IET Circuits, Devices & Systems

Research Artical

Journals
The Institution of Engineering and Technology

Design, Evaluation and Application of Approximate-Truncated Booth Multipliers

ISSN 1751-8644
doi: 0000000000
www.ietdl.org

Yuying Zhu[1], *Weiqiang Liu*[1*], *Peipei Yin*[1], *Tian Cao*[2], *Jie Han*[3], *Fabrizio Lombardi*[4]

[1] *College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, 211106 China*
[2] *MediaTek (MTK), Shanghai, China*
[3] *Department of Electrical and Computer Engineering, University of Alberta, Alberta, Canada*
[4] *Department of Electrical and Computer Engineering, Northeastern University, Boston, USA*
* *E-mail: liuweiqiang@nuaa.edu.cn*

Abstract: Approximate computing provides a promising way to achieve low power design at the cost of acceptable error. As a core component in a processor, the performance of the multiplier is important. This paper presents designs of approximate-truncated Booth multipliers (ATBMs) using proposed approximate modified radix-4 Booth encoders (AMBEs), approximate 4:2 compressors (ACs) and gradually truncated partial products. The accuracy of the ATBMs is adjustable with the so-called approximation factors that indicate the number of AMBEs and ACs used. The normalized mean error distance (NMED) and the product of the power and delay (PDP) are used to evaluate the error and the hardware performance of the multipliers. The results show that the proposed ATBMs outperform previous approximate Booth multipliers. Their validity is also shown with case studies of image processing, K-means clustering and handwritten digit recognition.

1 Introduction

With the continuous development of integrated circuits, power consumption has become a key issue that restricts the performance of digital integrated circuits [1]. It is very difficult to further improve the power consumption under the constraint of perfect accuracy. However, for a number of applications related to human perception, such as multimedia signal processing, wireless sensor networks, machine learning and pattern recognition, errors can be tolerated to a rather large extent. On the premise of not affecting the usability of the results, approximate computing [2, 3] has been proposed to reduce the power consumption and improve the performance of computing at the cost of acceptable errors. In those error tolerant applications, the approximate design can effectively reduce the power consumption and still produce reasonable results [4].

As important arithmetic units both approximate adders and multipliers have been studied quite extensively [5]. New metrics including error distance (ED), mean error distance (MED) and normalized error distance (NED) have been proposed for evaluating the approximate designs [6].

The variable latency speculative adder in [8] is an early design of approximate adders. The main idea is to generate each bit in an n-bit adder by k lower significant bits ($k < n$), so the delay of the adder is $log(k)$ in a carry lookahead adder. Different approximate segmenting adders are proposed in [9–11], where a n-bit adder is truncated into k-bit blocks so that the critical path is reduced. As the carry bit is calculated in parallel, the speed is improved significantly. Several approximate carry-select adders are designed in [12–16].

The structure of multipliers is more complex than that of adders. However, the design of multipliers can be regarded as the process of generating the final product by repeated summation of partial products (PPs). In this process, the operation of multipliers can be divided into three steps: PP generation, PP compression and final product summation.

Design of an approximate multiplier directly using approximate adders does not significantly improve the performance because the main computations in the multiplier are determined by the PP generation and the PP compression, which are commonly performed by encoders and compressors. Several designs of approximate multipliers are proposed in [17–19]. The main idea is that the module with a lower weight is made into a constant, and the truncated part is compensated. In [17], an approximate array multiplier is designed for neural networks. The multiplier ignores the PPs of the less significant bits and reduces the number of coding units and compressor units. [18] proposes a new high-speed and floating-point approximate multiplier. A 2×2 multiplier module is simplified in [19], which is used to build larger multipliers. [20] proposes approximate radix-8 Booth multipliers by approximately computing PPs. [23] proposes two efficient 4:2 approximate compressors. [22] proposes two approximate Booth encoders. [24] and [30] propose the design of approximate redundant binary multiplier. [7] proposes two 4×4 multipliers designed with different accuracies, which are used as building blocks for scaling up to 16×16 and 32×32 multipliers.

In this paper, we propose improved designs of approximate Booth multipliers, which use two approximate Booth encoders and two approximate 4:2 compressors with truncated parts. The proposed approximate-truncated multipliers are also applied to image processing and K-means clustering. The main contributions are summarized as follows:

- Two new approximate Booth encoders with low complexity are proposed;
- Two new 4:2 compressors with high accuracy are proposed;
- The proposed approximate modules are combined with truncated PP arrays for a better performance;
- Circuit-level designs for the proposed ATBMs are provided to evaluate the NMED and PDP;
- Comprehensive comparisons in case studies for three applications $i.e.$, image processing, K-means clustering and handwritten digit recognition are provided.

The rest of the paper is organized as follows: Section 2 reviews conventional Booth multipliers. Section 3 presents the design of approximate-truncated Booth multipliers (ATBMs), including the approximate Booth encoders and approximate 4:2 compressors. It also provides the error analysis and hardware evaluation of the proposed approximate multipliers. Section 4 presents the comparisons of the proposed multipliers with the state-of-the-art designs. Section 5 presents the case studies of image processing, K-means clustering and handwritten digit recognition. Section 6 concludes the paper.

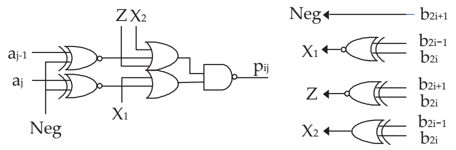

Fig. 1: MBE scheme: encoder and decoder [21].

Table 1 K-Map of MBE

$a_j\ a_{j-1}$ \ $b_{2i+1}\ b_{2i}\ b_{2i-1}$	000	001	011	010	110	111	101	100
00	0	0	0	0	1	0	1	1
01	0	0	1	0	1	0	1	0
11	0	1	1	1	0	0	0	0
10	0	1	0	1	0	0	0	1

2 Review of Conventional Booth Multipliers

This section reviews the conventional radix-4 Booth multiplier. The design of Booth multipliers mainly includes Booth encoder to generate the PPs, PP compression module and the final fast adder.

2.1 Modified Booth Encoding

In the designs of high-speed multiplier, modified Booth encoding (MBE) can reduce the number of partial products by half [21]. Suppose that the inputs of the multiplier are multiplicand A, where A=$a_N a_{N-1}...a_0$, and multiplier B, where B=$b_N b_{N-1}...b_0$. The multiplicand and multiplier are signed numbers and a_N, b_N are the signed bit. The original output is the product P with $2N$ bits. By dividing the multiplier into group $\{b_{2i+1}, b_{2i}, b_{2i-1}\}$, the Booth decoder selects 2A, A, 0, -A, -2A to generate PP rows. MBE reduces the number of PP rows from N to $N/2+1$.

Radix-4 Booth encoding is the logic circuit for PP selection. The circuit diagrams of the radix-4 Booth encoder and decoder are shown in Fig. 1 [21]. The encoder uses a 3-bit group to generate the Neg, X_1, X_2 and Z signals, which can be expressed as:

$$Neg = b_{2i+1} \tag{1}$$

$$X_1 = \overline{b_{2i-1} \oplus b_{2i}} \tag{2}$$

$$X_2 = b_{2i-1} \oplus b_{2i} \tag{3}$$

$$Z = \overline{b_{2i+1} \oplus b_{2i}} \tag{4}$$

Neg, X_1, X_2 and Z are encoded by three adjacent multiplier bits $\{b_{2i+1}, b_{2i}, b_{2i-1}\}$. Neg is a compensation bit. When the encoding result is -2A, -A and -0, the multiplicand needs to be complemented; so the compensation bit Neg is required for the complement of the negative multiplicand. When the encoding result is +2A, +A and +0, the complement of the multiplicand is the same as the original representation, and the Neg is 0. The Z signal is to prevent the left shift of the multiplicand when PP is selected as 0 and -0.

pp_{ij} is a PP in ith line and jth column, which is logically composed of multiplicand $\{a_j, a_j\}$ and multiplier $\{b_{2i+1}, b_{2i}, b_{2i-1}\}$. The logical expression of pp_{ij} is given by:

$$\begin{aligned} pp_{ij} =& (b_{2i} \oplus b_{2i-1})(b_{2i+1} \oplus a_j) + \\ & \overline{(b_{2i} \oplus b_{2i-1})}(b_{2i+1} \oplus b_{2i})(b_{2i+1} \oplus a_{j-1}) \end{aligned} \tag{5}$$

The Karnaugh-map (K-map) of PP is shown in Table 1 and the PP array of a conventional 8×8 radix-4 Booth multiplier is shown in Fig. 2.

2.2 Regular Partial Product Array

The regular PP array is based on the PP array encoded by the modified Radix-4 Booth. The number of PP rows of modified Radix-4 Booth has $N/2-1$ fewer than that of the non-Booth multiplier. The compensation bit Neg in the last row makes the PP arrays irregular. To ensure a precise final product, the design of the exact multiplier requires an extra compression level for the bit, which means that more compressors and longer critical paths are required [22].

In order to design a more regular PP array, the design of the approximate radix-4 Booth multiplier can directly truncate the compensation bit Neg in line $(N/2+1)$ (Fig. 3). The 8-bit multiplier, for example, ignoring the compensation of fifth lines, saves 16 compressors, which improves area and delay of the approximate multiplier significantly.

2.3 4-2 Compressor

As a basic unit of multiplier design, the compressor is directly related to the overall performance of the multiplier. The commonly used compressor structure is the Wallace structure composed of 4-2 compressors. The principle of the 4-2 compressor is to use two full adders in series [23], and its block diagram is shown in Fig. 4.

