Circuits and Systems, 2016, 7, 1701-1708
Published Online June 2016 in SciRes. http://www.scirp.org/journal/cs
http://dx.doi.org/10.4236/cs.2016.78147

Hybrid Segment Approximate Multiplication for Image Processing Applications

Jamuna Ramasamy[1], Sathishkumar Nagarajan[2]

[1]Department of Electronics & Communication Engineering, SNS College of Engineering, Coimbatore, India
[2]Department of Electronics & Communication Engineering, Sri Ramakrishna Engineering College, Coimbatore, India
Email: jamuname9@gmail.com, sathishkumar.n@srec.ac.in

Received 13 April 2016; accepted 1 May 2016; published 16 June 2016

Abstract

It is critical in terms of approximate computation errors in VLSI multiplier circuits are increasing with technology scaling. The most common method for fast and energy efficient execution of multiplication result is approximation of operands. But this traditional approximate result is not suitable for image processing applications. This paper proposes the two architectures of high accurate hybrid segment approximate multiplier (HSAM) and enhanced HSAM for image compression. Existing static segment method based approximate multiplier is not suitable for certain accurate applications and dynamic segment method based approximate multiplier is not suitable for cost efficient applications. The proposed work combines the advantages of both static segment method and dynamic segment method to drive the efficiency in accuracy and cost. The proposed approximate multipliers HSAM8 × 8 and EHSAM8 × 8 provide 99.85% and 99.999% accuracy respectively for various inputs. The proposed HSAM consumes less energy with small increase of area overhead. The proposed EHSAM consumes less energy without any area overhead. The proposed HSAM and EHSAM is improved the speed by 40% and 85% compared to the existing SSM8 × 8 technique.

Keywords

Communication, Timing Synchronization, FPGA, Multipliers, Parallel Processing, Power, Delay

1. Introduction

Energy consumption is one of the major challenges in VLSI technology, when the CMOS technology and the design complexity are scaled to deliver the desired functionality. Approximate Arithmetic Computing (AAC) is a solution for energy efficiency challenge with some computational error [1]. Because of this approximate

How to cite this paper: Ramasamy, J. and Nagarajan, S. (2016) Hybrid Segment Approximate Multiplication for Image Processing Applications. *Circuits and Systems*, **7**, 1701-1708. http://dx.doi.org/10.4236/cs.2016.78147

computing, reduction in energy dissipation and fast execution is possible at all levels of design hierarchy. It is suitable for the applications such as signal processing, image/video processing and machine learning and neuromorphic systems. This energy efficient VLSI circuits are mainly used in real time applications such as mobile and embedded computing devices to provide reliability and to improve the life span of VLSI systems [2].

Multiplication is the power, area and time consuming operation for large operands [3]. This is the bottleneck in many applications which requires large number of multipliers. But it is one of the basic arithmetic operations in DSP and image processing applications like convolution, correlation, finite Impulse Response, infinite impulse response, fast fourier transform, multiply accumulate unit, image compression and signal compression techniques. It is also used for designing arithmetic, image and signal processors [5]. To create a remedy for this power-hungry components, approximate computation is preferred which provides low energy consumption with computational error. DSP applications have more error tolerance than image processing applications. Previous studies provide various techniques to improve energy efficiency with some error tolerance. It can be divided into three types: 1) Aggressive voltage scaling [6] [7], 2) Truncation of bits [4] [6]-[8], 3) Inaccurate building blocks usage [9]. In [6], hardware is designed with a scalable effort using aggressive voltage scaling and truncation of bits to support error resilience. In [4], under-designed multiplier for 16 bits is proposed with a simple correction mechanism, during its operation in a critical mode. But the power consumption and computational accuracy is trade-off. It uses inaccurate 2 × 2 partial product generators to achieve minimum dynamic and leakage power with some accuracy loss. Better Signal to Noise Ratio (SNR) is achieved by this proposed multiplier which is not suffering from the overheads of advanced over-scaling techniques.

