Combining Restoring Array and Logarithmic Dividers into an Approximate Hybrid Design

Weiqiang Liu[1], Jing Li[1], Tao Xu[1], Chenghua Wang[1], Paolo Montuschi[2], Fabrizio Lombardi[3]
[1]College of EIE, Nanjing University of Aeronautics and Astronautics, Nanjing, 211106, China, E-mail: liuweiqiang@nuaa.edu.cn
[2]CCE Department, Politecnico di Torino, Torino, 10129, Italy, E-mail: paolo.montuschi@polito.it
[3]Department of ECE, Northeastern University, Boston, MA 02115, USA, Email: lombardi@ece.neu.edu

***Abstract*—This paper proposes a new design of an approximate hybrid divider (AXHD), which combines the restoring array and the logarithmic dividers to achieve an excellent tradeoff between accuracy and hardware performance. Exact restoring divider cells (EXDCrs) are used to generate the MSBs of the quotient for attaining a high accuracy; the other quotient digits are processed by a logarithmic divider as inexact scheme to improve figures of merit such as power consumption, area and delay. The proposed AXHD is evaluated and analyzed using error and hardware metrics. The proposed design is also compared with the exact restoring divider (EXDr) and previous approximate restoring dividers (AXDrs). The results show that the proposed design achieves very good performance in terms of accuracy and hardware; case studies for image processing also show the validity of the proposed designs.**

Keywords—Approximate Computing; Logarithmic Divider; Restoring Array Divider; Low Power.

I. INTRODUCTION

Power has been one of the main challenges in integrated circuit design. Computing systems are usually designed to generate outputs with the highest precision, which despite advances in feature size, also results in high power consumption and large hardware complexity (such as implementation area). In many applications (such as image processing, machine learning and computer vision), human perception mitigates the effects of computational errors. Approximate computing has been proposed as an innovative technique for low power and high performance systems in which errors can be tolerated [1]-[3].

As basic operations of an arithmetic processor, addition and multiplication are very important for achieving high performance; therefore, they have been extensively studied for approximate computing while also reducing power consumption [4]. Error metrics including the error rate (ER), error distance (ED), mean error distance (MED), normalized mean error distance (NMED) [5] have been proposed for evaluating the designs of approximate arithmetic circuits. Approximate adders have been extensively studied in the technical literature; among the many proposed schemes, approximate adder designs include speculative [6-7] and non-speculative transistor-level full adders [8-9].

The operation of multiplication is more complex than addition. Approximate design techniques can be applied to different parts of a conventional multiplier, such as operands [10-11], partial product (PP) generation [12] and PP tree [13-14]. Approximate dividers has received less attention in the technical literature. [15] has proposed the design of an approximate unsigned non-restoring divider (AXDnr); different AXDnrs have been proposed by replacing the logic primitives with approximate subtractors. Three types of approximate subtractors cell (AXSC) are then designed at transistor level. Exact subtractor cells (EXSCs), that are critical in the operation of the array divider, are truncated or replaced by AXSCs using various schemes. The ED and MED are then used to evaluate the error characteristics of the approximate divider. Both the restoring and non-restoring array dividers have been analyzed for approximate computing; [16] has shown that an approximate unsigned restoring divider (AXDr) has better performance than AXDnr with respect to power consumption while also introducing a small degradation in accuracy. The same replacement and truncation schemes of [15] have been applied to a restoring divider; [17] has proposed designs of an approximate high-radix divider, in which an approximate signed-digit adder cell is utilized to replace the exact signed-digit adder cell. The high-radix algorithm is usually applied to sequential circuits due to the extensive usage of look-up tables. A high-radix division scheme without a look-up table has been proposed in [18]; the replacement and truncation schemes from [15] have also been used to this high-radix exact array divider.

A divider based on dynamic approximation has been proposed in [19]. As the higher order bits are more significant than lower order bits, usually an approximate divider is designed by truncating the lower bits; for different lengths of input operands, leading one detectors and a barrel shifter are utilized to reduce the inaccuracy. The lengths of the bits processed by the accurate divider are adjustable to reduce inaccuracy and power dissipation.

Approximate division by transforming the operands into the logarithmic domain can significantly save power and area; however, accuracy is rather low. Conventional array dividers are more accurate but they dissipate more power; therefore, this paper proposes an approximate hybrid divider (AXHD) based on combining the restoring array and the logarithmic dividers. Exact restoring divider cells (EXDCrs) are used to correct the errors generated by conventional non-iterative logarithmic

dividers. The proposed divider designs can achieve excellent results for specific applications.

The paper is organized as follows. Both conventional restoring array dividers and conventional non-iterative logarithmic are reviewed in Section 2. Section 3 presents the proposed design of AXHD. Error analysis and hardware evaluation are provided in Section 4. Comparison with the exact restoring array dividers and other approximate dividers is also provided in this section. The application of the proposed approximate divider to image processing is presented in Section 5. The conclusion is provided in Section 6.

