J Sign Process Syst (2018) 90:641–654
DOI 10.1007/s11265-017-1280-4

Designs of Approximate Floating-Point Multipliers with Variable Accuracy for Error-Tolerant Applications

Peipei Yin[1] · **Chenghua Wang**[1] · **Weiqiang Liu**[1] · **Earl E. Swartzlander Jr.**[2] · **Fabrizio Lombardi**[3]

Received: 10 April 2017 / Revised: 11 July 2017 / Accepted: 1 September 2017 / Published online: 11 September 2017

Abstract Approximate/inexact computing has become an attractive approach for designing high performance and low power arithmetic circuits. Floating-point (FLP) arithmetic is required in many applications, such as digital signal processing, image processing and machine learning. Approximate FLP multipliers with variable accuracy are proposed in this paper; the accuracy and the circuit requirements of these designs are analyzed and assessed according to different metrics. It is shown that the proposed approximate FLP multiplier designs further reduce delay, area, power consumption and power-delay product (PDP) while incurring about half of the normalized mean error distance (NMED) compared with the previous designs. The proposed IFLPM24–15 is the most efficient design when considering both PDP and NMED. Case studies with three error-tolerant applications show the validity of the proposed approximate designs.

Keywords Approximate/inexact computing · Floating-point · Multiplier · Low power

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s11265-017-1280-4) contains supplementary material, which is available to authorized users.

✉ Weiqiang Liu
liuweiqiang@nuaa.edu.cn

[1] College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics, 29 Jiangjun Avenue, Nanjing, China

[2] Department of Electrical and Computer Engineering, University of Texas at Austin, 1 University Station C0803, Austin, TX, USA

[3] Department of Electrical and Computer Engineering, Northeastern University, 360 Huntington Avenue, Boston, MA, USA

1 Introduction

As an emerging paradigm, approximate (or inexact) circuit design [1] provides a new approach to address the high power consumption of integrated circuits. An approximate circuit can save both static and dynamic power by adopting a simplified design; moreover, speed can be increased due to a shorter critical path delay. Although accuracy is reduced, for most applications computational results using inexact circuits are still of a reasonable quality; for example, multimedia (such as audio, image and video) can tolerate errors due to the limits of human perception. Therefore, full accuracy is not always necessary, thus enabling inexact computing for these types of applications.

Floating-point (FLP) numbers provide a wider dynamic range of representable real numbers compared with fixed-point representations; however, FLP hardware incurs substantial power consumption, FLP multipliers in particular are very expensive in a processor's power dissipation. There are many applications that require a wide range of numbers and can be tolerant to some errors; so inexact FLP computation is also viable [1–3]. FLP multiplication is typically one of the most frequent power consuming operations; so, the focus of this paper is to design approximate FLP multipliers with low power and moderate errors. New metrics for evaluating the reliability and power efficiency of inexact designs have been proposed in [4].

Recent research on inexact fixed-point multipliers has shown that an inexact 8 × 8-bit multiplier with an average normalized mean error distance (NMED) of 0.18 achieves up to 37% improvement in power consumption [5]. Although inexact fixed-point arithmetic circuits have been extensively studied [6–14], FLP arithmetic circuits have not been fully considered for inexact computing. In [15–18], the power reduction of a FLP multiplier is accomplished by truncating the least significant mantissa bits of the input data; however, when this reduction is more than 66%, a tolerable accuracy is not preserved. An algorithmic-level simplification for FLP multiplication is proposed in [19]; for a 32-bit FLP multiplier, the 24 × 24-bit mantissa multiplication is replaced by a 25 × 25-bit addition. In [20], Mitchell's algorithm is applied to mantissa multiplication for FLP multiplication.

This paper is an extended version of a previous work [21], which introduced techniques to design approximate FLP multipliers to reduce power dissipation under a target accuracy. The approximation is realized by simplifying the mantissa multiplier and ignoring the rounding unit. Error and circuit-level metrics due to inexact schemes are also measured to evaluate the designs of inexact FLP multipliers. Compared with [21], the main differences between them are summarized as follows:

- A new approximate modified Booth encoding algorithm with a smaller error is proposed.
- A simpler inexact 4–2 compressor is proposed for partial products accumulation.
- Detailed analysis and simulation results for the FLP multiplier designs are presented.
- Case studies with three error-tolerant applications (i.e., low-pass equiripple finite impulse response filter, HDR image processing and data classifier) are provided.

The rest of this paper is organized as follows. Section 2 provides a brief review on FLP arithmetic and the exact FLP multiplier architecture. Section 3 presents the designs of inexact FLP multipliers. Section 4 assesses the accuracy and circuit-level metrics of different inexact designs. Section 5 shows three applications of the proposed FLP multiplier designs. Finally, conclusions are provided in Section 6.

2 Review

As per the IEEE 754 Standard [22], a single precision FLP number is represented as a string that is made of 23 mantissa bits, 8 exponent bits and 1 sign bit (Fig. 1). The exponent part has a bias of $2^{E-1}-1$, where E is the number of exponent bits. The actual mantissa bit length is 24 bits including the "hidden 1 bit" located before the binary point; the value of a normalized floating point number is calculated as

$$FP = (-1)^{Sign} \times 2^{Exponent-bias} \times (1.Mantissa). \qquad (1)$$

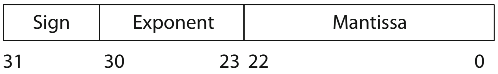


Figure 1 IEEE 754 floating-point format.

The architecture of a FLP multiplier is shown in Fig. 2. Multiplication of two FLP numbers requires a signed logic operation, exponent addition and mantissa multiplication. The sign bit of the result is calculated by an XOR operation on the sign bits of the operands. The result of the mantissa multiplication must be in normalized form to store the result in the IEEE standard format. Normalization is followed by the rounding unit; special cases (such as overflow, underflow, and not a number) are also detected and represented by flags.

3 Proposed Approximate FLP Multipliers

The inexact design of an FLP multiplier originates at an architectural level (Fig. 2); [15] summarizes the average power consumption of the FLP multiplier. The mantissa multiplier dominates the power dissipation (it accounts for over 80%) followed by the rounding unit with a power consumption of nearly 18%. Therefore, to reduce the power dissipation of a FLP multiplier, efforts are focused on the mantissa multiplier as well as the rounding unit. In the following subsections, the proposed circuit-level inexact designs are discussed in detail.

3.1 Mantissa Multiplier

In a FLP multiplier, the mantissa multiplier computes the product of two unsigned fixed-point numbers. A fast fixed-point multiplier is usually composed of three parts: partial product generation, a carry save adder (CSA) tree to reduce the partial product matrix to an addition of only two operands, and a carry propagation adder (CPA) to produce the binary result. Both the first and second parts significantly affect the delay, power consumption and circuit complexity of the entire multiplier. For the approximate FLP multipliers, an approximate modified Booth encoding (AMBE) algorithm is used to

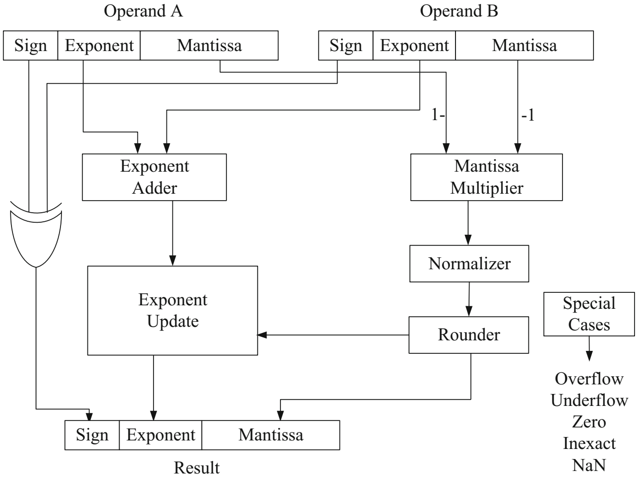


Figure 2 The accurate FLP multiplier architecture.

generate the inexact partial product (PP); an inexact 4–2 compressor is utilized to design the approximate array multiplier; a Wallace tree is used as tree structure.