An 4-2 compressor has five inputs (P_1, P_2, P_3, P_4, *Carry* and C_{in}) and three outputs (*Sum*, *Carry* and C_{out}). The compressor compresses the four rows of PP, *i.e.*, P_1, P_2, P_3, P_4 into the two rows of PP, *i.e.*, Sum and C_{out}. The outputs of the 4-2 compressor can be expressed as:

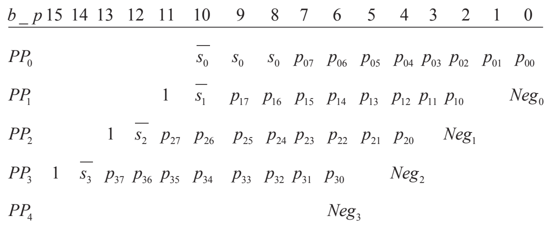

Fig. 2: PP array of an 8×8 Booth multiplier.

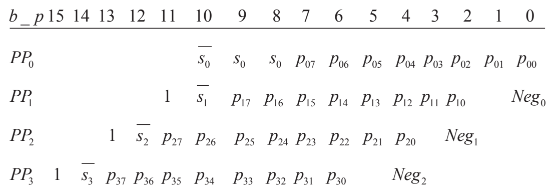

Fig. 3: Approximate regular PP array of an 8×8 Booth multiplier [22].

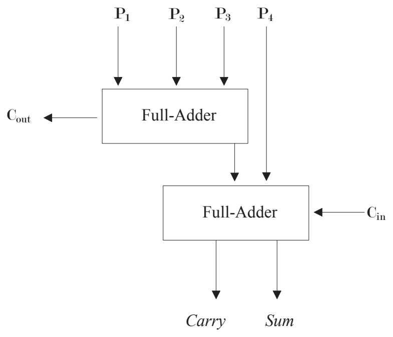

Fig. 4: Block diagram of 4-2 compressor [23].

$$Sum = P_1 \oplus P_2 \oplus P_3 \oplus P_4 \oplus C_{in} \tag{6}$$

$$C_{out} = (P_1 \oplus P_2)P_3 + \overline{(P_1 \oplus P_2)}P_4 \tag{7}$$

$$Carry = P_1 \oplus P_2 \oplus P_3 \oplus P_4 \oplus C_{in} + \overline{P_1 \oplus P_2 \oplus P_3 \oplus P_4}P_4 \tag{8}$$

Table 2 K-Map of AMBE-a

$a_j\ a_{j-1}$ \ $b_{2i+1}\ b_{2i}\ b_{2i-1}$	000	001	011	010	110	111	101	100
00	0	0	0	0	1	0	1	1
01	0	0	⓪	0	1	0	1	①
11	①	1	⓪	1	0	0	0	0
10	①	1	0	1	0	0	0	⓪

2.4 Error Metrics

For approximate designs, several metrics have been proposed to measure the error of approximate adders and multipliers including the maximum absolute error (MAE), the relative error distance (RED), and the mean error distance (MED) [6]. Three main error metrics, ($i.e.$, NMED, MAE and MRED) are used in this work to compare the design of approximate multipliers.

- NMED is defined as the normalized MED by the maximum output of the accurate design.
- MAE is the maximum absolute error of an approximate multiplier, which is used to measure error magnitude.
- MRED is the mean value of RED, which is used to evaluate the error distribution of approximate multipliers.

3 Design of Approximate Truncated Booth Multipliers (ATBMs)

In this section, the ATBMs along with two approximate Booth encoders, two approximate 4-2 compressors, an approximate regular PP array and combined with truncated units are proposed. The exact fast adder is used to obtain the final product.

3.1 Approximate Modified Booth Encoders (AMBEs)

The elements in the K-map are sorted by the Gray code. When the elements in the K-map are symmetrical, the complexity will be reduced.

3.1.1 Proposed Approximate Modified Booth Encoding-a (AMBE-a): The approximate Booth coding is designed based on the accurate MBE. Four elements are replaced in the K-map of accurate MBE by changing '1' to '0', thus the K-map of AMBE becomes symmetric.

Table 2 is the K-map for the first proposed approximate modified Booth encoding (AMBE-a), where ① denotes an entry in which a '0' is replaced by a '1' and ⓪ denotes an entry in which a '1' is replaced by a '0'. There are six entries modified to simplify the encoding. The approximate encoding uses one XOR and NAND function; therefore, the output of AMBE-a is given as follows:

$$\begin{aligned} app_{ij1} &= (a_j\overline{b_{2i+1}})\overline{b_{2i}b_{2i-1}} + \overline{a_j}b_{2i+1}\overline{b_{2i}b_{2i-1}} \\ &= (b_{2i+1} \oplus a_j)\overline{b_{2i}b_{2i-1}} \end{aligned} \tag{9}$$

Compared with the accurate MBE (5), AMBE-a can significantly reduce both the complexity and the critical path delay of Booth encoding. The error rate is defined as the probability of incorrect outputs when different inputs are provided. It is denoted by P_e, given by:

$$P_e = 6/32 = 18.75\% \tag{10}$$

The gate level structure of ABME-a is shown in Fig. 5, which only requires one XOR-2 gate, one NAND-2 gate and one AND-2 gate.

3.1.2 Proposed Approximate Modified Booth Encoding-b (AMBE-b): Table 3 is the K-map for the second proposed approximate modified Booth encoding (AMBE-b) after replacing the elements, where ten entries of ① and six entries of ⓪ are flipped. There

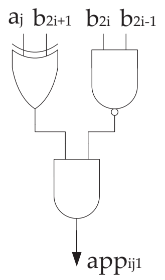

Fig. 5: The gate level circuit of AMBE-a.

Table 3 K-Map of AMBE-b

$a_j\ a_{j-1}$ \ $b_{2i+1}\ b_{2i}\ b_{2i-1}$	000	001	011	010	110	111	101	100
00	0	0	0	0	⓪	0	⓪	⓪
01	0	0	⓪	0	⓪	0	⓪	0
11	①	1	1	1	①	①	①	①
10	①	1	①	1	①	①	①	1

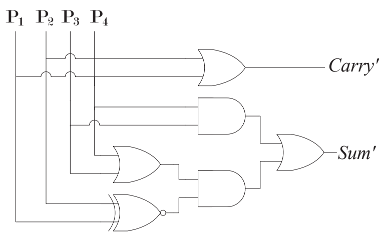

Fig. 6: The gate level circuit of AC-a.

are sixteen entries modified to simplify the encoding. The approximate encoding shows the output of Booth encoding is a_j; therefore, the output of AMBE-b is given as follows:

$$app_{ij2} = a_j \tag{11}$$

AMBE-b further reduce both the complexity and the critical path delay of Booth encoding; the error rate is given by:

$$P_e = 16/32 = 50\% \tag{12}$$

Although the error rate of AMBE-b is larger than AMBE-a, it is much more efficient in terms of hardware consumption.

3.2 Approximate 4-2 Compressors (ACs)

The compressor is usually designed by two full adders. The carry input C_{in} and the carry output C_{out} are ignored in the approximate compressors (ACs) in this work. The results of the proposed ACs are *Sum'* and *Carry'*.

3.2.1 Proposed Approximate Compressor-a (AC-a): The first proposed design of AC (AC-a) contains three OR-2 gates, two AND-2 gates and one NXOR-2 gate (Fig. 6). The logical expressions of AC-a are given as:

$$Sum' = \overline{(P_1 \oplus P_2)}\overline{(\overline{P_3} \cdot \overline{P_4})} + \overline{(\overline{P_3} + \overline{P_4})} \tag{13}$$

$$Carry' = \overline{\overline{P_1} \cdot \overline{P_2}} \tag{14}$$

Table 4 shows the K-map of AC-a. The difference between an approximate compressor and an accurate compressor under the same input is recorded as *Diff.* in the table. For example, if {P_1, P_2, P_3, P_4}=1111, the value in Table 4 is (*Carry'Sum'/Diff.*)=11/-1. In other words, the output of approximate compressor is $(1 \times 2^1 + 1 \times$

Table 4 K-Map of AC-a

$p_4 p_3$ \ $p_2 p_1$	00	01	11	10
	(Carry'Sum'/Diff.)			
00	00/0	10/+1	10/0	10/+1
01	01/0	10/0	11/0	10/0
11	01/-1	11/0	11/-1	11/0
10	01/0	10/0	11/0	10/0

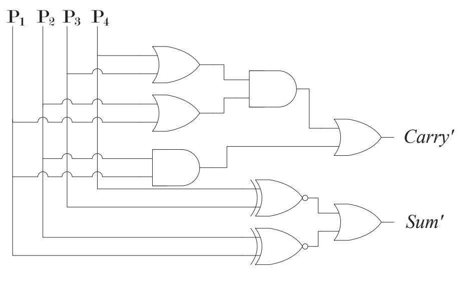

Fig. 7: The gate level circuit of AC-b.

Table 5 K-Map of AC-b

$p_4 p_3$ \ $p_2 p_1$	00	01	11	10
	(Carry'Sum'/Diff.)			
00	01/+1	01/0	11/+1	01/0
01	01/0	10/0	11/0	10/0
11	01/-1	11/0	11/-1	11/0
10	01/0	10/0	11/0	10/0

2^0=3), while the accurate output should be 4 and the *Diff.* value is (3-4=-1).

