IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

HYBRID APPROXIMATE ADDER FOR POWER, AREA, DELAY TRADE-OFF BASED ON BCSA AND ALA

[1]Ms. AKHILA R., [2]Ms. SINDHU T. V.
[1]Student, [2]Assistant Professor
[1]Department of Electronics and Communication Engineering
[1]IES College of Engineering
Thrissur-Kerala, India

Abstract: In application domains such as multimedia and image processing, digital signal processing, wireless communication, machine learning, and data mining which are inherently error-resilient, approximate computing can be used to overcome the thermal design power constraint. The technique may be used to attain more energy reduction and/or performance at the cost of some accuracy loss. Approximate adders can be broadly classified into two categories: block-based approximate adders (BAAs) and approximate least significant bit (LSB) adders (ALAs). In this paper, a hybrid approximate adder based on BCSA and ALA is proposed, for power, area and delay trade-off. In the proposed technology, both the BAA and ALA are used in a single 16 bit adder circuit. That is, the whole circuit is divided into two 8 bit parts and the MSB part is designed using Block based carry speculative Adders (BCSA), which improves the speed and the LSB part make use of approximate least significant bit Adders (ALAs) which results in area and power reduction. Compared to BAAs, ALAs are more hardware and power efficient and by using block based approximate adder, the overall delay can be reduced. Thus, power, area and delay trade off is achieved.

***Index Terms*: Approximate Computing, Hybrid Adder, Speculative Adder, Area Efficient, Low Power, Delay Efficient**

I. INTRODUCTION

Approximate computing is an emerging paradigm for energy-efficient and/or high performance design. It includes a plethora of computation techniques that return a possibly inaccurate result rather than a guaranteed accurate result, and that can be used for applications where an approximate result is sufficient for its purpose[1]. Approximate computing is based on the observation that in many scenarios, although performing exact computation requires large amount of resources, allowing bounded approximation can provide disproportionate gains in performance and energy, while still achieving acceptable result accuracy.

Approximate computing has been used in a variety of domains where the applications are error-tolerant, such as multimedia processing, machine learning, signal processing, scientific computing, etc. Therefore, approximate computing is mostly driven by applications that are related to human perception/cognition and have inherent error resilience. Many of these applications are based on statistical or probabilistic computation, such as different approximations can be made to better suit the desired objectives.

Basically approximate adders, multipliers, MAC units etc are used in signal processing (FIR Filters, IIR Filters, etc) and image processing applications (Image Sharpening, Image Smoothening, etc). In all these applications, the fundamental blocks are adders and multipliers. Normally a large number of adders and multipliers have to be used according to the circuit requirements. So, if we go for exact adders and multipliers in the above applications, power consumption, area and circuit complexity will be very high. In order to overcome this issue, we go for approximation. In approximate adders and multipliers, we divide the circuit design into two parts - Least Significant Bit (LSB) part and Most Significant Bit (MSB) part. The MSB part is kept unchanged or exact and the approximation is performed in the LSB part. This is because if we try to approximate the MSB part, the resulting error will be so high.

The main advantages of approximate circuits includes less area requirement, delay can be minimized and the overall power requirement will be less compared to exact circuits. The only problem with the approximate design is that since we are approximating the exact logic of the system, errors might occur at the final output. But, for all the above mentioned applications, errors occurring in the LSB part will be very minimum and can be negligible. These errors in LSB part will not affect the overall system function.

II. RELATED WORK

The need for approximate circuit designs mainly points to the simplification of arithmetic circuit units. There are different methodologies for approximating an adder and most of the approximate adders are used in error resilient applications. In a reconfigurable approximate adder explained in [3] has two operating modes and the structure is based on the conventional carry look ahead adder but the need for an external correction unit for the exact add operation is eliminated in the design.

Application specific integrated circuit implementations of error resilient applications like image and video compression is discussed in [4]. The most computationally intensive blocks in the implementation is made with approximate circuit design for improving the power consumption with little loss in the output quality.

The energy - quality trade off can be easily obtained by changing the operand bit width as mentioned in [5]. Dynamic, run time adaptation of effective bit length can be an effective tool for approximating a design.