The AMBE algorithm is achieved from the modified Booth encoding (MBE) algorithm [23]. The expression for the exact MBE algorithm is given as follows:

$$PP_j = (X_{2i} \oplus X_{2i-1})(X_{2i+1} \oplus Y_j) + \overline{(X_{2i} \oplus X_{2i-1})}(X_{2i+1} \oplus X_i)(X_{2i+1} \oplus Y_{j-1}). \tag{2}$$

To simplify the MBE, the AMBE algorithm introduces 6 incorrect outputs out of 16 outputs. Table 1 shows the Karnaugh map for the AMBE algorithm, where $\boxed{1}$ denotes a ‘0’ entry that has been replaced by a ‘1’ and $\boxed{0}$ denotes a ‘1’ that has been replaced by ‘0’. It also reveals the difference between the exact outputs of exact Booth encoding (PP) and the inexact outputs of approximate Booth encoding (APP). The proposed AMBE introduces 3 positive errors and 3 negative errors out of 16 outputs. The expression for the proposed AMBE is as follows:

$$PP_j^{'} = (X_{2i} + X_{2i-1})(X_{2i+1} \oplus Y_j). \tag{3}$$

An exact *N*-2 compressor consists of *N*-2 full adders; which are expressed as follows:

$$S_{um} = PP_1 \oplus PP_2 \oplus PP_3 \oplus PP_4 \oplus C_{in} \tag{4}$$

$$C_{out} = (PP_1 \oplus PP_2)PP_3 + \overline{PP_1 \oplus PP_2}PP_1 \tag{5}$$

$$C_{arry} = (PP_1 \oplus PP_2 \oplus PP_3 \oplus PP_4)C_{in} + \left(\overline{PP_1 \oplus PP_2 \oplus PP_3 \oplus PP_4}\right)PP_4, \tag{6}$$

where C_{in} is the carry bit from the position to the right, C_{out} is the carry bit into the higher position, and S_{um} and C_{arry} are the two output bits in positions *i* and *i* + 1, respectively. The inexact 4–2 compressors are designed without C_{in} and C_{out}. C_{out} is needed only when all inputs are 1 s. When all inputs are 1 s, two bits (11) are used as C_{arry} and S_{um} instead of three bits (100). In this new design, the expressions for approximate 4–2 compressor are as follows:

$$S_{um}^{'} = \overline{PP_3 \oplus PP_4} \tag{7}$$

$$C_{arry}^{'} = \overline{\overline{PP_1 + PP_2} + \overline{PP_3 + PP_4}}. \tag{8}$$

Table 1 Karnaugh map of the proposed AMBE.

$X_{2i+1}X_{2i}X_{2i-1}$ Y_jY_{j-1}	000	001	011	010	110	111	101	100
00	0	0	0	0	1	$\boxed{1}$	1	$\boxed{0}$
01	0	0	$\boxed{0}$	0	1	$\boxed{1}$	1	0
11	0	1	1	1	0	0	0	0
10	0	1	$\boxed{1}$	1	0	0	0	$\boxed{0}$

Table 2 shows the truth table of the approximate design for a 4–2 compressor; this table also shows the difference between the exact decimal value of the addition of the inputs and the decimal value of the outputs produced by the approximate compressor. This design has 1 positive error and 7 negative errors out of 16 outputs. Although this design has 8 incorrect outputs out of 16 outputs, the expression of S_{um} is simplified and the power dissipation and the delay are reduced.

Table 2 Truth table of proposed compressor.

PP_4	PP_3	PP_2	PP_1	$C_{arry}^{'}$	$S_{um}^{'}$	*Difference*
0	0	0	0	0	1	1
0	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	0	1	−1
0	1	0	0	0	0	−1
0	1	0	1	1	0	0
0	1	1	0	1	0	0
0	1	1	1	1	0	−1
1	0	0	0	0	0	−1
1	0	0	1	1	0	0
1	0	1	0	1	0	0
1	0	1	1	1	0	−1
1	1	0	0	0	1	-1
1	1	0	1	1	1	0
1	1	1	0	1	1	0
1	1	1	1	1	1	-1

In a single precision FLP multiplier, the mantissa multiplier is an unsigned 24-bit multiplier. For the 24-bit multiplier, the AMBE algorithm generates twelve unsigned digits; hence, twelve partial products are generated. The dot-notation of the partial products for the 24-bit multiplier is shown in Fig. 3; the extension elimination technique is used. In Fig. 3, a dot represents a bit of a partial product, and a box with a solid line represents a 4–2 compressor. The partial products are accumulated by a Wallace tree that reduces the partial product rows until only two rows remain. In this tree structure, the 48-bit result has to be rounded to 24-bits, so the lower part of the partial product (identified by the red dotted square in Fig. 3) is truncated (the truncation analysis is provided in Section 4). The approximate calculation from the 24th to the

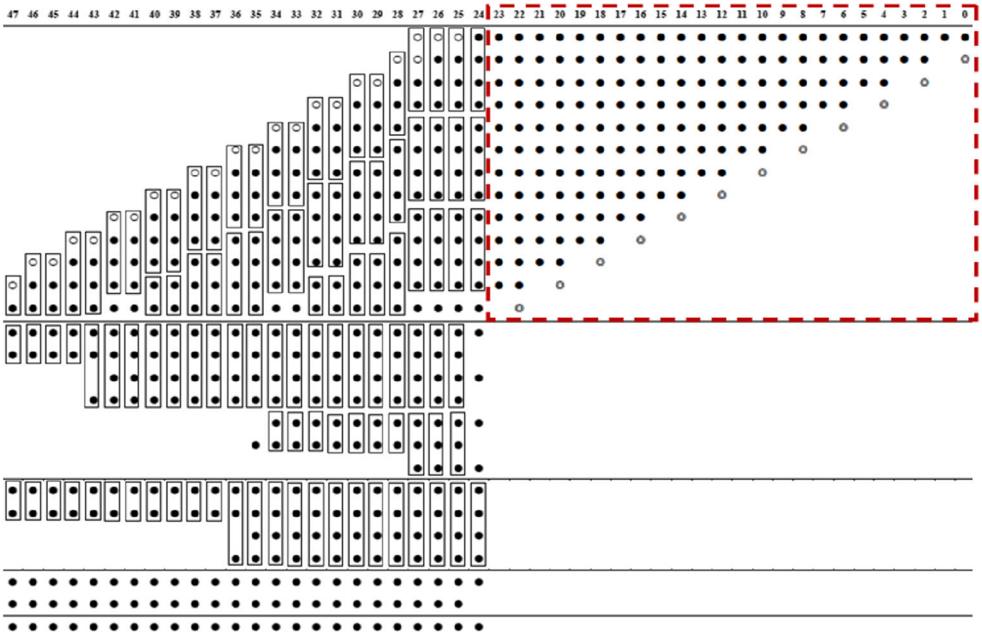


Figure 3 24 × 24 MBE partial product array.

47th bits can be varied such that if the accuracy requirement can be still relaxed further, more inexact bits can be utilized.

3.2 Rounder

The product of two n-bit numbers can be $2n$–bit wide; the result of the FLP multiplication however, must fit into the same n bits as the multiplier and the multiplicand, this often leads to a loss of precision. A rounder is needed to minimize the loss of precision; however, in an inexact design in which computation may be approximate, the rounding unit is not required. The rounding unit is also the second most energy consuming unit of a FLP multiplier; therefore, the rounding unit is ignored in an inexact design.

4 Simulation Results

In this section, the accuracy and hardware requirements of the proposed approximate mantissa multiplier and the single precision FLP multiplier designs are assessed. For comparison, the approximate FLP multiplier designs in [21] and two recent approximate multiplier designs in [19, 20] are also implemented and evaluated.

4.1 Accuracy

Inaccuracy in an inexact FLP design is due mainly from the mantissa. To quantify the accuracy of the inexact FLP multiplier, the error from the inexact mantissa multiplier is first measured. The acronyms of various mantissa multipliers and FLP multipliers as evaluated in this manuscript are shown in Tables 3 and 4. Approximate FLP multipliers with previous algorithm proposed in [21] and the two approximate FLP multipliers (namely, NAFLPM [19] and MAFLPM [20]) along with their mantissa multipliers (i.e., NAMM, MAMM) are simulated and assessed. The normalized mean error distance (NMED) is defined as the normalization of the mean error distance (MED) by the maximum output of the accurate design [4].

Assume that the input data is uniformly distributed in the full range of the 24-bit fixed-point format. The functions of the proposed inexact mantissa multipliers are simulated using C++, their NMEDs are reported in Table 5; so, the impact of truncation is less than 10^{-6} in the NMED and even when 19 bits in the mantissa multiplier are inexact, the NMED value is only 0.005581, which is less than the design of [19]. The NMEDs of the proposed approximate designs are almost half of previous approximate designs in [21].

