Middle-East Journal of Scientific Research 24 (9): 2772-2777, 2016
ISSN 1990-9233
© IDOSI Publications, 2016
DOI: 10.5829/idosi.mejsr.2016.24.09.23852

Approximate Carry Select Adder Based Multiplierless Multiplication (AMLM)

[1]R. Jamuna, [2]N. Sathish Kumar and [3]B. Sivasankari

[1]Department of ECE, SNS College of Engineering, Coimbatore 641 107, Tamilnadu, India
[2]Department of ECE, Sri Ramakrishna Engineering College, Coimbatore -641 022, Tamilnadu, India
[3]Assistant Professor, Department of ECE, SNS College of Technology, Coimbatore 641 035, Tamilnadu, India

Abstract: Approximate Computations are widely creating an attention, to minimize the power, area, delay and energy for image processing and digital signal processing. This paper presents an approximate carry select adder based multiplierless multiplication (AMLM), that has lower energy consumption compared to the existing one. The result of proposed scheme on Xilinx ISE 13.2 Kintex-7 FPGA device shows that the power consumption is reduced by 4.23% and number of Slices & LUTs are minimized which implies reduction in area.

Key words: Approximate Computations · Carry Select Adder (CSA) · Multiplierless multiplication

INTRODUCTION

This chapter describes about the design of approximate multiplier using approximate carry select adder (ACSA). The performance of these approximate carry select adder and approximate multiplierless multiplier are compared in terms of area, delay, power and energy.

This paper comprises the following sections. The basic approximation techniques are explained detailed in section II. Section III focuses on the introduction to the proposed approximate carry select adder based multiplierless multiplication (AMLM). In section IV, the simulation results are shown and the performance implementation is viewed in section V. Finally, the conclusion is presented in section VI [1].

Basic Approximation Techniques: Energy efficiency of multiplication is a critical objective. Since many digital applications use fixed- point arithmetic, it exhibits tolerance for computational errors. In this project, a multiplier is proposed that can tradeoff computational accuracy with energy consumption [2].

“Segmenting the original operands with significant bits and performing the multiplication only for those segments is the main principle of the project”.

The proposed technique performs approximate multiplication exploiting significant segments of operands unlike the existing techniques such as aggressive voltage scaling; truncation of bit-width and use of inaccurate building blocks for approximate multiplication.

Two different types of approximation techniques are,

- Dynamic Segment Method (DSM)
- Static Segment Method (SSM)

Dynamic Segment Method (DSM): Dynamic Segment Method (DSM) is a technique used for 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 which increases the area overhead with high accuracy [2].

In order to motivate and describe the proposed multiplier, an m-bit segment is defined as m contiguous bits starting with the leading one in an n-bit positive operand. This method is called as dynamic segment method (DSM) in contrast to static segment method (SSM) that will be discussed later in this section. With two m-bit segments from two n-bit operands, a multiplication is performed using an m ×m multiplier. In this method, 99.4% accuracy is achieved for a 16 × 16 multiplication even with an 8 × 8 multiplier. This method can capture m-bit segments starting from the exact leading one bit position [3].

Corresponding Author: R. Jamuna, Department of ECE, SNS College of Engineering,
Coimbatore 641 107, Tamilnadu, India.

0101 0101 0010 0010 X -------Segment starting from leading one

0001 0101 0000 0000 -------Segment starting from leading one

Fig. 1: Example of DSM

Such a multiplication approach has little negative impact on computational accuracy because it can eliminate redundant bits (i.e., sign extension bits) while feeding the most useful m significant bits to the multiplier. Furthermore, an m×m multiplier consumes much less energy than an n×n multiplier, because the complexity (and thus energy consumption) of multipliers quadratically increases with n. For example, the 4 × 4 and 8 × 8 multipliers consume almost 20 × and 5 × less energy than a 16×16 multiplier per operation on average [3].

Static Segment Method (SSM): 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 any one in the MSB, it uses the MSB segment and excludes the LSB segment which reduces the accuracy.

