IJMTST

Volume: 2 | Issue: 10 | October 2016 | ISSN: 2455-3778

Performance Improved Multipliers Based on Non-Redundant Radix-4 Signed-Digit Encoding

Yayathi Raju Thippani[1] | K. Swapna[2]

[1]PG Scholar, Vaagdevi College of Engineering, Telangana.
[2]Associate Professor, Vaagdevi College of Engineering, Telangana.

ABSTRACT

In this paper, we introduce an architecture of pre-encoded multipliers for Digital Signal Processing applications based on off-line encoding of coefficients. To this extend, the Non-Redundant radix-4 Signed-Digit (NR4SD) encoding technique, which uses the digit values {1; 0; +1; +2} or {-2; -1; 0; +1}, is proposed leading to a multiplier design with less complex partial products implementation. Extensive experimental analysis verifies that the proposed pre-encoded NR4SD multipliers, including the coefficients memory, are more area and power efficient than the conventional Modified Booth scheme.

KEYWORDS: *Multiplying circuits, Modified Booth encoding, Pre-Encoded multipliers, VLSI implementation*

I. Introduction

Multimedia and Digital signal processing (DSP) applications (e.g., Fast Fourier Transform (FFT), audio/video CoDecs) carry out a large number of multiplications with coefficients that do not change during the execution of the application. Since the multiplier is a basic component for implementing computationally intensive applications, its architecture seriously affects their performance.

Constant coefficients can be encoded to contain the least non-zero digits using the Canonic Signed Digit (CSD) representation [1]. CSD multipliers com-prise the fewest non-zero partial products, which in turn decreases their switching activity. However, the CSD encoding involves serious limitations. Folding technique [2], which reduces silicon area by time-multiplexing many operations into single functional units, e.g., adders, multipliers, is not feasible as the CSD-based multipliers are hard-wired to specific coef-ficients. In [3], a CSD-based programmable multiplier design was proposed for groups of pre-determined coefficients that share certain features. The size of ROM used to store the groups of coefficients is sig-nificantly reduced as well as the area and power consumption of the circuit.pre-determined coefficients attaining at the same time high efficiency.

Modified Booth (MB) encoding [4]–[7] tackles the aforementioned limitations and reduces to half the number of partial products resulting to reduced area, critical delay and power consumption. However, a dedicated encoding circuit is required and the partial products generation is more complex. In [8], Kim et al. proposed a technique similar to [3], for designing efficient MB multipliers for groups of pre-determined coefficients with the same limitations described in the previous paragraph.

In audio [11], [12] and video [13], [14] CoDecs, fixed coefficients stored in memory, are used as multiplication inputs. Since the values of constant coefficients are known in advance, we encode the coefficients off-line based on the MB encoding and store the MB encoded coefficients (i.e., 3 bits per digit) into a ROM. Using this technique [15]–[17], the encoding circuit of the MB multiplier is omitted. We refer to this design as pre-encoded MB multiplier. a condensed form (i.e,2 bits per digit).which the encoded coefficients need 3 bits per digit, the proposed NR4SD scheme reduces the memory size. Also, compared to the MB form, which uses five digit values {2; 1; 0; +1; +2g} the proposed NR4SD encoding uses four digit.

TABLE 1
Modified Booth Encoding

b_{2j+1}	b_{2j}	$b_{2j\ 1}$	b^{MB}_j	s_j	one_j	two_j
0	0	0	**0**	0	0	0
0	0	1	**+1**	0	1	0
0	1	0	**+1**	0	1	0
0	1	1	**+2**	0	0	1
1	0	0	**-2**	1	0	1
1	0	1	**-1**	1	1	0
1	1	0	**-1**	1	1	0
1	1	1	**0**	1	0	0

values. Thus, the NR4SD-based pre-encoded multipliers in-clude a less complex partial products generation cir-cuit. We explore the efficiency of the aforementioned pre-encoded multipliers taking into account the size of the coefficients' ROM.

