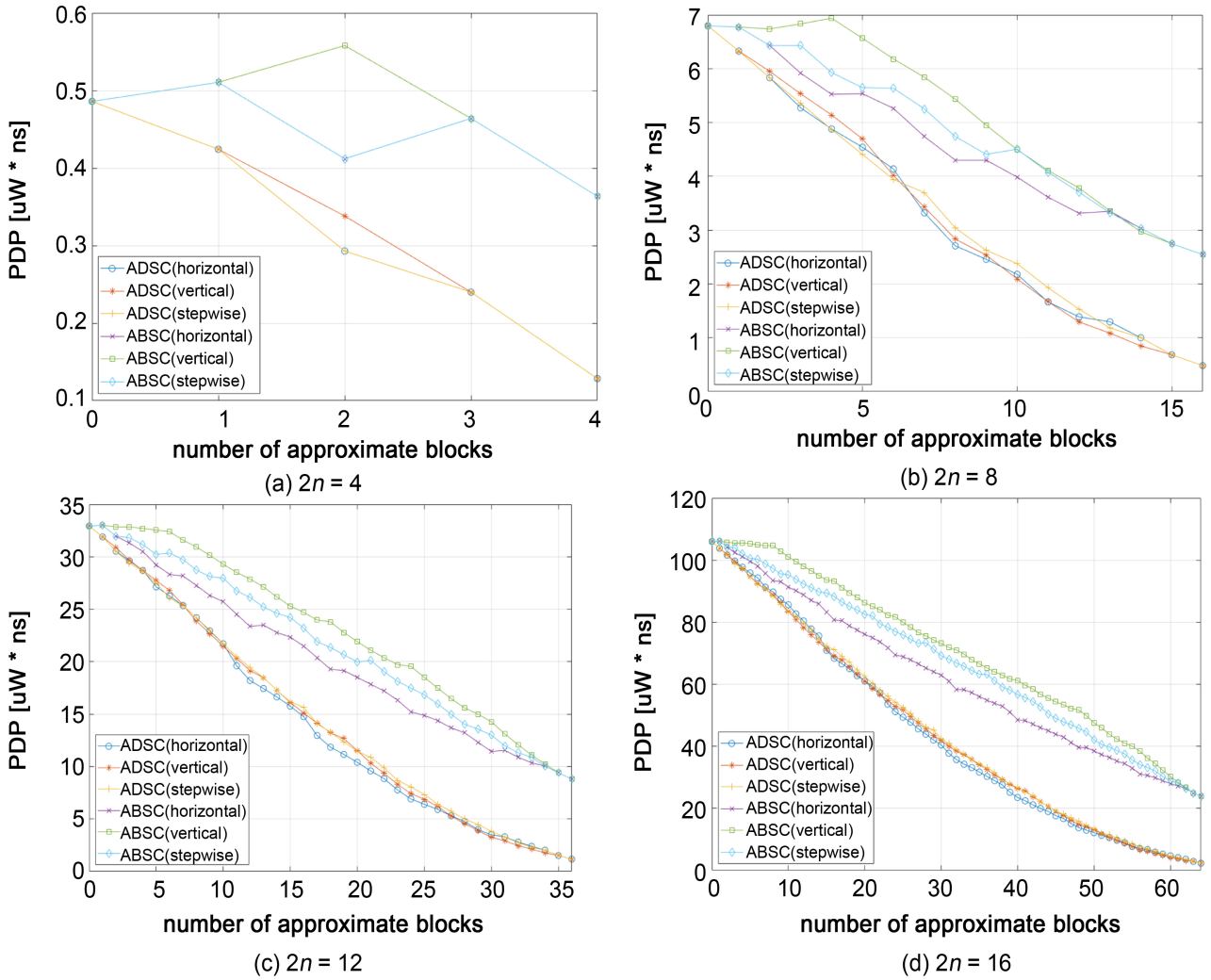


Figure 11. PDP of the approximate divider circuits.

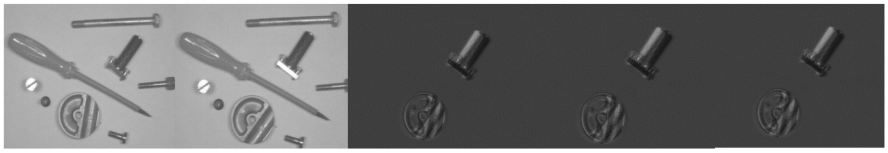


Figure 12. Change detection results for (a) the original input image and (b) the original image with two minor changes (the disk is rotated and one screw has different shading from the first image) using an (c) accurate divider, (d) ABSC with SR and $p = 4$, and (e) ADSC with SR and $p = 4$.

output images, two metrices are an approximation to human perception of reconstruction quality. The PSNR and SSIM for all designs are summarized in **Table 6**. The best results for all p values are shown in red. The best PSNR and SSIM design for $p = 6$ is the proposed ABSC method. Although the best PSNR and SSIM design are AXDr3 for both $p = 4$ and 8, **Figure 12(d)** & **Figure 12(e)** are visually similar to **Figure 12(c)**, the accurate result.