The AC has an independent carry and it is possible to perform compression in parallel, which reduces the critical path and improves the performance. The error rate is given by:

$$P_e = 4/16 = 25\% \tag{15}$$

3.2.2 Proposed Approximate Compressor-b (AC-b): The second proposed design of AC (AC-b), contains four OR-2 gates, two AND-2 gates and two NXOR-2 gate (Fig. 7). The logical expressions of AC-b are given as:

$$Sum' = \overline{(P_1 \oplus P_2)} + \overline{(P_3 \oplus P_4)} \tag{16}$$

$$Carry' = \overline{\overline{P_1} + \overline{P_2}} + \overline{\overline{P_1 + P_2} + \overline{P_3 + P_4}} \tag{17}$$

Table 5 shows the K-map of AC-b. The error rate is given by:

$$P_e = 4/16 = 25\% \tag{18}$$

3.3 Proposed Approximate Truncated Booth Multipliers (ATBMs)

In an approximate Booth multiplier, the proposed approximate Booth encodings, $i.e.$, AMBE-a and AMBE-b are used for generating the approximate PPs; the proposed AC are used for compressing the approximate PPs generated by the AMBEs and the approximate regular PP array is be used. An approximation factor p (p=1, 2, ..., 2N) has been proposed in [22] and is also used in this work. This is defined as the number of least significant PP columns that are generated by the approximate Booth encoders.

According to the approximation factor p, the number of truncated units $i.e.$ d, is introduced ($p > d$), which can be seen in Fig. 8. The units in the truncated parts are ignored, which do not consume any hardware resources. Fig. 8 presents the structure of ATBM-8-8-3. The first number, 8, represents the number of bit width of the

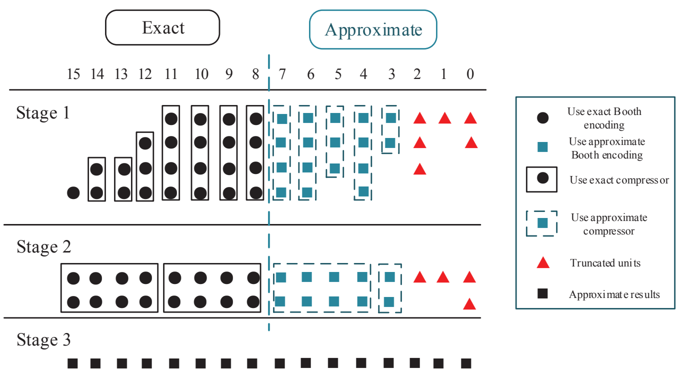

Fig. 8: Structure of ATBM-8-8-3.

Table 6 Four Designs of ATBMs (♦ Denotes a Used Unit, while ◇ Denotes an Unused Unit)

Multiplier	AMBE-a	AMBE-b	AC-a	AC-b
ATBM1	♦	◇	♦	◇
ATBM2	◇	♦	♦	◇
ATBM3	♦	◇	◇	♦
ATBM4	◇	♦	◇	♦

input, the second represents the approximate bit width p and the last number, 3, represents the bit width of truncated parts d.

In the truncated part of ATBM, the lower bits are truncated one by one. The range of d is from 1 to $p - 1$.

Four types of ATBMs are proposed as follows and the different approximate modules of each design can be seen in Table 6. The first and third ATBMs (ATBM1 and ATBM3) use AMBE-a to produce the approximate PPs for lower p bits; however, they use AC-a and AC-b, respectively, to compress the approximate PPs. The second and fourth OABTMs (ATBM2 and ATBM4) use AMBE-b to produce the approximate PPs for lower p bits, which use AC-a and AC-b, respectively, to compress the approximate PPs. The most significant (2N-p) bits of all the four designs are processed with accurate Booth encoder and the exact compressor.

3.4 Hardware and Error Evaluation of ATBMs

Since some PPs of ATBMs are truncated, the consumption of hardware resources is less than most multipliers, relatively. In this section, the error and hardware evaluation of ATBMs will be discussed. At first, 8-bit ATBMs are evaluated to find the regular pattern of ATBMs. which will be applied to 16-bit ATBMS.

All designs are described at gate-level in Verilog HDL and verified by Synopsys VCS and then synthesized by the Synopsys Design Compiler using the NanGate 45 nm Open Cell Library. In the simulation of each design, a supply voltage of 1.25 V and room temperature are assumed. The average power consumption is found using the Synopsys Power Compiler with a back annotated switching activity file generated from the random input vectors. During synthesis, the maximum area and delay are set to 0 μm^2 and 0.2 ns, respectively. The simulation setting for the time scale is 1 ns/100 ps in the Design Compiler script.

In Table 7, power-delay product (PDP) and NMED of 8-bit ATBMs are presented. The ranges of p are from 2 to 14 and the numbers of d increase one by one. We can find that PDPs decrease as the d increases due to the truncated part. The change of NMED is different. When $p > 6$ (ATBM2 and ATBM4) or $p > 8$ (ATBM1 and ATBM3), NMEDs reduce to the least value and increase again. Compared with non-truncated ones, the PDP of ATBMs is reduced by about 80% and the NMED is reduced by about 11% when $p = 14$ and $d = 11$.

In order to visualize the characteristics of ATBMs, Figs. 9-12 show the NMEDs depending on p (larger than 6) for ATBMs and compare with the multipliers (non-truncated) that use the same

Table 7 NMEDs and PDPs of 8-bit ATBMs at Different Approximation Factors

		ATBM1		ATBM2		ATBM3		ATBM4	
p	d	NMED (10^{-2})	PDP (pJ)	NMED (10^{-2})	PDP (pJ)	NMED (10^{-2})	PDP (pJ)	NMED (10^{-2})	PDP (pJ)
2	1	0.0765	0.0994	0.0767	0.0934	0.0764	0.0993	0.0767	0.0934
	1	0.0809	0.0956	0.0866	0.0956	0.0811	0.0962	0.0865	0.0955
4	2	0.0829	0.0920	0.0868	0.0968	0.0828	0.0922	0.0867	0.0967
	3	0.0931	0.0913	0.0938	0.0873	0.0931	0.091	0.0937	0.0873
	1	0.113	0.0841	0.145	0.0726	0.111	0.0844	0.145	0.0742
	2	0.114	0.0829	0.145	0.0719	0.112	0.0823	0.145	0.0733
6	3	0.116	0.0751	0.143	0.0723	0.113	0.0763	0.143	0.0721
	4	0.121	0.0746	0.139	0.0718	0.118	0.0756	0.140	0.0720
	5	0.165	0.0696	0.164	0.0676	0.163	0.0698	0.167	0.0673
	1	0.397	0.0642	0.621	0.0527	0.391	0.0653	0.619	0.0528
	2	0.396	0.0629	0.619	0.0519	0.390	0.0643	0.617	0.0521
	3	0.393	0.0591	0.613	0.0520	0.387	0.0594	0.611	0.0503
8	4	0.389	0.0571	0.602	0.0503	0.383	0.0566	0.600	0.0503
	5	0.383	0.0546	0.571	0.0489	0.376	0.0562	0.571	0.0429
	6	0.408	0.0511	0.525	0.0466	0.401	0.0508	0.527	0.0489
	7	0.582	0.0458	0.585	0.0458	0.576	0.0461	0.592	0.0463
	1	1.347	0.0542	2.214	0.0392	1.325	0.0539	2.166	0.0408
	2	1.346	0.0530	2.212	0.0389	1.324	0.0514	2.165	0.0402
	3	1.343	0.0492	2.206	0.0390	1.321	0.0467	2.159	0.0357
	4	1.338	0.0455	2.194	0.0324	1.316	0.0483	2.147	0.0354
10	5	1.325	0.0414	2.154	0.0302	1.304	0.0385	2.111	0.0361
	6	1.308	0.0352	2.077	0.0301	1.288	0.0385	2.038	0.0304
	7	1.312	0.0320	1.946	0.0277	1.294	0.0329	1.923	0.0284
	8	1.502	0.0288	1.177	0.0266	1.504	0.0291	1.773	0.0268
	9	1.984	0.0233	1.980	0.0228	1.844	0.0242	1.916	0.0222
	1	4.668	0.0419	6.726	0.0290	4.457	0.0454	6.611	0.0290
	2	4.668	0.0402	6.725	0.0285	4.456	0.0437	6.610	0.0285
	3	4.664	0.0366	6.718	0.0265	4.453	0.0390	6.604	0.0276
	4	4.658	0.0356	6.707	0.0236	4.448	0.0372	6.594	0.0251
	5	4.638	0.0302	6.666	0.0239	4.435	0.0315	6.559	0.0215
12	6	4.605	0.0270	6.587	0.0194	4.412	0.0275	6.487	0.0203
	7	4.553	0.0235	6.418	0.0188	4.386	0.0241	6.343	0.0197
	8	4.506	0.0204	6.093	0.0172	4.411	0.0205	6.046	0.0171
	9	4.484	0.0158	5.743	0.0142	4.746	0.0149	5.533	0.0133
	10	4.874	0.0136	5.529	0.0125	5.263	0.0138	5.312	0.0122
	11	6.289	0.0106	6.315	0.0102	5.359	0.0106	5.359	0.0103
	1	13.175	0.0404	21.678	0.0266	9.807	0.0441	19.686	0.0298
	2	13.174	0.0398	21.677	0.0261	9.806	0.0428	19.685	0.0291
	3	13.168	0.0359	21.670	0.0228	9.804	0.0393	19.680	0.0263
	4	13.160	0.0337	21.660	0.0227	9.800	0.0371	19.673	0.0241
	5	13.130	0.0311	21.625	0.0215	9.792	0.0314	19.646	0.0241
	6	13.077	0.0241	21.559	0.0190	9.778	0.0289	19.595	0.0208
14	7	12.969	0.0210	21.404	0.0163	9.764	0.0246	19.474	0.0193
	8	12.782	0.0194	21.033	0.0152	9.805	0.0233	19.177	0.0188
	9	12.446	0.0114	20.752	0.0105	10.178	0.0136	18.592	0.0111
	10	12.178	0.0094	19.586	0.0080	10.631	0.0136	17.871	0.0089
	11	12.103	0.0071	19.083	0.0062	10.743	0.0082	17.094	0.0073
	12	19.280	0.0058	20.689	0.0055	23.498	0.0070	22.875	0.0063
	13	34.757	0.0033	31.632	0.0030	34.757	0.0033	31.632	0.0030

