Power Efficient Design and Implementation of a Novel Constant Correction Truncated Multiplier

Yu Ren, Dong Wang, Leibo Liu, Shouyi Yin and Shaojun Wei
Tsinghua University, Beijing
E-mail: reneereny@gmail.com Tel: +86-158-1033-3606
E-mail: doonny1212@gmail.com Tel: +86-10-62781553

***Abstract*—Multipliers are the fundamental arithmetic units in most high speed signal and multimedia processing SoCs. In most cases, the final products of the multiplications can be rounded with lower precisions to avoid the continuous growth of word length while the computation carries on. A class of modified multipliers which is called truncated multipliers could significantly reduce the hardware cost and energy consumption of the final implementation. This paper proposes an optimized design of truncated multipliers based on comprehensive analysis of statistical properties of the input operands. The proposed design architectures are modeled in Matlab codes using Fixed-point toolbox and in RTL using Verilog. The synthesis results on FPGA show that the proposed multiplier architectures could achieve almost 37% reduction on logic resource and consume about 33% less dynamic power over the full-size multipliers. The average error can be reduced by 30% to 50% compared to the traditional constant correction truncated multipliers. This paper also explores the efficiency of the proposed design in actual systems. Simulation results show that the proposed design has obvious advantages on power efficiency in applications against traditional truncated multipliers.**

I. INTRODUCTION

Multiplication has been pervasively used in Digital Signal Processing (DSP) applications like filtering, convolution and compression [1]. For most DSP systems, the exact result with full precision word length is not always required and modified design of standard multipliers could reduce power dissipation and increase throughput [2]. Since a full-precision $n \times n$ multiplier computes the $2n$ output, rounding the result to n bits could avoid the continuous growth of word length while the computation carries on.

Several approaches have been proposed in the literature [2-13] to reduce the area and improve the latency on the critical path for the implementation of truncated multipliers. The basic idea of these techniques is to remove the least significant bits of the partial products and introduces adjusting circuit to compensate for the errors introduced. It was shown that by saving corresponding hardware in the multipliers' reduction array and final carry propagation chain, the dynamic power dissipation can also be proportionally reduced. However, the traditional schemes usually introduce a non-zero DC (Direct Current) component on the final product for all data inputs [7], which would cause obvious mistakes when one of the multiplicand is zero.

To address this issue, this paper proposes a new algorithm and corresponding hardware design of the truncated multiplier based on traditional CCT (Constant Correction Truncated) multiplier algorithm [4], which could further improve the area and power efficiency while minimizing the average and maximum error by 30% to 50% compared to CCT multiplier. The proposed scheme can be applied to both signed and unsigned multipliers [14],[15]. We also verify the new scheme in designing power efficiency multipliers in several real applications, which could reduce the logic resource by 36.9% than a 16-bit standard multiplier and 9.3% than a 16-bit traditional CCT multiplier. As to power dissipation, the proposed scheme consumes 33.0% less power than a standard multiplier and 30.2% less a traditional CCT multiplier.

II. OVERVIEW OF THE CONSTANT CORRECTION TRUNCATED MULTIPLIERS

The CCT multiplier is first introduced in [4]. Assuming multiplication of two n-bit numbers A and B results in a $2n$-bit product P, which are of the form

$$A=\sum_{i=0}^{n-1} a_i \cdot 2^{-n+i} \quad B=\sum_{i=0}^{n-1} b_i \cdot 2^{-n+i} \quad P=\sum_{i=0}^{2n-1} p_i \cdot 2^{-2n+i}$$

Where a_i and b_i denote each bit of A and B, p_i denotes each bit of P. The relationship between the partial product bit $\pi_{ij}=a_j \cdot b_i$ and the final products p_i in the multiplication matrix is shown as follows:

			×	a_{n-1}	⋯	a_1	a_0
				b_{n-1}	⋯	b_1	b_0
				$\pi_{0,n-1}$	⋯	$\pi_{0,1}$	$\pi_{0,0}$
			$\pi_{1,n-1}$	$\pi_{1,n-2}$	⋯	$\pi_{1,0}$	
			⋯	⋯	⋯		
		$\pi_{n-2,n-1}$	⋯	$\pi_{n-2,1}$			
	$\pi_{n-1,n-1}$	⋯	$\pi_{n-1,1}$	$\pi_{n-1,0}$			
p_{2n-1}	p_{2n-2}	⋯	p_n	p_{n-1}	⋯	p_1	p_0

Fig. 1 $n \times n$ Multiplication Matrix

The n^2 partial products π_{ij} are summed to compute the $2n$-bit product P, which is usually then rounded to n bits in most applications to avoid continuous growth of the word-length of the results. Truncation could also be conducted to reduce the area and power consumption by omitting the least significant columns of the multiplication matrix and summing only the $(n+k)$ most significant columns, i.e. column $(n-k)$ to column

This paper is supported by China Postdoctoral Science Foundation NO. **20110490427**.

