논문 2020-15-27

Analysis of Reduced-Width Truncated Mitchell Multiplication for Inferences Using CNNs

HyunJin Kim*

Abstract : This paper analyzes the effect of reduced output width of the truncated logarithmic multiplication and application to inferences using convolutional neural networks (CNNs). For small hardware overhead, output width is reduced in the truncated Mitchell multiplier, so that fractional bits in multiplication output are minimized in error-resilient applications. This analysis shows that when reducing output width in the truncated Mitchell multiplier, even though worst-case relative error increases, average relative error can be kept small. When adopting 8 fractional bits in multiplication output in the evaluations, there is no significant performance degradation in target CNNs compared to existing exact and original Mitchell multipliers.

Keywords : Approximate computing, Convolutional neural network, Logarithmic multiplication, Mitchell algorithm, Reduced-width multiplier

Ⅰ. Introduction

In CNN models, when the high accurate floating-point format is replaced by the fixed-point format, the error resilience property makes inference accuracy not degraded. Besides, by reducing data size in fixed-point format, low-cost computational operations are achieved. For better classification performance, more layers have been added in recent CNN models, which also increases the number of multiply-accumulate (MAC) operations. Since the hardware complexity of MAC operations is relatively large, simplified approximate multiplication helps develop the lightweight implementation of neural networks. Compared to the quantization techniques, the approximate multiplication has a different trend of errors and cost-effective hardware structure [1-7]. Notably, the logarithmic multiplier approximates multiplication only using adders and shifters so that hardware costs are reduced significantly.

Many existing logarithmic multiplications were based on Mitchell multiplier [8]. In truncated Mitchell multiplier, fractions are truncated internally, so that additional cost is reduced at the expense of accuracy. When applying the original and truncated Mitchell multipliers to inferences using CNNs, there is no significant performance degradation in [6, 9]. Even though the truncated Mitchell multiplier shows tremendous hardware cost reduction, the dramatical cost increase in MAC operations requires other cost-reducing method. In the modified Booth multiplier [10-13], the fixed-width or reduced-width property was adopted. Given two n-bit inputs, the fixed-width multiplier generates n-bit output, which can be affordable in its target applications. In the reduced-width multiplier, the number of output bits $m \neq n$ can be smaller than $2n$. Even though the reduced-width output

*Corresponding Author (hyunjin2.kim@gmail.com)
Received: Aug. 11, 2020, Revised: Sep 15, 2020, Accepted: Sep. 24, 2020.
HyunJin. Kim: Dankook University (Assoc. Prof.)
※ The EDA tool was supported by the IC Design Education Center(IDEC), Korea. This research was results of a study on the "HPC Support" Project, supported by the 'Ministry of Science and ICT' and NIPA.

ⓒ IEMEK J. Embed. Sys. Appl. 2020 Oct. 15(5) 235-242
ISSN : 1975-5066
http://dx.doi.org/10.14372/IEMEK.2020.15.5.235

can reduce hardware costs much more, previous researches for the logarithmic multiplier did not consider its property. When the number of truncated bits increases, so-called reduced-width Mitchell multiplier decreases errors from the reduced output with smaller hardware complexity.

This paper analyzes the reduced-width truncated logarithmic multiplication and applies it to the inferences using CNNs. Hardware overhead is reduced significantly by truncating fractions and then adopting reduced output bit width. The unbiased design makes the average relative error small without significant changes according to output bit width. Even though the cost-effective approximations increase the cases with worst-case relative error, experimental results show that when 8 fractional bits are adopted in the reduced-width truncated Mitchell multiplier, there is no significant the performance degradation in inferences of well-known CNNs.

II. Reduced-Width Truncated Mitchell Multiplication

1. Logarithmic Truncated Multiplication

In the logarithmic numerical system, an n-bit positive integer A is expressed as:

$$A=(1+x_A)\cdot 2^{k_A}, x_A\in[0,1). \tag{1}$$

In (1), k_A and x_A are the characteristic number of the most significant bit (MSB) with the value of `1' and the fraction in the logarithmic representation of A. Multiplication of two numbers A and B is formulated as:

$$A\cdot B=(1+x_A)\cdot(1+x_B)\cdot 2^{k_A+k_B}. \tag{2}$$

By applying logarithmic conversion to $A\cdot B$,

$$\log_2(A\cdot B)= k_A+k_B+\log_2((1+x_A)\cdot(1+x_B)). \tag{3}$$

For $C=A\cdot B$, when k_C and x_C are calculated, C is obtained in anti-logarithmic conversion.