The accuracy of the results can be made configurable as in [6] where the required accuracy changes during the runtime, for maximizing the benefits of approximate design. An error detection and correction unit can be used for minimizing the overall error, which can be implemented using several AND gates and the error can be corrected by adding a 1 to the approximate or inaccurate output.

The carry speculative adder discussed in [7] divides an n-bit adder into several smaller blocks which can operate independently. Separate carry predictor circuits are used for each block and those bits near the MSB are only used by the carry predictor circuit. This results in the reduction of critical path.

The error tolerant adders discussed in [2] and [8] can be seen as a potential solution for the trade-off between speed and power. For attaining improvement in both speed and power performances, some of the overall output accuracy is sacrificed. The circuit divides the input operands to accurate and inaccurate parts, whose length need not be equal. No carry signal will be generated for the inaccurate part for eliminating the carry propagation path, which itself is the main cause for the delay in the conventional adder circuits.

III. CARRY SPECULATIVE ADDERS

Certain approximation targets timing and certain approximation targets power and area according to the circuit requirement. Block based adder divides the whole circuit into small blocks. For example, consider a 16 bit adder. If we are designing it as a block based adder, either we can design it as four 4 bit adders (4 blocks) or two 8 bit adders (2 blocks) as shown in fig 1[1].

When we design the whole circuit as block based adder, each block will produce a carry output. That is in case of a 8 bit adder there will be 2 blocks of 4 bits each. In the first block, a_0,a_1,a_2,a_3 and b_0,b_1,b_2,b_3 are added and a carry out is produced say c_1. This c_1 is given to the second block as carry in along with bits a_4,a_5,a_6,a_7 and b_4,b_5,b_6,b_7. The second block also produces a carryout which will be the final carry out. If the designed circuit is a normal block based adder, the second block will be active only once the first block finishes the addition and produces the carry out. This may case delay in the overall operation especially when the adders is of higher bit size.

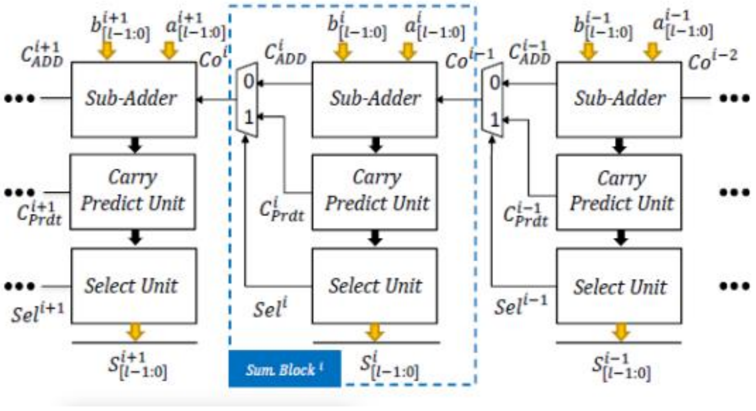


Fig 1. General architecture of an approximate adder equipped with carry prediction unit.

In case of speculative adders, the circuit will not wait for the proper carry from the first block for activating the second block. Instead, with some logic we will speculate the carry from the first stage even before the addition takes place, and the speculated carry is given to the second block and is thus activated before the first stage is completed. This reduces the overall delay of the circuit.

IV. PROPOSED TECHNOLOGY

A hybrid adder is designed by combining BCSA and ALA, which can improve the overall speed of the circuit along with reducing power and area requirement.

Figure 2 shows the basic block diagram of the proposed hybrid adder using BCSA and ALA. Approximate adders can be broadly classified into two categories: block based approximate adders (BAAs) and approximate least significant bit (LSB) adders (ALAs). In ALA, instead of diving into blocks, they will divide the whole adder into two – MSB and LSB. The bits in MSB and LSB are not fixed and are variable according to the circuit requirement. For example, consider a 16 bit adder.

