To appear in the *International Journal of Electronics*
Vol. 00, No. 00, January 2013, 1–9

International Journal of Electronics
Coarse Grain Reconfigurable Multiplier Architectures For Signal Processing Applications

Kishore. P, Kowshick. B, Vivekanandan. B, Abhinav. M and Moorthi. S*

VLSI Systems Research Laboratory, Department of Electrical and Electronics Engineering, National Institute of Technology, Tiruchirappalli, Tamilnadu, INDIA

(*v4.0 released January 2013*)

There has been tremendous increase in the usage of multimedia hand-held devices owing to the developments in electronic technology. These battery operated devices with no cooling fan demand reduction in power consumption for increased performance and longer battery life. These devices perform intense digital signal processing which are highly multiplication centric. Development of low power multiplier designs for Signal Processing applications will significantly contribute towards the performance of mobile devices. In this paper, a low power reconfigurable multi-precision multiplier architecture is presented and its performance is compared with Booth Radix-4 and Wallace-Tree multipliers. The multiplier configures itself based on the effective size of the input operands preventing un-necessary power loss. The effect of word-length of the fundamental block and the adder used for adding partial products on the performance is studied by implementing variants of the proposed architecture. The functional performance of proposed multiplier is evaluated by enhancing images by unsharp masking using high pass FIR filter.

Keywords: reconfigurable multiplier; truncation multiplier; Signal Processing; Truncation adder; high pass FIR filter;

1. Introduction

With rapid advancements in technology over the last decade there has been an exponential growth in the use portable devices and mobile phones. Smart phones and tablets are increasingly used for complex multimedia applications like gaming, high-quality audio playing and High-Definition video streaming apart from their regular functions. These applications require intense digital signal processing. Being the largest silicon consumer in a digital processor, multipliers are a very important component in Digital Signal Processing followed by adders. Designing low power and fast multiplier and adder is instrumental for the development of hand-held multi-media devices with better real-time performance and longer battery-life. There have been previous attempts to improve the power, area and speed performances of multipliers. Some of the previous research works attempt to use alternative structure and algorithm for computing the product. Yuan (2010) talks about a multiplier where the area has been optimised using Horner's scheme

*Corresponding author. Email: srimoorthi@nitt.edu and Canonical Signed Digit numeric format. However, their power and speed performances have not been analysed. Kim (1998) presents a modified Booth algorithm where operands are swapped to reduce power consumption. While Hwang (2006) presents a low power multiplier design using column addressing schemes at the cost of circuit overhead.

Other researches aimed at enhancing the performances of the multipliers intend to do so by compromising one of the factors like accuracy, speed and power consumption. One way to reduce power consumption and delay is by reducing the word length of the inputs. The two most common ways to reduce word-length - right shift method and truncation method are compared in Guia (2010). It states that truncation multipliers have better relation between power saved to accuracy than right shift reduction. Similar results are provided in Han (2005) which explains the application of word-length reduction techniques in multi-precision multipliers. Signed right shift method and truncation method have been applied on 16-bit Radix-4 and Wallace multipliers and their performances were compared leading to a conclusion that truncation method gives better results. The dynamic power loss which is the largest contributor for power losses depends on the number of bits (of multiplier) being switched, Han (2005) shows that normally for a n-bit multiplier, the expected value of switching is n/2. While for truncation it is (n-m)/2, where m is the number of truncated bits (having value 0).

Apart from reducing word-length, research over using reconfigurable architectures for enhancing performance is also being done. Haynes (1998) presents a reconfigurable multiplier array which is more area efficient than equivalent multiplier implemented in conventional FPGA structure. Sakthi (2011) presents a multiplier which uses 4-N/2 bit multipliers to achieve N bit multiplication and can be operated in 6 different modes. Koutroumpezis (2002) presents a reconfigurable multiplier which decomposes a higher word length multiplication into smaller ones and the weighted partial products are added to obtain the final result. A multiplier design which can be configured for different word-lengths is presented in Moni (2011). But this work did not deal with the effect of the adder used for partial product addition and the word length size of the fundamental building blocks on the performance of the multiplier.