II. Review

A. Conventional Restoring Array Divider

Let A and B to be the two operands of the integer division, and the results of division are the quotient Q and the remainder R. A is then given by:

$$A = B\,Q + R \tag{1}$$

where, the dividend A and the remainder R have the same sign and |R|<|B| [20]. The restoring divider is a widely used type of array divider. The trial subtraction is performed in each row; the corresponding quotient digit is 1 (0) if the trial difference is positive (negative). If the quotient digit is 1, the trial difference is moved as the partial remainder to the next row; otherwise, the partial remainder is immediately dropped. The exact restoring divider cell (EXDCr) and an 8 by 4 exact restoring divider (EXDr) are shown in Fig. 1 and Fig. 2, respectively [16]. It is worth observing that A conventional restoring divider overflows when A is large and B is small, while the logarithmic divider does not.

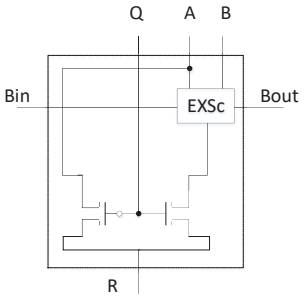


Fig. 1 An exact restoring divider cell [16].

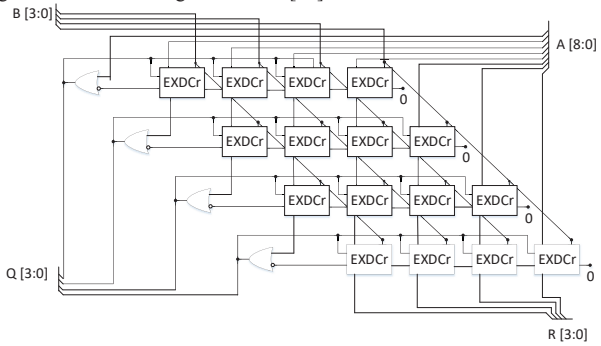


Fig. 2 A conventional unsigned 8 by 4 restoring array divider [16].

B. Non-Iterative Logarithmic Divider

The operands of a logarithmic divider are given by the dividend *A* and the divisor *B*, *A* and *B* are both positive integers, different from zero, as the cases where either *A*, or *B* or both are zero are usually handled separately. k is the exponent, and x is the fractional part of the mantissa. They are expressed as follows:

$$A = 2^{k_1}(1 + x_1) \tag{2}$$

$$B = 2^{k_2}(1 + x_2) \tag{3}$$

where, k_1 and k_2 are the so-called characteristics of *A* and *B*, respectively and represent the position of the leading most significant bits. x_1 and x_2 are in the range of [0,1). The logarithm of a quotient, i.e. *Q*, is equal to the difference of the logarithms of the dividend and divisor.

$$Q = \frac{A}{B} = 2^{k_1 - k_2}\frac{1 + x_1}{1 + x_2} \tag{4}$$

$$\log_2 Q = k_1 - k_2 + \log_2(1 + x_1) - \log_2(1 + x_2) \tag{5}$$

As $\log_2(1 + x) \approx x$ when $0 \le x < 1$, the approximate quotient can be expressed as follows:

$$\log_2 Q \approx k_1 - k_2 + x_1 - x_2 \tag{6}$$

Therefore, (6) can be calculated by subtractions. There are four steps in the conventional logarithmic divider proposed by Mitchell [10]: leading one detection, binary-to-logarithm conversion, mantissa subtraction and logarithm-to-binary conversion. The leftmost one bit is detected by the leading one detector (LOD). The binary-to-logarithm converter (BLC) converts the input operands into logarithms. Only the most significant bit is converted to logarithm, the logarithm of the mantissa is approximated by itself. The division is performed by a mantissa subtraction by the Subtractor in Fig. 3 in the logarithmic domain. The logarithm-to-binary converter (LBC) converts the subtraction result back to a binary number as the final quotient. The logarithmic divider is shown in block form in Fig. 3 [10].

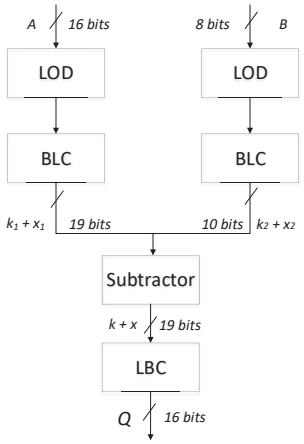


Fig. 3 A 16 by 8 unsigned non-iterative logarithmic divider [10].

III. Proposed Approximate Hybrid Divider

A logarithm divider consumes significantly less power and requires less chip area than a conventional array divider [10] because the rows used to perform the subtraction operations consume more hardware resources and area compared with the simple subtraction in logarithm divider; however, when the operand width is large, the accuracy decreases due to the approximation in the logarithmic conversion. Therefore, a new

approximate hybrid design (AXHD) combining an exact restoring and logarithmic dividers is proposed in this work.

As mentioned above, the conventional restoring divider will overflow when X is large and Y is small, while the logarithmic divider won't. The proposed AXHD combines the array and the logarithmic dividers as shown, at block level, in Fig. 4. Extra rows are added to the array to avoid overflowing by computing more trial subtractions. For a 16 by 8 EXDr, 8 extra rows are added, in which both the quotient and the remainder are 16-bit wide. Its design principle is to use the exact division for the most significant bits and an approximate logarithmic division for the remaining bits. The trade-off is between accuracy, area, speed and power.