Z. Babic *et al.* proposed an iterative logarithmic multiplier [3]. It uses Mitchell's algorithm without the use of logarithmic approximation. This method uses two parallel circuits for error correction. It affords less computational error with area overhead and high power consumption. In [10], a low error and high performance multiplexer based truncated multiplier is proposed with a Pseudo carry Compensation Truncation (PCT) scheme. It is implemented on TSMC 0.18 um technology and it provides low Mean Square Error (MSE).

Ritupriya Jha proposed an 8-bit shift and approximate multiplier which is suitable for embedded DSP applications [5]. It is based on Vedic mathematics, the word Vedic derived from vedha which means the store house of all knowledge. It uses Urthva Tiryakbhyam sutra which operates in vertical and crosswise manner. Due to parallel processing, it increases the speed. But it increases the area when the number of bits is increased. In [11], Mitchell's logarithmic multiplier is designed using iterative multiplier.

This iterative multiplier is mainly used in convolution process for multiplying the pixels of image with the kernel value. Because of its iteration, it reduced the delay. But this iterative process leads to the accurate result.

An approximate multiplier is presented in [12], which uses the accurate multiplier for MSB and accurate plus approximate multiplier for LSB. Since it provides little computational error, it is suitable for multimedia and wireless applications. In [13], an approximate multiplier is proposed with an approximate adder and an OR gate based error recovery module for error reduction. When compared to the wallace tree, it reduced the critical path delay with the cost of quality degradation. An approximate multiplier using 4:2 compressors is presented in [14]. It is based on recursive multiplication technique. It provides the reduction in power and delay, high accuracy with area overhead.

The rest of this paper is structured as follows. Section 2 explains the multiplier less multiplier. Different types of proposed approximation techniques are presented in Section 3. Section 4 gives the detail discussion about the results like energy consumption, power, delay and area of different approximate multipliers. The case study on image processing applications is given in Section 5. Finally Section 6 concludes the paper.

2. Proposed Multiplier Less Multiplier (MLM)

The main components of the proposed multiplier less multiplier (MLM) are shifter, multiplexer and adder. The shifter is used to perform the right shift operation of one of the input operand. The shifted values and zeros are given as the inputs to the multiplexer which select the appropriate values based on the other input operand is also called as control operand. One of the inputs of the multiplexer is always zero. If the control operand bit of the multiplexer is zero then the output of the multiplier is zero, otherwise it selects the multiplicand as the output (**Figure 1**).

Finally, these parallel outputs are added to get the appropriate output. The fastest adder named as carry select adder is used in this proposed multiplier for high speed operation. When compared to the conventional multipliers like array multiplier, Wallace tree and dadda multiplier, the proposed multiplier provides high speed and

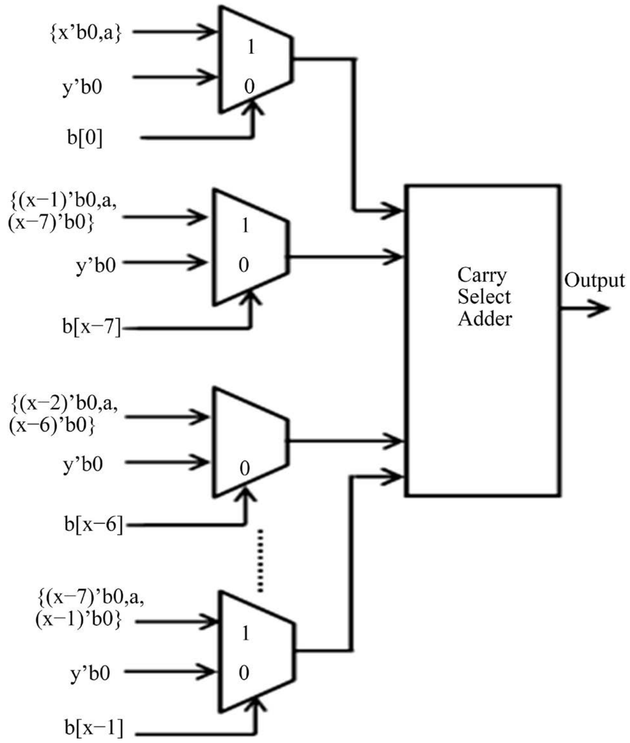


Figure 1. Proposed multiplier less multiplication.

low energy consumption. This high speed and energy efficient proposed multiplier is used in the approximate multiplier, which is suitable for mobile, embedded computing and image processing applications.