In Mitchell multiplier [8], depending on the range of x_A+x_B, k_C and x_C are differently approximated as:

$$\begin{cases} k_C=k_A+k_B+1, x_C=x_A+x_B-1 \\ \quad if\ \$x_A+x_B\geq 1 \\ k_C=k_A+k_B, x_C=x_A+x_B \\ \quad if\ x_A+x_B<1. \end{cases} \tag{4}$$

The truncated Mitchell multiplier can reduce hardware costs by adopting only a few high-order bits of fractions. In n-bit Mitchell multiplier, the number of bits for representing x_A or x_B is $n-1$. In the n'-bit truncated multiplication, high-order n' bits of x_A and x_B are adopted, which is denoted as x'_A and x'_B.

Because $1>x_A>x'_A$ and $1>x_B>x'_B$, $rerr$ is positively biased. There have been several techniques to make the multiplication unbiased [3, 9]. Especially, if $2^{-n'}$ is added, x_A+x_B is replaced by $x'_A+x'_B+2^{-n'}$ for the unbiased design, which makes (4) modified as:

$$\begin{cases} k_C=k_A+k_B+1, x_C=x'_A+x'_B+2^{-n'}-1 \\ \quad if\ x'_A+x'_B+2^{-n'}\geq 1, \\ k_C=k_A+k_B, x_C=x'_A+x'_B+2^{-n'} \\ \quad if\ x'_A+x'_B+2^{-n'}<1. \end{cases} \tag{5}$$

2. Reduced-Width Multiplication

When n-bit A is multiplied by n-bit B, typical exact multiplier generates $2n$-bit multiplication output with full accuracy. Each bit width of integer and fraction in a n-bit fixed-point format is denoted as $<IL, FL>$. Formally, typical n-bit multiplier has $2n$-bit $<2IL, 2FL>$ output. In usual fixed-point format MAC operations in CNNs, a bundle of $<2IL, 2FL>$ multiplication outputs can be accumulated. Even though $2n$-bit multiplication output provides high-accurate MAC operations, its hardware overhead can be significant

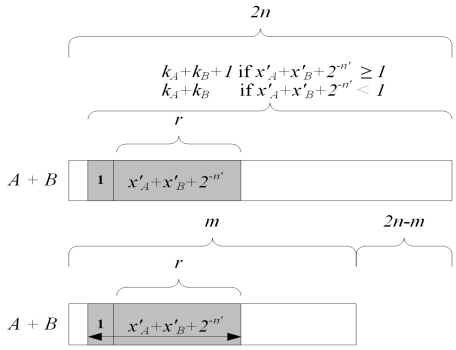


Fig. 1 Reduced output in truncated Mitchell multiplication.

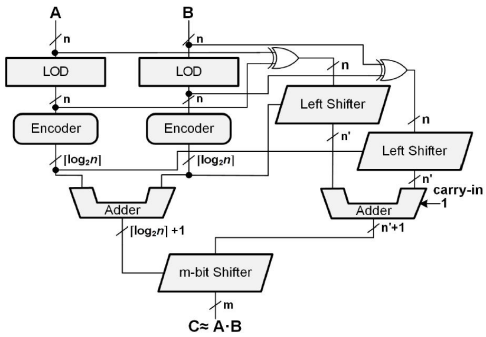


Fig. 2 Structure of reduced-width truncated Mitchell multiplier.

compared to that of reduced-width multiplication. In the error-resilient applications, even though output bit width is reduced, performance degradation does not happen or is negligible. If the $<2IL, 2FL>$ output is excessively accurate for its target application and m output bits are enough, $2n-m$ low-order output bits are eliminated.