In the proposed design of this approximate divider, p rows of EXDCrs are used to calculate the most significant quotient digits to improve the accuracy; p should be selected according to the accuracy requirement of a specific application and therefore, 16-p is the replacement depth. The partial remainder generated by EXDCrs is then computed by the logarithmic divider for the remaining quotient digits; p rows of EXDCrs make up an exact divider whose input dividend is p-bit wide. In particular, p=0 corresponds to a full logarithmic divider and p=16 corresponds to an exact restoring array divider. If p is sufficiently small, then a very simple hardware can be expected for implementing the exact division part. Prior to considering the 16 by 8 AXHD implementation, we explore the feasibility of our design by considering first an 8 by 4 AXHD. Fig. 5 depicts the scheme for p=2. The first two lines of the EXDC blocks correspond to p=2 and the rightmost cascaded part is shown Fig. 3, which is drawn now for the case of 8 by 4.

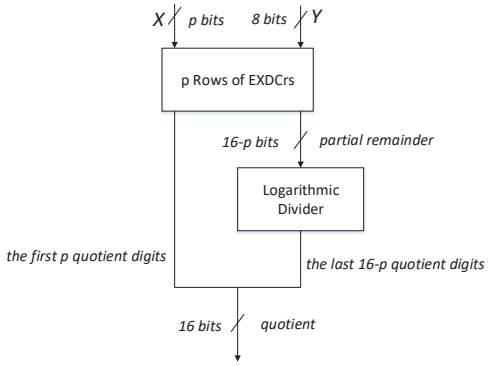


Fig. 4 16 by 8 AXHD based on restoring and logarithmic dividers.

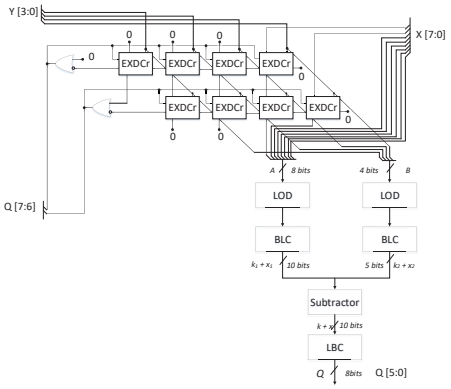


Fig. 5. 8 by 4 AXHD based on restoring and logarithmic dividers for p=2.

The length of the quotient digits required by EXDCrs can be adjusted by the number of EXDCr rows, *i.e., p*. A large p results in more accurate results at the cost of both a higher power consumption and hardware complexity. As a logarithmic divider is used for the last quotient bits, the remainder is not available. However, the computation of the remainder is naturally not in the purposes of approximate division [15].

Fig. 6 shows a detailed example of the proposed 8 by 4 AXHD. The dividend is 156 and the divisor is 3 with p =2. The final result shows that the difference between the exact quotient (*i.e.*, Q=52) and the approximate quotient (*i.e.*, Q=55) is only 3.

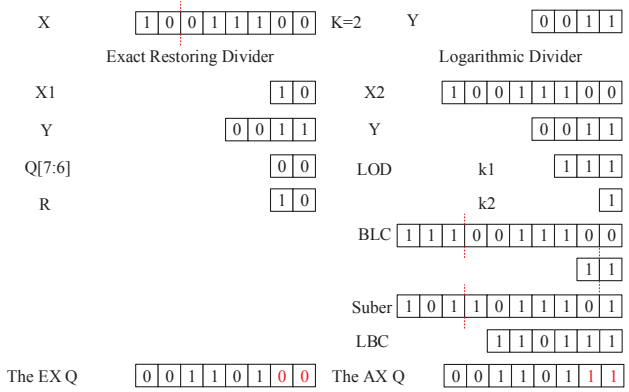


Fig. 6 An example of the proposed 8 by 4 AXHD.

IV. Error Analysis and Hardware Evaluation

The results of the normalized mean error distance (NMED), power, delay, area and PDP are provided in this section. The NMED is defined as the MED normalized by the maximum output of the accurate design. The error results are generated using exhaustive simulation. The NMED power product (NMPP) are also provided to evaluate the tradeoff between power and accuracy.

The proposed AXHD design is compared with [16], in which the rows of EXDCrs are replaced by approximate restoring divider cells (AXDCrs) by four replacement schemes, *i.e.*, vertical replacement (VR), horizontal replacement (HR), square replacement (SR) and triangle replacement (TR). The approximate restoring dividers with HR and TR schemes have a better performance than those with VR and SR schemes [16]. Furthermore, the structure of the HR scheme is similar to the proposed scheme. Therefore, the proposed divider is compared with the three approximate restoring array dividers (AXDrs) with HR and TR schemes (i.e. AXDr1, AXDr2 and AXDr3) of [16] and the conventional exact restoring divider. These divider have been assessed by utilizing various replacement depths.

A. Error Analysis

TABLE I. NMED (10^{-6}) OF AXHD AND AXDRS.