(*2n-1*) in Fig. 1. However, both the rounding and truncation would introduce errors to the final products.

To calculate the expected value of reduction error, it is assumed that each bit of the input operand a_i or b_i has an equal probability of being '1' or '0', which means each partial product bit π_{ij} has an expected value of 1/4, and the positional weight is $2^{-2n+i+j}$. The reduction error, which is the sum of all partial products in columns 0 to (*n-k-1*), is calculated as

$$E_{reduct} = -\frac{1}{4} \cdot \sum_{q=0}^{n-k-1} (q+1) \cdot 2^{-2n+q} \quad (1)$$

Rounding error occurs because the (*n+k*)-bit product is rounded to *n* bits. To estimate the expected value of rounding error, it is assumed the probability of each product bit, p_{n-1} to p_{n-k}, being one is 0.5, and the expected value of rounding error is

$$E_{round} = -\frac{1}{2} \cdot \sum_{q=n-k}^{n-1} 2^{-2n+q} = -2^{-n-1} \cdot (1 - 2^{-k}) \quad (2)$$

The expected value of the total error is the sum of reduction error E_{reduct} and rounding error E_{round}, and correction constant is selected to be the inverse value of the total error with (*n+k*) bits precision, which is given as below:

$$C = -\frac{round(2^{n+k} \cdot E_{total})}{2^{n+k}} \quad (3)$$

where *round(x)* indicates *x* is rounded to the nearest integer. The expected value of the average error is given as

$$E_{avg} = C + E_{total} \quad (4)$$

Two assumptions are made in previous conduction of (3) and (4). However, in previous studies, we have found these assumptions were not reasonable. By simulating *A* and *B* from 0 to 2^N-1 using Fixed-Point Toolbox in Matlab R2009a and comparing the average error with theoretical results, we get

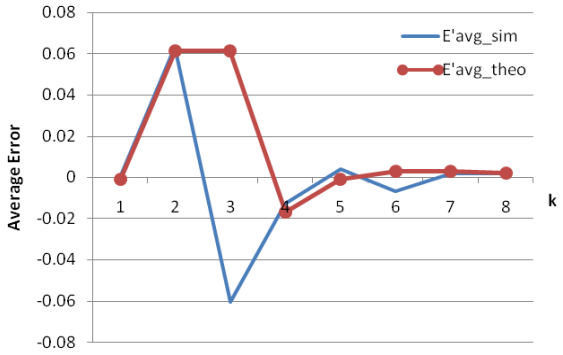

Fig. 2 Average Error Comparison of 8-bit Multiplier between Simulation and Theoretical result

From Fig. 2, we could see that there is big deviation between theoretical and true value for some values of *k*. The reason for the inconsistence is that the assumption made to calculate the reduction and rounding error is unwarranted.

In the traditional scheme[4], to calculate the expected value of rounding error, it is assumed that the probability of each output bit being one is 0.5, while in fact, p_i is the sum of partial products in column *i*, and the partial products are relevant to multiplicands. Simulation result of p_i being one in an 8-bit multiplier is shown in Fig. 3, and it is clearly that the output bit is not uniformly distributed.

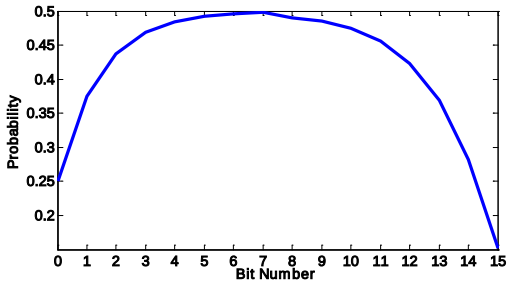

Fig. 3 Probability of an 8-bit Multiplier Product Bit Being One

When it comes to the expected value of reduction error, it is also assumed that the probability of each input bit being one is 0.5. While in fact, input data is not distributed uniformly, either.