In this brief, an approximate carry select adder is proposed for image processing and Digital Signal Processing applications. It uses X number of bits from Y bit operand in a successive way. Here X should satisfy the following condition [4].

$$X \geq Y/2 \text{ for Positive number} \qquad (1)$$

With the help of Leading One Detector (LOD), the leading one position of the operand is detected for the X bit segment. When compared to the truncation of LSB bits, this technique leads to high accuracy, due to its effective bit capturing. The X bit segment is taken from the starting of leading one bit from the Y bit operand. So it takes expensive LODs and shifters, navigate them to an X × X multiplier and expand 2X bits to 2Y bits. The proposed approach is different from the existing techniques by restricting the X bit segment's starting bit position to two or three, eliminating the LODs by OR gate and trading the shifters with multiplexers [4].

The proposed method for approximate multiplication is the Static Segment Method (SSM). Regardless of m and n, we have four possible combinations of taking two m-bit segments from two n-bit operands for a multiplication using the m-bit SSM. For a multiplication, we choose the m-bit segment that contains the leading one bit of each operand and apply the chosen segments from both operands to the m×m multiplier.

Two design architectures using two different numbers of bits for the segment is given below:

- SSM_8X8 – where the segment size (m) is 8 bit

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Fig. 2: SSM Segmentation when m=8 bit

- SSM_10X10 – where the segment size (m) is 10 bit

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Fig. 3: Segmenting in SSM where m=10

Approximate Carry Select Adder Based Mlm (AMLM): Approximate multiplierless multiplier is implented using approximate Carry Select

Adder. Approximate Carry Select Adder reduces the area, power consumption and energy.

Block Diagram of AMLM: The block diagram of AMLM is shown in Fig.4. Here the inputs are segmented into higher segment and lower segment. The higher segment of the input is given as input to the OR gate and the lower segment of the input is given as input to the multiplexer. The output of the bitwise OR gate is considered as select line. Based on this select line the data from the multiplexers are selected. If the OR gate's output is 1, the higher segment is selected, otherwise lower segment data is selected from the multiplexer. The multiplexed outputs are given as input to the accurate multiplier. Then this product is shifted into three terms. Finally the shifted values are selected based on the output of the OR gate [5].

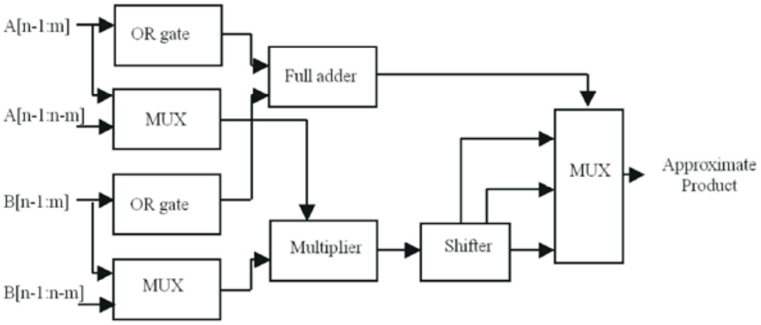


Fig. 4: Block Diagram of AMLM

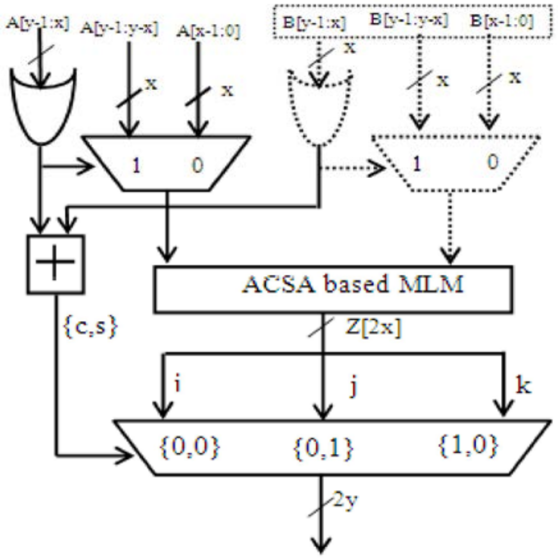


Fig. 5: Architecture of AMLM

Architecture of AMLM: The architecture of AMLM is shown in Fig. 5. Here x and y indicates the bit width of the inputs. The bit width of x is equal to half of the y. A and B are two inputs. A, B are input operands, Z is the 2x bit product, c and s denotes the carry and sum of the adder and i, j, k are concatenated outputs of the zeroes and Z.