Table 3. Delay, area, power, and combined metrics for proposed approximate 4/2 and 16/8 dividers.

Design	r_{apprx}	*Delay* (ns)		*Area* (μm²)		*Power* (μW)		*PDP* (fJ)	
		$n = 2$	8	2	8	2	8	2	8
Exact	-	0.45	3.63	15.21	246.2	1.08	29.2	0.486	106.0
ABSC-HR	0.25	0.46	2.99	15.33	240.1	1.11	27.0	0.516	80.73
	0.5	0.40	2.34	14.98	240.6	1.03	24.9	0.412	58.27
	0.75	0.40	1.72	15.56	244.3	116	23.0	0.464	39.56
ABSC-VR	0.25	0.46	3.26	15.33	244.3	1.11	28.6	0.516	93.24
	0.5	0.45	2.68	15.91	244.3	1.24	26.5	0.558	71.02
	0.75	0.4	2.12	15.56	244.3	1.16	24.4	0.464	51.73
ABSC-SR	0.25	0.46	3.13	15.33	245.8	1.11	28.2	0.516	88.27
	0.5	0.4	2.54	14.98	245.4	1.03	26.3	0.412	66.80
	0.75	0.4	1.90	15.56	245.4	1.16	24.1	0.464	45.79
ADSC-HR	0.25	0.42	2.91	14.27	209.7	1.01	23.5	0.424	68.39
	0.5	0.38	2.03	11.47	172.7	0.81	17.6	0.308	35.73
	0.75	0.32	1.18	10.06	135.7	0.75	11.7	0.240	13.81
ADSC-VR	0.25	0.42	2.88	14.27	211.1	1.01	24.0	0.424	69.10
	0.5	0.35	2.21	12.87	174.6	0.81	17.4	0.284	38.45
	0.75	0.32	1.36	10.06	137.6	0.75	10.9	0.240	14.82
ADSC-SR	0.25	0.42	2.99	14.27	211.2	1.01	23.8	0.424	71.16
	0.5	0.38	2.21	11.47	173.8	0.81	17.6	0.308	38.90
	0.75	0.32	1.37	10.06	137.6	0.75	11.6	0.240	15.89

Table 4. Accuracy comparisons for the quotients of the proposed approximate 16/8 dividers with previous state-of-the-art methods.

Design	*p*	*ER* (%)	*MRED* (10^{-2})	*NED* (10^{-2})	RMSE
ASC-SR	4	3.55	0.079	0.014	0.190
	6	17.44	0.375	0.072	0.464
	8	60.72	1.596	0.321	1.361
ASC-HR	4	3.31	0.063	0.026	0.185
	6	23.40	0.409	0.203	0.602
	8	71.29	1.914	1.151	2.441
AXRD-M1 [11]	4	9.26	0.226	0.037	0.232
	6	23.47	0.494	0.101	0.604
	8	66.29	1.909	0.449	2.124
AXDr1 [5]	4	0.69	0.015	0.009	0.084
	6	9.20	0.183	0.038	1.118
	8	50.93	1.207	0.292	1.330
AXDr2 [5]	4	9.34	0.216	0.038	0.325
	6	50.72	1.306	0.249	0.985
	8	96.44	6.355	1.366	4.015
AXDr3 [5]	4	1.58	0.030	0.006	0.127
	6	11.71	0.204	0.048	0.380
	8	48.46	0.993	0.254	1.229

Table 5. Delay, area, power, and PDP comparisons of the proposed approximate 16/8 dividers with previous state-of-the-art approximate dividers.