II. Modified Booth Algorithm

Modified Booth (MB) is a redundant radix-4 encoding technique [6], [7]. Considering the multiplication of the 2's complement numbers A, B, each one consisting of n=2k bits, B can be represented in MB form as:

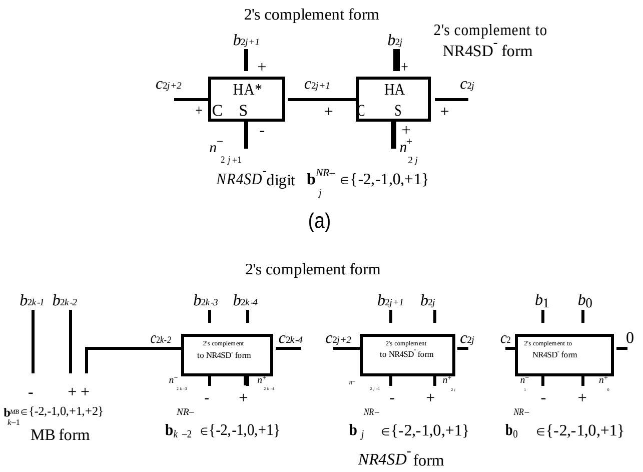

Fig. 2. Block Diagram of the NR4SD⁺ Encoding Scheme at the (a) Digit and (b) Word Level.

TABLE 2
NR4SD Encoding

2's complement			NR4SD form			Digit	NR4SD Encoding		
b_{2j+1}	b_{2j}	c_{2j}	c_{2j+2}	n_{2j+1}	n^+_{2j}	b^{NR}_j	one^+_j	one_j	two_j
0	0	0	0	0	0	**0**	0	0	0
0	0	1	0	0	1	**+1**	1	0	0
0	1	0	0	0	1	**+1**	1	0	0
0	1	1	1	1	0	**-2**	0	0	1
1	0	0	1	1	0	**-2**	0	0	1
1	0	1	1	1	1	**-1**	0	1	0
1	1	0	1	1	1	**-1**	0	1	0
1	1	1	1	0	0	**0**	0	0	0

TABLE 3
NR4SD⁺ Encoding

2's complement			NR4SD⁺ form			Digit	NR4SD⁺ Encoding		
b_{2j+1}	b_{2j}	c_{2j}	c_{2j+2}	n^+_{2j+1}	n_{2j}	b^{NR+}_j	one^+_j	one_j	two^+_j
0	0	0	0	0	0	**0**	0	0	0
0	0	1	0	1	1	**+1**	1	0	0
0	1	0	0	1	1	**+1**	1	0	0
0	1	1	0	1	0	**+2**	0	0	1
1	0	0	0	1	0	**+2**	0	0	1
1	0	1	1	0	1	**-1**	0	1	0
1	1	0	1	0	1	**-1**	0	1	0
1	1	1	1	0	0	**0**	0	0	0

1a). The outputs c_{2j+2} and n_{2j+1} of the HA* relate to its inputs as follows:

$$2c2j+2 \quad n_{2j+1} = b2j+1 + c2j+1:$$

The following Boolean equations summarize the HA* operation:

$$c2j+2 = b2j+1 \_ c2j+1; \; n_{2j+1} = b2j+1 \quad c2j+1:$$

Step 4: Calculate the value of the $\mathbf{b}^{NR}_j$ digit.

$$\mathbf{b}_j^{NR} = 2n_{2j+1} + n_2{}^+{}_j: \qquad (5)$$

Equation (5) results from the fact that n_{2j+1} is nega-tively signed and n^+_{2j} is positively signed.

Step 5: j := j + 1.

where b 1 = 0. Each MB digit is represented by the bits s, one and two (Table 1). The bit s shows if the digit is negative (s=1) or positive (s=0). One shows if the absolute value of a digit equals 1 (one=1) or not (one=0). Two shows if the absolute value of a digit equals 2 (two=1) or not (two=0). Using these bits, we calculate the MB digits $\mathbf{b}^{MB}_j$ as follows:

$$b_{MB}{}^j = (-1)^{sj} \; . (onej + 2twoj) \qquad (3)$$

Equations (4) form the MB encoding signals.

$$\begin{aligned} s_j = b_{2j+1}, \quad one_j = b_{2j-1} \oplus b_{2j}, \\ two_j = (b_{2j+1} \oplus b_{2j}) \wedge \overline{one}_j. \end{aligned} \qquad (4)$$

NR4SD⁺ Algorithm

Step 1: Consider the initial values j = 0 and c_0=0.