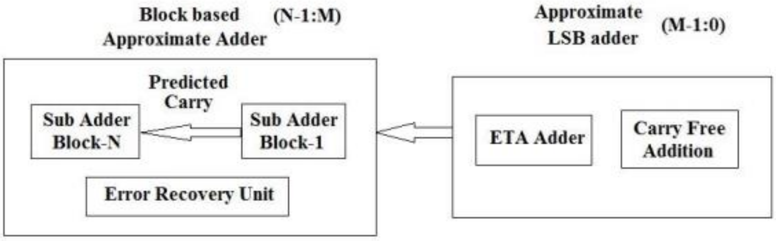


Fig 2. Block Diagram of Proposed Hybrid Adder

It is not necessary to divide the MSB and LSB as 8 bits each in ALA. Instead, we can divide the MSB as, say 4 bits and the LSB as 12 bits or we can divide the MSB as 10 bits and the LSB as remaining 6 bits. That is, in general the number k is variable, where k is the approximating bits.

If the error is very high, k can be varied so as more number of bits is kept exact (MSB bits) and least number of bits are being approximated (LSB bits). Compared to BAAs, ALAs are more hardware and power efficient. When we go for block based approximate adders, the overall delay can be reduced as explained in the previous slide. When ALAs are used, the circuit will be more area and power efficient. So, in the proposed technology, both the BAA and ALA are used in a single 16 bit adder circuit. That is, the whole circuit is divided into two 8 bit parts and the MSB part is designed using Block based carry speculative Adders (BCSA), which improves the speed and the LSB part make use of approximate least significant bit Adders (ALAs) which results in area and power reduction.

4.1 Block Carry Speculative Adders

In this structure, the adder is partitioned into some non-overlapped parallel blocks, which in the worst-case, the carry output of a block is dependent on the carry output of the previous block. To reduce the critical path more, an approach to predict the carry output of a block based on its signals as well as of the next block has been formulated. The structure has a low hardware complexity leading a low delay (on average, about one block) and a rather high quality.

By adding an error recovery unit as shown in fig 3[1], the overall error produced due to the approximation can be minimized to a large extend.

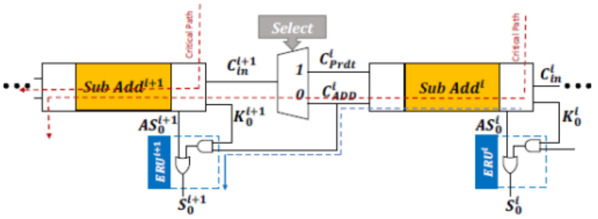


Fig 3. Structure of BCSA with error recovery unit

4.2 Error Tolerant Adder

ETA attains great improvement in both the power consumption and speed performance by sacrificing some of the accuracy of the overall output[2].

The main cause of delay in the conventional adder circuits is caused by the carry propagation chain along the critical path of the adder from LSB to MSB. Glitches in this carry propagation results in significant increase in power consumption. So, if the carry propagation can be eliminated or controlled in the critical path of the adder, it may result in greater improvement in terms of speed and power consumption.

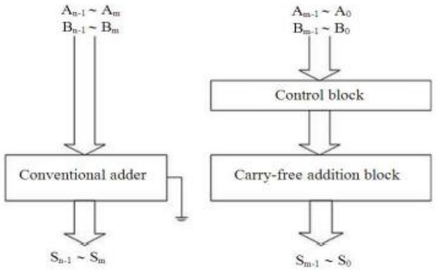


Fig 4. Architecture of proposed ETA

In ETA, the input operands are first split into two parts - the accurate part and the inaccurate part - and the length of each part need not to be equal. In the accurate part, the addition is performed using the normal addition method for preserving the correctness. But in the inaccurate part, a special addition mechanism which curtails the generation of carry signal will be applied. For minimizing the errors occurring due to the elimination of carry propagation path, a special strategy can be used as follows:

a. Check every bit position from left to right (MSB to LSB);

b. If both input bits are “0” or different, normal one-bit addition is performed and the operation proceeds to next bit position;

c. If both input bits are “1,”the checking process stopped and from this bit onward, all sum bits to the right are set to “1.” Table 1 and 2 shows the result obtained for RCA.