Design	*p*	*Delay* (ns)	*Area* (μm²)	*Power* (μW)	PDP (fJ)
Exact	-	3.63	246.17	29.2	106.00
ASC-SR	4	3.32	245.47	28.7	95.28
	6	2.95	246.52	27.8	82.01
	8	2.43	245.93	26.0	63.18
ASC-HR	4	3.20	224.64	26.0	83.20
	6	2.76	199.60	21.8	60.17
	8	2.04	165.44	16.2	33.05
AXRD-M1 [11]	4	3.35	223.00	27.0	90.45
	6	3.02	199.60	23.5	70.97
	8	2.52	157.48	18.5	46.62
AXDr1 [5]	4	3.65	248.04	29.2	106.58
	6	3.62	251.08	28.8	104.26
	8	3.59	255.29	28.0	100.52
AXDr2 [5]	4	3.68	244.53	29.5	108.56
	6	3.71	243.01	29.3	107.70
	8	3.75	241.25	29.1	109.13
AXDr3 [5]	4	3.64	247.34	29.1	105.92
	6	3.61	249.33	28.4	102.52
	8	3.58	252.02	27.2	97.38

Table 6. PSNR and SSIM of 16/8 bits approximate dividers for change detection and foreground extraction.

Design	*p*	*Change Detection*		*Foreground Extraction*	
		PSNR (dB)	SSIM	PSNR (dB)	SSIM
ABSC-SR	4	68.5	0.9999	63.8	0.9997
	6	62.5	0.9998	59.2	0.9992
	8	52.1	0.9982	47.3	0.9982
ADSC-SR	4	67.1	0.9999	58.0	0.9991
	6	58.1	0.9995	46.3	0.9924
	8	48.3	0.9957	40.6	0.9469
AXRD-M1	4	60.2	0.9996	51.8	0.9957
	6	57.7	0.9993	48.9	0.9920
	8	51.3	0.9959	42.1	0.9765
AXDr1	4	70.3	0.9999	62.2	0.9996
	6	57.4	0.9993	49.1	0.9936
	8	52.2	0.9982	42.6	0.9868
AXDr2	4	62.1	0.9998	53.9	0.9980
	6	51.9	0.9981	42.9	0.9901
	8	41.6	0.9912	35.3	0.9593
AXDr3	4	76.4	0.9999	70.5	0.9999
	6	60.9	0.9997	57.1	0.9988
	8	54.8	0.9986	46.4	0.9859

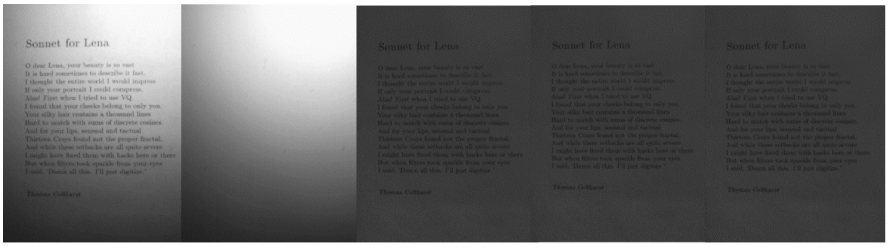


Figure 13. Foreground extraction results using (a) the original blurry input image and (b) the background noise image are shown using an (c) accurate divider, (d) ABSC with SR and $p = 4$, and (e) ADSC with SR and $p = 4$.

5.2. Foreground Extraction

A useful method for clearly viewing an image blurred by background noise or poor lighting conditions is to simply extract the foreground part of the image. This is referred to as foreground extraction and can be implemented by simply dividing the pixels of the original noisy image by a second image composed of only the background noise. This is demonstrated using two 8-bit grayscale images as illustrated in **Figure 13**.

The PSNR and SSIM of the foreground application for all designs are summarized in **Table 6** along with the change detection results. The best PSNR and SSIM design for $p = 6$ and $p = 8$ is the proposed ABSC method. Although the best PSNR and SSIM design are AXDr3 for $p = 4$, the values for ABSC-SR is the second-best design with slightly reduced PSNR and SSIM values. In addition, it can be seen that **Figure 13(d)** and **Figure 13(e)**, produced using the proposed methods, are visually similar to **Figure 13(c)**.

6. Conclusion

This paper has proposed approximate 2-D unsigned restoring divider designs, named ABSC and ADSC, based on two types of novel approximate subtractor cell designs. These approximate subtractor cells were designed to break the "borrow chain" that resulted in excessive delays and power dissipation in a traditional accurate 2-D restoring divider design. Since the proposed designs are cells, they can be applied to research on other circuits, such as square root, where the same cell can be used, and there is a possibility that they can be applied to sequential design for much longer bit operations. It can also be extended to an application-specific design by applying analysis to the data or algorithm used. If biased data are used, there will be an efficient cell arrangement accordingly, which is worth studying. Comprehensive comparisons were made with an accurate design as well as previous state-of-the-art designs. The results show that the largest ADSC 16/8 approximate divider has a Mean Relative Error Distance (MRED) of 6.30×10^{-4} to 1.91×10^{-2}, Normalized Error Distance (NED) of 2.60×10^{-4} to 1.15×10^{-2}, and Root Mean Squared Error (RMSE) of 0.185 to 2.441 with 67.2% of the area, 55.5% of the power usage, and 56.2% of the propagation delay when compared to a completely accurate design. In general, the