Depth	2	4	6	8	10	12	14	16
AXHD	0.32	2.17	8.99	33.5	178.3	282	359	377
AXDr1 HR	1.07	9.05	40.8	160	423.0	630	1,443	1,443
AXDr2 HR	7.86	33.6	137	566	2,747	13,747	122,187	508,714
AXDr3 HR	1.01	8.64	37.9	146	370.2	590	1,537	1,635
AXDr1 TR	0.00	0.12	1.44	9.11	40.70	152	450	721
AXDr2 TR	0.19	1.39	6.61	27.6	107.4	393.2	1341	4256
AXDr3 TR	0.02	0.36	2.20	10.2	41.9	155	496	818

AXHD can calculate the quotient only; therefore, the NMED of the integer quotient is provided in Table I. The NMEDs of AXHD at different approximation depth are shown in Figs. 7-8; a comparison with AXDrs using HR and TR schemes is also provided. As the error of HR and TR grows exponentially by increasing the replacement depth, a logarithmic axis is used in the figures.

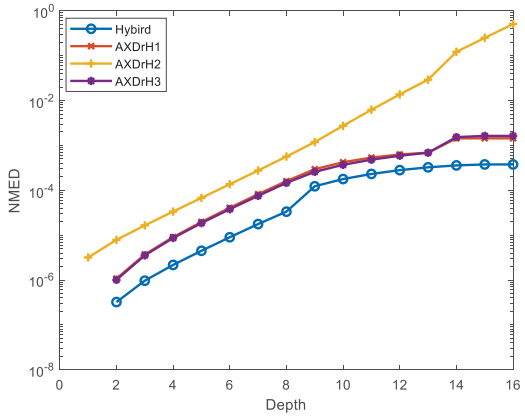


Fig. 7 The NMED of the proposed AXHD and AXDrs using HR schemes for various replacement depth.

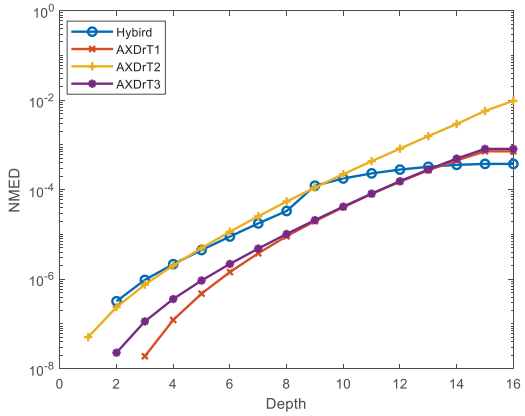


Fig. 8 The NMED of the proposed AXHD and AXDrs using TR schemes for various replacement depth.

The proposed hybrid divider has a smaller NMED compared with previous AXDrs using HR [16] at the same replacement depth. Compared with AXDrs with a TR scheme at a small depth, the NMED of the proposed AXHD is slightly larger compared with AXDrs with a TR scheme. However, as the replacement depth is increased, the NMED of the AXHD increases slowly and is lower than AXDrs when the replacement depth is larger than 12.

B. Hardware Evaluation

The proposed design and AXDrs are described at gate-level in Verilog HDL and verified by Synopsys VCS. All designs are then synthesized by the Synopsys Design Compiler using the NanGate 45 nm Open Cell Library. The average power consumption is found using the Synopsys Power Compiler with a back annotated switching activity file generated from the random input vectors. The critical path delay, area, power consumption and PDP are reported in this section at different values for replacement depth.

TABLE II. POWER (μW) OF AXHD AND AXDRS.

Depth	2	4	6	8	10	12	14	16
AXHD	2675	2130	1724	1279	885	712	616	500
AXDr1 HR	1930	1867	1785	1657	1672	1623	1520	1506
AXDr2 HR	1922	1869	1752	1720	1785	1919	1959	2059
AXDr3 HR	1833	1701	1504	1348	1201	1141	1086	1032
AXDr1 TR	1953	1953	1936	1832	1765	1645	1572	1545
AXDr2 TR	1960	1941	1892	1817	1747	1688	1706	1755
AXDr3 TR	1969	1876	1789	1595	1411	1288	1151	1060

Table II reports the power consumption of the AXHD and AXDrs. As shown in Figs. 9-10, by increasing the replacement depth, the power of AXHD decreases nearly linear. When the replacement depth is larger than 8, the proposed design performs better than AXDrs [16]. However, when the replacement depth is smaller than 5, AXHD consumes more power than EXDrs. When the replacement depth is smaller than 8, AXDrs outperform AXHD in terms of power consumption.

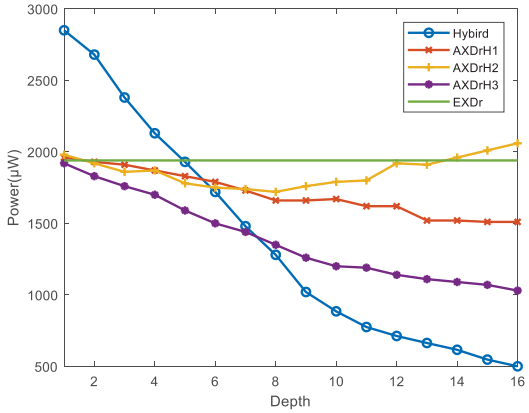


Fig. 9 Power of the proposed AXHD, EXDr, and AXDrs using HR schemes at various replacement depths.