3. Proposed Approximate Multipliers

Basically two different types of approximation techniques are available. Dynamic Segment Method (DSM) is one of the techniques used for accurate approximate multiplication. It uses the eight bit segment from the leading one position of the operands. To perform these operations, it uses the leading one detector (LOD) which increases the cost with high accuracy. But the Static Segment Method (SSM) uses the OR gate instead of LOD to reduce the hardware requirement. It provides high accuracy for the LSB segments and low accuracy for MSB segments. If there is one present in the MSB, it uses the MSB segment and excludes the LSB segment which reduces the accuracy. This proposed work combines the advantages of both static segment method and dynamic segment method to drive the efficiency in accuracy and cost. In addition to that the new multiplier less multiplier is used instead of existing multipliers like wallace tree or dadda.

3.1. Hybrid Segment Approximate Multiplier (HSAM)

The proposed hybrid approximate multiplier takes m consecutive bits (*i.e.*, an m-bit segment) of an n-bit operand and selected two m-bit segments are applied to MLM method for obtaining cost efficiency. The advantages of both static segment and dynamic segment are combined, which is called as hybrid segment technique. To improve the accuracy of an approximate multiplier, hybrid segment technique is used (**Figure 2**).

It uses LOD and shifters like dynamic segment method, with the slight changes in its logic functions which provides good accuracy and low energy consumption. By reducing LOD and shifters, it reduces the area. It also provides high speed due to the proposed MLM technique.

3.2. Enhanced Hybrid Segment Approximate Multiplier (EHSAM)

To improve the accuracy of the proposed HSAM, one of the segments is checked and if the magnitude of the segment is less than 15, then value 1 is added with the segment. If the condition is true 1 added segment is selected.

Otherwise the input segment is taken as one of the inputs of the proposed multiplier. When compared to the SSM8 × 8, SSM10 × 10 and the HSAM technique, it provides better accuracy, less energy consumption and high speed (**Figure 3**).

3.3. Working Example for Approximation Method

Two 16 bit operands are given namely

$$A = (0101\ 0101\ 1111\ 1111)_2 = (22015)_{10}$$

$$B = (0000\ 0000\ 1111\ 1111)_2 = (255)_{10}$$

Product C = $(5613825)_{10}$

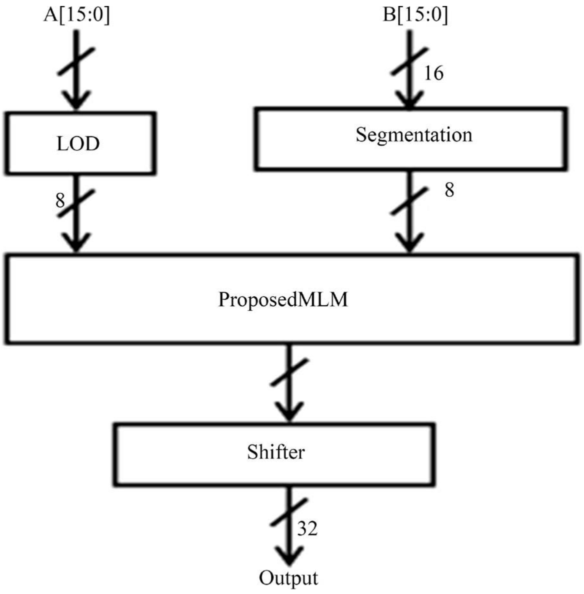


Figure 2. Proposed Hybrid segment approximate multiplier (HSAM).