approximation modules except the truncated units. ATBM1 and ATBM3 have similar characteristics and ATBM2 is similar as ATBM4 due to that the same Booth encoding is applied, which indicate that the Booth encoding plays an important role in the design of Booth multiplier. For ATBM1, the sharp changing point is at $d = p - 3$; for ATBM3, the sharp changing point is at $d = p - 3$ $(p \neq 12)$ and $d = p - 4$ $(p = 12)$; for ATBM2 and ATBM4, the sharp changing point is at $d = p - 2$ $(p \neq 14)$ and $d = p - 3$ $(p = 14)$. Overall, the ATBMs show a better performance in error evaluation.

From the above graphs, it can be found that the sharp change always happens when $p \simeq d - 3$. If the truncated bits are larger than $p - 3$, the error of ATBMs will increase immediately. As shown in Fig. 13, the more significant bits larger than $p - 3$ have larger effect on the accuracy of the multiplier.

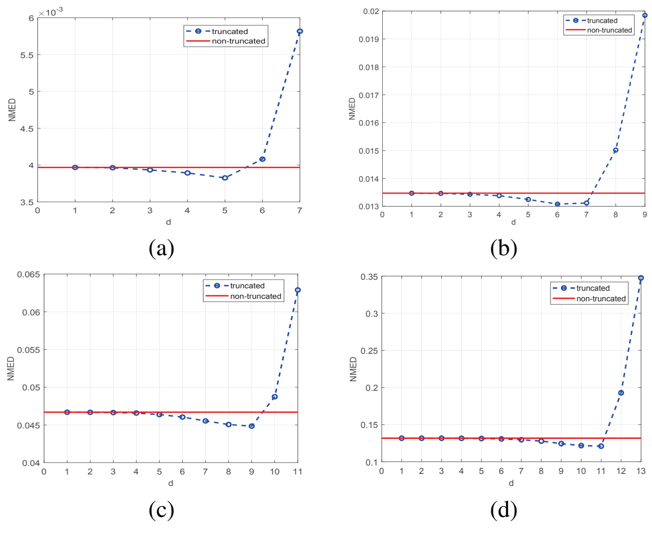

Fig. 9: NMEDs for ATBM1-8: (a) ATBM1-8-8, (b) ATBM1-8-10, (c) ATBM1-8-12, (d) ATBM1-8-14.

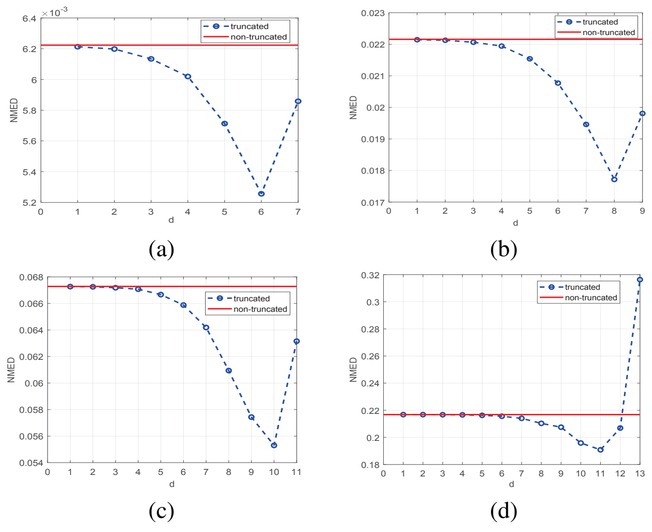

Fig. 10: NMEDs for ATBM2-8: (a) ATBM2-8-8, (b) ATBM2-8-10, (c) ATBM2-8-12, (d) ATBM2-8-14.

3.5 *Top Designs of ATBMs*

Following the evaluation of 8-bit ATBMs, the best designs of 8-bit ATBMs are found for constructing 16-bit designs. PDP versus NMED for 8-bit ATBMs (ATBM1, ATBM2, ATBM3, ATBM4) are plotted in Figs. 14-17 for $p = 8$ to $p = 14$. From Figs. 14 and 16, the distribution of ATBM1 and ATBM3 with different p is similar. The NMEDs of ATBM1 and ATBM3 decrease slowly and have a sharp changing point when PDPs decrease. ATBM2 and ATBM4 are similar. The changing trend of them is different from the ATBM1 and ATBM3. The reason is that the NMEDs of ATBM2 and ATBM4 decrease faster, which makes scatter plots sparse.

For ATBM1, the efficient designs are obtained at $d = 5$ and $d = 6$ for $p = 8$; at $d = 6$ and $d = 7$ for $p = 10$; at $d = 8$ and $d = 9$ for $p = 12$;at $d = 10$ and $d = 11$ for $p = 14$. For ATBM2, the efficient design is obtained at $d = 6$ for $p = 8$; at $d = 8$ for $p = 10$; at $d = 10$ for $p = 12$; at $d = 10$, $d = 11$ and $d = 12$ for $p = 14$. For ATBM3, the efficient design is obtained at $d = 6$ for $p = 8$; at $d = 7$ for $p = 10$; at $d = 7$ and $d = 8$ for $p = 12$; at $d = 9$, $d = 10$ and $d = 11$ for $p = 14$. For ATBM4, the efficient design is obtained at $d = 6$ for $p = 8$; at $d = 8$ for $p = 10$; at $d = 10$ and $d = 11$ for $p = 12$; at $d = 9$, $d = 10$ and $d = 11$ for $p = 14$. The efficient ATBM designs for different p and corresponding d are shown in Figs. 18-21 (red

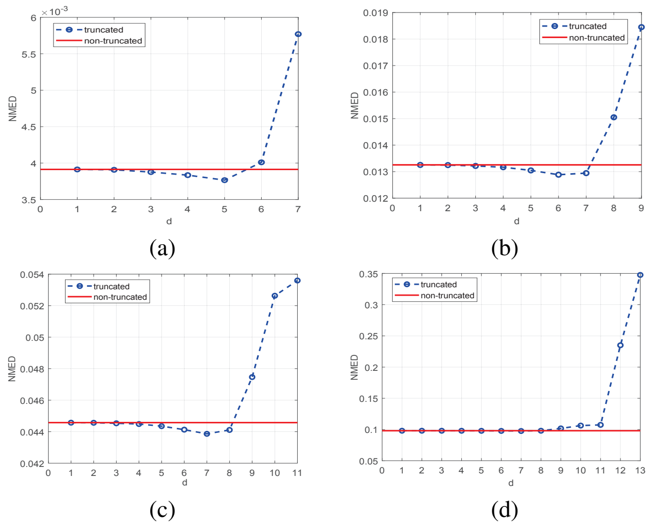
Fig. 11: NMEDs for ATBM3-8: (a) ATBM3-8-8, (b) ATBM3-8-10, (c) ATBM3-8-12, (d) ATBM3-8-14.

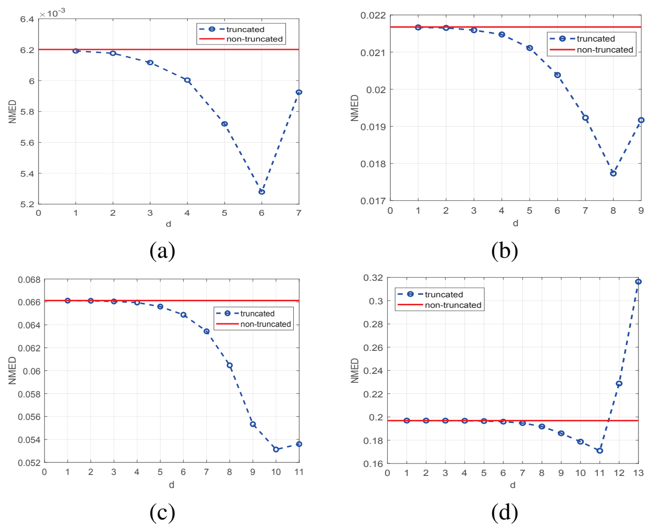
Fig. 12: NMEDs for ATBM4-8: (a) ATBM4-8-8, (b) ATBM4-8-10, (c) ATBM4-8-12, (d) ATBM4-8-14.

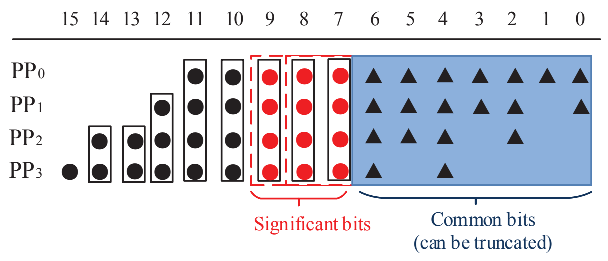

Fig. 13: The PP array of ATBM-8-10-7.

circles). The PDP versus NMED for 8-bit ATBMs are shown in Fig. 22.