Table 3 Acronyms of mantissa multipliers.

Acronym	Description
MM	Accurate mantissa multiplier
IMM*i*	Inexact mantissa multiplier with i-bit truncation in partial product
IMM*i-j*	Inexact mantissa multiplier with i-bit truncation and j-bit inexact design in partial product using proposed algorithm in this paper
IMM*i-j* [21]	Inexact mantissa multiplier with i-bit truncation and j-bit inexact design in partial product using previous algorithm in [21]
NAMM [19]	Approximate mantissa multiplier neglecting an additional term
MAMM [20]	Approximate mantissa multiplier using Mitchell's algorithm

Table 4 Acronyms of FLP multipliers.

Acronym	Description
FLPM	Accurate single precision FLP multiplier
IFLPM*i*	Inexact FLP multiplier with i-bit truncation in partial product of mantissa multiplier
IFLPM*i-j*	Inexact FLP multiplier with i-bit truncation and j-bit inexact design in partial product of mantissa multiplier using proposed algorithm in this paper
IFLPM*i-j* [21]	Inexact FLP multiplier with i-bit truncation and j-bit inexact design in partial product of mantissa multiplier using previous algorithm in [21]
NAFLPM [19]	Approximate FLP multiplier neglecting an additional term in mantissa multiplier
MAFLPM [20]	Approximate FLP multiplier using Mitchell's algorithm in mantissa multiplier

Also the inexact FLP multipliers using the above inexact mantissa multipliers are simulated. In many DSP applications (such as image processing and voice recognition), the data range is usually limited to $2^4 \sim 2^{10}$. In this paper, the data range is fixed to $0 \sim 10^5$ i.e., the dynamic range of luminance that a human can perceive. The NMEDs are reported in Table 6; the trend of the NMEDs with different numbers of inexact bits for an approximate FLP multiplier is shown in Fig. 4. Table 6 and Fig. 4 also show the comparison of NMEDs among the proposed FLP multipliers, previous FLP multipliers and two recent FLP multipliers.

- By increasing the number of inexact bits, the NMED grows slowly up to 39 inexact bits (24-bit truncated and 15-bit inexact); the NMED of the inexact FLP multiplier with 15 inexact bits and 24 truncated bits in a mantissa multiplier is 0.001195 which is between that of NAFLPM and MAFLPM.
- When the number of inexact bits is more than 39 where bit length is 48, NMED is increased rapidly; and the NMED of its FLP multiplier is 0.1185 for IFLPM24–24 whose inexact bit length is 48.
- Compared with NAFLPM which uses a 25 × 25-bit addition to replace the 24 × 24-bit mantissa multiplication, the NMEDs of the proposed FLP multipliers are smaller except IFLPM24–19 and IFLPM24–24 (the number of inexact bits is more than 43).

Table 5 NMED of approximate mantissa multipliers

Mantissa Multipliers	NMED
IMM5	2.0047E-12
IMM10	8.755E-12
IMM20	1.397E-8
IMM22	6.147E-8
IMM24	2.757E-7
IMM24–5	3.308E-6
IMM24–7	3.161E-5
IMM24–11	1.954E-4
IMM24–15	1.991E-3
IMM24–19	2.009E-2
IMM24–24	2.994E-1
IMM24–5 [21]	8.572E-6
IMM24–7 [21]	3.099E-5
IMM24–11 [21]	3.887E-4
IMM24–15 [21]	5.377E-3
IMM24–19 [21]	5.581E-2
IMM24–24 [21]	5.200E-1
NAMM [19]	8.333E-2
MAMM [20]	2.976E-3

Table 6 NMED of approximate FLP multipliers.

FLP Multipliers	NMED
IFLPM5	1.232E-8
IFLPM10	1.232E-8
IFLPM20	2.028E-8
IFLPM22	3.545E-8
IFLPM24	1.237E-7
IFLPM24–5	1.218E-6
IFLPM24–7	5.226E-6
IFLPM24–11	8.922E-5
IFLPM24–15	1.195E-3
IFLPM24–19	1.372E-2
IFLPM24–24	1.185E-1
IFLPM24–5 [21]	3.617E-6
IFLPM24–7 [21]	1.299E-5
IFLPM24–11 [21]	2.199E-4
IFLPM24–15 [21]	2.531E-3
IFLPM24–19 [21]	2.864E-2
IFLPM24–24 [21]	2.322E-1
NAFLPM [19]	1.715E-2
MAFLPM [20]	5.928E-4

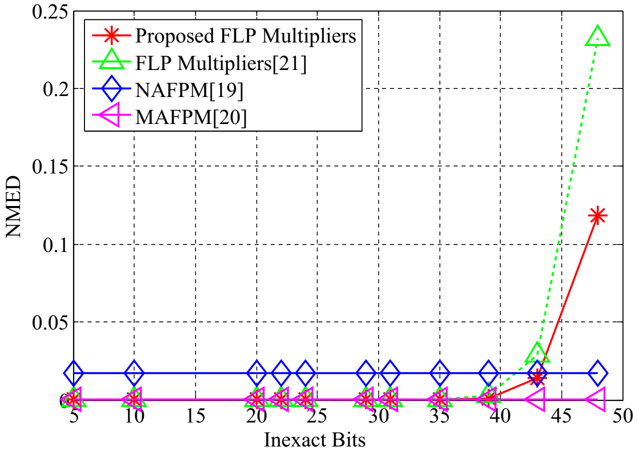


Figure 4 NMEDs of inexact FLP multipliers.

- Compared with MAFLPM which used logarithm algorithm to simplify the mantissa multiplication, among the proposed FLP multipliers except multipliers with only truncation used, there are three multipliers (IFLPM24–5, IFLPM24–7, IFLPM24–11) whose NMED is smaller.
- Compared with previous FLP multipliers in [21], the NMEDs of proposed FLP multipliers are all lower possibly due to the compensation of negative and positive error.

4.2 Circuit Assessment

All designs are implemented and simulated by Synopsys Design Compiler using the Nangate 45 nm Open Cell Library. NAFLPM and MAFLPM along with the mantissa multipliers NAMM and MAMM are also synthesized. In the simulation of each circuit design, a supply voltage of 1.25 V and room temperature are assumed. The power consumption is simulated by using the Synopsys Power Complier. The values of the power-delay product (PDP) and the area-delay product (ADP) are calculated as comprehensive metrics.

Table 7 summarizes the delay, area and power consumption of the proposed and previous mantissamultipliers. The first design is the exact 24-bit unsigned multiplication (MM); the 2nd to the 6th designs are the approximate multipliers using different numbers of truncated bits in the least significant bits of the partial products. The following six designs are the new approximate multipliers with truncation by utilizing the inexact algorithm proposed in Section 3.1. The next six designs are the previous approximate multipliers with truncation by utilizing the inexact algorithm proposed in [21]. NAMM and MAMM are the two approximate multipliers using an algorithmic-level simplification [19, 20]. Table 8 summarizes the delay, area and power consumption of the FLP multipliers using the above mantissa multipliers when the rounding unit is ignored. The delay, area, power and PDP metrics are compared.

- When all bits in the mantissa multiplier are inexact, the power of the proposed FLP multiplier (shown in Fig. 5) is 680.3 μW (reduction by 67%), which is lower than that of NAMM (i.e. it reduces the power by 66%), but higher than that of NAFLPM (reduces the power by 79%).
- Even when all bits are inexact, the area of all proposed FLP multipliers (shown in Fig. 6) is still higher than that of NAFLPM and MAFLPM.
- Compared with MAFLPM (reduction in delay of 22%), the delay of the proposed FLP multipliers (shown in Fig. 7) is reduced significantly. When the truncated bit length is 5, the delay of the proposed FLP multiplier is already reduced up to 23%. Moreover when all bits are inexact, the delay of the proposed FLP multiplier can be reduced up to 50% which is a little higher than that of NAFLPM (at a reduction in delay of 49%).

Table 7 Circuit assessment of mantissa multipliers.