Step 2: Calculate the carry positively signed c_{2j+1} (+) and the negatively signed sum n_{2j} (-) of a HA* with inputs b_{2j} (+) and c_{2j} (+) (Fig. 2a). The carry c_{2j+1} and the sum n_{2j} of the HA* relate to its inputs as follows:

The outputs of the HA* are analyzed at gate level in the following equations:

$$c_{2j+1} = b_{2j} \_ c_{2j}; \; n_{2j} = b_{2j} \quad c_{2j}:$$

Step 3: Calculate the carry c_{2j+2} and the sum n^+_{2j+1} of a HA with inputs b_{2j+1} and c_{2j+1}.

$$c2j+2 = b2j+1 \wedge c2j+1; \; n^+{}_{2j+1} = b2j+1 \quad c2j+1:$$

Step 4: Calculate the value of the $\mathbf{b}^{NR}{}_j{}^+$ digit.

$$\mathbf{b}_j^{NR+} = 2n_2{}^{+}{}_{j+1} \quad n_{2j}: \qquad (7)$$

Equation (7) results from the fact that n+ $_{2j+1}$ is positively
signed and n−2j is negatively signed.
Step 5: j := j + 1.

Step 6: If (j < k-1), go to Step 2. If (j = k-1), encode the most significant digit according to MB algorithm and considering the three consecutive bits to be b2k−1, b2k−2 and c2k−2 (Fig. 2b). If (j = k), stop. Table 3 shows how the NR4SD+ digits are formed. Equations (8) show how the NR4SD+ encoding signals one+ j , one−j and two+ j of Table 4 are generated.

Table 4
Numerical Examples of the Encoding techniques

2's Complement	10000000	10011010	01011001	01111111
Integer	-128	-102	+89	+127
Modified Booth	2000	2212	1221	2001
NR4SD-	2000	1212	2221	2001
NR4SD+	2000	2122	1121	2001

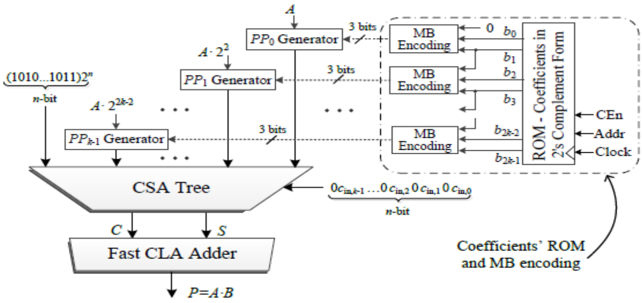

Fig. 3. System Architecture of the Conventional MB Multiplier.

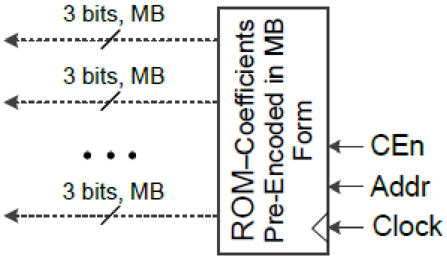

Fig. 5. The ROM of Pre-Encoded Multiplier with Standard Coefficients in MB Form.

The generation of the ith bit pj;i of the partial product PPj is illustrated at gate level in Fig. 4a [6], [7]. For the computation of the least and most significant bits of PPj, we consider a $_{1}$=0 and a$_n$ = a$_n$ $_{1}$, respectively.

After shaping the partial products, they are added, properly weighted, through a Carry Save Adder (CSA) tree along with the correction term (COR):

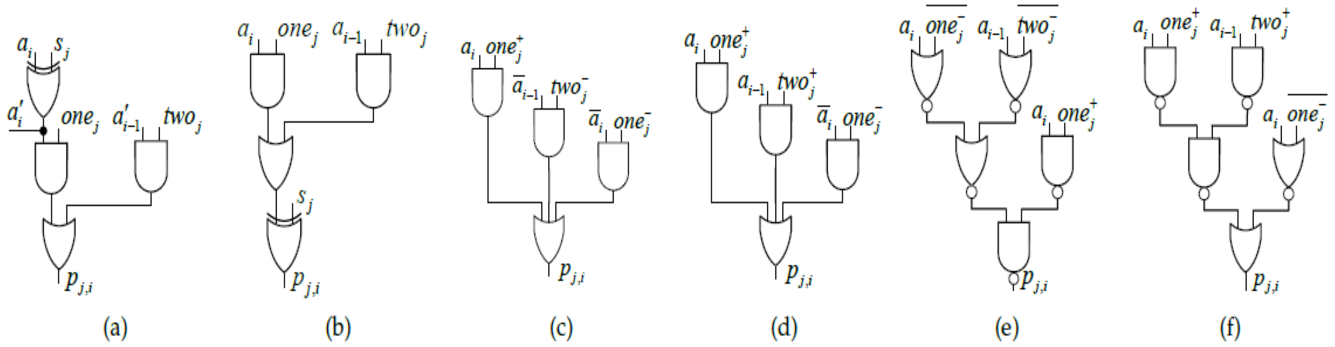
a_i s_j
a_i' one_j a_{i-1}' two_j
$p_{j,i}$
(a)
a_i one_j a_{i-1} two_j
s_j
$p_{j,i}$
(b)
a_i one_j^+
$\bar{a}_{i-1}$ two_j^-
$\bar{a}_i$ one_j^-
$p_{j,i}$
(c)
a_i one_j^+
a_{i-1} two_j^+
$\bar{a}_i$ one_j^-
$p_{j,i}$
(d)
a_i $\overline{one_j^-}$ a_{i-1} $\overline{two_j^-}$
a_i one_j^+
$p_{j,i}$
(e)
a_i one_j^+ a_{i-1} two_j^+
a_i $\overline{one_j^-}$
$p_{j,i}$
(f)