Therefore, if $2FL \geq m$, the reduced-width multiplication has $<2IL, 2FL-m>$ output. In previous works, n-bit reduced-width booth multiplier has n-bit output, which is cost-effective in digital signal processing [10-12]. Because $2n-m$ low-order output bits are eliminated, small inputs and weights are ignored, so that the worst-case relative error ($rerr_{worst}$) can be 100%

Fig. 1 describes shift of $x'_A+x'_B+2^{-n'}$ with m-bit output with $(2n-m)$-bit reduction, where `1' means the leading one in the logarithmic representation. Because aligned fractions are summed and then shifted, the number of bits for valid information can be small. Thus, when $x'_A+x'_B+2^{-n'}$ are shifted to the left sufficiently, the lost valid information caused by reducing output bits can be ignored or small. Besides, it is expected that hardware costs decrease by reducing output bit width.

Fig. 2 describes the structure of the reduced-width truncated Mitchell multiplier. When n-bit numbers A and B are multiplied, two leading-one detectors (LODs) find k_A and k_B. Like LODs in [2, 6, 14-16], the output of each LOD for A and B consists of n bits, where the output bit in the same position with leading `1' in $A$ and $B$ is `1'; other output bits are `0.' The output of LODs for A and B are encoded into $\lceil \log_2 n \rceil$-bit k_A and k_B.

By shifting $A-2^{k_A}$ and $B-2^{k_B}$ from XOR gates, fractions are expressed with n bits, being aligned to the left and truncated into n' bits by removing $n-n'$ least significant bits (LSBs). Then, n'-bit adder calculates $(n'-1)$-bit $x'_A+x'_B+2^{-n'}$ by adding carry-in `1.' so that the unbiased design is simply implemented.

Finally, m-bit shifter generates m-bit multiplication output, where k_C is determined by the carry-out of the n'-bit adder as shown in (5). When $k_C < 2n-m$, output C is zero by reducing output bit width; therewise, the output is as follows: if $k_C-(2n-m) < n'$, only $(n'+1-(k_C-2n+m))$ high-order bits of $1+x_C$ are shifted to the left by k_C after reducing the output width; if $k_C-(2n-m) \geq n'$, $(n'+1)$ bits of $1+x_C$ are shifted to the left by k_C. For example, let's assume that $A=1110_2(14_{10})$ and $B=1111_2(15_{10})$ are multiplied in an 8-bit Mitchell multiplier with $m=12$ and $n'=2$, where $k_A=3$, $k_B=3$, $x_A=0.11_2$, and $x_B=0.111_2$. Because $n'=2$, $x'_A=0.11_2$, $x_B=0.11_2$, and $2^{-n'}=0.01_2$, $x'_A+x'_B+2^{-n'}=1.11_2 \geq 1$, so that

$k_C = 7$ and $x_C = 0.11_2$. Because $k_C - (2n-m) = 3 \geq n'$, 111_2 for representing $1 + x_C = 1.11_2$ is shifted to the left by `1,' so that $C = 11100000_2 = 224_{10}$.

In another example, when $A = 10001_2(17_{10})$ and $B = 11_2(3_{10})$, $k_A = 4$, $k_B = 1$, and $x'_A + x'_B + 2^{-n'} = 0.11_2 < 1$, so that $k_C = 5$ and $x_C = 0.11_2$. In this example, $k_C - (2n-m) = 1 < n'$, so that $n' - (k_C - 2n + m) = 2$. Therefore, 2 bits of $1 + x_C$ are shifted for $C = 110000_2(48_{10})$, where 1 low-order bit for representing $1 + x_C$ is truncated. When $A = 11_2(3_{10})$ and $B = 10_2(2_{10})$, $k_A = k_B = 1$ and $k_C < 2n - m$, making C zero.