Mantissa Multipliers	Power(μW)	Area(μm^2)	Delay(ns)	PDP(fJ)	ADP($\mu m^2 \cdot ns$)
MM	1464.1	6003.9	1.68	2459.7	10,080.7
IMM5	1441.2	5628.3	1.57	2262.7	8836.4
IMM10	1265.7	5120.0	1.50	1743.8	7363.4
IMM20	926.3	3743.7	1.42	1315.3	5316.1
IMM22	818.9	3336.9	1.58	1293.9	5272.3
IMM24	718.0	2956.9	1.50	1077.0	4435.4
IMM24–5	574.5	2419.8	1.39	798.6	3393.5
IMM24–7	523.0	2226.4	1.32	690.4	2938.8
IMM24–11	416.0	1876.9	1.25	520.0	2346.1
IMM24–15	354.7	1628.2	1.07	379.5	1742.2
IMM24–19	308.8	1464.6	1.09	336.6	1596.4
IMM24–24	286.9	1371.5	1.07	307.0	1467.5
IMM24–5 [21]	583.8	2503.9	1.39	811.5	3480.4
IMM24–7 [21]	513.9	2263.9	1.72	833.9	3893.9
IMM24–11 [21]	436.3	2028.2	1.50	654.5	3042.3
IMM24–15 [21]	352.3	1800.0	1.40	493.2	2520.0
IMM24–19 [21]	319.3	1681.9	1.35	431.1	2270.6
IMM24–24 [21]	296.7	1589.6	1.34	397.6	2130.1
NAMM [19]	48.2	196.0	1.48	71.3	290.1
MAMM [20]	635.1	1969.7	1.64	1041.6	3230.3

Table 8 Circuit assessment of FLP multipliers.

FLP Multipliers	Power(μW)	Area(μm^2)	Delay(ns)	PDP(fJ)	ADP($\mu m^2 \cdot ns$)
FLPM	2055.7(100%)	7997.3(100%)	9.71(100%)	19,960.8(100%)	77,653.8(100%)
IFLPM5	1941.0(94%)	7440.6(93%)	7.46(77%)	14,479.9(73%)	55,506.9(71%)
IFLPM10	1798.2(87%)	7002.4(88%)	7.06(73%)	12,695.3(64%)	49,437.0(64%)
IFLPM20	1311.4(64%)	5528.3(69%)	6.25(64%)	8196.3(41%)	34,550.0(44%)
IFLPM22	1278.0(62%)	5228.0(65%)	6.06(62%)	7744.7(39%)	31,681.7(41%)
IFLPM24	1162.9(57%)	4843.1(61%)	5.86(60%)	6814.6(34%)	28,380.6(37%)
IFLPM24–5	990.7(48%)	4079.4(51%)	5.59(58%)	5538.0(27%)	22,803.8(30%)
IFLPM24–7	933.6(45%)	4057.6(51%)	5.29(54%)	4938.7(25%)	21,464.7(28%)
IFLPM24–11	834.2(41%)	3727.7(47%)	4.98(51%)	4154.3(21%)	19,019.6(24%)
IFLPM24–15	759.0(37%)	3465.0(43%)	4.91(51%)	3726.7(19%)	17,013.2(22%)
IFLPM24–19	705.3(34%)	3291.0(41%)	4.89(50%)	3449.0(17%)	16,093.0(21%)
IFLPM24–24	680.3(33%)	3182.7(40%)	4.85(50%)	3299.5(17%)	15,436.1(20%)
IFLPM24–5 [21]	1004.3(49%)	4306.3(54%)	5.41(56%)	5433.3(27%)	23,297.1(30%)
IFLPM24–7 [21]	950.9(46%)	4121.9(51%)	5.30(55%)	5039.8(25%)	21,846.1(28%)
IFLPM24–11 [21]	859.8(42%)	3819.2(48%)	4.98(51%)	4281.8(21%)	19,019.6(24%)
IFLPM24–15 [21]	784.9(38%)	3576.1(45%)	4.92(51%)	3861.7(19%)	17,594.4(23%)
IFLPM24–19 [21]	739.6(36%)	3418.1(43%)	4.90(50%)	3624.0(18%)	16,748.7(22%)
IFLPM24–24 [21]	716.1(35%)	3317.6(41%)	4.86(50%)	3480.2(17%)	16,123.5(21%)
NAFLPM [19]	434.2(21%)	1924.8(24%)	4.95(51%)	2149.3(11%)	9527.8(12%)
MAFLPM [20]	708.1(34%)	2903.7(36%)	7.58(78%)	5367.4(27%)	22,010.0(28%)

- When the inexact bit length grows to 31 (24-bit truncated and 7-bit inexact), the PDPs of the proposed FLP multipliers (shown in Fig. 8) are lower than that of MAFLPM (reduces the PDP by 73%). Even when all bits are inexact, the PDP of the proposed FLP multiplier is still higher than that of NAFLPM, (i.e. it reduces the PDP by 89%).
- The trend for the PDP provided in Fig. 8 shows that an increase in the number of inexact bits for the proposed approximate FLP multiplier leads to a reduction in PDP.
- The ADPs in Table 8 show almost the same trend as the PDPs for all of the approximate multipliers.
- Compared with previous FLP multipliers in [21], the delay, area, power consumption and PDP of the proposed FP multipliers are improved.

The NMEDs and PDPs of all approximate FLP multipliers using truncation and the inexact algorithm (except IFLPM24–24 with a high NMED) are shown in Fig. 9. Among the single precision FLP multipliers, NAFLPM is more power efficient than the other approximate multipliers; however, its NMED is larger than the others, except IFLPM24–19. Although

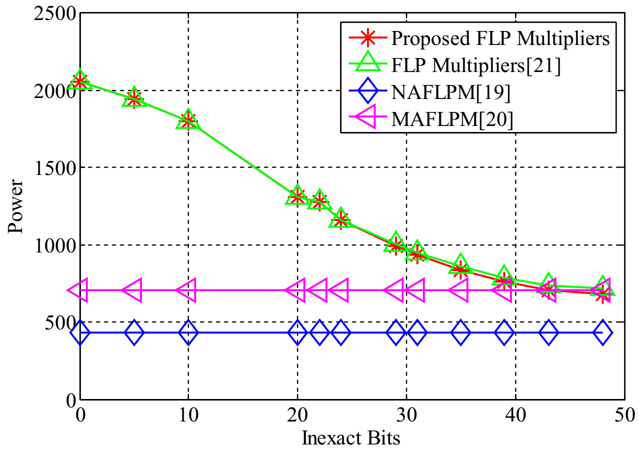


Figure 5 Power comparison of inexact FLP multipliers.

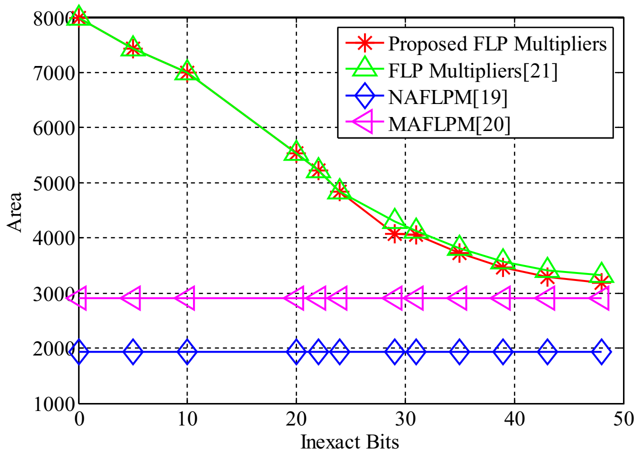


Figure 6 Area comparison of inexact FLP multipliers.

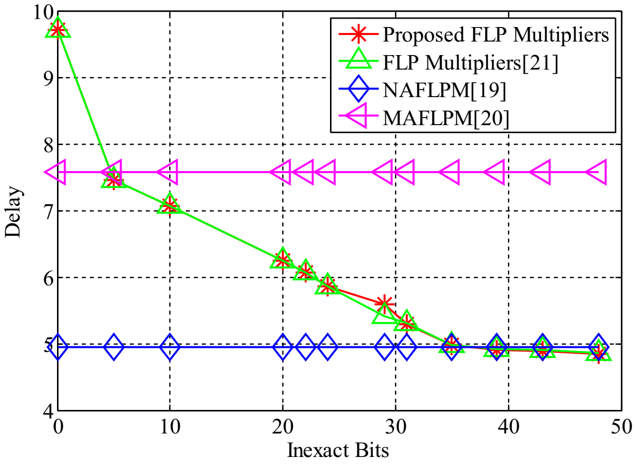


Figure 7 Delay comparison of inexact FLP multipliers.