Fig. 4. Generation of the ith Bit p$_{j;i}$ of PP$_j$ for a) Conventional, b) Pre-Encoded MB Multipliers, c) NR4SD , d) NR4SD$^+$ Pre-Encoded Multipliers, and e) NR4SD , f) NR4SD$^+$ Pre-Encoded Multipliers after reconstruction.

1) to be stored in the ROM. Since the n-bit coefficient B needs three bits per digit when encoded in MB form, the ROM width requirement is 3n/2 bits per coefficient. Thus, the width and the overall size of the ROM are increased by 50% compared to the ROM of the conventional scheme (Fig. 3).

4.3 Pre-Encoded NR4SD Multipliers Design

The system architecture for the pre-encoded NR4SD multipliers is presented in Fig. 6. Two bits are now stored in ROM: n$_{2j+1}$, n$^+_{2j}$ (Table 2) for the NR4SD or n$^+_{2j+1}$, n$_{2j}$ (Table 3) for the NR4SD$^+$ form. In this way, we reduce the memory requirement to n+1 bits per coefficient while the corresponding memory required for the pre-encoded MB scheme is 3n/2 bits per coefficient. Thus, the amount of stored bits is equal to that of the conventional MB design, except for the most significant digit that needs an extra bit as it is MB encoded. Compared to the pre-encoded MB multiplier, where the MB encoding blocks are omitted, the pre-encoded NR4SD multipliers need extra hardware to generate the signals of (6) and (8) for the NR4SD and NR4SD$^+$ form, respectively. The NR4SD encoding blocks of Fig. 6 implement the circuitry of Fig. 7.

Each partial product of the pre-encoded NR4SD and NR4SD$^+$ multipliers is implemented based on Fig. 4c and 4d, respectively, except for the $PP_{k\ 1}$ that corresponds to the most significant digit. As this digit is in MB form, we use the PPG of Fig. 4b applying the change mentioned in Section 4.2 for the s_j bit. The partial products, properly weighted, and the correction term (COR) of (11) are fed into a CSA tree. The input carry $c_{in;j}$ of (11) is calculated as $c_{in;j}$ = two_j _ one_j and $c_{in;j}$ = one_j for the NR4SD and NR4SD$^+$ pre-encoded multipliers, respectively, based on Tables 2 and 3. The carry-save output of the CSA tree is finally summed using a fast CLA adder.

III. Implementation Results

We implemented in Verilog the multiplier designs of Table 5.

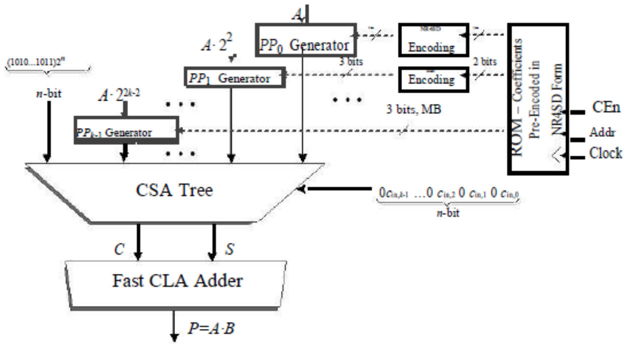

Fig. 6. System Architecture of the NR4SD Multipliers.