III. Error and Cost Analysis

1. Error Analysis

For error analysis, simulations were performed with different m and n', where one million pairs of two inputs were randomly selected and applied. Outputs are averaged for $n = 8$ and $n = 32$ by varying m (16~8 for $n = 8$ and 64~36 for $n = 32$) and n' (4~8). The relative error $rerr$ is calculated as:

$$rerr = (MUL_{exact} - MUL_{appr}) / ML_{exact}. \quad (6)$$

In Table 1, when $n = 8$, averaged relative error $rerr_{avg}$ decreases sharply with m up to 2.47(%). In Table 2, when $n = 32$, the differences between $rerr_{avg}$s in each n' and m are negligible, where $rerr_{avg}$ are 3.85(%). In Table 3, $rerr_{avg}$ without adopting the unbiased design are summarized. Unlike Table 2, $rerr_{avg}$ are much varied from depending on n', where $rerr_{avg}$ is ranged in 7.92~4.11(%), so that it is concluded that the unbiased design can enhance the performance of the reduced-width truncated Mitchell multiplication. Thus, the unbiased design is expected that error-resilient applications like inferences using CNNs can tolerate these reduced m and n' without performance degradation. By decreasing m and n', the maximum relative error $rerr_{\max}$ can increase up to 100(%), and variation of $rerr$s increases, which will be evaluated with several well-known CNN models in the later part of this paper. In the analysis in [6], when the number of bits in the fixed-point format is 32, the maximum performance of accuracies was achieved in inferences using CNNs. In the later part of this paper, both cost analysis and inference accuracies are evaluated for $n = 32$.

Table 1. $rerr_{avg}$ for 8×8 multiplication

$rerr_{avg}$(%)	$n' = 4$	$n' = 5$	$n' = 6$	$n' = 7$	$n' = 8$
$m = 16$	2.47	2.73	2.98	3.25	3.50
$m = 15$	2.48	2.73	3.00	3.26	3.51
$m = 14$	2.50	2.76	3.03	3.29	3.55
$m = 13$	2.55	2.83	3.11	3.38	3.64
$m = 12$	2.67	2.98	3.26	3.55	3.82
$m = 11$	2.93	3.29	3.59	3.91	4.18
$m = 10$	3.49	3.87	4.25	4.60	4.91
$m = 9$	4.54	5.03	5.48	5.88	6.25
$m = 8$	6.48	7.18	7.71	8.24	8.62

Table 2. $rerr_{avg}$ for 32×32 multiplication

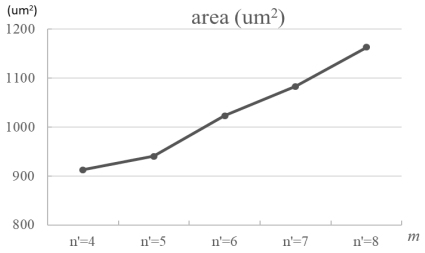
$rerr_{avg}$(%)	$n' = 4$	$n' = 5$	$n' = 6$	$n' = 7$	$n' = 8$
$m = 64$	3.85	3.85	3.85	3.85	3.85
$m = 60$	3.85	3.85	3.85	3.85	3.85
$m = 56$	3.85	3.84	3.85	3.85	3.85
$m = 52$	3.85	3.85	3.85	3.84	3.84
$m = 48$	3.85	3.85	3.85	3.85	3.84
$m = 44$	3.84	3.84	3.85	3.85	3.85
$m = 40$	3.85	3.85	3.85	3.84	3.84
$m = 36$	3.85	3.85	3.85	3.85	3.85

Table 3. $rerr_{avg}$ for 32×32 multiplication without unbiasing

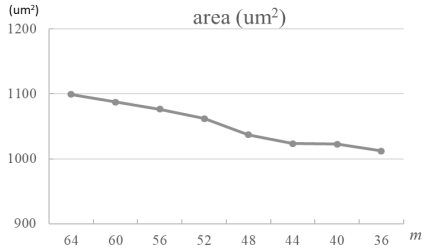
$rerr_{avg}$(%)	$n' = 4$	$n' = 5$	$n' = 6$	$n' = 7$	$n' = 8$
$m = 64$	7.92	5.91	4.89	4.37	4.11
$m = 60$	7.94	5.91	4.88	4.37	4.11
$m = 56$	7.92	5.91	4.89	4.37	4.11
$m = 52$	7.93	5.91	4.88	4.37	4.11
$m = 48$	7.92	5.90	4.88	4.37	4.11
$m = 44$	7.92	5.91	4.89	4.37	4.11
$m = 40$	7.93	5.91	4.89	4.37	4.11
$m = 36$	7.92	5.91	4.89	4.37	4.11