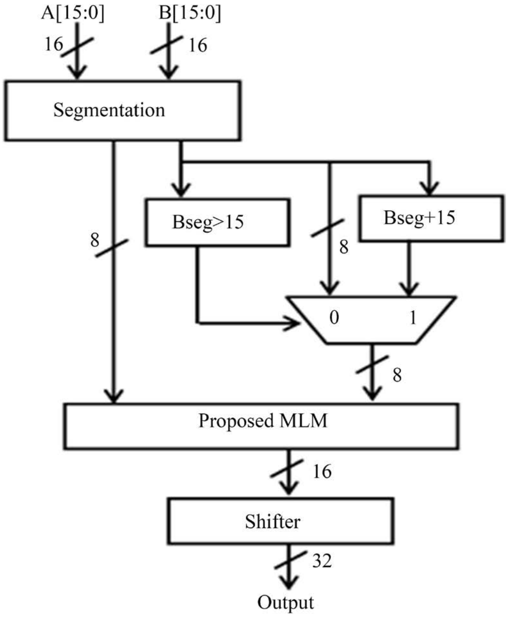


Figure 3. Enhanced Hybrid segment approximate multiplier (EHSAM).

The multiplication technique used in SSM method is given in steps as follows:

Step 1: Segmentation

A = 0101 0101 1111 1111 ➔01010101
B = 0000 0000 1111 1111 ➔11111111

Step 2: 8 × 8 multiplication

$$A_{seg} = 01010101_2 = 85_{10} \quad B_{seg} = 11111111_2 = 255_{10}$$

Let $Z = A_{seg} * B_{seg} = 01010101 * 11111111$

Z = 0101 0100 1010 1011_2 (16 bit)

Step 3: Expanding the product

Step 3: Expanding the product

The product is C_{ssm} = Z * (2^8)

C_{ssm} = 0000 0000 0101 0100 1010 1011 0000 0000_2 (32 bit) = $(5548800)_{10}$

The accuracy of an approximate multiplier using SSM technique is computed by

$$\text{Accuracy}(\%) = \left(\frac{C_{ssm}}{C}\right) * 100 = 98.84\% \tag{1}$$

4. Results and Discussion

The implementation of proposed approximate multipliers is carried out in Xilinx ISE tool and the Kintex-7 FPGA is used as the target platform. To evaluate the performance of the proposed approximate multipliers Cost and accuracy results are compared with the existing works.

The accuracy of an approximate multiplier is calculated using HSAM and EHSAM technique is computed using Equation (2) and Equation (3). **Table 1** shows that comparison results of existing and proposed approximate multiplier techniques. The proposed HSAM8x8 and EHSAM 8 × 8 approximate multipliers provide 98.85% and 99.999% accuracy respectively for various inputs.

$$\text{Accuracy}(\%) = \left(\text{HSAM } 8\times 8_{\text{output}} / \text{MLM } 16\times 16\right) * 100 \tag{2}$$

$$\text{Accuracy}(\%) = \left(\text{EHSAM } 8\times 8_{\text{output}} / \text{MLM } 16\times 1\right) * 100 \tag{3}$$

The cost analysis of an existing and proposed approximate multiplier is calculated in terms of area, delay and energy. Area of different approximate multipliers is computed in terms of number of slice LUTs and number of occupied slices is depicted in **Table 2**. To fairly compare the proposed technique, the existing SSM 8 × 8 is taken in to account due to approximation of 8 × 8. The proposed HSAM has additional area overhead by 7% but EHSAM reduces the area overhead by 19.8% compared to existing SSM 8 × 8 technique. The proposed HSAM

Table 1. Accuracy analysis of approximate multiplier using various SSM techniques.

Technique	Accuracy
SSM (8 × 8) [13]	98.84
SSM (10 × 10) [13]	98.90
HSAM (8 × 8) [pro]	99.85
EHSAM (8 × 8) [pro]	99.999

Table 2. Cost analysis.

Approximate multipliers	Number of Slice LUTs	Slices	Delay (ns)	Energy consumption (nJ)
Existing SSM 8 × 8 [13]	116	46	22.624	0.452
Existing SSM10 × 10 [13]	170	57	19.71	0.545
Proposed HSAM [pro]	125	42	16.104	0.305
Proposed EHSAM [pro]	93	32	12.094	0.217

consumes less energy with small increase of area overhead. The proposed EHSAM consumes less energy without any area overhead. The proposed HSAM and EHSAM is improved the speed by 40% and 85% compared to the existing SSM 8 × 8 technique. The proposed HSAM and EHSAM is reduces the energy consumption by 70% and 108% compared to the existing SSM 8 × 8 technique.