In Figs. 18-21, the numbers represent the truncated modules and approximate units. For example, when $p = 8$, '1-7' means $d = 1$ (the truncated units) and $p - d = 7$ (the approximate units). Meanwhile, red circles indicate the best designs for ATBMs. Orange ones indicate that the ATBM designs with NMEDs less than the non-truncated multipliers and blue ones indicate the opposite. Generally, the best design of ATBMs can be achieved when $d > \frac{p}{2}$.

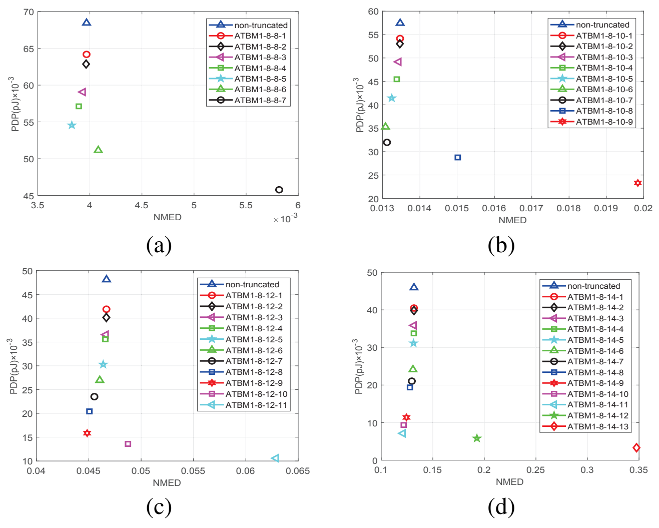
Fig. 14: PDP versus NMED for ATBM1-8: (a) ATBM1-8-8, (b) ATBM1-8-10, (c) ATBM1-8-12, (d) ATBM1-8-14.

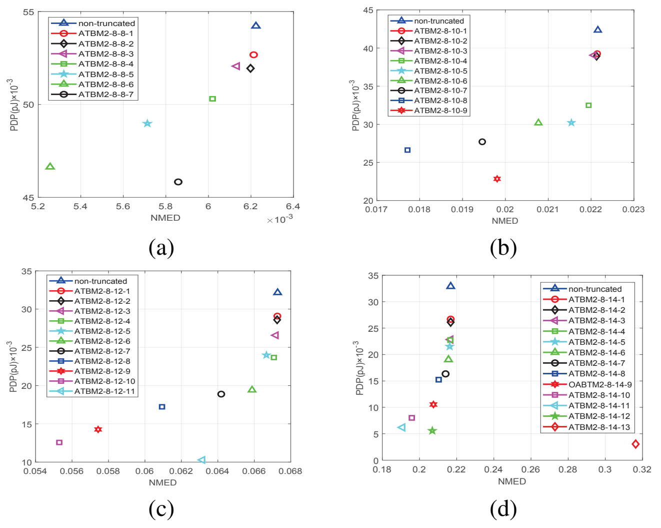
Fig. 15: PDP versus NMED for ATBM2-8: (a) ATBM2-8-8, (b) ATBM2-8-10, (c) ATBM2-8-12, (d) ATBM2-8-14.

Fig. 22 shows the comparisons of 8-bit ATBMs with better performance, including ATBM2 and ATBM4 with $p = 6$ and $d = 4$. The best designs of 8-bit ATBMs are ATBM2-8-12-10, ATBM4-8-12-10 and ATBM4-8-12-11 when considering both PDP and NMED.

As shown in the evaluation of 8-bit ATBMs above, the best designs are achieved around $d = p - 3$ when p is larger than 6. Hence, the performance of 16-bit ATBMs are presented in Table 8. The PDPs keep decreasing when d increases and NMEDs have a sharp change when $d = 3$ (ATBM1 and ATBM3) and $d = 2$ (ATBM2 and ATBM4). Compared with 16-bit approximate multipliers which use the same approximation modules except the truncated units, the PDP of ATBMs is reduced by about 85% and the NMED is reduced by about 48% when $p = 28$ and $d = 25$.

4 Comparison with the State-of-the-Art Designs

The proposed designs at large p values have large NMEDs, but small PDPs; moreover, the proposed 16 bit ATBM1 and ATBM2 with p<20 show a good tradeoff between PDP and NMED. Therefore,the proposed 16-bit designs with p=12, 14, and 16, are compared with previous approximate 16-bit Booth multipliers proposed in [25] (R4ABM04), [26] (R4ABM11), [27] (R4ABM12), [20] (R8ABM1,

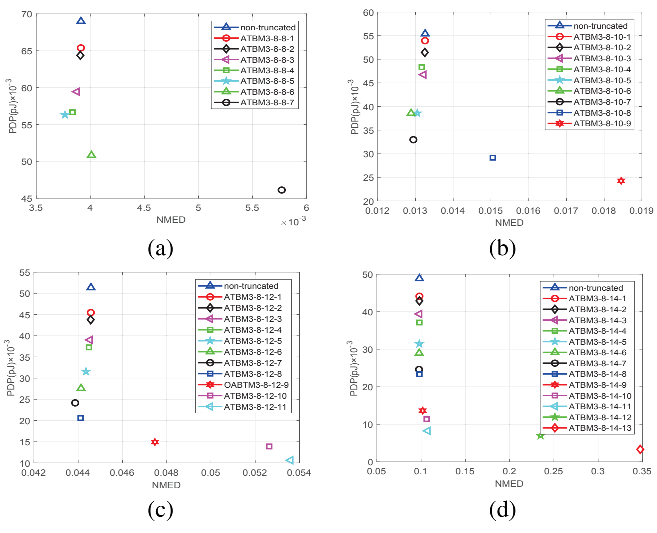
Fig. 16: PDP versus NMED for ATBM3-8: (a) ATBM3-8-8, (b) ATBM3-8-10, (c) ATBM3-8-12, (d) ATBM3-8-14.

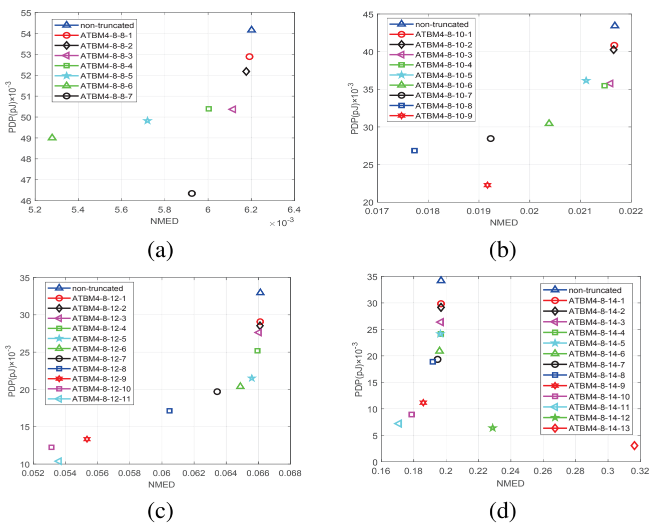
Fig. 17: PDP versus NMED for ATBM4-8: (a) ATBM4-8-8, (b) ATBM4-8-10, (c) ATBM4-8-12, (d) ATBM4-8-14.

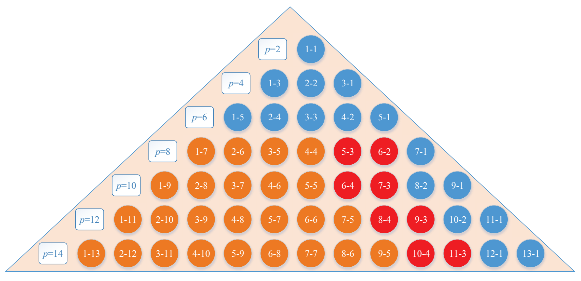
Fig. 18: The top designs of ATBM1.

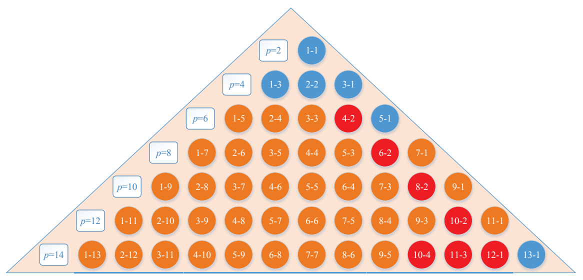
Fig. 19: The top designs of ATBM2.

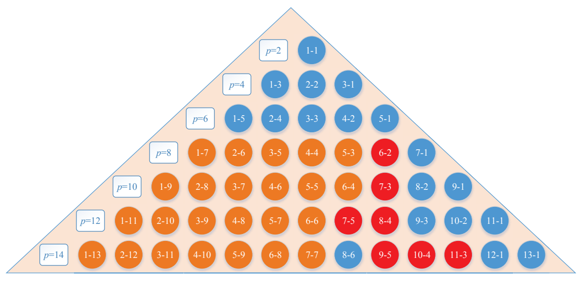
Fig. 20: The top designs of ATBM3.