Where

$$i = \{(2 * (y - x))\ b0,\ Z\} \tag{2}$$

$$j = \{((y - x))\ b0,\ Z\ (y - x)'\ b0\} \tag{3}$$

$$k = \{Z,\ (2 * (y - x))'\ b0\} \tag{4}$$

The major advantage of the proposed approximate multiplier is its scaling property. Because of this scalability for X and Y, the complexity, area and energy consumption is reduced.

The inputs are separated into two segments named as higher segment and lower segment. The higher segment of the two input are bitwise ORed. The input segments are selected based on the output of the OR gate. The multiplexed output is multiplied by multiplier. Then the outputs are shifted into three terms. Finally the approximate product is selected by using sum as a select line.

Architecture of Proposed Multiplier: The architecture of proposed multiplier is shown in Fig. 6. The shifter mostly performs the right shift operation. The shifted values are given as the inputs to the multiplexer. One of the inputs of the multiplexer is always zero. If the bit of the multiplier is zero, then zero is the output of the multiplexer, otherwise it selects the multiplicand as the output. Finally, these are added by approximate carry select adder to get the approximate output. When compared to the conventional multipliers like array multiplier, Wallace tree and dada multiplier, the proposed multiplier provides high speed and low energy consumption with less accuracy. 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 [5].

Approximate CSA: Approximate Carry Select Adder is an area, power and energy efficient adder which provides the approximate sum. This approximate adder is suitable for some applications such as image processing, signal processing applications.

The block diagram of ACSA is shown in Fig. 7 Here the inputs are segmented into higher segment and lower segment. The higher segments of the inputs are given as input to the OR gate and the lower segments of the inputs are given as input to the multiplexer. The output of the bitwise OR gate is considered as a select line. Based on this select line the data from the multiplexers are selected.

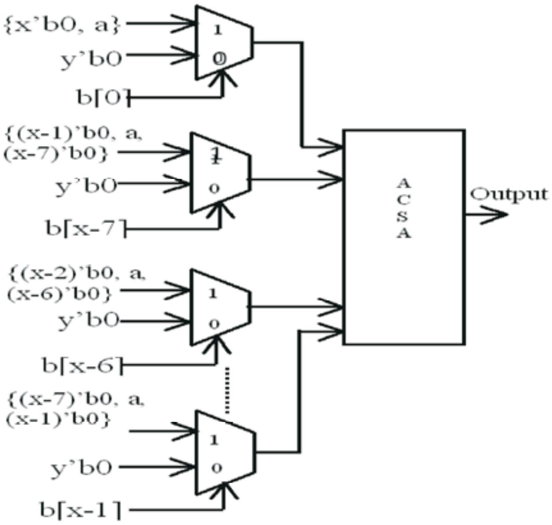


Fig. 6: Architecture of Proposed Multiplier

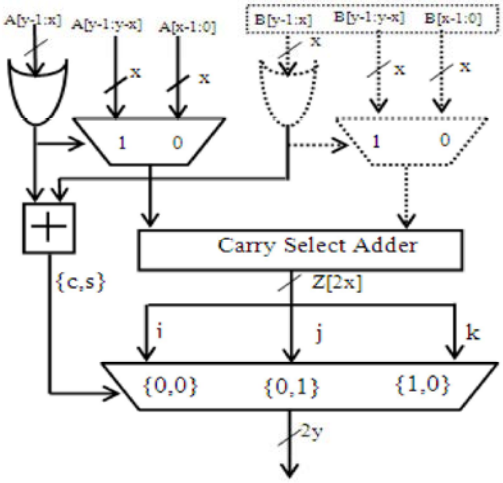


Fig. 7: Approximate Carry Select Adder

If the OR gate's output is 1, the higher segment is selected, otherwise lower segment data is selected from the multiplexer.

The multiplexed outputs are given as input to the approximate carry select adder. Then approximate sum is shifted into three terms. Finally the shifted values are selected based on the output of the OR gate.