IFLPM24–5, IFLPM24–5 [21] and MAFLPM have similar PDP (about 27% of FLPM), MAFLPM has a larger NMED (5.928×10^{-4}). Among the proposed single precision FLP multipliers, IFLPM24–15 is the most efficient design with moderate values of NMED and PDP. IFLPM24–5 and IFLPM24–7 have a better NMED, but a larger PDP, while IFLPM24–19 shows the opposite features. Therefore, according to the accuracy requirements, different designs with variable accuracy can be applied. The results show that the proposed designs accomplish significant reductions in power dissipation and delay compared to an exact design.

5 Application Evaluation

In this section, the proposed inexact FLP multiplier designs and previous inexact FLP multiplier designs are applied to applications including digital signal processing, image processing and data classification. The applications are a 73-tap low-pass equiripple Finite Impulse Response (FIR) filter, a HDR image processor and a data classifier. These applications are chosen due to their inherent tolerance to inexactness. All of the applications are simulated by MATLAB.

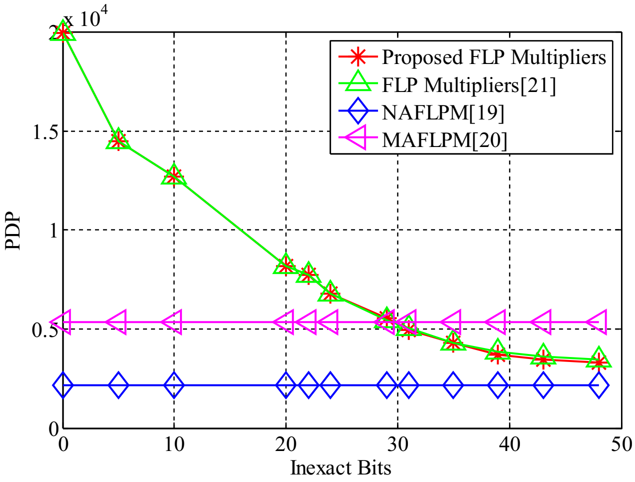


Figure 8 PDP comparison of inexact FLP multipliers.

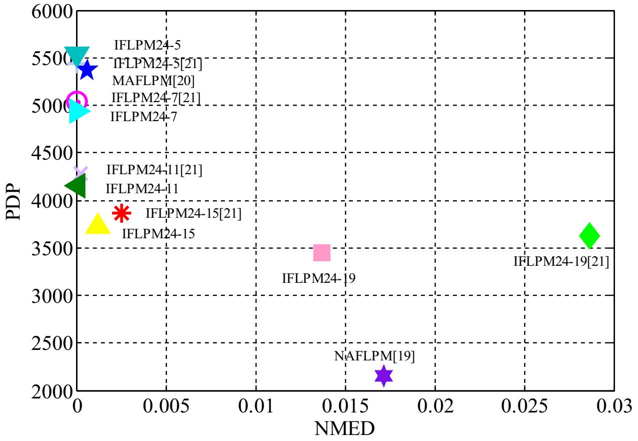


Figure 9 NMEDs and PDPs of inexact FLP multipliers.

5.1 FIR Filter

The FIR filter is designed by the Filter Design and Analysis (FDA) Tool in MATLAB. The pass-band and stop-band frequencies of the filter are 8 *kHz* and 15 *kHz*, respectively. The input of the FIR filter is the sum of three sinusoidal variables $x_1(n)$, $x_2(n)$ and $x_3(n)$ with 1 *kHz*, 15 *kHz*, and 20 *kHz* frequencies respectively, and a White Gaussian Noise $w(n)$ with -30*dBW* power, i.e., $x(n) = x_1(n) + x_2(n) + x_3(n) + w(n)$. White Gaussian Noise is used to simulate the random effects found in nature.

The approximate single precision FLP multipliers are applied to compute the output of the filter, while the adders used here are accurate. To assess the performance of the approximate multipliers for the FIR filter operation, the input signal-to-noise ratio (SNR_{in}) and output signal-to-noise ratio (SNR_{out}) are used. The same input signal with SNR_{in} of −3.01 dB (due to a randomly generated White Gaussian Noise) is utilized for all operations.

The simulation results of the FIR filter operation are shown in Fig. 10, in which the SNR_{out}s are sorted in descending order. Overall, the multipliers with truncation (IFLPM5, IFLPM10, IFLPM20, IFLPM22 and IFLPM24) and small number inexactness with 24-bit truncation (IFLPM24–5, IFLPM24–7, IFLPM24–11, IFLPM24–5 [21], IFLPM24–7 [21] and IFLPM24–11 [21]) have similar SNR_{out} value with the exact multiplier design (FLPM). By increasing the number of inexact bits, SNR_{out} for the proposed FLP multipliers decreases slowly up to 39 inexact bits (24-bit truncated and 15-bit inexact). The SNR_{out} of IFLPM24–19 is 26.03 dB, higher than for NAFLPM but lower than for MAFLPM. The SNR_{out}s using the proposed FLP multipliers are slightly higher than using the previous FLP multipliers with the same number

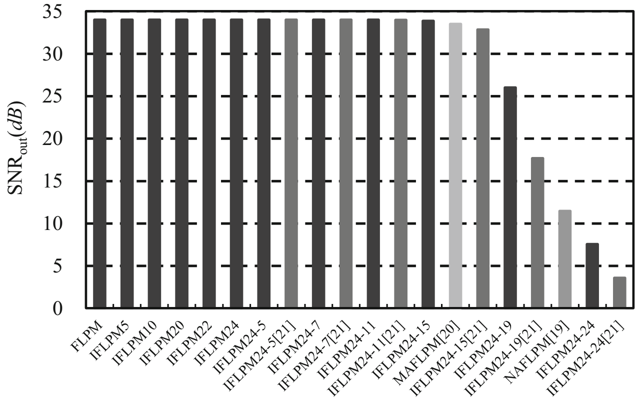


Figure 10 SNR$_{out}$ for the exact and inexact FLP multipliers.

inexact bits. The trend of the SNR$_{out}$ is opposite of the NMEDs, mainly because the loss in SNR occurs due to the large error distance of these multipliers detailed in Section 4.1.

Table 9 Overall Visibility (P_det) and Quality Scores (Q_MOS) for inexact FLP multipliers using 'StillLife.exr'.

Inexact FLP Multiplier	P_det	Q_MOS
IFLPM5 ~ 24	0	99.9996
IFLPM24–5	0	99.9996
IFLPM24–7	0	99.9996
IFLPM24–11	2.96114E-010	99.9996
IFLPM24–15	1.65349e-006	99.9996
IFLPM24–19	0.0189922	99.9861
IFLPM24–24	1	99.2024
IFLPM24–5 [21]	0	99.9996
IFLPM24–7 [21]	0	99.9996
IFLPM24–11 [21]	1.43462E-009	99.9996
IFLPM24–15 [21]	3.25882E-005	99.9994
IFLPM24–19 [21]	0.168052	99.9527
IFLPM24–24 [21]	1	97.2166
NAFLPM [19]	0.152252	99.9637
MAFLPM [20]	5.78118E-006	99.9995

5.2 HDR Image Application

The proposed inexact FLP multiplier designs are applied to two HDR OpenEXRimages ('StillLife.exr' and 'Ocean.exr'). OpenEXR is a HDR image file format developed by Industrial Light & Magic [24]; it is widely used in computer imaging applications. It supports both the IEEE half and single precision FLP formats. The High Dynamic Range Visible Difference Predictor (HDR-VDP) [25] is a visual metric to evaluate the inexact FLP multipliers targeting high dynamic range image processing applications. It compares a pair of images (reference and test images) and predicts the probability that their visual difference is visible to an average observer. HDR-VDP works within the complete range of luminance that the human eye sees and produces subjective comparison results.

'StillLife.exr' and 'Ocean.exr' are used in this paper and they have typical dynamic ranges in the OpenEXR image set; their data ranges are $2^{-9} \sim 2^{9}$ and $2^{-12} \sim 2^{12}$, respectively. The overall visibility, i.e., P_det, is defined as the probability that the differences between the images are visible for an average observer; the quality, i.e., Q_MOS, is defined as the degradation with respect to the reference image, expressed as a mean-opinion-score. P_det has a range of 0 to 1 and Q_MOS has a range of 0 to 100. A higher value of P_det means that it is more likely a difference can be observed; a higher value of Q_MOS means that the image has a better quality. Therefore, Q_MOS is more relevant when evaluating the quality of a processed image. In this simulation, it is assumed that the diagonal display size is 12 in., resolution is 1024 by 768, the viewing distance is 0.5 m, and the color encoding is a sRGB display.