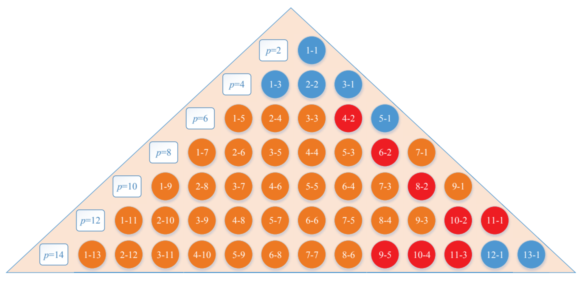
Fig. 21: The top designs of ATBM4.

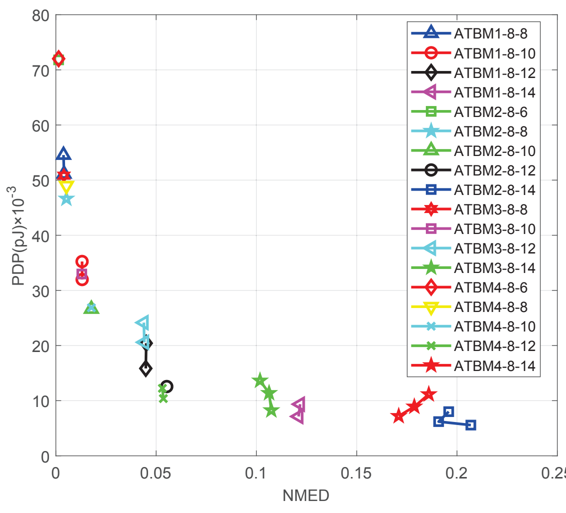

Fig. 22: PDP versus NMED for 8-bit ATBMs.

R8ABM2-C9 and R8ABM2-C15), [22] (R4ABM1 and R4ABM2), [24] (ARBM) and [30] (R4ARBM1, R4ARBM2, R4ARBM3 and R4ARBM4).

R4ABM1, R4ABM2, ARBM, R4ARBM1, R4ARBM2, R4ARBM3 and R4ARBM4 are compared with ATBMs when p=12, 14, and 16. R8ABM1, R8ABM2-C9 and R8ABM2-C15 are all radix-8 approximate Booth multipliers with no truncation, 9-bit truncation and 15-bit truncation, respectively. All these designs are also synthesized by the Synopsys Design Compiler using the NanGate 45 nm Open Cell Library.

The power consumption, critical path delay, area, PDP and NMED are presented in Table 9. R4ABM04, R4ABM11, R4ABM12, R8ABM1, R8ABM2-C9 and R8ABM2-C15 perform well in PDPs but have large NMEDs. The proposed multipliers are significantly more efficient and faster than R4ABMs. The PDP is

Table 8 NMEDs and PDPs of 16bit ATBMs at Different Approximation Factors

		ATBM1		ATBM3				ATBM2		ATBM4	
p	d	NMED (10^{-5})	PDP (pJ)	NMED (10^{-5})	PDP (pJ)	p	d	NMED (10^{-5})	PDP (pJ)	NMED (10^{-5})	PDP (pJ)
8	5	0.254	0.505	0.254	0.505	8	5	0.254	0.519	0.254	0.519
	6	0.254	0.518	0.254	0.522		6	0.254	0.505	0.254	0.506
	8	0.267	0.428	0.266	0.445		9	0.291	0.415	0.292	0.420
12	9	0.266	0.430	0.266	0.436	12	10	0.290	0.411	0.290	0.413
	10	0.273	0.424	0.275	0.422		11	0.339	0.412	0.339	0.412
	10	0.626	0.371	0.613	0.389		11	0.751	0.351	0.733	0.353
14	11	0.618	0.368	0.615	0.369	14	12	0.704	0.326	0.705	0.345
	12	0.6529	0.364	0.678	0.354		13	0.859	0.328	0.914	0.334
	12	2.419	0.308	2.341	0.310		13	3.170	0.277	3.050	0.279
16	13	2.287	0.301	2.259	0.301	16	14	2.757	0.276	2.756	0.279
	14	2.393	0.284	2.407	0.283		15	2.825	0.271	3.008	0.267
	14	8.962	0.232	8.824	0.238		15	11.21	0.209	11.23	0.212
18	15	8.597	0.225	8.680	0.221	18	16	10.28	0.206	10.45	0.205
	16	9.353	0.215	9.749	0.210		17	10.51	0.197	11.64	0.199
	16	34.14	0.180	33.51	0.180		17	43.32	0.150	42.59	0.156
20	17	32.28	0.164	32.54	0.172	20	18	37.33	0.151	37.80	0.155
	18	33.54	0.166	34.88	0.163		19	39.19	0.151	44.12	0.152
	20	460.8	0.0876	447.6	0.0874		21	462.9	0.0707	485.7	0.0740
24	21	422.7	0.0778	441.57	0.0811	24	22	414.9	0.0682	477.1	0.0688
	22	481.9	0.0729	512.6	0.0746		23	511.1	0.0656	740.6	0.0657
	24	7419	0.0314	6842	0.0309		25	6090	0.0228	7222	0.0243
28	25	7326	0.0260	7089	0.0275	28	26	5555	0.0220	5555	0.0217
	26	6208	0.0232	5637	0.0231		27	5619	0.0178	5619	0.0179

reduced by about 21% ($p = 16$) and the NMED is reduced by about 4% ($p = 12$). Compared with R4ARBMs, ATBMs have smaller PDP values (reduced by about 27%) while the values of NMED are close. Fig. 23 shows that the proposed ATBM2 ($p = 14$) and ATBM4 ($p = 14$) are the best designs when considering both PDP and NMED, while ATBM2 ($p = 12$) and ATBM4 ($p = 12$) are the second best designs. These results confirm that the proposed designs with $p = 12$ and $p = 14$ are better than previous approximate Booth multipliers when considering both PDP and NMED. Table 10 shows a qualitative comparison between the above approximate booth multipliers (using 3 qualitative measures: low, medium, high). All proposed ATBMs have low or medium power, delay, area, PDP and NMED.

Table 9 Comparative Evaluation of 16-bit Approximate Booth Multipliers

Approximate Booth Multipliers	Power (uW)	Delay (ns)	Area (um^2)	PDP (pJ)	NMED (10^{-5})
R4ABM04 [25]	427.3	0.95	1939	0.406	5.31
R4ABM11 [26]	404.4	0.94	1859	0.380	2.18
R4ABM12 [27]	394.6	0.95	1808	0.374	2.26
R8ABM1 [20]	376.7	1.23	1516	0.463	1.92
R8ABM2-C9 [20]	332.6	1.22	1332	0.406	4.43
R8ABM2-C15 [20]	217.3	1.18	912	0.256	5.73
R4ABM1 (p=12) [22]	525.6	0.94	2127	0.508	0.31
R4ABM1 (p=14) [22]	516.6	0.95	2169	0.490	0.93
R4ABM1 (p=16) [22]	452.9	0.87	1923	0.451	3.10
R4ABM2 (p=12) [22]	515.4	0.92	2086	0.484	0.27
R4ABM2 (p=14) [22]	479.7	0.92	2004	0.441	0.62
R4ABM2 (p=16) [22]	424.0	0.86	1764	0.412	3.02
ARBM (p=12) [24]	656.6	0.90	2641	0.590	0.33
ARBM (p=14) [24]	619.7	0.91	2512	0.564	0.74
ARBM (p=16) [24]	596.8	0.88	2412	0.525	2.72
R4ARBM1 (p=12) [30]	607.3	0.94	2430	0.571	0.17
R4ARBM1 (p=14) [30]	590.9	0.89	2349	0.526	1.16
R4ARBM1 (p=16) [30]	551.9	0.87	2223	0.48	3.62
R4ARBM2 (p=12) [30]	588.4	0.95	2358	0.559	0.15
R4ARBM2 (p=14) [30]	580.1	0.89	2294	0.516	0.76
R4ARBM2 (p=16) [30]	518.9	0.88	2101	0.457	3.21
R4ARBM3 (p=12) [30]	602	0.94	2405	0.566	0.27
R4ARBM3 (p=14) [30]	592	0.89	2346	0.527	0.71
R4ARBM3 (p=16) [30]	559.4	0.88	2214	0.492	2.93
R4ARBM4 (p=12) [30]	587.7	0.94	2362	0.552	0.17
R4ARBM4 (p=14) [30]	570.2	0.89	2278	0.507	0.7
R4ARBM4 (p=16) [30]	511.4	0.88	2092	0.45	3.18
ATBM1 (p=12 d=9)	462.4	0.93	1882.7	0.430	0.26
ATBM1 (p=14 d=11)	423.1	0.87	1717.5	0.368	0.61
ATBM1 (p=16 d=13)	358.4	0.84	1486.6	0.301	2.28
ATBM2 (p=12 d=10)	456.7	0.90	1847.1	0.415	0.29
ATBM2 (p=14 d=12)	408.5	0.80	1675.0	0.326	0.70
ATBM2 (p=16 d=14)	333.6	0.83	1398.6	0.276	2.75
ATBM3 (p=12 d=9)	490.9	0.89	1972.6	0.436	0.26
ATBM3 (p=14 d=11)	429.1	0.86	1747.3	0.369	0.61
ATBM3 (p=16 d=13)	358.5	0.84	1480.8	0.301	2.25
ATBM4 (p=12 d=10)	459.8	0.90	1857.7	0.413	0.29
ATBM4 (p=14 d=12)	401.5	0.86	1638.5	0.345	0.70
ATBM4 (p=16 d=14)	336.4	0.83	1409.2	0.279	2.75

5 Case Studies

The proposed approximate multipliers are applied to image processing, K-mean clustering and handwritten digit recognition and compared with R4ABM2 [22] in this section, since R4ABM2 has a similar structure and a good tradeoff between NMED and PDP.