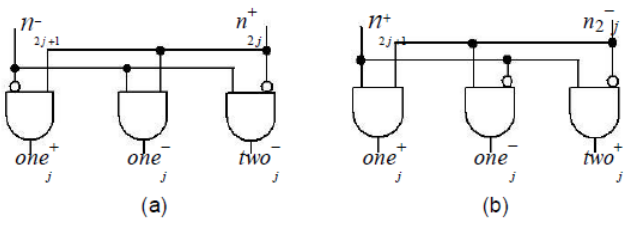

Fig. 7. Extra Circuit Needed in the NR4SD Multipliers to Complete the (a) NR4SD and (b) NR4SD$^+$

Encoding of a negative digit. In order to avoid these inverters and, thus, reduce the area/power/delay of NR4SD , NR4SD$^+$ pre-encoded multipliers, the PPGs for the NR4SD , NR4SD$^+$ multipliers were designed based on primitive NAND and NOR gates, and replaced by Fig. 4e, 4f, respectively.

TABLE 5
Multiplier Designs

Design		Input A	Input B Encoding		ROM width
			Type	Technique	
Conventional MB		2's complement n-bit	MB	MB encoding	n-bit
Pre-Encoded	MB		MB	Fully Pre-Encoded	3n/2-bit
	NR4SD		NR4SD	Partially Pre-Encoded	(n+1)-bit
	NR4SD$^+$		NR4SD$^+$		(n+1)-bit

memory compiler of the same library provided the physical ROMs for the coefficients. Since the ROMs re-quired for the pre-encoded multipliers are larger than the one for the conventional MB scheme, access time is increased. However, the pre-encoded designs may achieve lower clock periods than the conventional MB one because the encoding circuits that are included in the critical path, are omitted or less complex. We first synthesized each design at the lowest achievable clock period and then, each pre-encoded design at the clock period achieved by the conventional MB scheme. We also synthesized all designs at higher clock periods targeting to explore their behavior under different timing constraints in terms of area and power consumption. For each clock period, we simulated all designs using Modelsim and 20 different sets of 512 ROM words. For the conventional MB multiplier, the 2's complement inputs were randomly generated with equal possibility of a bit to be 0 or 1. Using a high level programming language, we generated the pre-encoded values of B which we then stored in the ROMs of pre-encoded designs. Finally, we used Synopsys Prime Time to calculate power consumption.

The performance of the proposed designs is considered with respect to the width of the input numbers, i.e., 16, 24 and 32 bits. Table 6 summarizes the performance of each architecture at minimum possible clock period. We observe that the pre-encoded NR4SD architectures are more area efficient than the conventional or pre-encoded MB designs with respect to their performance in the lowest possible clock periods. Regarding power dissipation, the pre-encoded NR4SD scheme consumes the least power which, in the cases of 16 and 24 bits of input width, is equal to the power consumed by the pre-encoded MB design.

TABLE 6
Performance at Lowest Clock Periods

Design		D^a	A^b	A D	P^c	P D
16 bits						
Conventional MB		2.01	18945	38079	11.20	22.51
Pre-Encoded	MB	1.95	19079	37205	11.00	21.45
	NR4SD	1.95	18357	35796	11.00	21.45
	$NR4SD^+$	1.95	18485	36045	11.10	21.65
24 bits						
Conventional MB		2.26	32354	73121	18.80	42.49
Pre-Encoded	MB	2.18	30251	65948	16.80	36.62
	NR4SD	2.18	28822	62832	16.80	36.62
	$NR4SD^+$	2.18	29573	64470	17.40	37.93
32 bits						
Conventional MB		2.34	50199	117465	29.50	69.03
Pre-Encoded	MB	2.28	47490	108277	27.30	62.24
	NR4SD	2.28	43779	99816	26.40	60.19
	$NR4SD^+$	2.29	46318	106068	27.50	62.98

a. Delay for ROM+Multiplier in ns.
b. Area occupation for ROM+Multiplier in um^2.
c. Power consumption for ROM+Multiplier in mW.

IV. Conclusion

In this paper, new designs of pre-encoded multi-pliers are explored by off-line encoding the stan-dard coefficients and storing them in system mem-ory. We propose encoding these coefficients in the Non-Redundant radix-4 Signed-Digit (NR4SD) form. The proposed pre-encoded NR4SD multiplier designs are more area and power efficient compared to the conventional and pre-encoded MB designs. Extensive experimental analysis verifies the gains of the pro-posed pre-encoded NR4SD multipliers in terms of area complexity and power consumption compared to the conventional MB multiplier.