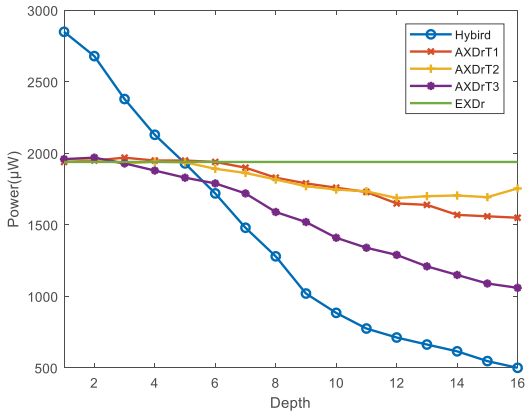


Fig. 10 Power of the proposed AXHD, EXDr, and AXDrs using TR schemes at various replacement depths.

Table III shows the delay results of AXHD and AXDrs. As shown in Figs. 11-12, the delay of AXHD decrease linearly; the

AXDrs with HR or TR schemes have no significant improvement in terms of delay. In the proposed design, the delay has been improved significantly when the replacement depth is large because the delay of the logarithmic divider is small. When the replacement depth is small, the delay is mainly determined by the EXDCrs.

TABLE III. DELAY (ns) OF AXHD AND AXDRS.

Depth	2	4	6	8	10	12	14	16
AXHD	9.01	7.83	6.75	5.69	4.44	3.24	2.2	1.44
AXDr1 HR	8.62	8.5	8.59	8.56	8.54	8.55	8.63	8.61
AXDr2 HR	8.43	8.47	8.39	8.26	8.3	8.34	8.09	8.19
AXDr3 HR	8.51	8.46	8.3	8.25	8.14	8.13	8.23	8.02
AXDr1 TR	8.84	8.89	8.85	8.76	8.84	8.76	8.75	8.78
AXDr2 TR	8.8	8.85	8.74	8.79	8.67	8.57	8.6	8.56
AXDr3 TR	8.86	8.84	8.79	8.75	8.74	8.57	8.5	8.33

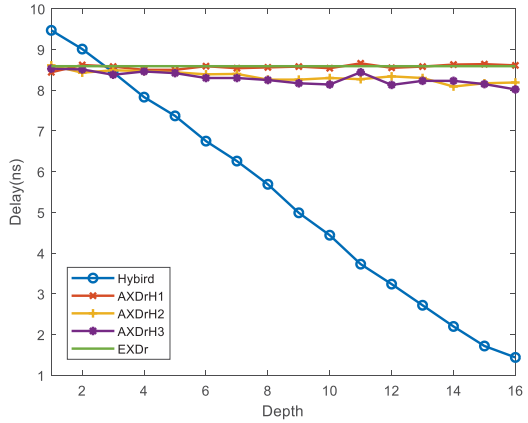


Fig. 11 Delay of the proposed AXHD, EXDr, and AXDrs using HR schemes at various replacement depths.

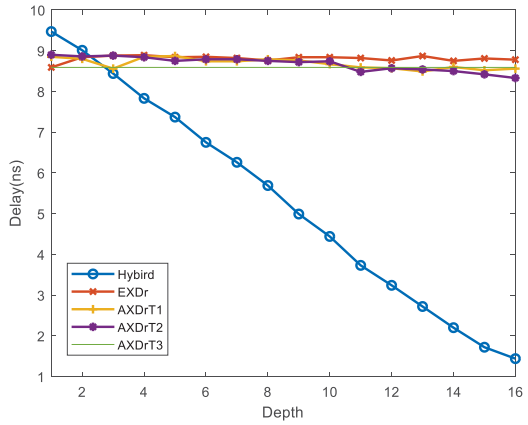


Fig. 12 Delay of the proposed AXHD, EXDr, and AXDrs using TR schemes at various replacement depths.

The area results are provided in Table IV. It can be seen in Figs. 13-14 that the area of the AXHD decreases steadily with an increase of the replacement depth, which is again much smaller than AXDrs when the replacement depth is larger than 8. When the replacement is smaller than 6, the AXHD has a larger area than EXD and when the replacement is smaller than 8, AXDrs have a smaller area.

TABLE IV. AREA (μM^2) OF AXHD AND AXDRS.

Depth	2	4	6	8	10	12	14	16
AXHD	1694	1523	1361	1184	928	775	631	433
AXDr1 HR	1324	1310	1293	1278	1283	1266	1245	1207
AXDr2 HR	1294	1231	1163	1096	1033	979	899	843
AXDr3 HR	1292	1252	1198	1145	1099	1063	995	945
AXDr1 TR	1362	1362	1364	1340	1334	1317	1304	1303
AXDr2 TR	1363	1328	1277	1219	1158	1085	1020	967
AXDr3 TR	1359	1316	1290	1224	1153	1126	1074	1025

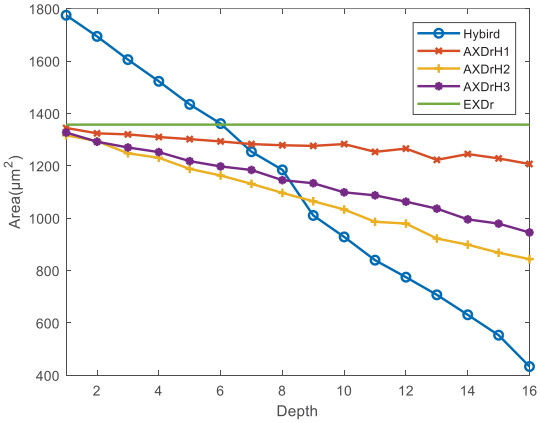


Fig. 13 Area of the proposed AXHD, EXDr, and AXDrs using HR schemes at various replacement depths.