In a conventional adder circuit, the delay is mainly attributed to the carry propagation chain along the critical path, from the least significant bit (LSB) to the most significant bit (MSB). Meanwhile, a significant proportion of the power consumption of an adder is due to the glitches that are caused by the carry propagation. Therefore, if the carry propagation can be eliminated or curtailed, a great improvement in speed performance and power consumption can be achieved. Zhu (2010) addresses the issue caused by carry propagation in various adder architectures and discusses the possible solution to eliminate the carry without compensating much on the accuracy.

The paper proposes a reconfigurable multi-precision multiplier architecture which configures itself based on the effective size of the input operands. The proposed architecture is developed to reduce power consumption and increase computation speed. Multiple instances of the same fundamental blocks are used to achieve the multiplication. Two variants of the architecture have been developed with different word-lengths of fundamental blocks and their performances have been studied. Further, different adder algorithms are used to add the partial products and their performances are evaluated. In most of the signal processing computations, one of the multiplier operand is smaller than the other as shown in Walters (2011), such patterns of input operands are exploited in proposed design to reduce error while truncation.

Section 2 explains the proposed multiplier design and the variants are explained in its subsections. Section 3 consists of the implementation details. Sections 4 and 5 present the performances of the proposed design and the conclusions inferred by comparing their performance against other multiplier designs.

2. Proposed Multiplier Architecture

The proposed multiplier is a scalar multi-precision multiplier capable of performing signed integer multiplication. It receives inputs in signed two's complement format. The proposed design consists of multiplier blocks of smaller word-length which work together to perform the final multiplication. The number of these blocks used to compute the final result depends on the word-length of the fundamental multiplier block and the actual size of the operands. A multiplier consisting of n-bit fundamental blocks will require $2^{(N/n)}$ blocks to perform N-bit multiplication. Two design variants have been proposed a) Architecture-I using N/2-bit fundamental multiplier blocks b) Architecture-II using N/4-bit fundamental blocks where N is the word-length of the multiplier.

Unsigned multiplication algorithms are used for the fundamental blocks since they work perfectly with the proposed design where operands are decomposed into smaller blocks and multiplied. Negative operands are accommodated by obtaining their absolute value. The effective size of the positive operands obtained is computed. Effective size is the minimum word-size required to store a number. Based on the effective sizes of the operands the word-length of the multiplication to be performed is decided and the required fundamental blocks are enabled. The partial products obtained are appropriately weighted and added to receive the unsigned product. Depending on the sign of the input operands the unsigned product is negated to obtain the final result.

2.1 *Architecture-I*

In architecture-I, N/2 bit fundamental blocks are used to perform the multiplication, N is the word-length of the multiplier. Depending on the effective size of the operands, N-bit or N/2-bit multiplication is performed. All the 4 fundamental blocks are enabled to perform the N-bit multiplication while only single fundamental block is enabled for performing N/2-bit multiplication thus saving power and increasing speed. When N/2 bit multiplication is performed, the output is left-shifted appropriately to accommodate the truncation performed. Fig. 1 shows the block diagram of Architecture-I.

2.2 *Architecture-II*

In this architecture N/4-bit fundamental blocks are used which allows N/4-bit, N/2-bit, 3N/4-bit and N-bit multiplication to be performed. The effective size of operands determines the kind of multiplication to be performed, the word-length