5.1 Image Processing

The 8-bit approximate multiplier proposed in this paper is applied to image processing to verify its function. The two images are multiplied pixel by pixel to mix into a single output image. As Booth multipliers perform signed multiplication, the pixel values have been shifted from $0 \sim 255$ to $-128 \sim 127$. For ATBMs, ATBM2-8-12-10, ATBM4-8-12-10 and ATBM4-8-12-11 perform better. They are applied to digital image multiplication as shown in the Fig. 24.

The quality of the processed image deteriorates with an increase of p. The output image is still viable when p is smaller than 12, thus confirming the error analysis presented previously.

The Mean Structural Similarity Measure (MSSIM) [29] shows that the image segmentation result is similar to the average local structure of the reference image, and the value is between 0 and 1. The greater the value, the better the quality of the segmentation. The corresponding MSSIM and PSNR values are provided in Table 11, using the percentage of PDP (an approximate design over its exact counterpart). The quality of processed images does not decrease much when $p \leq 10$ (with MSSIM>0.95). ATBM2-8-12-10, ATBM4-8-12-10 and ATBM4-8-12-11 are compared with R4ABM2 [22], under the same value of p. ATBM4-8-12-10 achieves a similar MSSIM and a higher PSNR than R4ABM2(p=12) with a significantly smaller PDP value.

5.2 K-Means Clustering

K-means clustering is a method for cluster analysis in data mining. It partitions n observations into K clusters with the nearest mean [31]. The proposed 16-bit ATBM4 (ATBM4-16-12-10, ATBM4-16-14-12 and ATBM4-16-16-14) are applied to calculate the squared deviation between points belonging to different clusters. The F-measure value [32] is used as the metric to evaluate the clustering results. It considers both the precision and the recall of the test. So, the F-measure score can be interpreted as a weighted average of the precision and recall. The best value of the F-measure score is 1 and its worst value is 0. Each F-measure value is the average of 50 experiments for each data set. In this work, several University of California Irvine (UCI) benchmark datasets [34] are selected to test the K-means clustering using ATBMs. The F-measure results are listed in Table 12.

The performance of data sets except Customers do not have obvious change because the data of them are very close. The effect of K-means clustering is very stable using the proposed multipliers and the values of F-measure are the same. On the contrary, the

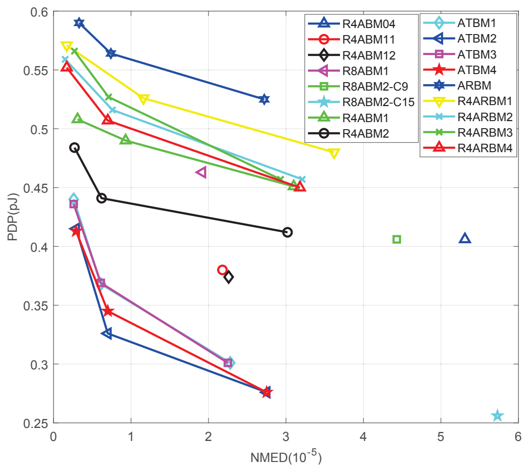

Fig. 23: PDP versus NMED for 16-bit Approximate Booth Multipliers.

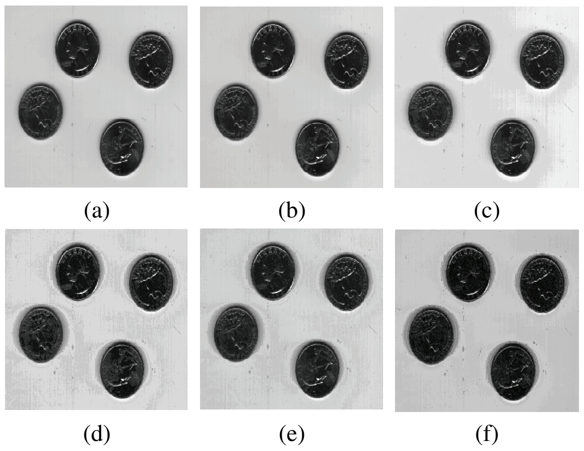

Fig. 24: Image multiplication results using multipliers with different p: (a) R4ABM2 (p=8), (b) R4ABM2 (p=10), (c) R4ABM2 (p=12), (d) ATBM2-8-12-10, (e) ATBM4-8-12-10, (f) ATBM4-8-12-11.

Table 10 Qualitative Comparison between 16-bit Approximate Booth Multipliers

Approxiamte Multipliers	Power	Delay	Area	PDP	NMED
R4ABM04 [25]	medium	medium	medium	medium	high
R4ABM11 [26]	medium	low	medium	medium	medium
R4ABM12 [27]	medium	medium	medium	medium	medium
R8ABM1 [20]	medium	high	medium	medium	low
R8ABM2-C9 [20]	low	high	low	medium	high
R8ABM2-C15 [20]	low	high	low	low	high
R4ABM1 (p=12) [22]	high	low	high	high	low
R4ABM1 (p=14) [22]	high	medium	high	high	low
R4ABM1 (p=16) [22]	medium	low	medium	medium	medium
R4ABM2 (p=12) [22]	high	low	high	high	low
R4ABM2 (p=14) [22]	medium	low	medium	medium	low
R4ABM2 (p=16) [22]	medium	low	medium	medium	medium
ARBM (p=12) [24]	high	low	high	high	low
ARBM (p=14) [24]	high	low	high	high	low
ARBM (p=16) [24]	high	low	high	high	medium
R4ARBM1 (p=12) [30]	high	low	high	high	low
R4ARBM1 (p=14) [30]	high	low	high	high	low
R4ARBM1 (p=16) [30]	high	low	high	high	medium
R4ARBM2 (p=12) [30]	high	medium	high	high	low
R4ARBM2 (p=14) [30]	high	low	high	high	low
R4ARBM2 (p=16) [30]	high	low	high	medium	medium
R4ARBM3 (p=12) [30]	high	low	high	high	low
R4ARBM3 (p=14) [30])	high	low	high	high	low
R4ARBM3 (p=16) [30]	high	low	high	high	medium
R4ARBM4 (p=12) [30]	high	low	high	high	low
R4ARBM4 (p=14) [30]	high	low	high	high	low
R4ARBM4 (p=16) [30]	high	low	high	medium	medium
ATBM1 (p=12 d=9)	medium	low	medium	medium	low
ATBM1 (p=14 d=11)	medium	low	medium	medium	low
ATBM1 (p=16 d=13)	low	low	low	low	medium
ATBM2 (p=12 d=10)	medium	low	medium	medium	low
ATBM2 (p=14 d=12)	medium	low	medium	low	low
ATBM2 (p=16 d=14)	low	low	low	low	medium
ATBM3 (p=12 d=9)	medium	low	medium	medium	low
ATBM3 (p=14 d=11)	medium	low	medium	medium	low
ATBM3 (p=16 d=13)	low	low	low	low	medium
ATBM4 (p=12 d=10)	medium	low	medium	medium	low
ATBM4 (p=14 d=12)	medium	low	medium	low	low
ATBM4 (p=16 d=14)	low	low	low	low	medium

data in Customers are dispersed and the F-measure values are different. ATBM4 results in similar F-measure values as R4ABMs for the same value of p, but using less hardware resources.

5.3 *Handwritten Digit Recognition*

The handwritten digit recognition utilizes a deep neural network LeNet-5 [34] that classifies the MNIST database [35]. The LeNet-5 consists of two convolutional layers (Layer 1 (C1) and Layer 3

Table 11 MSSIM and PSNR of Processed Images using 8-bit ATBMs with Different Approximate Factors

Design	R4ABM2 [22]		
	(p = 8)	(p = 10)	(p = 12)
PDP (%)	55.72	48.09	39.69
MSSIM	0.9977	0.9786	0.8945
PSNR (dB)	49.08	33.98	22.41
Design	ATBM2-8-12-10	ATBM4-8-12-10	ATBM4-8-12-11
PDP (%)	9.54	9.31	7.86
MSSIM	0.7791	0.8325	0.7950
PSNR (dB)	27.28	29.94	23.79

Table 12 F-measure Values of Clustering by K-mean using 16-bit ATBMs with Different Approximate Factors (NS: No. of Samples, NC: No. of Clusters)

Datasets	Iris	Glass	Hayes-roth	Balance-scale	Customers
NS	150	214	132	625	440
NC	3	7	4	3	3
Design	F-measure value				
R4ABM1 [22]	0.505	0.421	0.524	0.603	0.717
R4ABM2 [22]	0.505	0.421	0.524	0.603	0.717
ATBM4-16-12-10	0.505	0.421	0.524	0.603	0.577
ATBM4-16-14-12	0.505	0.421	0.524	0.603	0.578
ATBM4-16-16-14	0.505	0.421	0.524	0.603	0.590

(C3)), each of which is followed by a max pooling layer, two fully connected layers (Layer 5 (C5) and Layer 6 (F6)), and a softmax layer. The activation function is tanh. The training set and the test set include 60,000 and 10,000 images from the MNIST database, respectively. After LeNet-5 is trained the double-precision floating-point exact multipliers are replaced with an approximate design in the inference phase. Note that only multiplication operations in convolutional or fully connected layers are performed using approximate 8-bit fixed-point multipliers. The recognition rate is used as a metric to evaluate accuracy. To achieve a high recognition rate, the approximate designs with p=6 (ATBM2-8-6-4 and ATBM4-8-6-4) along with the previous design R4ABM2 [22] with p = 4 and 6 are considered.