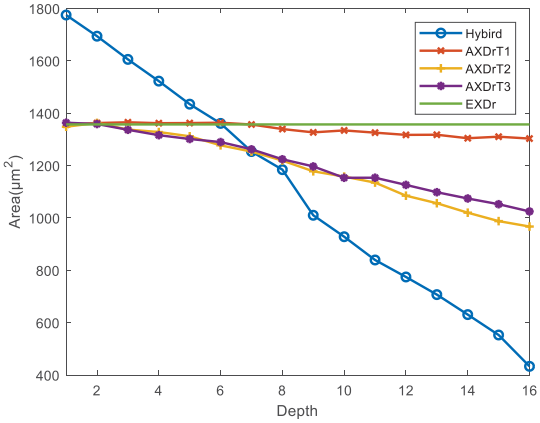


Fig. 14 Area of the proposed AXHD, EXDr, and AXDrs using TR schemes at various replacement depths.

The power delay product (PDP) is used to evaluate the overall hardware performance of the designs (Table V). As shown in Fig. 15-16, AXHD has significant advantages when the replacement depth is larger than 6. AXDr3 with a TR scheme is close to the proposed design. When the replacement depth is smaller than 4, EXDr and AXDrs outperform AXHD.

TABLE V. PDP (fJ) OF AXHD AND AXDRS.

Depth	2	4	6	8	10	12	14	16
AXHD	24102	16675	11636	7280	3929	2308	1355	721
AXDr1 HR	16640	15871	15334	14186	14281	13872	13118	12968
AXDr2 HR	16204	15828	14697	14207	14817	16007	15850	16862
AXDr3 HR	15601	14390	12485	11118	9775	9276	8935	8277
AXDr1 TR	17265	17361	17134	16050	15601	14414	13752	13567
AXDr2 TR	17248	17181	16539	15972	15145	14465	14668	15020
AXDr3 TR	17444	16579	15724	13953	12332	11042	9780	8828

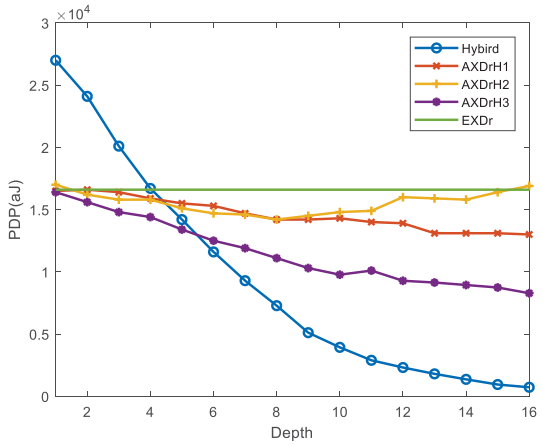


Fig. 15 PDP of the proposed AXHD, EXDr, and AXDrs using HR schemes at various replacement depths.

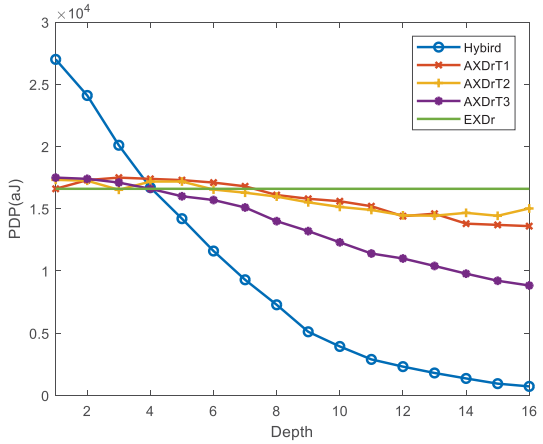


Fig. 16 PDP of the proposed AXHD, EXDr, and AXDrs using TR schemes at various replacement depths.

C. *NMED Power Product (NMPP)*

The NMPP is also used to evaluate the trade-off between accuracy and power consumption (Table VI). As shown in Fig. 17 the proposed AXHD shows a better performance under this metric than AXDrs with a HR scheme for all replacement depths. However (Fig. 18) only when the replacement depth is larger than 12, AXHD is better than AXDrs under the TR scheme.

So under all previous comparative metrics, AXHD has advantages in terms of accuracy and hardware performance when the replacement depth is larger, so the proposed design is suitable for approximate dividers with large bit-width operands.

TABLE VI. NMPP (nW) RESULTS OF AXHD AND AXDRS.