Taking the input statistical characteristic into consideration will contribute to more accurate estimation of total error and improvement in precision by selecting correction constant wisely. Therefore, by following this idea, this paper proposes a novel method to estimate the error and select correction constant to solve the problems above.

III. INPUT-STATISTICALLY-BASED (ISB) CCT MULTIPLIER

A. *Theoretical Analysis*

The multiplication matrix of Fig. 1 can be rearranged as the multiplication array [2] shown in Fig. 4, where we take n=8 for instance.

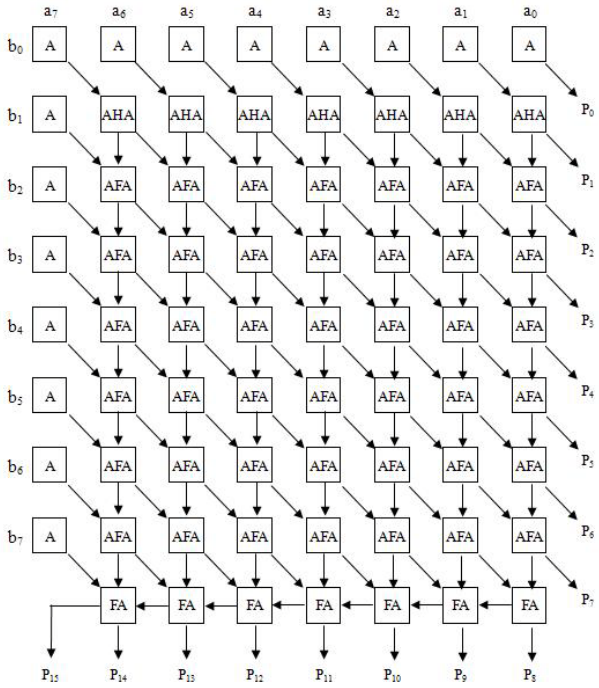

Fig. 4 An 8×8 Multiplier Array

Where “A” represents AND gate, “FA” represents full adder, “HA” represents half adder in the picture above.

Let $p(a_i)$ and $p(b_j)$ be the probabilities of input bit being one, $p(i,j)$ be the probability of partial product at column i and row j being one. Thus $p(i,j)= p(a_i) \cdot p(b_j)$. Let $p_s(i,j)$ denote the probability that the sum bit of the adder at the ith column and jth row is one and, $p_c(i,j)$ the carry bit being one. $q(i,j)=1-p(i,j)$

For $0 \le i \le N-2$, $j=1$, we have

$$p_c(i,j) = p(i,j) \cdot p(i+1, j-1) \tag{5}$$

$$p_s(i,j) = p(i,j) \cdot q(i+1, j-1) + q(i,j) \cdot p(i+1, j-1) \tag{6}$$

For $0 \le i \le N-2$, $2 \le j \le N-1$

$$\begin{aligned} p_c(i,j) = & p(i,j)p_s(i+1,j-1)p_c(i,j-1) \\ & + p(i,j)p_s(i+1,j-1)q_c(i,j-1) \\ & + p(i,j)q_s(i+1,j-1)p_c(i,j-1) \\ & + q(i,j)p_s(i+1,j-1)p_c(i,j-1) \end{aligned} \tag{7}$$

$$\begin{aligned} p_s(i,j) = & p(i,j)p_s(i+1,j-1)p_c(i,j-1) \\ & + p(i,j)q_s(i+1,j-1)q_c(i,j-1) \\ & + q(i,j)q_s(i+1,j-1)p_c(i,j-1) \\ & + q(i,j)p_s(i+1,j-1)q_c(i,j-1) \end{aligned} \tag{8}$$

As has been analyzed in Section 2, the reduction error is the sum of the partial products in 0 to (n-k-1) diagonals, which gives a formula of the expected value of

$$E_{reduct} = -\sum_{q=0}^{n-k-1}\sum_{j=0}^{q} p(q-j, j) \cdot 2^{-2n+q} \tag{9}$$

The expected value of rounding error is the sum of product p_{n-1} to p_{n-k} .

$$E_{round} = -\sum_{q=n-k}^{n-1} p_s(0,q) \cdot 2^{-2n+q} \tag{10}$$

And new form of correction constant is calculated as

$$C = -\frac{round(2^{n+k} \cdot E_{total})}{2^{n+k}} \tag{11}$$

B. *Mathematical Models of the ISB-CCT-Multiplier*

Fig.5 illustrates the mathematical model of the proposed truncated multiplier. M and N are the input length. L is the output length. K is number of additional partial product columns. First of all, analyze the M-bit and N-bit input database and obtain the probability of each input bit being one, P_A and P_B. Calculate output bit distribution P_C according to formula (5)-(8), and total error and correction constant using (9)-(11). Summing up the (L+k) most significant columns of multiplication matrix and (L+k) bit correction constant, and the final result is rounded to L bits.