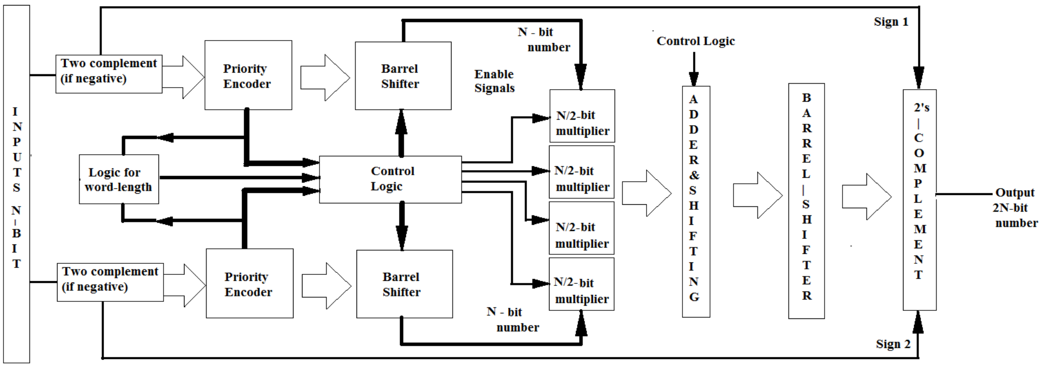


Figure 1. Block diagram of Multiplier Architecture-I

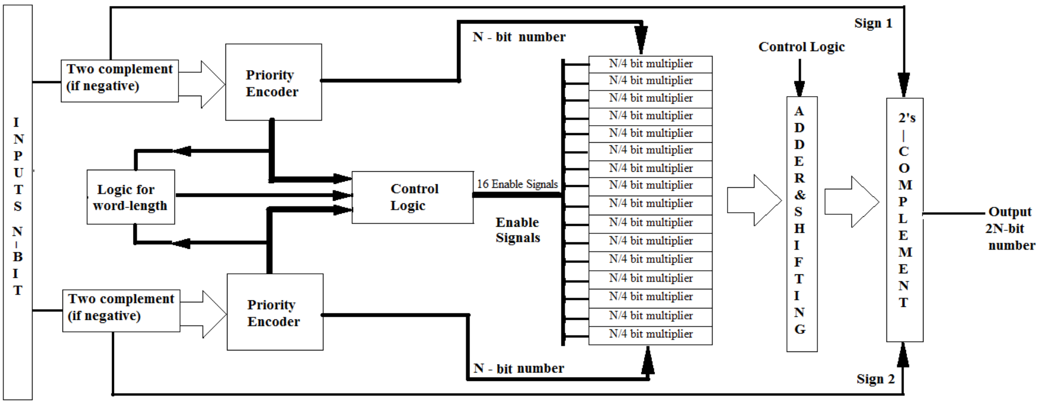


Figure 2. Block diagram of Multiplier Architecture-II

is selected such that the input operands are not truncated and accuracy is not lost. Appropriate number of fundamental blocks required to perform the multiplication are enabled. Fig. 2 shows the block diagram of Architecture-II.

3. Implementation

The proposed multiplier architectures have been implemented using Verilog hardware description language. 8-bit and 32-bit multipliers of both the variants (design-I and design-II) have been implemented using 180nm CMOS technology and their performances are compared with Wallace Tree and Radix-4 Booth multiplier algorithms of corresponding word-lengths. Further, different adder algorithms have been used to perform the addition of partial products in the adder blocks shown in figures 1 and 2. The performance of multipliers using carry select adder, carry save adder and modified version of truncation adder presented in Zhu (2010) is evaluated and tabulated. Cadence RTL compiler and SoC encounter are used to simulate the functionality of the proposed multipliers and to compute their power, area and worst case delay characteristics along with their layouts.

Functional performances of the proposed multiplier designs are also evaluated and compared with that of Booth Radix-4 multipliers. The functional performance is evaluated by enhancing images by unsharp-masking using a high pass FIR filter.

Table 1. Performance characteristics of multipliers with 8-bit input

Multiplier type	Fundamental Block	Adder Type	Power(μw)	Delay(ps)	Area (sq μm)
Radix 4	N/A	N/A	347.66	5514	3536
Wallace	N/A	N/A	412.71	5718	4747
Architecture-1	N/2 (4-bits)	Truncation Error Adder	220.74	9406	11283
Architecture-1	N/2 (4-bits)	Carry Save Adder	222.39	8250	11310
Architecture-1	N/2 (4-bits)	Carry Select Adder	226.601	9112	12105
Architecture-2	N/4 (2-bits)	Truncation Error Adder	271.57	7379	10079
Architecture-2	N/4 (2-bits)	Carry Save Adder	289.76	6108	8589
Architecture-2	N/4 (2-bits)	Carry Select Adder	326.06	7816	12773