5. Case Study on Image Processing Applications

The proposed approximate multiplier is suitable for image compression applications. It is simulated using Matlab image processing tool box.

5.1. Implementation Flow

The implementation flow of image processing applications using high accurate static segment based approximate multiplier is shown in **Figure 4**. Input images are converted into text file using Matlab programming.

The text file is given as the input to VLSI implementation. This is done by Xilinx ISE tool, which provides the text file as output. Finally the output text file is converted into the image using Matlab programming.

5.2. Simulation Output

The various approximate multipliers are designed to support two 16 bit inputs and 32 bit output. These multipliers are implemented on Spartan 6 with a speed grade-3 using Verilog HDL. The simulation output of novel approximate multiplier is shown in **Figure 5**. Here a, b are two 16 bit inputs and P is 32 bit product.

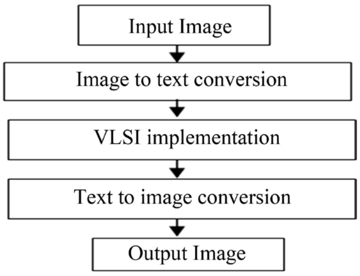
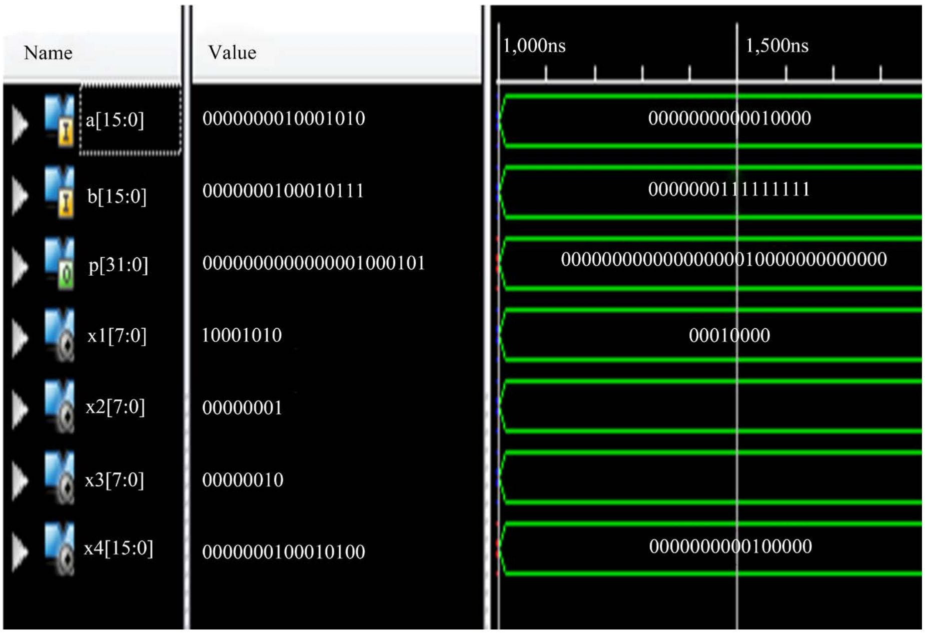


Figure 4. Implementation flow.



```
Name       Value                     1,000ns          1,500ns
a[15:0]    0000000010001010          0000000000010000
b[15:0]    0000000100010111          0000000111111111
p[31:0]    0000000000000001000101    00000000000000000010000000000000
x1[7:0]    10001010                  00010000
x2[7:0]    00000001
x3[7:0]    00000010
x4[15:0]   0000000100010100          0000000000100000
```

Figure 5. Simulation output of approximate multiplier.

The simulation output of image compression using Matlab tool is shown in **Figure 6**. Here the column 1 denotes the original image and column 2 denotes the compressed images, which are obtained by proposed approximate multipliers.