Tables 9 and 10 shows the overall visibility P_det and quality Q_MOS of inexact FLP multipliers for 'StillLife.exr' and 'Ocean.exr' respectively. The results for the overall visibility of IFLPM5 ~ 24 (IFLPM5, IFLPM10, IFLPM20, IFLPM22 and IFLPM24), IFLPM24–5,IFLPM24–7, IFLPM24–5 [21] and IFLPM24–7 [21] are 0. These results show that the differences between the exact and inexact results are so small that observers cannot perceive them. When all bits in the multipliers are inexact, the P_det is 1, so the differences are easy to find. The trend of P_det is similar to NMED. The Q_MOSs for all inexact multipliers using the image

Table 10 Overall Visibility (P_det) and Quality Scores (Q_MOS) for inexact FLP multipliers Using 'Ocean.exr'.

Inexact FLP Multiplier	P_det	Q_MOS
IFLPM5 ~ 24	0	99.9996
IFLPM24–5	0	99.9996
IFLPM24–7	0	99.9996
IFLPM24–11	4.10116E-009	99.9996
IFLPM24–15	1.70448e-005	99.9995
IFLPM24–19	0.0618347	99.9702
IFLPM24–24	1	98.5514
IFLPM24–5 [21]	0	99.9996
IFLPM24–7 [21]	0	99.9996
IFLPM24–11 [21]	1.65287E-008	99.9996
IFLPM24–15 [21]	3.08757E-004	99.9989
IFLPM24–19 [21]	0.799399	99.8706
IFLPM24–24 [21]	1	88.0136
NAFLPM [19]	0.602509	99.9085
MAFLPM [20]	1.55393E-005	99.9995

'StillLife.exr' are only slightly lower than 100, so the quality for all inexact FLP multipliers are very good. For the even higher dynamic range image (i.e. 'Ocean.exr'), the results for the overall visibility are all lower than that of 'StillLife.exr', i.e., the differences between the exact and inexact results are more likely to be observed. The quality of 'Ocean.exr' is still very good except IFLPM24–24 [21] whose quality score is 88.0136.

The HDR-VDP detection maps of 'StillLife.exr' and 'Ocean.exr' for the exact and inexact multiplied images by the inexact FLP multiplier designs are shown in Figs. 11 and 12 respectively. The default map is represented as a multiple color (blue, cyan, green, yellow and red) picture. Red denotes high probability and blue denotes low probability. As shown in Fig. 11, except IFLPM24–24 and IFLPM24–24 [21], the differences for inexact FLP multipliers are not visible to the average human observer. From Fig. 12, two other multipliers IFLPM24–19 [21] and NAFLPM have a small probability to observe the differences.

5.3 Data Classifier

To evaluate the proposed FLP multipliers further, a classifier designed by a feedforward neural network is simulated. The neural network [26, 27] with acyclic connection from an input layer to an output layer is a common structure due to its simplicity compared to a recurrent neural network. In each layer, a neuron has multiple input connections called weights from some (or all) of neurons in the previous layer. The output of the network is determined by feedforward from the input layer to the output layer with all weights. To train the network, each weight is updated to minimize the error in the output layer generally with a given desired output. To update the weight, backpropagation (BP) algorithm is used in this network.

In this classifier, three applications from the UCI Machine Learning repository to serve as a "benchmark" suite [28] are selected. The benchmarks used in this paper include Iris, Wine and Breast. The two selection criteria were application diversity and having less than 100 attributes. For each application,

Figure 11 The HDR-VDP detection map of 'StillLife.exr' for the two multiplied images.

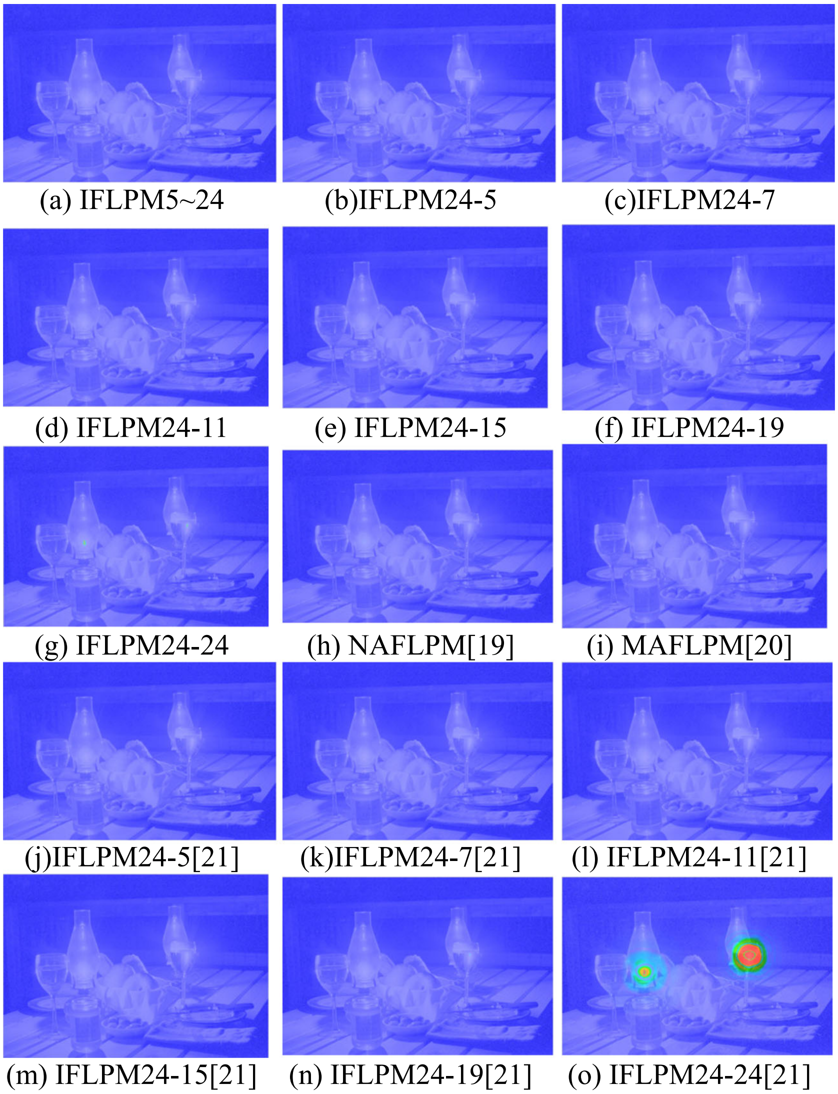
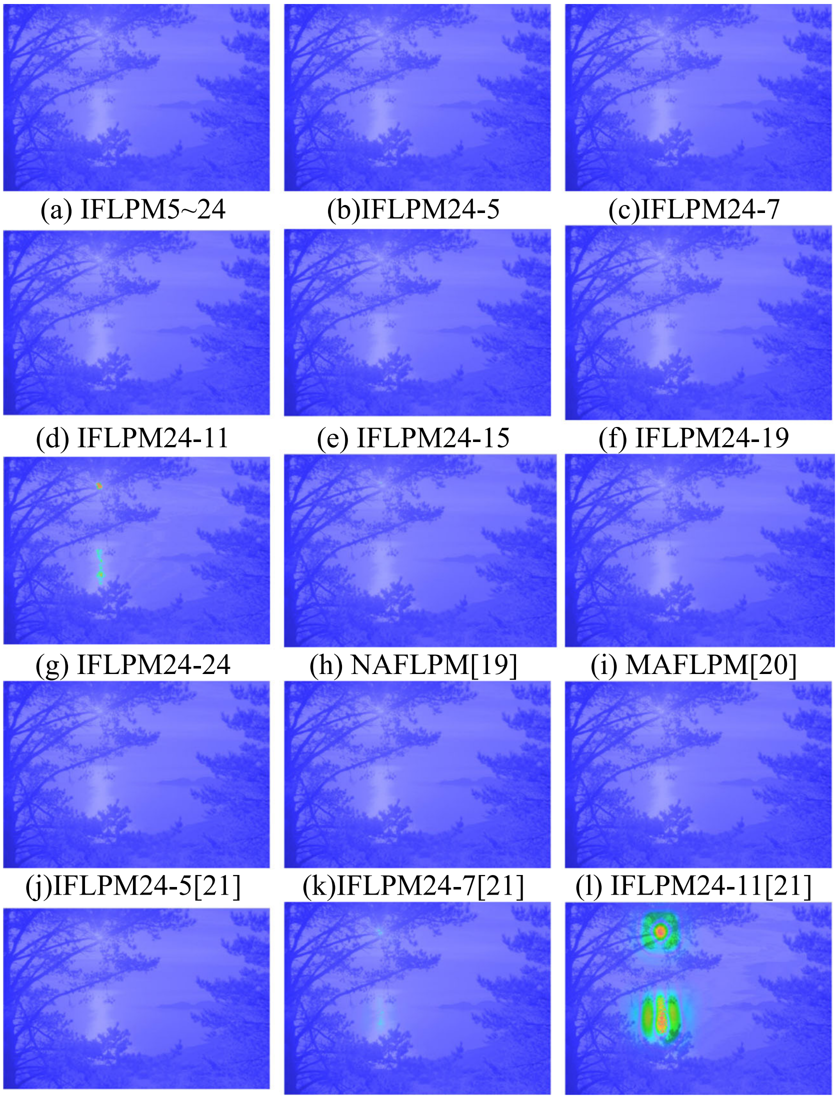