V. EXPERIMENTAL RESULTS

ModelSim software is used for coding and simulating the existing and proposed architectures. ISE Design suite can also be used but the ModelSim software is more user friendly and it has an inbuilt simulation environment, which most of the ISE design suite older versions failed to provide. The code has been written in Verilog Description language and simulated successfully for obtaining the output.

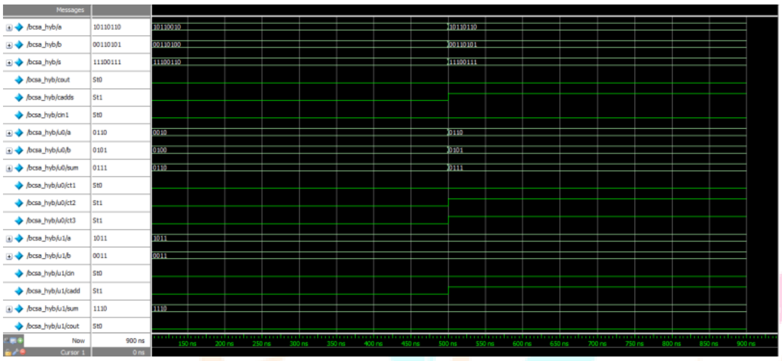


Fig 5. Simulated output of 8 bit hybrid adder

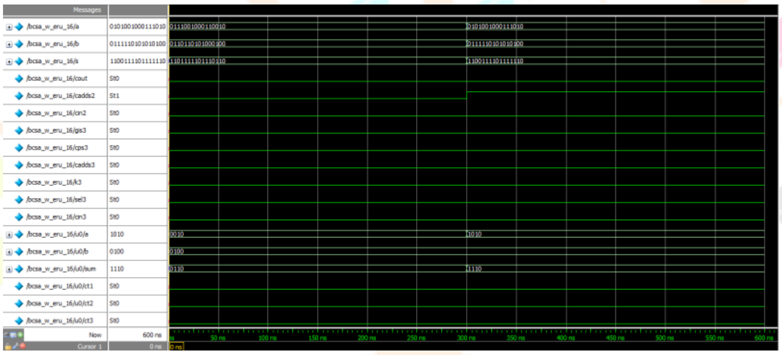

Messages
/bcsa_w_eru_16/a 0101001000111010
/bcsa_w_eru_16/b 0111110101010100
/bcsa_w_eru_16/s 1100111101111110
/bcsa_w_eru_16/cout St0
/bcsa_w_eru_16/cadds2 St1
/bcsa_w_eru_16/cin2 St0
/bcsa_w_eru_16/gis3 St0
/bcsa_w_eru_16/cps3 St0
/bcsa_w_eru_16/cadds3 St0
/bcsa_w_eru_16/k3 St0
/bcsa_w_eru_16/sel3 St0
/bcsa_w_eru_16/cin3 St0
/bcsa_w_eru_16/u0/a 1010
/bcsa_w_eru_16/u0/b 0100
/bcsa_w_eru_16/u0/sum 1110
/bcsa_w_eru_16/u0/ct1 St0
/bcsa_w_eru_16/u0/ct2 St0
/bcsa_w_eru_16/u0/ct3 St0
Now 600 ns
Cursor 1 0 ns

Fig 6. Simulated output of 16 bit hybrid adder

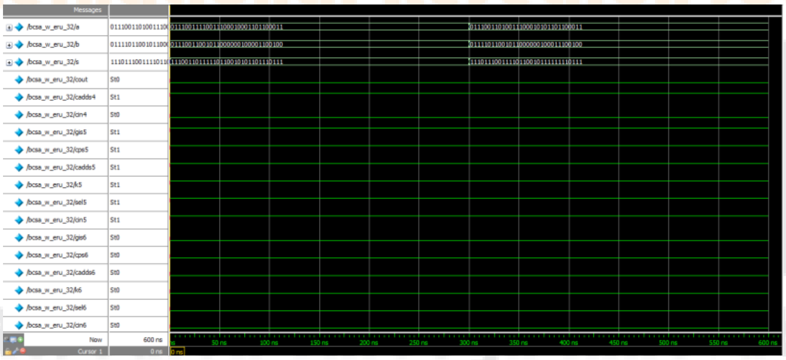
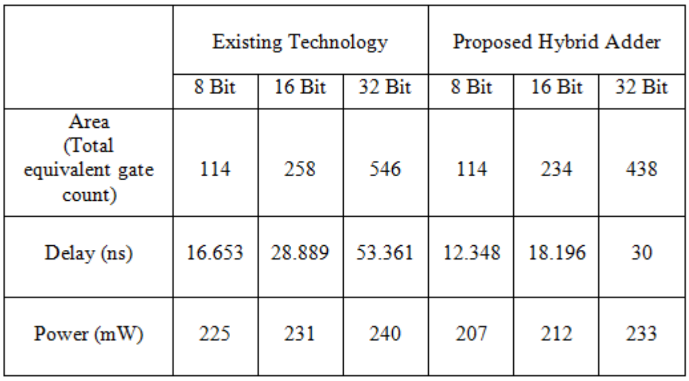