ABSC design had slightly better accuracy than the ADSC design. When compared to previous state-of-the-art designs, the proposed ADSC design had the best hardware characteristics, in terms of delay, area, and power usage.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Han, J. and Orshansky, M. (2013) Approximate Computing: An Emerging Paradigm for Energy-Efficient Design. *Proceedings of the* 2013, 18*th IEEE European Test Symposium* (*ETS*), Avignon, 27-30 May 2013, 1-6. https://doi.org/10.1109/ETS.2013.6569370

[2] Venkataramani, S., Chakradhar, S.T., Roy, K. and Raghunathan, A. (2015) Approximate Computing and the Quest for Computing Efficiency. *Proceedings of the* 2015*,* 52*nd ACM/EDAC/IEEE Design Automation Conference* (*DAC*), San Francisco, CA, 8-12 June 2015, 1-6. https://doi.org/10.1145/2744769.2751163

[3] Parhami, B. (2000) Computer Arithmetic: Algorithms and Hardware Designs. 2nd Edition, Oxford University Press, London.

[4] Chen, L., Han, J., Liu, W. and Lombardi, F. (2015) Design of Approximate Unsigned Integer Non-Restoring Divider for Inexact Computing. *Proceedings of the* 25*th Edition on Great Lakes Symposium on VLSI*, Pittsburgh, PA, 20-22 May 2015, 51-56. https://doi.org/10.1145/2742060.2742063

[5] Chen, L., Han, J., Liu, W. and Lombardi, F. (2016) On the Design of Approximate Restoring Dividers for Error-Tolerant Applications. *IEEE Transactions on Computers*, **65**, 2522-2533. https://doi.org/10.1109/TC.2015.2494005

[6] Jiang, H., Liu, L., Lombardi, F. and Han, J. (2018) Adaptive Approximation in Arithmetic Circuits: A Low-Power Unsigned Divider Design. *Proceedings of the* 2018 *Design*, *Automation & Test in Europe Conference & Exhibition* (*DATE*), Dresden, 19-23 March 2018, 1411-1416. https://doi.org/10.23919/DATE.2018.8342233

[7] Jiang, H., Liu, L., Lombardi, F. and Han, J. (2019) Low-Power Unsigned Divider and Square Root Circuit Designs Using Adaptive Approximation. *IEEE Transactions on Computers*, **68**, 1635-1646. https://doi.org/10.1109/TC.2019.2916817

[8] Behroozi, S., Li, J., Melchert, J. and Kim, Y. (2019) SAADI: A Scalable Accuracy Approximate Divider for Dynamic Energy-Quality Scaling. *Proceedings of the* 24*th Asia and South Pacific Design Automation Conference*, Tokyo, 21-24 January 2019, 481-486. https://doi.org/10.1145/3287624.3287668

[9] Melchert, J., Behroozi, S., Li, J. and Kim, Y. (2019) SAADI-EC: A Quality-Configurable Approximate Divider for Energy Efficiency. *IEEE Transactions on Very Large Scale Integration Systems*, **27**, 2680-2692. https://doi.org/10.1109/TVLSI.2019.2926083

[10] Jeong, J. and Kim, Y. (2021) ASADRD: Accuracy Scalable Approximate Divider Based on Restoring Division for Energy Efficiency. *Electronics*, **10**, Article 31. https://doi.org/10.3390/electronics10010031

[11] Adams, E., Venkatachalam, S. and Ko, S.B. (2019) Approximate Restoring Dividers Using Inexact Cells and Estimation from Partial Remainders. *IEEE Transactions Computers*, **69**, 468-474. https://doi.org/10.1109/TC.2019.2953751

[12] Liang, J., Han, J., Lombardi, F. and Han, J. (2013) New Metrics for the Reliability of Approximate and Probabilistic Adders. *IEEE Transactionson Computers*, **62**, 1760-1771. https://doi.org/10.1109/TC.2012.146

[13] Hore, A. and Ziou, D. (2010) Image Quality Metrics: PSNR vs. SSIM. *Proceedings of the* 20*th International Conference on Pattern Recognition*, Istanbul, 23-26 August 2010, 2366-2369. https://doi.org/10.1109/ICPR.2010.579