Figure 12 The HDR-VDP detection map of 'Ocean.exr' for the two multiplied images.



(a) IFLPM5~24 (b)IFLPM24-5 (c)IFLPM24-7

(d) IFLPM24-11 (e) IFLPM24-15 (f) IFLPM24-19

(g) IFLPM24-24 (h) NAFLPM[19] (i) MAFLPM[20]

(j)IFLPM24-5[21] (k)IFLPM24-7[21] (l) IFLPM24-11[21]

(m) IFLPM24-15[21] (n) IFLPM24-19[21] (o) IFLPM24-24[21]

the number of inputs and outputs are selected based on the typical number of attributes and classes in the examples of the UCI repository. The data characteristics are summarized in Table 11. The inexact FLP multipliers are used to replace the exact FLP multipliers multiplying the weights and neurons. The accuracy is defined as the percentage of matches between the assigned class and the predicted output class.

Table 11 Data characteristics.

Application	Description	#Input, #Output	η, #Hidden
Iris	Plants classification	4,3	0.2,6
Wine	Wine origin based on chemicals	13,3	0.2,4
Breast	Breast cancer diagnostic	30,2	0.1,14

Table 12 summarizes the classification accuracy using exact and inexact FLP multipliers in the network after same training iteration number. From Table 12, it is shown that even the design that uses the accurate multiplier cannot always classify all data correctly. Figure 13 shows the accuracy comparison for the accurate design, the proposed designs and the previous designs. Consider the Iris application first, the accuracies of all multipliers except IFLPM24–24 [21] whose accuracy is 76.7% are all higher than 93%. Due to the inherently error-tolerant capability of the neural network, the performence for IFLPM24–19 [21] is better than other multipliers, although its NMED is slightly higher. In the Wine application, except IFLPM24–24 whose accuracy is close to 90%, the accuracies of all multipliers are higher than 94%. For over half of the Wine applications, the accuracy remains 100%. In the Breast application, the accuracies produced by the inexact multipliers except IFLPM24–24 and IFLPM24–24 [21] are close to that produced by the exact multiplier.

Table 12 Classification accuracy using exact and inexact FLP multipliers in the BP network

	Iris	Wine	Breast
FLPM	96.7%	100%	96.7%
IFLPM5	96.7%	100%	96.7%
IFLPM10	96.7%	100%	96.7%
IFLPM20	96.7%	100%	96.7%
IFLPM22	96.7%	100%	96.7%
IFLPM24	96.7%	98.3%	96.7%
IFLPM24–5	96.7%	100%	97.3%
IFLPM24–7	96.7%	100%	96.0%
IFLPM24–11	96.7%	98.3%	96.0%
IFLPM24–15	96.7%	100%	97.3%
IFLPM24–19	96.7%	98.3%	97.3%
IFLPM24–24	93.3%	89.7%	88.7%
IFLPM24–5 [21]	96.7%	100%	96.0%
IFLPM24–7 [21]	96.7%	100%	97.3%
IFLPM24–11 [21]	96.7%	100%	97.3%
IFLPM24–15 [21]	96.7%	100%	96.7%
IFLPM24–19 [21]	100%	98.3%	96.7%
IFLPM24–24 [21]	76.7%	94.8%	88.7%
NAFLPM [19]	93.3%	96.6%	94.0%
MAFLPM [20]	96.7%	100%	96.0%

For all three applications, as long as the error is such that it does not affect the classification, the accuracy for the inexact multipliers can remain high enough. The error tolerance capability of neural networks improves the degree of inaccuracy that can be tolerated and thus, results in greater energy savings. Overall, the classifiers using the proposed approximate multipliers can even have better accuracy than the exact multiplier design.

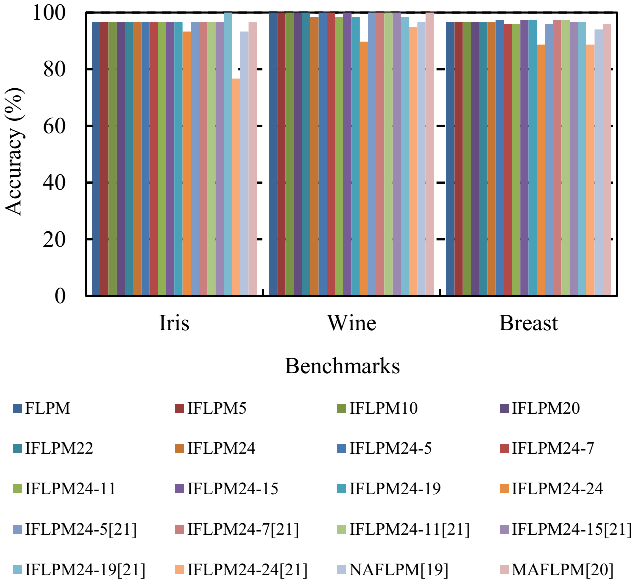


Figure 13 Accuracy comparision for the three benchmarks.

6 Conclusions

In this paper, FLP multipliers with inexact 24-bit fixed-point multipliers using inexact partial product generators and inexact partial product compressors have been proposed. The accuracy and the hardware requirements of approximate 24-bit multipliers and single precision FLP multipliers have also been assessed by simulation. The simulation results have shown that according to the accuracy requirements, different designs with varied inexact bits can be selected. The proposed designs accomplish significant reductions in power dissipation and delay compared to an exact design. All proposed designs have also been applied to three different applications of digital signal processing, image processing and data classification. The results show that the quality of applications using the proposed inexact FLP multiplier designs except IFLPM24–24 is not notably affected and the degradation is tolerable.

Acknowledgements This work is supported by grants from National Natural Science Foundation of China (No. 61401197), Natural Science Foundation of Jiangsu Province (BK20151477) and Fundamental Research Funds for the Central Universities China (NS2017024).

References

1. Venkataramani, S., Chakradhar, S.T., Roy, K., & Raghunathan, A. (2015). Computing approximately, and efficiently. *Design, Automation & Test in Europe Conference & Exhibition* (pp. 748–751).
2. Palem, K., & Ingamneni, A. (2013). Ten years of building brokenchips: The physics and engineering of inexact computing. *ACM Transactions on Embedded Computing Systems, 12*(5), 1–23.
3. Liu, W., Chen, L., Wang, C., O'Neill, M., & Lombardi, F. (2016). Design and analysis of inexact floating-point adders. *IEEE Transactions on Computers, 65*(1), 308–314.
4. Liang, J., Han, J., & Lombardi, F. (2013). New metrics for the reliability of approximate and probabilistic adders. *IEEE Transactions on Computers, 62*(9), 1760–1771.
5. Liu, W., Qian, L., Wang, C., Jiang, H., Han, J., & Lombardi, F. (2017). Design of Approximate Radix-4 booth multipliers for error-tolerant computing. *IEEE Transactions on Computers, 66*(8), 1435–1441.
6. Liu, C., Han, J., & Lombardi, F. (2014). A low-power, high-performance approximate multiplier with configurable partial error recovery. *Design, Automation and Test in Europe Conference and Exhibition (DATE)* (pp. 14).
7. Momeni, A., Han, J., Montuschi, P., & Lombardi, F. (2015). Design and analysis of approximate compressor for multiplication. *IEEE Transactions on Computers, 64*(4), 984–994.
8. Bhardwaj, K., Mane, P.S., & Henkel, J. (2014). Power-and area-efficient approximate Wallace tree multiplier for error-resilient

systems. *15th International Symposium on Quality Electronic Design (ISQED)* (pp. 263269).