Fig 7. Simulated output of 32 bit hybrid adder

VI. Performance comparison

Xilinx ISE 8.1i was used to obtain the power, delay and area characteristics of the adders. Table I compared the area, delay and power requirements in the existing block carry speculative adder with error recovery unit and the proposed hybrid adder.

TABLE 1. Comparison of area, delay and power in existing technology and proposed hybrid adder



	Existing Technology			Proposed Hybrid Adder		
	8 Bit	16 Bit	32 Bit	8 Bit	16 Bit	32 Bit
Area (Total equivalent gate count)	114	258	546	114	234	438
Delay (ns)	16.653	28.889	53.361	12.348	18.196	30
Power (mW)	225	231	240	207	212	233

VII. Conclusion

A hybrid approximate adder based on BCSA and ALA has been designed and simulated successfully using ModelSim software, for power, area and delay trade-off. In the proposed technology, both the BAA and ALA are used in 8 bit, 16 bit and 32 bit adder circuits. That is, the whole circuit is divided into two MSB part and LSB part and the MSB part is designed using Block based carry speculative Adders (BCSA), which improves the speed and the LSB part make use of approximate least significant bit Adders (ALAs) which results in area and power reduction. Compared to BAAs, ALAs are more hardware and power efficient and by using block based approximate adder, the overall delay is thus reduced.

References

[1] Farhad Ebrahimi-Azandaryani, Omid Akbari1, Mehdi Kamal, Ali Afzali-Kusha, Massoud Pedram, “Block-based Carry Speculative Approximate Adder for Energy-Efficient Applications,” 2018 IEEE.

[2] Jayanthi, A.N. and C.S. Ravichandran, “Design of an Error Tolerant Adder,” American Journal of Applied Sciences 9 (6): 818-824, 2012 ISSN 1546-9239, 2012 Science Publications

[3] Omid Akbari, Mehdi Kamal, Ali Afzali-Kusha, and Massoud Pedram, “RAP-CLA: A Reconfigurable Approximate Carry Look-Ahead Adder,” IEEE Transactions on Circuits And Systems—Ii: Express Briefs, Vol. 65, No. 8, August 2018

[4] Vaibhav Gupta, Debabrata Mohapatra, Anand Raghunathan, Kaushik Roy, “Low-Power Digital Signal Processing Using Approximate Adders”, IEEE Transactions On Computer□Aided Design Of Integrated Circuits And Systems, Vol. 32, No. 1, January

[5] Jie Han, Michael Orshansky, "Approximate Computing: An Emerging Paradigm For Energy-Efficient Design", 18th IEEE European Test Symposium (ETS) 2013.

[6] Andrew B. Kahng and Seokhyeong Kang, "Accuracy-Configurable Adder for Approximate Arithmetic Designs", 2012

[7] Ing-Chao Lin, Yi-Ming Yang, and Cheng-Chian Lin, "High Performance Low-Power Carry Speculative Addition With Variable Latency", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2014

[8] Ning Zhu, Wang Ling Goh, Weija Zhang, Kiat Seng Yeo, and Zhi Hui Kong, "Design of Low-Power High Speed Truncation-Error-Tolerant Adder and Its Application in Digital Signal Processing", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 18, No. 8, August 2010.