2. Cost Analysis

Several existing designs and the reduced-width truncated multiplier were coded



(a)



(b)

Fig. 3 Summary of multiplier area; (a) sweeping n' for $m=44$ (b) sweeping m for $n'=6$.

as combinational multipliers in Verilog hardware description language (HDL). The codes were synthesized using Synopsys Design Compiler and 28/32nm standard cell library, where the timing constraint was 250MHz frequency in Ultra mode. To analyze the effects of varied m and n' in the reduced-width truncated multiplier, data shown in Fig. 3 were evaluated when $n=32$. In Fig. 3, area increases almost linearly with respect to m and n'. Therefore, it is expected that area costs in the truncated Mitchell multiplier decrease by reducing the output width m.

In Table 4, $rerr$ and costs are compared, where $m=44$ and $n'=6$ were adopted in the target design. Exact radix-4 Booth multiplier [17] (Booth), Mitchell multiplier [8], two-stage iterative multiplier [1] (IM) were also compared with our reduced-width Mitchell multiplier (RTMM) by synthesizing them in the same environments. In Table 4, area and power are reduced by 89.9-69.2(%) and

Table 4. Comparisons of hardware costs for 32×32 multiplication

design	$rerr_{\max}$	$rerr_{avg}$	area(um2)	power(uW)
Booth	0	0	10,139	1,290
MM	11.11	3.76	3,320	198
IM	6.25	0.83	8.791	553
RTMM	-	3.85	1,023	50

Table 5. Top-1 accuracy comparison with existing multipliers on CNN models

Model	Multiplier	Accuracy(%)
NiN Top-1	FLOAT	89.6
	FIXED	89.6
	MM	88.9
	TMM	88.9
	RTMM	88.9
AlexNet Top-5	FLOAT	79.9
	FIXED	79.9
	MM	79.9
	TMM	79.9
	RTMM	79.9
GoogLeNet Top-5	FLOAT	89.1
	FIXED	89.1
	MM	88.5
	TMM	88.5
	RTMM	88.5
ResNet50 Top-5	FLOAT	91.2
	FIXED	91.2
	MM	90.0
	TMM	88.8
	RTMM	88.8

96.2-75.0(%) compared to other existing designs. The two-stage iterative multipliers [1] has large areas to meet the timing constraint with 4 ns clock speed. Especially, even though $rerr_{\max}$ and $rerr_{avg}$ are considerably small, area is 8.6 times that of the target design. Considering analysis done in Fig. 3 and Table 4, the reduced-width truncated Mitchell multiplier can have less area cost than other well-known multiplications and can further reduce area costs.

IV. Experiments on CNN Inferences

The reduced output increases output error in the reduced-width truncated Mitchell multiplier. Effects of the increased output error were evaluated on CNNs using the Caffe deep

Table 6. NiN model Top-1 accuracies

Top-1	$n'=4$	$n'=5$	$n'=6$	$n'=7$	$n'=8$
$m=46$	88.7%	88.8%	88.9%	88.9%	88.9%
$m=44$	88.7%	88.8%	88.9%	88.9%	88.9%
$m=42$	88.8%	88.8%	88.8%	88.8%	88.7%
$m=40$	88.8%	88.8%	88.8%	88.8%	88.7%
$m=38$	88.2%	87.9%	87.6%	87.7%	87.5%
$m=36$	64.7%	62.9%	60.7%	59.7%	59.4%
$m=34$	14.3%	12.2%	11.7%	11.6%	11.5%

Table 7. AlexNet model Top-5 accuracies

Top-5	$n'=4$	$n'=5$	$n'=6$	$n'=7$	$n'=8$
$m=46$	79.7%	79.7%	79.7%	79.7%	79.7%
$m=44$	79.7%	79.7%	79.7%	79.7%	79.7%
$m=42$	79.7%	79.7%	79.7%	79.7%	79.7%
$m=40$	79.6%	79.7%	79.7%	79.7%	79.7%
$m=38$	79.1%	79.1%	79.7%	79.0%	79.0%
$m=36$	68.9%	68.1%	67.3%	66.8%	66.4%
$m=34$	0.2%	9.1%	0.0%	0.0%	0.0%