References

[1] G. W. Reitwiesner, "Binary arithmetic," Advances in Computers, vol. 1, pp. 231–308, 1960.

[2] K. K. Parhi, VLSI Digital Signal Processing Systems: Design and Implementation. John Wiley & Sons, 2007.

[3] K. Yong-Eun, C. Kyung-Ju, J.-G. Chung, and X. Huang, "Csd-based programmable multiplier design for predetermined coefficient groups," IEICE Trans. Fundam. Electron. Commun. Comput. Sci., vol. 93, no. 1, pp. 324–326, 2010.

[4] O. Macsorley, "High-speed arithmetic in binary computers," Proc. IRE, vol. 49, no. 1, pp. 67–91, Jan. 1961.

[5] W.-C. Yeh and C.-W. Jen, "High-speed booth encoded parallel multiplier design," IEEE Trans. Comput., vol. 49, no. 7, pp. 692– 701, Jul. 2000.

[6] Z. Huang, "High-level optimization techniques for low-power multiplier design," Ph.D. dissertation, Department of Com-puter Science, University of California, Los Angeles, CA, 2003.

[7] Z. Huang and M. Ercegovac, "High-performance low-power left-to-right array multiplier design," IEEE Trans. Comput., vol. 54, no. 3, pp. 272–283, Mar. 2005.

[8] Y.-E. Kim, K.-J. Cho, and J.-G. Chung, "Low power small area modified booth multiplier design for predetermined coeffi-cients," IEICE Trans. Fundam. Electron. Commun. Comput. Sci., vol. E90-A, no. 3, pp. 694–697, Mar. 2007.

[9] C. Wang, W.-S. Gan, C. C. Jong, and J. Luo, "A low-cost 256-point fft processor for portable speech and audio appli-cations," in Int. Symp. on Integrated Circuits (ISIC 2007), Sep. 2007, pp. 81–84.

[10] A. Jacobson, D. Truong, and B. Baas, "The design of a recon-figurable continuous-flow mixed-radix fft processor," in IEEE Int. Symp. on Circuits and Syst. (ISCAS 2009), May 2009, pp. 1133–1136.

[11] Y. T. Han, J. S. Koh, and S. H. Kwon, "Synthesis filter for mpeg-2 audio decoder," Patent US 5 812 979, Sep., 1998.

[12] M. Kolluru, "Audio decoder core constants rom optimization," Patent US 6 108 633, Aug., 2000.

[13] H.-Y. Lin, Y.-C. Chao, C.-H. Chen, B.-D. Liu, and J.-F. Yang, "Combined 2-d transform and quantization architectures for h.264 video coders," in IEEE Int. Symp. on Circuits and Syst. (ISCAS 2005), vol. 2, May 2005, pp. 1802–1805.

[14] G. Pastuszak, "A high-performance architecture of the double-mode binary coder for h.264.avc," IEEE Trans. Circuits Syst. Video Technol., vol. 18, no. 7, pp. 949–960, Jul. 2008.

[15] J. Park, K. Muhammad, and K. Roy, "High-performance fir filter design based on sharing multiplication," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 11, no. 2, pp. 244–253, Apr. 2003.

[16] K.-S. Chong, B.-H. Gwee, and J. S. Chang, "A 16-channel low-power nonuniform spaced filter bank core for digital hearing aids," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 53, no. 9, 853–857, Sep. 2006.

[17] B. Paul, S. Fujita, and M. Okajima, "Rom-based logic (rbl) design: A low-power 16 bit multiplier," IEEE J. Solid-State Circuits, vol. 44, no. 11, pp. 2935–2942, Nov. 2009.

[18] M. D. Ercegovac and T. Lang, "Multiplication," in Digital Arithmetic. San Francisco: Morgan Kaufmann, 2004, pp. 181– 245.

[19] N. Weste and D. Harris, CMOS VLSI Design: A Circuits and Systems Perspective, 4th ed. USA: Addison-Wesley Publishing Company, 2010.

[20] "Dual dsp plus micro for audio applications," Feb. 2003, TDA7503 Datasheet, STMicroelectronics.

[21] C. Xu, X. Dong, N. Jouppi, and Y. Xie, "Design implications of memristor-based rram cross-point structures," in Design, Automation Test in Europe Conf. Exhibition (DATE), Mar. 2011, pp. 1–6.