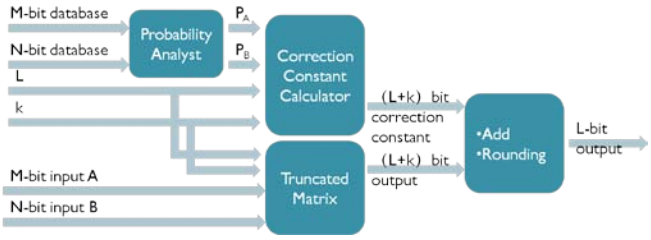

Fig. 5 Model of input-statistically-based truncated multiplier

C. *Simulation Results*

Simulate two multiplicands from 0 to 2^N-1 (N=8) and compare the difference between the true value and theoretical value of total error using traditional and proposed method. We can get

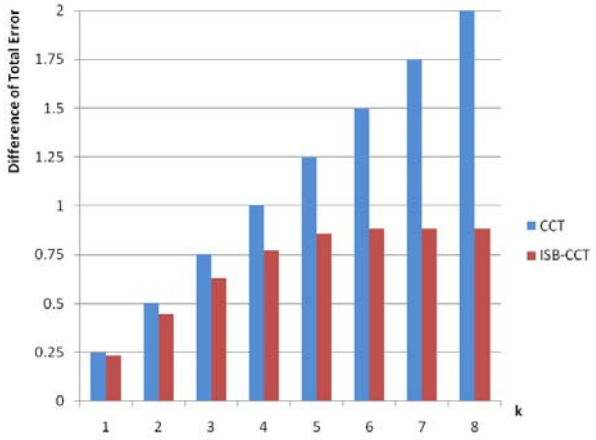

Fig. 6 Difference of Total Error between True Value and Theoretical Value of an 8-bit Multiplier

Fig.6 shows that compared to traditional CCT, the proposed design estimates the error more accurately, and the difference could be reduced by 30%-50%.

IV. Implementation of the ISB-CCT Multiplier

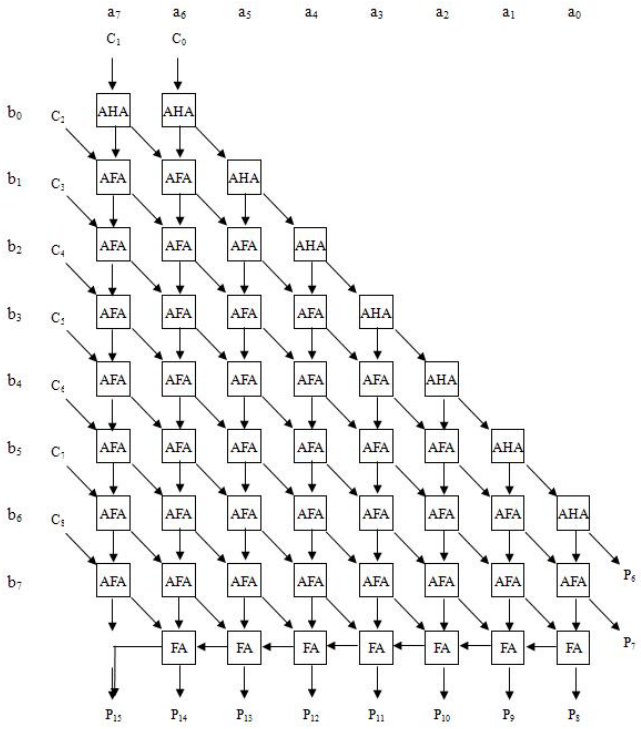

Fig. 7 A constant correction truncated 8×8 multiplier

Fig. 7 shows the architecture of a constant correction truncated 8×8 multiplier with k=2. We have implemented it in Verilog HDL and synthesized using Altera Quartus II 8.0 Web Edition with EP2S15F484C3 device in Stratrix II family.