learning framework [18]. Multipliers were emulated in the matrix multiplication by modifying the Caffe GPU version. For the fixed-point format multiplication, 16 integer bits and 16 fractional bits were adopted for the multiplication input in target neural networks. In all designs, multiplications with unsigned numbers were assumed, so that 2's complement conversion was used for the computations with signed numbers in neural networks. Inferences were performed using Network in Network (NiN) model [19] for CIFAR-10 [20] test dataset and AlexNet [21], GoogeLeNet [22], and ResNet50 [23] for ImageNet [24] ILSVRC2012 validation dataset. Table 5 lists inference accuracies on CNN models comparing with other existing multipliers. There is no difference between floating-point (FLOAT) and 32-bit fixed-point models (FIXED). The truncated Mitchell multiplier (TMM) has $m=64$ for $2n$-bit output and $n'=6$. Our design denoted as RTMM reduced the output bit width by $m=44$, but except for the case of ResNet50, the accuracies were nearly close to those of the original Mitchell multiplier. On ResNet50 model, the accuracy dropped in the truncated Mitchell multiplier (TMM) compared to that of the original Mitchell multiplier (MM). But the accuracies of the truncated and reduced-width truncated Mitchell multipliers were not different. Therefore, logarithmic approximation and fraction truncation mainly caused a decrease in accuracy for the ResNet50 model. The truncated Mitchell multiplier with $m=44$ did not show significant accuracy drop.

Tables 6 and Tables 7 list the inference results by varying m and n'. The Top-1 accuracy means the rate of object with the highest score for the correct answer. The Top-5 accuracy depends on whether its correct answer can be classified into one of top five scores or not. In the inferences, when $m \geq 40$, the accuracies were nearly close to that of the original Mitchell multiplier [8]. By decreasing n', the accuracies were slightly degraded. But when m was not large enough, there was serious drop in accuracies. Because 16 integer bits and 16 fractional bits for $n=32$ were assumed, $2n$-bit multiplication output had 32 integer bits and 32 fractional bits. With more than 8 fractional bits in multiplication output, there was no significant performance degradation. Therefore, it is concluded that the inference accuracy depends on the fractional bits of multiplication output and m.

V. Conclusion

We analyze the reduced-width truncated Mitchell multiplier on CNN inferences. The analysis shows that even though output bit width decreases, the inference accuracies obtained from several CNN models are not affected by the reduced output width in the truncated Mitchell multiplier. Compared with low-bit quantized CNN, the hardware reductions are not great. But on target CNNs, there is no significant performance degradation using 8 fractional bits in multiplication output. With the reduced output bit width, hardware costs and number of output bits can be reduced. Therefore, it is expected that the

reduced-width truncated Mitchell multiplier can significantly minimize the cost of inferences.

References

[1] Z. Babic, A. Avramovic, P. Bulic, "An Iterative Mitchell's Algorithm Based Multiplier," Proc. of ISSPIT 2008, pp. 303-308, 2008.

[2] Z. Babi´c, A. Avramovi´c, P. Buli´c, "An Iterative Logarithmic Multiplier," Microprocessors and Microsystems, Vol. 35, No. 1, pp. 23-33, 2011.

[3] S. Hashemi, R. I. Bahar, S. Reda, "Drum:A Dynamic Range Unbiased Multiplier for Approximate Applications," Proc. of ICCAD, pp. 418-425, 2015.

[4] R. Zendegani, M. Kamal, M. Bahadori, A. Afzali-Kusha, "Roba Multiplier: A Tounding-based Approximate Multiplier for High-speed yet Energy-efficient Digital Signal Processing," IEEE Trans. VLSI Systems, Vol. 25, No. 2, pp. 393-401, 2017.

[5] W. Liu, L. Qian, C. Wang, H. Jiang, J. Han, F. Lombardi, "Design of Approximate Radix-4 Booth Multipliers for Error-tolerant Computing," IEEE Transactions on Computers, Vol. 66, No. 8, pp. 1435-1441, 2017.

[6] M.S. Kim, A.A. Del Barrio, R. Hermida, N. Bagherzadeh, "Low-power Implementation of Mitchell's Approximate Logarithmic Multiplication for Convolutional Neural Networks," Proc. of ASP-DAC, pp. 617-622, 2018.