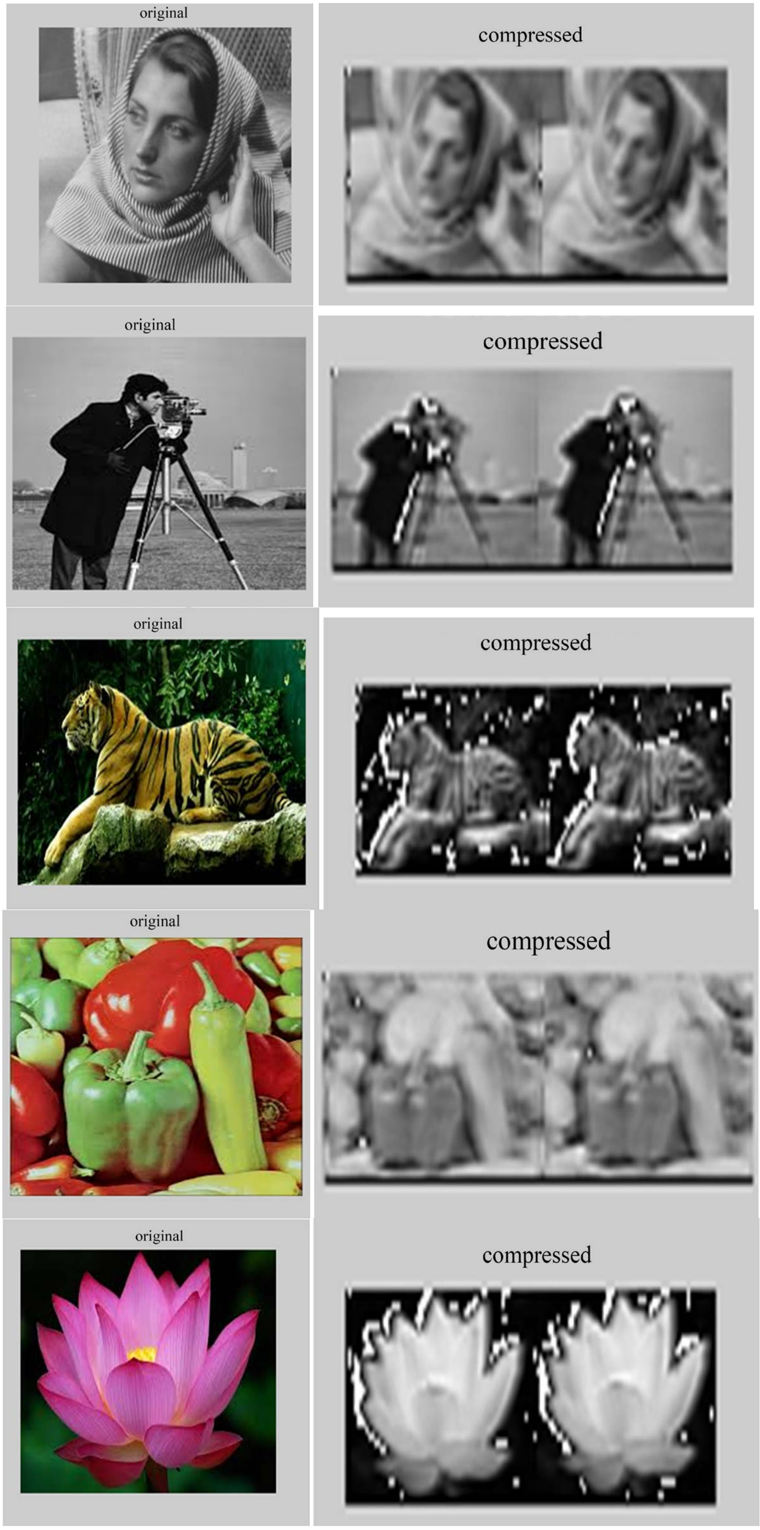


Figure 6. Simulation output for image compression.