The synthesis results show that 8-bit standard multiplier (which is not truncated, as shown in Fig. 4) needs 110 ALUTs and 48.35mW, and 16-bit standard multiplier needs 510 ALUTs and 270.07mW. The detailed results for traditional CCT and proposed scheme are shown in Table I, where the hardware cost is measured in the number of LUTs (Look-Up-Tables) used including both combinational and register elements. The power consumption is reported in the value of dynamic power dissipation since the static power is the same with identical FPGA devices.

TABLE I
LOGIC RESOURCE AND POWER COMPARISON BETWEEN TRADITIONAL CCT AND ISB-CCT MULTIPLIERS

n=8	Logic Resource (LUTs)		Dynamic Power (mW)	
k	CCT	ISB	CCT	ISB
1	78	78	39.2	32.38
2	88	86	39.56	35.54
3	102	100	45.24	46.88
4	107	100	41.66	48.41
5	110	105	50.5	51.17
6	114	105	43.18	45.6
7	118	109	42.54	48.72
8	119	118	49.49	44.56
n=16	Logic Resource (LUTs)		Dynamic Power (mW)	
k	CCT	ISB	CCT	ISB
2	343	342	230.38	228.16
4	413	386	243.77	250.83
6	413	386	305.84	238.89
8	491	456	319.97	252.39
10	522	486	316.1	253.13
12	527	504	259.98	270.1
14	541	518	259.01	287.92
16	544	519	298.23	257.11

Compared to standard multiplier, 8-bit ISB-CCT consumes 29.1% less logic resource, and 16-bit 36.9%. Besides, the proposed scheme consumes 7.9% less logic resource for 8-bit and 9.3% less for 16-bit than traditional CCT multiplier.

As to power dissipation, compared to standard multiplier, ISB-CCT consumes 33.0% lower power for 8-bit and 31.95% for 16-bit. As the proposed scheme occupies less logic resource, the power dissipation is lower than traditional CCT with 8-bit 17.4% less and 16-bit 30.2%.

V. APPLICATION OF THE TRUNCATED MULTIPLIER IN DSP SYSTEMS

In this section, we apply the proposed truncated multipliers to implement the JPEG baseline sequential codec [16] and weighted filter to demonstrate the validity of the proposed scheme.

A. DCT/IDCT Simulation

Test images are divided into 8×8 blocks for processing. 2-D DCT could be implemented by performing 1-D DCT on each row, then performing it again on each column, which could be depicted as $C=DPD^T$, where C is the matrix after transformation, P is the image for processing, and D is the DCT operator matrix. For IDCT, it is given as $P=D^TCD$.

For comparison, performing DCT/IDCT using the methods given as below: 1) Matlab dct2/idct2 function; 2) double precision operation; 3) fixed-point operation using standard multiplier; 4) fixed-point operation using Non-Correction Truncated (NCT) multiplier; 5) fixed-point operation using traditional CCT multiplier; 6) fixed-point operation using ISB-CCT multiplier. The PSNR (Peak Signal-to-Noise Ratio) of the transformed picture is shown in Table II.

TABLE II
PSNR USING DIFFERENT METHODS FOR DCT/IDCT(DB)

Matlab func	31.4713	k	1	2	3
Double Precision	31.4713	NCT	25.2662	29.8745	31.0590
Standard Mul	31.1184	CCT	26.2466	30.3544	29.7303
		ISB	26.2466	30.3544	29.7303

The data shows that in DCT/IDCT, the proposed architecture has the same result as traditional CCT. The reason is that the novel scheme is based on the probability distribution of the input operands, and it takes four multiplications before obtaining the final results in DCT/IDCT. In fact, the intermediate results has an uninformed distribution for each bit, which coincides with the assumption in traditional CCT.

B. Weighted Filter Implementation

Now implement the truncated multiplier into another application in DSP, weighted filter, which could remove noise in real images.

The filtering template is given as

$$\begin{bmatrix} 1/32 & 1/32 & 1/32 \\ 1/32 & 3/4 & 1/32 \\ 1/32 & 1/32 & 1/32 \end{bmatrix}$$

Taking double-precision operation, standard multiplier, non-correction truncated multiplier, traditional CCT multiplier and ISB-CCT multiplier to filter the noise in images, and the PSNR is shown in Table III.