Table 2. Performance characteristics of multipliers with 32-bit input

Multiplier type	Fundamental Block	Adder Type	Power(μw)	Delay(ps)	Area (sq μm)
Radix 4	N/A	N/A	12818.52	23592	54500
Architecture-1	N/2 (16-bits)	Truncation Error Adder	2483.874	36226	109229
Architecture-1	N/2 (16-bits)	Carry Save Adder	2316.174	28810	105763
Architecture-1	N/2 (16-bits)	Carry Select Adder	2401.698	28801	113251
Architecture-2	N/4 (8-bits)	Truncation Error Adder	3247.153	26601	108856
Architecture-2	N/4 (8-bits)	Carry Save Adder	3183.318	23690	104745
Architecture-2	N/4 (8-bits)	Carry Select Adder	3427.855	22094	120502

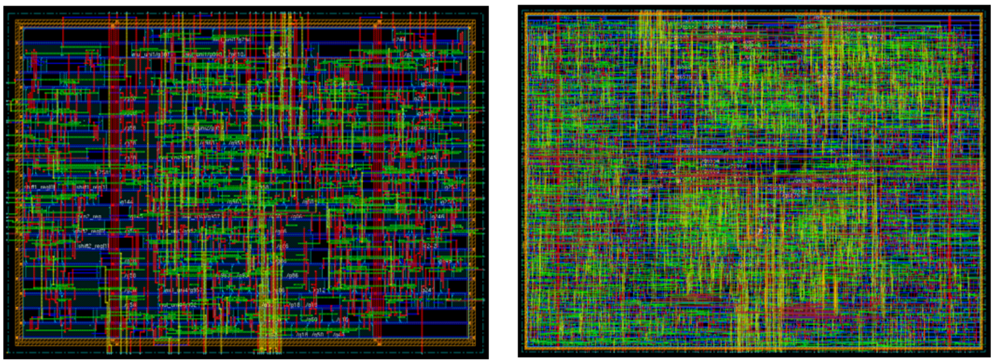
A 64 tap high pass FIR filter is designed which is used to segregate the high frequency components of the input image. The cut-off frequency of the designed FIR high pass filter is 0.04. Since the co-efficients of the filter range between -1 to +1. The co-efficients are multiplied with 2^{16} using the proposed multiplier design to obtain the modified co-efficients. The co-efficients thus obtained are convoluted with input image. The multiplication involved in this convolution is performed by the proposed multiplier. The filtered image is superimposed over original image to obtain the enhanced image. The images sharpened using proposed multiplier and other multiplier algorithms are compared and presented in forthcoming results section.

4. Results

The power, area and worst path delay of the proposed multiplier is tabulated and compared with that of Wallace tree and Radix-4 Booth multiplier algorithm. The power, worst path delay and area of 8-bit and 32-bit multipliers are tabulated in table 1 and 2 respectively. The layouts of the proposed multipliers obtained using Cadence SoC encounter tool are shown in fig. 3. The sample image considered is shown in fig. 4. The high pass filtered image and enhanced image obtained using Radix-4 Booth multiplier is shown in fig. 5(a) and fig. 5(b) respectively.

The filtered and sharpened image obtained using proposed multiplier architecture-I with truncation adder is shown in fig 6(a) and fig 6(b) respectively. Fig 6(c) and fig 6(d) shows the filtered and sharpened images obtained using proposed multiplier architecture-I with carry save adder.

Figures 7(a),7(b),7(c) and 7(d) show the filtered and the sharpened images obtained by unsharp masking implemented with proposed multiplier architecture-II with truncation error adder and carry save adder respectively.