Table 13 shows the recognition rate of the handwritten digit recognition application. When the multiplication operations in the F6 layer are performed approximately, the recognition rate is given as in the 3rd row in Table 13. The other rows show the corresponding recognition rates with the approximate multipliers for the layers. Note that when recognition is based on the exact floating-point multiplier, the recognition rate is 98.38%. When the floating-point

Table 13 Recognition Rate (%)

Multipliers	PDP (%)	F6	F6/C5	F6/C5/C3	F6/C5/C3/C1
R4ABM2 [22] (p=4)	74.05	98.37	98.37	98.38	98.38
R4ABM2 [22] (p=6)	61.07	98.35	97.31	97.01	97.01
ATBM2-8-6-4	54.81	98.32	97.78	97.64	97.64
ATBM4-8-6-4	54.96	98.33	97.73	97.63	97.63

multiplier is replaced by the approximate designs, the recognition rates are similar or slightly lower. For ATBM2-8-8-6 and ATBM4-8-8-6, the recognition rates are similar or ever slightly higher than for R4ABM2 with p = 6 and their PDPs are smaller. For R4ABM2 with p = 4, the recognition rate is still 98.38%, but the PDP is fairly high. In general, the recognition rates for the proposed approximate designs (ATBM2-8-6-4 and ATBM4-8-6-4) remain high.

6 Conclusion

Designs of approximate-truncation Booth multipliers (ATBMs) have been studied in this paper. Two approximate Booth encoders have been proposed to reduce the complexity of MBE by introducing incorrect terms in the truth table. Meanwhile, two approximate compressors are also designed that achieve good performance. Based on the two proposed approximate Booth encoders, $i.e.$, AMBE-a and AMBE-b, and two proposed approximate compressors, $i.e.$, AC-a and AC-b, combined with truncated parts which truncate the partial products, four improved approximate Booth multipliers, i.e. ATBMs, have been designed with different approximation factors. The error metrics of the proposed designs have been studied at different values in the approximation factor (p and d) to achieve a good tradeoff between accuracy and complexity, $i.e.$, PDP vs NMED. The proposed designs have been compared with previous approximate Booth multipliers. Results presented in this paper have shown that ATBM2 is the best design when considering both PDP and NMED. The proposed designs have also been applied to image processing, K-mean clustering and handwritten digit recognition, resulting in a very small loss of accuracy.

7 Acknowledgements

This work is supported by grants from National Natural Science Foundation of China (61871216), the Fundamental Research Funds for the Central Universities China (NE2019102), and the Six Talent Peaks Project in Jiangsu Province (2018XYDXX-009).

Fabrizio Lombardi is supported by NSF under CCF-1953961 and 1812467 grants.

Jie Han is supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada under Project RES0025211.

8 References

1 S. Venkataramani, S. T. Chakradhar, K. Roy, A. Raghunathan: "Computing approximately and efficiently", *Proc. Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2015, pp. 748-745
2 W. Liu, F. Lombardi, M. ShulteS: "A Retrospective and Prospective View of Approximate Computing", *Proceedings of the IEEE*, 2020, 108, (3), pp. 394 - 399
3 S. Venkataramani, V. K. Chippa, and S. T. Chakradhar: "Quality programmable vector processors for approximate computing", *Proc. the 46th IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, 2013, pp.1-12
4 V. Chippa, S. Chakradhar, K. Roy and A. Raghunathan: "Analysis and characterization of inherent application resilience for approximate computing", *Proc. the 50th Annual Design Automation Conference (DAC)*, 2013, pp, 1-9
5 S. Venkataramani, S. Chakradhar, K. Roy and A. Raghunathan: "Approximate computing and the quest for computing efficiency". *Proc. the 52nd Annual Design Automation Conference (DAC)*, 2015, pp, 1-6
6 J. Liang, J. Han, and F. Lombardi: "New metrics for the reliability of approximate and probabilistic adders", *IEEE Trans. Computers*, 2013, 63, (9), pp. 1760-1771
7 M. S. Ansari, H. Jiang, B. F. Cockburn and J. Han: "Low-Power Approximate Multipliers Using Encoded Partial Products and Approximate Compressors", *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 2018, 8, (3), pp. 404-416
8 A. K. Verma, P. Brisk and P. Ienne: "Variable latency speculative addition: A new paradigm for arithmetic circuit design", *Proc. Design, Automation & Test in Europe (DATE)*, 2008, pp, 1250-1255
9 A. B. Kahng and S. Kang: "Accuracy-configurable adder for approximate arithmetic designs", *Proc. the 49th Annual Design Automation Conference (DAC)*, 2012, pp, 820-825
10 D. Mohapatra, V. K. Chippa, A. Raghunathan and K. Roy: "Design of voltage-scalable meta-functions for approximate computing", *Proc. Design, Automation & Test in Europe (DATE)*, 2011, pp, 1-6
11 N. Zhu, W. L. Goh and K. S. Yeo: "An enhanced low-power high-speed adder for error-tolerant application", *Proc. the 12th Int. Symp. Integrated Circuits*, 2009, pp, 69-72
12 K. Du, P. Varman and K. Mohanram: "High performance reliable variable latency carry select addition", *Proc. Design, Automation & Test in Europe (DATE)*, 2012, pp, 1257-1262
13 Y. Kim, Y. Zhang and P. Li: "An energy efficient approximate adder with carry skip for error resilient neuromorphic VLSI systems", *IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, 2013, pp, 130-137
14 L. Li and H. Zhou: "On error modeling and analysis of approximate adders", *IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, 2014, pp, 511-518
15 C. Lin, Y. M. Yang and C. C. Lin: "High-performance low-power carry speculative addition with variable latency", *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, 2015, 23, (9), pp. 1591-1603
16 R. Ye, T. Wang, F. Yuan, R. Kumar and Q. Xu: "On reconfiguration-oriented approximate adder design and its application", *IEEE/ACM Int. Conf. Computer-Aided Design (ICCAD)*, 2013, pp, 48-54
17 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 and Systems I: Regular Papers*, 2010, 57, (4), pp. 850-862
18 P. Yin, C. Wang and W. Liu: "Design and performance evaluation of approximate floating-point multipliers", *IEEE Conmputer Society Annual Symp. VLSI (ISVLSI)*, 2016, pp, 296-301
19 P. Kulkarni, P. Gupta and M. D. Ercegovac: "Trading accuracy for power in a multiplier architecture", *Journal of Low Power Electronics*, 2011, 7, (4), pp. 490-501
20 H. Jiang, J. Han, F. Qiao and F. Lombardi: "Approximate radix-8 Booth multipliers for low-power and high-performance operation", *IEEE Trans. Computers*, 2016, 65, (8), pp. 2638-2644
21 W. Yeh and C. Jen: "High-speed Booth encoded parallel multiplier design", *IEEE Trans. Computers*, 2000, 49, (7), pp. 692-701
22 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*, 2017, 66, (8), pp.1435-1441
23 A. Momeni, J. Han, P. Montuschi, and F. Lombardi: "Design and analysis of approximate compressors for multiplication", *IEEE Trans. Computers*, 2015, 64, (4), pp. 984-994
24 T. Cao, W. Liu, C. Wang, X. Cui and F. Lombardi: "Design of approximate redundant binary multipliers". *Proc. IEEE/ACM Int. Symp. Nanoscale Architectures (NANOARCH)*, 2016, pp. 31-36
25 K. J. Cho, K. C. Lee, J. G. Chung, and K. K. Parhi: "Design of low error fixed-width modified Booth multiplier", *IEEE Trans. VLSI Systems*, 2004, 12, (5), pp. 522-531
26 J. P. Wang, S. R. Kuang, and S. C. Liang: "High-accuracy fixed-width modified Booth multipliers for lossy applications", *IEEE Trans. VLSI Systems*, 2011, 19, (1), pp. 52-60
27 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*, 2012, 59, (3), pp. 594-603
28 C. Liu, J. Han, and F. Lombardi: "A low-power, high-performance approximate multiplier with configurable partial error recovery". *Proc. Design Automation & Test in Europe (DATE)*, 2014, pp. 95-98
29 Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli: "Image quality assessment: from error visibility to structural similarity", *IEEE Trans. Image Processing*, 2004, 13, pp. 600-612
30 W. Liu , T. Cao , P. Yin , Y. Zhu , C. Wang , Earl E. Swartzlander and F. Lombardi: "Design and Analysis of Approximate Redundant Binary Multipliers", *IEEE Trans. Computers*, 2019, 68, (6), pp.804 -819
31 E. Forgy: "Cluster analysis of multivariate data: Efficiency versus interpretability of classifications", *Biometrics*, 1965, 21, pp. 768-769
32 L. Rushi, D. Snehlata, and M. Latesh: "Class imbalance problem in data mining: Review", *Int. J. Comput. Sci.Netw.*, 2013, 2, pp. 83-87
33 K. Bache and M. Lichman: "UCI machine learning repository", 2013, Available at http://archive.ics.uci.edu/ml
34 Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner: "Gradient-based learning applied to document recognition", *Proceedings of the IEEE*, 1998, 86, (11), pp. 2278-2324
35 Y. LeCun, C. Cortes, and C. J.C. Burges: "The MNIST database of handwritten digits", 2010, Available at http://yann.lecun.com/exdb/mnist/