Depth	2	4	6	8	10	12	14	16
AXHD	0.86	4.62	15.5	42.8	157.8	200.8	221.4	188.8
AXDr1 HR	2.07	16.9	72.9	266	707.5	1021.8	2192.8	2172.6
AXDr2 HR	15.1	62.8	239	974	4905	26384.2	239389.2	1047341
AXDr3 HR	1.85	14.7	57.1	197	444.5	672.9	1668.8	1687.4
AXDr1 TR	0	0.24	2.8	16.7	71.9	249.3	707.9	1114.5
AXDr2 TR	0.38	2.7	12.5	50.2	187.6	663.7	2286.4	7467.5
AXDr3 TR	0.05	0.68	3.9	16.3	59.1	199.5	571.1	866.4

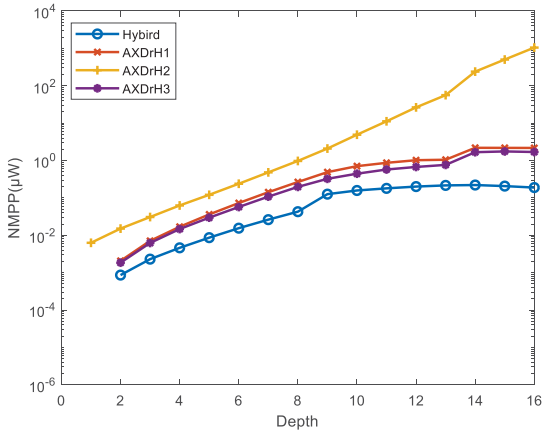


Fig. 17 NMPP of the proposed AXHD and AXDrs using HR schemes at various replacement depths.

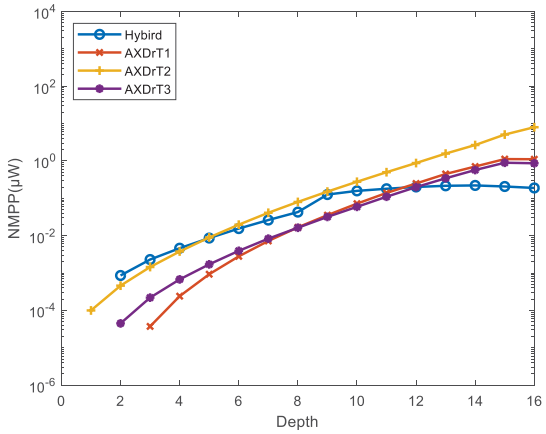


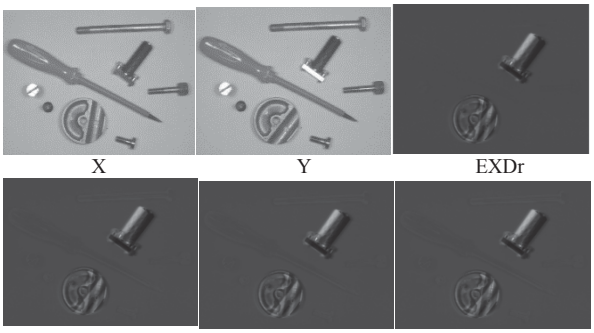
Fig. 18 NMPP of the proposed AXHD and AXDrs using TR schemes at various replacement depths.

V. APPLICATIONS

In this section, two applications of image processing involving pixel division (only for quotient calculation, namely, change detection and background removal) are studied using the proposed AXHD. 16-to-8 AXHD and EXDr are used to compute the inputs (X and Y) of 8-bit grayscale images (Figs. 19 and 20) with replacement depths of 3, 6 and 9. To implement the pixel division operation in an integer divider, the dividend is multiplied by a constant for the mean value of the output image to be equal to the mean value of the example image prior to division. The peak signal-noise ratio (PSNR) at different replacement depths is provided (Tables VII and VIII) to show the difference between exact and approximate results.

TABLE VII. PSNR OF AXHD WITH DIFFERENT REPLACEMENT DEPTHS FOR CHANGE DETECTION.

Depth	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
PSNR /dB	Infinite	Infinite	84.69	64.99	61.32	54.17	39.87	39.71
Depth	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
PSNR /dB	39.71	39.71	39.71	39.71	39.71	39.71	39.71	39.71



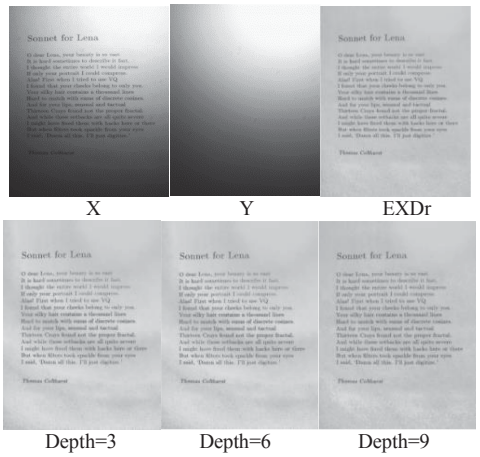
X Y EXDr

Depth=3 Depth=6 Depth=9

Fig. 19 Change detection using EXDr and AXHDs.

TABLE VIII. PSNR OF AXHD WITH DIFFERENT REPLACEMENT DEPTHS FOR BACKGROUND REMOVAL.