(a) Layout of 8 bit multiplier using error adder (b) Layout of 32 bit multiplier using error adder

Figure 3. Layouts of Multiplier Architecture-I obtained using Cadence

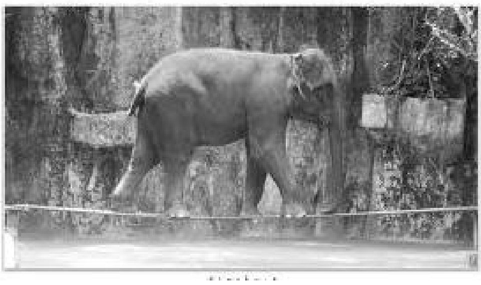
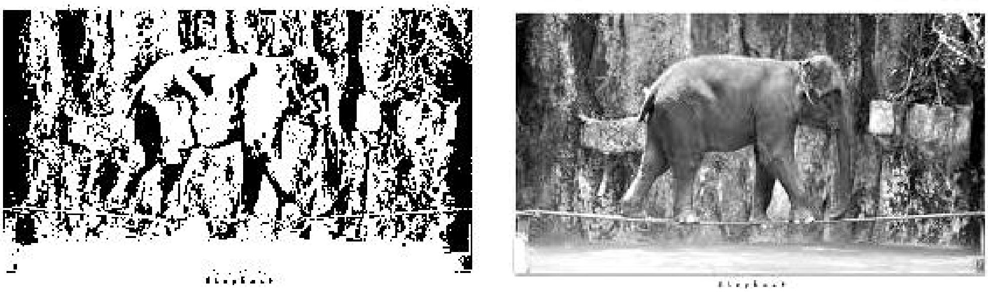


Figure 4. Original Image

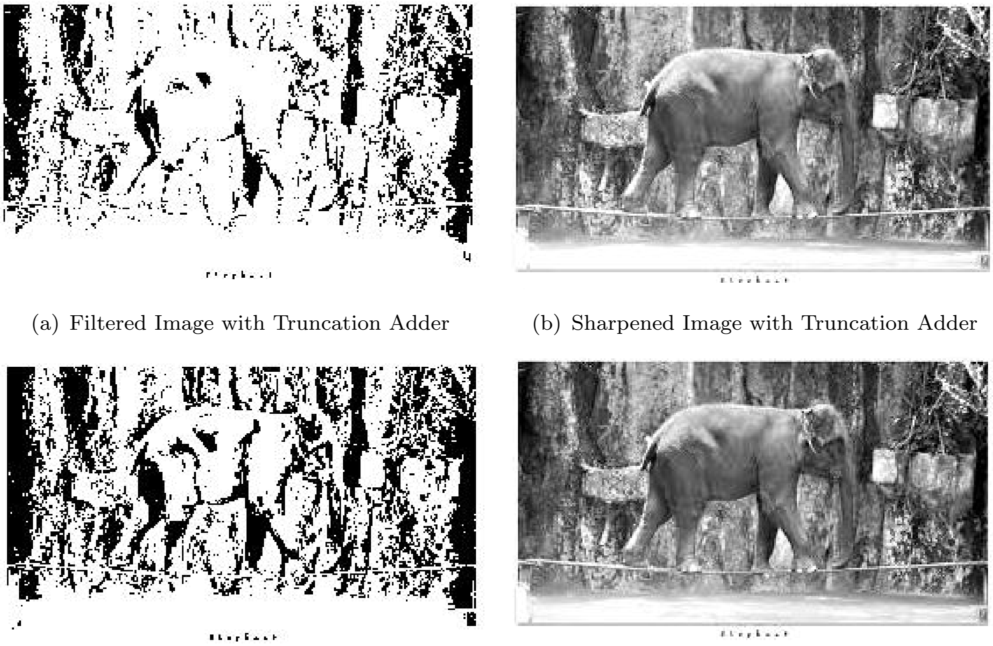


(a) Filtered Image with Booth multiplier (b) Sharpened Image with Booth multiplier