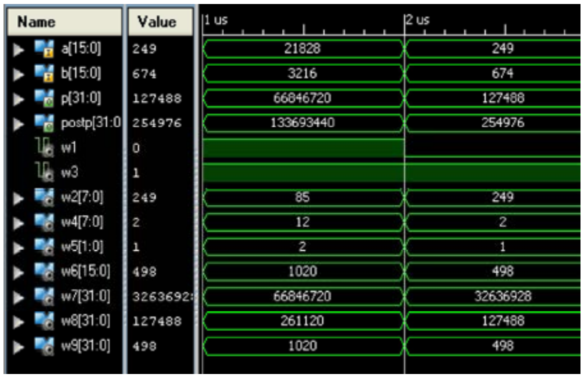
The architecture of approximate carry select adder is shown in Fig. 7. The major components of the approximate carry select adder are OR gate, multiplexer, full adder and carry select adder. Here x and y indicates the bit width of the inputs. The bit width of x is equal to half the bit width of y. A and B are two inputs. A, B are input operands, Z is the 2x bit product, c and s denotes the carry and sum of the adder and i, j, k are concatenated outputs of the zeroes and Z. The value of i,j,k are similar to equations 2,3 and 4 [6].

The total error can be computed by the following equation

$$T_e = (|R_t - R_e|)/R_t \quad (5)$$

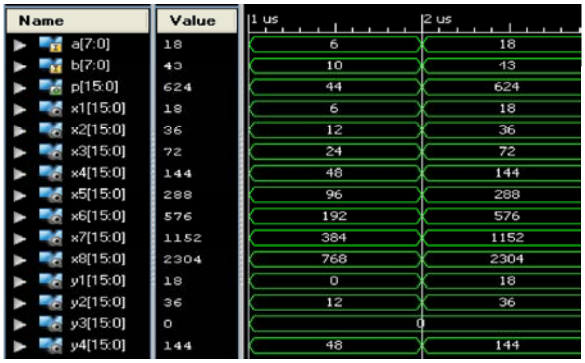
Here R_t indicates the true result and R_e indicates the error result.

Simulation Result: The design of approximate multiplierless multiplier using approximate carry select adder is synthesized on Kintex 7 FPGA device and simulated in Xilinx ISE 13.2. This architecture is coded using Verilog HDL.



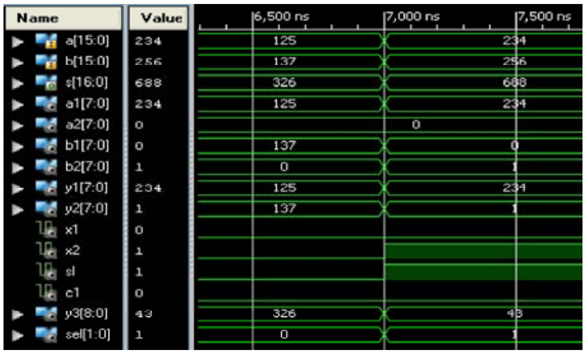
```
Name          Value       |1 us                    |2 us
a[15:0]       249          21828                    249
b[15:0]       674          3216                     674
p[31:0]       127488       66846720                 127488
postp[31:0    254976       133693440                254976
w1            0
w3            1
w2[7:0]       249          85                       249
w4[7:0]       2            12                       2
w5[1:0]       1            2                        1
w6[15:0]      498          1020                     498
w7[31:0]      3263692:     66846720                 32636928
w8[31:0]      127488       261120                   127488
w9[31:0]      498          1020                     498
```

Fig. 8: Simulation Result of SSM 8x8



```
Name          Value       |1 us                    |2 us
a[7:0]        18           6                        18
b[7:0]        43           10                       43
p[15:0]       624          44                       624
x1[15:0]      18           6                        18
x2[15:0]      36           12                       36
x3[15:0]      72           24                       72
x4[15:0]      144          48                       144
x5[15:0]      288          96                       288
x6[15:0]      576          192                      576
x7[15:0]      1152         384                      1152
x8[15:0]      2304         768                      2304
y1[15:0]      18           0                        18
y2[15:0]      36           12                       36
y3[15:0]      0                        0
y4[15:0]      144          48                       144
```

Fig. 9: Simulation Result of AMLM using ACSA



```
Name          Value       |6,500 ns        |7,000 ns        |7,500 ns
a[15:0]       234           125                              234
b[15:0]       256           137                              256
s[16:0]       688           326                              688
a1[7:0]       234           125                              234
a2[7:0]       0                              0
b1[7:0]       0             137                              0
b2[7:0]       1             0                                1
y1[7:0]       234           125                              234
y2[7:0]       1             137                              1
x1            0
x2            1
sl            1
c1            0
y3[8:0]       43            326                              43
sel[1:0]      1             0                                1
```

Fig. 10: Simulation Result of ACSA

Performance Analysis: The performance of these architectures is compared in terms of area, delay, power consumption and energy consumption. The performance can also be computed in terms of accuracy. The area is compared in terms of number of occupied slices and number of slice LUTs. Table I shows the analysis of area in terms of number of occupied slices and number of slice Look Up Tables (LUTs) [7].