[7] I. Alouani, H. Ahangari, O. Ozturk, S. Niar, "A Novel Heterogeneous Approximate Multiplier for Low Power and High Performance," IEEE Embedded Systems Letters, Vol. 10, No. 2, pp. 45-48, 2018.

[8] J.N. Mitchell, "Computer Multiplication and Division Using Binary Logarithms," IRE Trans. Electronic Computers, No. 4, pp. 512-517, 1962.

[9] M.S. Kim, A.A. Del Barrio, L.T. Oliveira, R. Hermida, N. Bagherzadeh, "Efficient Mitchell's Approximate Log Multipliers for Convolutional Neural Networks," IEEE Trans. Computers, Vol. 68, No. 5, pp. 660-675, 2018.

[10] S.J. Jon, H.H. Wang, "Fixed-width Multiplier for Dsp Application," Proc. of IEEE International Conference on Computer Design, pp. 318-322, 2000.

[11] S.J. Jou, M.H. Tsai, Y.L. Tsao, "Low-error Reduced-width Booth Multipliers for Dsp Applications," IEEE Trans. Circuits and Systems I: Fundamental Theory and Applications, Vol. 50, No. 11, pp. 1470-1474, 2003.

[12] K.J. Cho, K.C. Lee, J.G. Chung, "Design of Low Error Fixed-width Modified Booth Multiplier," IEEE Trans. VLSI Systems, Vol. 12, No. 5, pp. 522-531, 2004.

[13] S.S. Bhusare, V.K. Bhaaskaran, "Fixed-width Multiplier with Simple Compensation Bias," Procedia Materials Science, Vol. 10, pp. 395-402, 2015.

[14] K.H. Abed, R.E. Siferd, "Cmos Vlsi Implementation of a Low-power Logarithmic Converter," IEEE Trans. Computers, Vol. 52, No. 11, pp. 1421-1433, 2003.

[15] K. Kunaraj, R. Seshasayanan, "Leading one Detectors and Leading one Position Detectors an Evolutionary Design Methodology," Canadian journal of electrical and computer engineering, Vol. 36, No. 3, pp. 103-110, 2013.

[16] S.E. Ahmed, S. Kadam, M. Srinivas, "An Iterative Logarithmic Multiplier with Improved Precision," Proc. of IEEE Symposium Computer Arithmetic, pp. 104-111, 2016.

[17] I. Koren, Computer arithmetic algorithms. AK Peters/CRC Press, 2001.

[18] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, "Caffe: Convolutional Architecture for Fast Feature Embedding," Proc. ACM Int'l Conf. on Multimedia, pp. 675-678, 2014.

[19] M. Lin, Q. Chen, S. Yan, "Network in Network," arXiv:1312.4400, 2013.

[20] A. Krizhevsky G. Hinton, "Learning Multiple

Layers of Features from Tiny Images," technical report, Citeseer, 2009.

[21] A. Krizhevsky, I. Sutskever, G.E. Hinton, "Imagenet Classification with Deep Convolutional Neural Networks," Proc. of Advances in neural information processing systems, pp. 1097-1105, 2012.

[22] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, "Going Deeper with Convolutions," Proc. of the IEEE conference on computer vision and pattern recognition, pp. 1-9, 2015.

[23] K. He, X. Zhang, S. Ren, J. Sun, "Deep Residual Learning for Image Recognition," Proc. of IEEE conference on computer vision and pattern recognition, pp. 770-778, 2016.

[24] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," International Journal of Computer Vision, Vol. 115, No. 3, pp. 211-252, 2015.

HyunJin Kim (김 현 진)

HyunJin Kim is the associate professor in the School of Electronics and Electrical Engineering at the Dankook University, Republic of Korea. He received Ph.D in Electronics and Electrical Engineering (2010) from Yonsei University, Republic of Korea. He was a senior engineer in the field of flash memory controller project at the Memory Division of Samsung Electronics (2010.04~2011.08). His current research interests reside in the realm of the approximate & stochastic computing for neural network implementation and energy-aware embedded system.

Email: hyunjin2.kim@gmail.com