Figure 5. Images using booth multiplier

5. Conclusion

8-bit and 32-bit versions of both the proposed low power multiplier architectures are successfully implemented and verified. Their performances are compared with that of the commonly used multiplier algorithms namely Wallace tree and Radix-4 Booth algorithm and are tabulated in table 1 and 2. It is seen that the performances of developed 8-bit and 32-bit multiplier architectures are superior to commonly used multipliers when power consumption is concerned. Proposed Architecture-I consumes 80% less than the total power consumed by radix-4 booth multiplier. Proposed Architecture-II saves 72% of the power compared to radix-4 booth multiplier. The worst case delay for proposed architecture-2 (using carry-save adder) is 0.4% more than radix multipliers for 32-bit and is 8.7% more in the case of 8-bit multipliers. The performances of both 8 and 32-bit multipliers show that



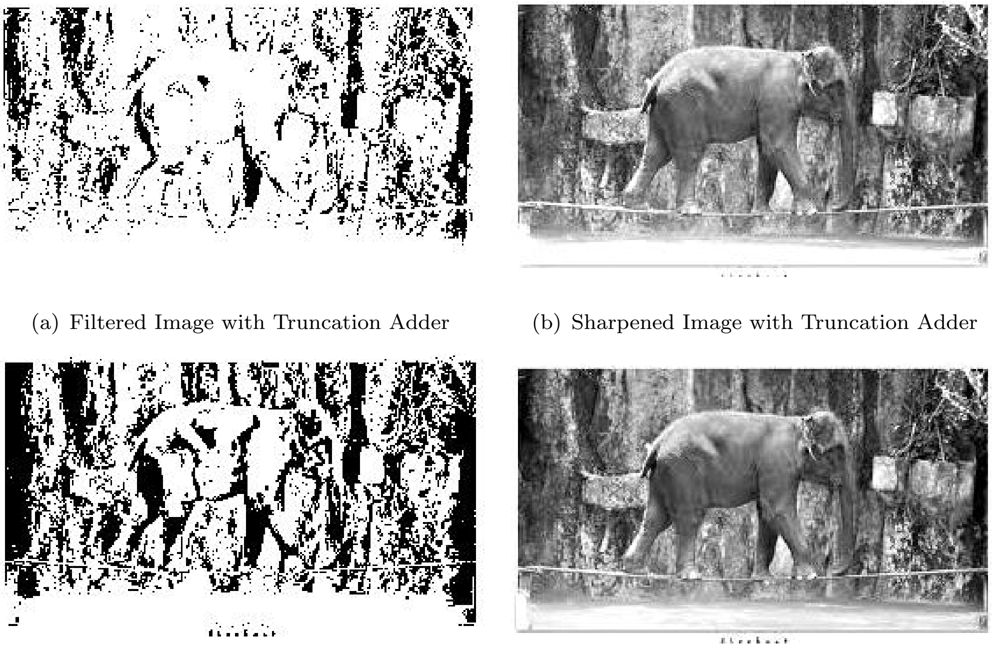
(a) Filtered Image with Truncation Adder

(b) Sharpened Image with Truncation Adder

(c) Filtered Image Carry Save Adder

(d) Sharpened Image with Carry Save Adder

Figure 6. Image obtained using Proposed Multiplier architecture-I



(a) Filtered Image with Truncation Adder

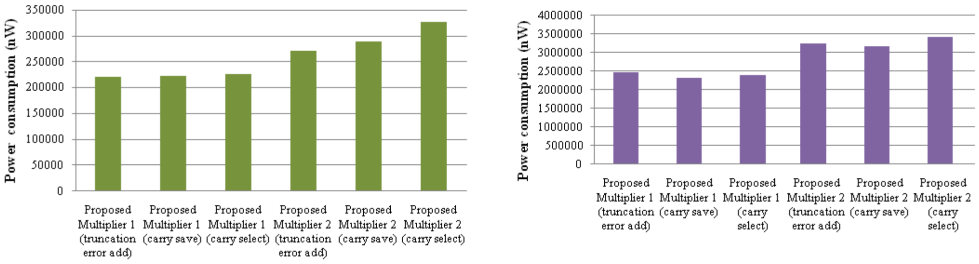
(b) Sharpened Image with Truncation Adder