6. Conclusion

In this paper, a hybrid segment based approximate multiplier is designed for error critical applications. The proposed hybrid approximate multiplier takes m consecutive bits (*i.e.*, an m-bit segment) of an n-bit operand and selected two m-bit segments are applied to MLM method for obtaining cost efficiency. The proposed HSAM has additional area overhead by 7% but EHSAM reduces the area overhead by 19.8% compared to existing SSM 8 × 8 technique. The proposed HSAM and EHSAM is improved the speed by 40% and 85% compared to the existing SSM 8 × 8 technique. The proposed HSAM and EHSAM is reduces the energy consumption by 70% and 108% respectively compared to the existing SSM 8 × 8 technique.

References

[1] Shao, B.T. and Li, P. (2015) Array-Based Approximate Arithmetic Computing: A General Model and Applications to Multiplier and Squarer Design. *IEEE Transactions on Circuits and Systems I*: *Regular Papers*, **62**, 1081-1090.

[2] Hegde, R. and Shanbhag, N.R. (1999) Energy-Efficient Signal Processing via Algorithmic Noise-Tolerance. Coordinated Science Laboratory University of Illinois, Urbana, IL.

[3] Babic, Z., Avramovic, A. and Bulic, P. (2011) An Iterative Logarithmic Multiplier. *Microprocessors and Microsystems*, **35**, 23-33.

[4] Chang, C.-H. and Satzoda, R.K. (2010) A Low Error and High Performance Multiplexer-Based Truncated Multiplier. *IEEE Transactions on Very Large Scale Integration* (*VLSI*) *Systems*, **18**, 1767-1771.

[5] Jha, R. and Paul, R.K. (2015) Swift and Approximate 8 bit Multiplier for Embedded DSP Applications: A Review. *International Journal of Industrial Electronics and Electrical Engineering*, **3**, 40-52.

[6] Chippa, V.K., Mohapatra, D., Raghunathan, A., Roy, K. and Chakradhar, S.T. (2010) Scalable Effort Hardware Design: Exploiting Algorithmic Resilience for Energy Efficiency. *Proceedings of* 47*th IEEE*/*ACM Design Automation Conference* (*DAC*), Anaheim, CA, 13-18 June 2010, 555-560. http://dx.doi.org/10.1145/1837274.1837411

[7] Mohapatra, D., Karakonstantis, G. and Roy, K. (2009) Significance Driven Computation: A Voltage-Scalable, Variation-Aware, Quality-Tuning Motion Estimator. *Proceedings of* 14*th IEEE*/*ACM International Symposium on Low Power Electronics and Design* (*ISLPED*), San Fancisco, 19-21 August 2009, 195-200.
http://dx.doi.org/10.1145/1594233.1594282

[8] Babic, Z., Avramovic, A. and Bulic, P. (2011) An Iterative Logarithmic Multiplier. *Microprocessors and Microsystems*, **35**, 23-33. http://dx.doi.org/10.1016/j.micpro.2010.07.001

[9] Kulkarni, P., Gupta, P. and Ercegovac, M. (2011) Trading Accuracy for Power with an Underdesigned Multiplier Architecture. *Proceedings of* 24*th Internatioal Conference on VLSI Design* (*VLSID*), 2-7 January 2011, 346-351.
http://dx.doi.org/10.1109/vlsid.2011.51

[10] Sunil, M. and Harish, B. (2014) High Performance Multiplexer Based Pseudo-Carry Compensation Truncated Muliplier. ISRASE eXplore Digital Library.

[11] Shetty, D.R. and Patil, S. (2013) Improving Accuracy in Mitchell's Logarithmic Multiplication Using Iterative Multiplier for Image Processing Application. *International Journal of Soft Computing and Engineering* (*IJSCE*), **3**, 187-190.

[12] Anand, S. and Saxena, A.K. (2013) Design of Approximate Multiplier for Error-Tolerant Applications. *International Journal of Engineering Research and Applications*, **3**, 880-883.

[13] Liu, C., Han, J. and 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*), Dresden, 24-28 March 2014, 1-4.

[14] Maheshwari, N., Yang, Z.X., Han, J. and Lombardi, F. (2015) A Design Approach for Compressor Based Approximate Multipliers. 28*th International Conference on VLSI Design* (*VLSID*), Bangalore, 3-7 January 2015, 209-214.