Table I: Area analysis

Technique	Number of occupied slices	Number of Slice
LUTs		
MLM	95	318
AMLM	77	140

From the above table, it is clear that in terms of number of occupied slices, AMLM reduces the area by 18.94% and in terms of number of slice LUTs, AMLM reduces the area by 55.97%.

Table II: Delay and Power analysis

Technique	Delay (ns)	Power (ns)
MLM	4.479	0.118
AMLM	8.436	0.113

From the above table, it is observed that AMLM reduces the power consumption by 4.23%. Since AMLM reduces the area and power consumption, it increases the delay [8-14].

Table III: Accuracy analysis

Technique	Accuracy (%)
MLM	100
AMLM	99.98

From the above table it is clear that, compared to MLM, AMLM reduces the accuracy by 0.02% which is considerable for some image processing applications.

CONCLUSION

In this paper, approximate multiplierless multiplier using approximate carry select adder and proposed multiplier are discussed. This is suitable for image processing applications and digital signal processing applications. From the evaluated results, it is shown that the proposed design has good performance with better delay, area and power consumption.

REFERENCES

1. Hsu Chang-Ling, 2014. A Cloud Service Selection model based on user-specified quality of service level, ICAIT, ICDIPV, ITCSE, NC, pp: 43-54.
2. Liu Yongwen, Moez Esseghir and Leila Merghem Boulahia, 2016. Evaluation of Parameters Importance in Cloud Service Selection Using Rough Sets, Journal of Applied Mathematics, 7: 527-541.
3. Ruby Annette, J. and W. Aisha Banu, 2014. Cloud Broker for Reputation-Enhanced and QoS based IaaS Service Selection, Proc. of Int. Conf. on Advances in Communication, Network and Computing, CNC, © Elsevier.
4. Qu Lie, Yan Wang and Mehmet A. Orgun, Ling Liu and Athman Bouguettaya, 2014. Context-aware cloud service selection based on comparison and aggregation of user subjective assessment and objective performance assessment, In Web Services (ICWS), 2014 IEEE International Conference on, IEEE, pp: 81-88.
5. Filial Fatima Zohra and Belabbes Yagoubi, 2015. Global Trust: A Trust Model for Cloud Service Selection, International Journal of Computer Network and Information Security, 5: 41-50.
6. ABOUREZQ Manar and Abdellah IDRISSI, 2015. ntegration of Qos Aspects in the Cloud Serv Research and Selection System, (IJACSA) International Journal of Advanced Computer Science and Applications, 6(6): 111-122.
7. Fan, Wenjuan, Yang, Shanlin, Perros and Harry Pei, Jun, 2015. A Multi-dimensional trust-aware cloud service selection mechanism based on Evidential Reasoning Approach, International Journal of Automation and Computing, 12(2): 208-219.
8. Kapgate Deepak, 2014. Weighted Moving Average Forecast Model based Prediction Service Broker Algorithm for Cloud Computing, International Journal of Computer Science and Mobile Computing, 3(2): 71-79.
9. Zhang, M., R. Ranjan, A. Haller, D. Georgakopoulos and P. Strazdins, 2012. Investigating decision support techniques for automating Cloud service selection,Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on, Taipei, pp: 759-764.
10. Gui Zhipeng, Chaowei Yang, Jizhe Xia, Qunying Huang, Kai Liu, Zhenlong Li, Manzhu

11. Manikandan, R. and G. Kousalya, 2016. A framework for an Intelligent Broker Model of Cloud Service Selection, Asian Journal of Information Technology, 15(11): 1776-1784.
12. Goscinski Andrzej and Michael Brock, 2010. Toward dynamic and attribute based publication, discovery and selection for cloud computing, Future Generation Computer Systems, 26(7): 947-970.
13. Yu, Min Sun, Nanyin Zhou and Baoxuan Jin, 2014. A Service Brokering and Recommendation Mechanism for Better Selecting Cloud Services, PloS ONE, 9(8): 29.
14. Cloud Service Measurement Index Consortium (CSMIC), SMI framework. URL: http://beta-www.cloudcommons.com/ servicemeasurementindex.