9. Kulkarn, P., Gupta, P., & Ercegovac, M. (2011). Trading accuracy for power with an Underdesigned multiplier architecture. *24th International Conference on VLSI Design (VLSI Design),* (pp. 346351).
10. Sullivan, M.B., & Swartzlander, E.E. Jr. (2012). Truncated error correction for flexible approximate multiplication. *Forty-Sixth Asilomar Conference on Signals, Systems and Computers,* (pp. 355359).
11. Park, H., & Swartzlander, Jr. E.E. (2006). Truncated multiplication with symmetric correction. *Fortieth Asilomar Conference on Signals, Systems and Computers,* (pp. 931934).
12. Swartzlander, Jr. E.E. (1999). Truncated multiplication with approximate rounding. *Thirty-Third Asilomar Conference on Signals, Systems and Computers,* (pp. 14801483).
13. King, E.J., & Swartzlander, Jr. E.E. (1997). Data-dependent truncation scheme for parallel multipliers. *31st Asilomar Conference on Signals, Systems and Computers*, (pp. 11781182).
14. Schulte M.J., & Swartzlander, Jr. E.E. (1993). Truncated multiplication with correction constant. *IEEE Workshop on VLSI Signal Processin,* (pp. 388-396).
15. Tong, J. Y., Nagle, D., & Rutenbar, R. (2000). Reducing power by optimizing the necessary precision/range of floating-point arithmetic. *IEEE Transactions on Very Large Scale Integrated System, 8*(3), 273–286.
16. Fang, F., Chen, T., & Rutenbar, R. (2002). Floating-point bit-width optimization for low-power signal processing applications. *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP),* (pp.32083211).
17. Eilert, J., Ehliar, A., & Liu, D. (2004). Using low precision floating point numbers to reduce memory cost for MP3 decoding. *Proc. 6th IEEE Workshop on Multimedia Signal Processing,* (pp.119122).
18. Gupta, A., Mandavalli, S., Mooney, V., Ling, K., Basu, A., Johan, H., & Tandianus, B. (2011). Low power probabilistic floating point multiplier design. *2011 I.E. Computor Society Annual Symposium on VLSI,* (pp. 182187).
19. Zhang, H., Putic, M., & Lach, J. (2014). Low power GPGPU computation with imprecise hardware. *51st ACM/EDAC/IEEE Design Automation Conference (DAC),* (pp.16).
20. Zhang, H., Zhang, W., & Lach, J. (2014). A low-power accuracy-configurable floating PointMultiplier. *32nd IEEE International Conference on Computer Design (ICCD),* (pp. 4854).
21. Yin, P., Wang, C., Liu, W., & Lombardi, F. (2016). Design and performance evaluation of approximate floating-point multipliers. *2016 I.E. Computer Society Annual Symposium on VLSI,* (pp. 296301).
22. IEEE Computer Society (2008) *IEEE standard for floating-point arithmetic*. IEEE Standard 754-2008.
23. Yeh, W., & Jen, C. (2000). High-speed booth encoded parallel multiplier design. *IEEE Transactions on Computers, 49*(7), 692–701.
24. Industrial Light & Magic, http://www.openexr.com. 2014.
25. Mantiuk, R., Kim, K., Rempel, A., & Heidrich, W. (2011). HDR-VDP-2: A calibrated visual metric for visibility and quality predictions in all luminance conditions. *ACM Transactions on Graphics, 30*(4), 76–79.
26. Kung, J., Kim, D., & Mukhopadhyay, S. (2015). A power-aware digital feedforward neural network platform with backpropagation driven approximate synapses. *2015 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, (pp. 8590).
27. Du, Z., Lingamneni, A., Chen, Y., Palem, K., Temam, O., & Wu, C. (2015). Leveraging the error resilience of neural Networksfor designing highly energy efficient accelerators. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 34*(8), 1223–1235.
28. Asuncion, A., & Newman, D. UCI machine learning repository. http://archive.ics.uci.edu/ml/index.php. 2007.

Peipei Yin received the B.Sc. and M.Sc. degrees from Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China, in 2010 and 2013, respectively. Since Sep. 2015, she has been pursuing the Ph.D. degree in the College of Electronic and Information Engineering, NUAA, Nanjing, China. Her research interests include computer arithmetic, fault tolerance systems, and low power technologies in approximate computing.

Chenghua Wang received the B.Sc. and M.Sc. degrees from Southeast University, Nanjing, China, in 1984 and 1987, respectively. In 1987, he joined the College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, where he became a full Professor in 2001. He has published 6 books and over 100 technical papers in journals and conference proceedings. He is the recipient of more than ten teaching and research awards at the provincial and ministerial level. His current research interests include testing of integrated circuits, and circuits & systems for communications.

Weiqiang Liu received the B.Sc. degree in Information Engineering from Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China and the Ph.D. degree in Electronic Engineering from the Queen's University Belfast (QUB), Belfast, UK, in 2006 and 2012, respectively. In Dec. 2013, he joined the College of Electronic and Information Engineering, NUAA, where he is currently an Associate Professor. He was a Research Fellow in the Institute of Electronics, Communications and Information Technology (ECIT) at QUB from Aug. 2012 to Nov. 2013. He serves as an

Associate Editor of *IEEE Transactions on Computers (TC)*, the leader of The Multimedia Team at TC Editorial Board, the Guest Editors of two special issues of *IEEE Transactions on Emerging Topics in Computing.* He has been a technical program committee member for several international conferences including the IEEE International Symposium on Circuits and Systems (ISCAS), IEEE Symposium on Computer Arithmetic (ARITH), the Annual IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP) and the International Conference on Neural Information Processing (ICONIP). He has published one research book by Artech House and over 40 leading journal and conference papers. His paper was finalist in the Best Paper Contest of ISCAS 2011 and he is the co-author of a Best Paper Candidate of GLSVLSI 2015. His research interests include computer arithmetic, emerging technologies in computing systems, and cryptographic hardware. He is a Senior Member of IEEE.

Earl E. Swartzlander Jr. received the B.S. degree from Purdue University in 1967, the M.S. degree from the University of Colorado in 1969, and the Ph.D. degree from the University of Southern California in 1972, all in electrical engineering. He is a professor of electrical and computer engineering at the University of Texas at Austin. In this position, he and his students conduct research in computer engineering with emphasis on application-specific processor design, including high-speed computer arithmetic, embedded processor architecture, VLSI technology, and nanotechnology. As of June 2017, he has supervised 46 Ph.D. students. He is the author of two books, editor of 11 books and the author or coauthor of 86 refereed journal papers, 41 book chapters, and 310 conference papers. He was the editor-in-chief of the *IEEE Transactions on Computers* from 1990 to 1994 and was the founding editor-in-chief of the *Journal of VLSI Signal Processing*. In addition, he has served as an associate editor for the *IEEE Transactions on Computers*, the *IEEE Transactions on Parallel and Distributed Systems*, and the *IEEE Journal of Solid-State Circuits.* He has been a member of the Board of Governors of the IEEE Computer Society (1987–1991), the IEEE Signal Processing Society (1992–1994), and the IEEE Solid-State Circuits Council/Society (1986–1991). He has been a member of the IEEE History Committee (1996–2004), the IEEE Fellows Committee (2000–2003), the IEEE James H. Mulligan, Jr., Education Medal Committee (2007–2011), various IEEE Awards Board subcommittees (2011 to present), and the IEEE Awards Board (2016). He has chaired a number of conferences. He is a Life Fellow of the IEEE. He has been honored with the IEEE Third Millennium Medal, the Distinguished Engineering Alumnus Award from the University of Colorado, the Outstanding Electrical Engineer and Distinguished Engineering Alumnus Awards from Purdue University, and the IEEE Computer Society Golden Core Award.

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