(c) Filtered Image Carry Save Adder

(d) Sharpened Image with Carry Save Adder

Figure 7. Image obtained using Proposed Multiplier Architecture-II

architecture-I provides higher power saving (8% more) than architecture-II as can be seen from fig. 8. However, for mobile and hand-held devices which require lower worst case delay and power consumption, multiplier architecture-II has the best overall performance. On examining the effect of adders on the performance of the multipliers, it is found that using truncation adder results in higher power saving



(a) 8-bit multiplier architectures

(b) 32-bit multiplier architectures

Figure 8. Power comparison of multiplier architectures

but considering delay and power consumption which is required for mobile applications, using carry save adder results in better performance. The performance of the proposed multiplier designs are further evaluated by performing unsharp masking to sharpen blurred images using developed multiplier and its performance is compared with that of Radix-4 Booth multiplier. Fig. 4 shows an input image which is enhanced. The filtered images obtained using proposed multiplier architecture-I and II with truncation adder differs from the image obtained by using Booth Radix-4 multiplier algorithm. However, there is no significant difference between the outputs from proposed multipliers using carry save adder and Booth Radix-4 multiplier.

In this work emphasis is given on reconfigurability and truncation logic for increasing power saving and speed. However more power saving can achieved by exploring faster and better multiplier algorithms to implement the fundamental multiplier block.

References

Guia Solaz, M. & Richard, Conway (2010). Comparative study on wordlength reduction and truncation for low power multipliers,*MIPRO, 2010 Proceedings of the 33rd International Convention*,(pp. 84-88), Opatija, Croatia.

Han, K., Evans, B.L. & Swartzlander Jr., E.E (2005). Low power multipliers with data wordlength reduction. Retrieved from http://users.ece.utexas.edu/ bevans/papers/2005/lowPower/LowPowerMultAsilomar2005Paper.pdf.

Haynes, S.D & Cheung, P.Y.K (1998). A reconfigurable multiplier array for video image processing tasks, suitable for embedding in an FPGA structure,*IEEE symposium on FPGAs for custom computing machines*,(pp. 226-234), Napa Valley, Canada.

Hwang, Y.T., Lin,J.F., Sheu, M.H. & Sheu, C.J (2006). Low power multiplier designs based on improved column bypassing schemes, *IEEE Asia Pacific Conference on Circuits and Systems*,(pp. 594-597), Singapore.

Kim, J.H., Lee, J.S. & Cho, J.D. (1998). A low power booth multiplier based on operand swapping in instruction level, *Journal of Korean Physical Society*,(Vol. 33, pp. 258-261).

Koutroumpezis, G., Tatas, K. , Soudris, D., Blionas, S., Masselos, K. & Thanailakis, A. (2002). Architecture Design of a Reconfigurable Multiplier for Flexible Coarse-Grain Implementations, *Field-Programmable Logic and Applications: Reconfigurable Computing Is Going Mainstream*,(Vol. 23438, pp. 1027-1036).

Moni, D.J. & Sophia, P.E. (2011). Radix 4 configurable booth multiplier for low power and high speed applications, *IOSR Journal of Electronics and Communication Engineering*,(Vol. 4, Issue 2, pp. 31-77).

Sakthi, S.S & Kayalvizhi, N. (2011). Power Aware Reconfigurable Multiplier for DSP Applications, *IEEE Recent Advances in Intelligent Computational Systems*,(pp. 352-356), Trivandrum.

Walters, E.G. & Schulte, M.J. (2011). Truncated-Matrix multipliers with coefficient shifting,*Forty Fifth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*,(pp. 176-180), Pacific Grove, California.

Yuan, X. , Ying, T. & Chunpeng, T. (2010). Improved design of the multiplier in the design of the filter, *IEEE International Conference on Computer and Communication Technologies in Agricultural Engineering*,(pp. 13-16), Chengdu.

Zhu, N., Goh, W.L , Zhang, W., Yeo, K.S & Kong, Z.H (2010). Design of Low-Power High-Speed Truncation-Error-Tolerant Adder and Its Application in Digital Signal Processing, *Very Large Scale Integration (IEEE Transactions on Very Large Scale Integration (VLSI) Systems*,(Vol. 18, Issue 8, pp. 1225-1229).