TABLE III
PSNR USING DIFFERENT METHODS FOR FILTERING(DB)

Original Noise	20.1562	k	1	2	3
Double Precision	22.3445	NCT	22.1420	22.1420	22.1420
Standard Mul	22.3445	CCT	22.1054	22.2654	22.3179
		ISB	22.3374	22.3374	22.3374

The filtering results show that the PSNR of the ISB-CCT is better than that achieved with standard multipliers and traditional CCT multipliers, which is due to the non-uniformed distribution of the input operand bit in filtering

matrix. The novel algorithm takes the distribution of input bit into consideration, thus it estimates the error more accurately.

The use of truncated multiplier in the implementation of JPEG compression using DCT/IDCT and weighted filter has been examined and it is found that the proposed design has obvious advantages against traditional truncated multipliers in applications based on non-uniformed input distribution, while the advantages are not as obvious in uniformly distributed input cases. Under the same precision, the ISB-CCT multiplier has a higher PSNR than traditional CCT multiplier. On the other hand, within a certain precision, the ISB-CCT multiplier consumes less power than the traditional one.

VI. CONCLUSIONS

This paper proposes an optimized design of truncated multipliers based on comprehensive analysis of statistical properties of the input operands. By giving detailed error analysis on both reduction error and rounding error, we have revealed that the basic assumption of the equalized distribution of the probability of the bit in the partial products and the input operands are not reasonable. Therefore, a new statistic scheme is introduced and an optimized correction algorithm is generalized. We also propose the hardware implementation of the architecture, verify and synthesize the design on FPGA. At last, two applications in DSP are given. In summary, our proposed multiplier achieves lower error, less area, lower power dissipation than traditional CCT multiplier, and it could be used for the implementation of digital filter, DCT and IDCT hardware accelerators and so on.

REFERENCES

[1] U. M. Baese, *Digital Signal Processing With Field Programmable Gate Arrays*. Berlin, Germany: Springer,2004.

[2] S. S. Kidambi, F. El-Guibaly, A. Antonion, "Area-efficient multipliers for digital signal processing applications", in *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 43, no. 2, pp. 90-05, Feb 1996.

[3] J. E. Stine and O. M. Duverne, "Variations on truncated multiplication", in *Proc. Euro. Symp. Digital Syst. Design*, pp. 112-119 Sept. 1-6 2003.

[4] E. E. Swartzlander, Jr. "Truncated multiplications with approximate rounding", *Conference Record of the Thirty-Third Asilomar Conference on Signals, Systems, and Computers*, vol. 2, pp. 1480-1483, 1999

[5] C-H. Chand and R. K. Satzoda, "A low error and high performance multiplier-based truncated multiplier", *IEEE Transactions on VLSI Systems*, vol. 18, no. 12, pp. 1767-1771, Dec. 2010

[6] M. J. Schulte and E. E. Swartzlander, Jr., "Truncated multiplication with correction constant," in *VLSI Signal Processing, VI*, Oct. 1993, pp. 388-396.

[7] E. J. King and E. E. Swartzlander, Jr., "Data-dependent truncation scheme for parallel multipliers," in *Proc. 31st Asilomar Conf. Signals, Syst. Comput.*, Nov. 1997, vol. 2, pp. 1178-1182.

[8] K. Bircherstaff, M. J. Schulte, and E. E. Swartzlander, Jr., "Parallel reduced area multipliers," in *Journal of VLSI signal processing*, vol. 9, pp. 181-191, 1995.

[9] C. Lemonds and S. S. Shetti, "A low power 16 by 16 multiplier using transition reduction circuitry," in *Proceedings of the International Workshop on Low Power Design*, pp. 139-142,1994.

[10] E. de Angel and E. E. Swartzlander, Jr., "Low power parallel multipliers," in *VLSI Signal Processing*, IX., pp. 199-208,1997.

[11] S.-R. Kuang and J.-P. Wang, "Design of power-efficient configurable Booth multiplier," in *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 3, pp. 568-580, March 2010

[12] Y. Lim, "Single precision multiplier with reduced circuit complexity for signal processing applications," in *IEEE Transactions on Computers*, vol. 41, no. 10, pp. 1333-1336, 1992.

[13] A. D. Booth, "A signed binary multiplication technique," *Q. J. Mech. Appl. Math.*, vol. 4, pp. 236-240,1951.

[14] C. R. Baugh and B. A. Wooley, "A two's complement parallel array multiplication algorithm," in *IEEE Transactions on Computers*, vol. C-22, pp. 1045-1047, 1973

[15] E. G. Walters III, M. G. Arnold, and M. J. Schulte, "Using truncated multipliers in DCT and IDCT hardware accelerators", *Proc. SPIE 5205*, 573(2003).