Depth	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
PSNR /dB	Infinite	Infinite	74.17	62.95	56.07	50.32	43.04	27.12
Depth	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
PSNR /dB	27.12	27.12	27.12	27.12	27.12	27.12	27.12	27.12



X Y EXDr

Depth=3 Depth=6 Depth=9

Fig. 20 Background removal using EXDr and AXHDs.

For both applications, there is no loss when the replacement depth is 1 and 2; the PSNR decreases when the replacement depth increases. When the replacement depth is larger than 7, the PSNRs converge to 39.71 and 27.12 for change detection and background removal respectively, as the input dividend is dropped totally into the logarithmic divider.

VI. CONCLUSION

The designs of hybrid approximate dividers (AXHDs) have been presented in this paper. The non-iterative logarithmic divider is combined with a conventional restoring array divider, because it has the advantage of low power consumption. By replacing the last rows of exact cells (EXDrs) by the logarithmic divider, a very good tradeoff between accuracy and hardware performance (*i.e.* power delay and area) is achieved. The proposed dividers are analyzed with error and hardware metrics; the results show that the proposed AXHD design performs better than other approximate dividers, especially when the replacement depth is large. Case studies for two image processing applications confirm the validity of the proposed AXHD design.

REFERENCES

[1] J. Han and M. Orshansky, "Approximate computing: an emerging paradigm for energy-efficient design," in *Proc. 18th IEEE European Test Symposium (ETS)*, 2013, pp.1-6.

[2] S. Venkataramani, S. Chakradhar, K. Roy and A. Raghunathan, "Approximate computing and the quest for computing efficiency," in *Proc. 52nd Annual Design Automation Conference (DAC)*, 2015, Article 120, 6 pages.

[3] Q. Xu, N.-S. Kim and T. Mytkowicz, "Approximate Computing: A Survey," *IEEE Design & Test*, vol. 33, no.1, pp. 8-22, 2016.

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

[5] J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders," *IEEE Trans. Computers*, vol. 63, pp. 1760-1771, 2013.

[6] S.-L. Lu, "Speeding up processing with approximation circuits," *Computer*, vol. 37, pp. 67-73, 2004.

[7] N. Zhu, W. L. Goh, W. Zhang, K. S. Yeo and Z. H. Kong, "Design of low-power high-speed truncation error tolerant adder and its application in digital signal processing," *IEEE Trans. VLSI Syst*., vol. 18, pp.1225-1229, 2010.

[8] H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie and C. Lucas, "Bio-inspired imprecise computational blocks for efficient VLSI implementation of soft-computing applications," *IEEE Trans. Circuits Syst.: Part I Regular Papers*, vol. 57, pp. 850-862, 2010.

[9] V. Gupta, D. Mohapatra, A. Raghunathan and K. Roy, "Low power digital signal processing using approximate adders," *IEEE Trans. CAD*, vol. 32, pp. 124-137, 2013.

[10] J. Mitchell, "Computer multiplication and division using binary logarithms", *IRE Trans. Electron. Comput.*, vol. 11, pp. 512-517, 1962.

[11] S. Hashemi, R. Bahar and S. Reda, "DRUM: A dynamic range unbiased multiplier for approximate applications," in *Proc. IEEE/ACM Int. Conf. Computer Design (ICCD),* 2015, pp. 418 - 425.

[12] W. Liu, L. Qian, C. Wang, H. Jiang, J. Han and F. Lombardi, "Design of approximate radix-4 Booth multipliers for error-tolerant computing," *IEEE Trans. Computers*, vol. 66, pp.1435-1441, Aug. 2017.

[13] Y.-H. Chen and T.-Y. Chang, "A high-accuracy adaptive conditional probability estimator for fixed-width Booth multipliers," *IEEE Trans. Circuits Syst. I: Reg. Papers*, vol. 59, pp. 594-603, 2012.

[14] G. Zervakis, K. Tsoumanis, S. Xydis, N. Axelos and K. Pekmestzi, "Approximate multiplier architectures through partial product perforation: power-area tradeoffs analysis," in *Proc. ACM Great Lake Symp. VLSI*, 2015, pp. 229-232.

[15] L. Chen, Jie Han, W. Liu, and F. Lombardi, "Design of approximate unsigned integer non-restoring divider for inexact computing." *In Proc. Great Lakes Symp. VLSI*, pp. 51-56, 2015.

[16] L. Chen, Jie Han, W. Liu, and F. Lombardi, "On the design of approximate restoring dividers for error-tolerant applications." *IEEE Trans. Computers*, vol. 65, pp. 2522-2533, 2016.

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

[18] T. Aoki, K. Nakazawa, and T. Higuchi. "High-radix parallel VLSI dividers without using quotient digit selection tables." In *Proc. 30th IEEE Int. Symp. Multiple-Valued Logic (ISMVL)*, pp. 345-352, 2000.

[19] S. Hashemi, R. Bahar, and S. Reda. "A low-power dynamic divider for approximate applications." In *Proc. 53rd Annual Design Automation Conference*, p. 105, 2016.

[20] B. Parhami, Computer Arithmetic: Algorithms and Hardware Designs: Oxford University Press, 2000.