Variability-Aware Pruning of Approximate CMOS Logic

Adnan Aquib Naseer
Department of EECS
University of Central Florida
Orlando, Florida

***Abstract*—Approximate computing has in recent years emerged as an important approach for energy saving mechanisms. Approximate computing involves reducing the number of transistors in a circuit to reduce delay and increase power savings. The power savings are at the cost of reduced accuracy. In most Signal Processing cases it is not necessary for the digital circuit to produce a precise output, the field of approximate computing takes advantage of the reduced accuracy requirements to improve speed and power consumption. This paper reviews methods used to approximate circuits and their performance in real-life applications. We also look at the effects of Process Variation on Approximate circuits where we analyze the effect of Variability on the signal delay of the critical path in approximate circuits.**

Keywords—inexact arithmetic units, approximate computing, adder, variable accuracy, low power, mirror adder, adders, error distance, power consumption, process variation ,delay, power reduction

I. Introduction

Power consumption has become a major factor in current computing applications. The need for higher energy efficiency in recent times has pushed the industry and academia to find various other avenues to decrease power consumption. Some of the common methods of reducing energy consumptions include Voltage Scaling, Probabilistic CMOS (PCMOS), Algorithmic Truncation and Approximation [6]. These methods are reviewed briefly in this paper. Various Mechanisms are used to reduce errors due to Voltage Overscaling, one of the mechanisms of particular interest is shown in [1]. In [1] Slack redistribution is used to gracefully degrade the circuit. Use of PCMOS is another way of reducing the energy cost without compromising on accuracy by a large margin as shown in [2]. [2] States that all MOS devices are inherently probabilistic rather than deterministic, it takes advantage of this probability in cases where noise is present to yield accurate results. [2] Implements PCMOS in System on Chips (SoC) to form Probabilistic System on Chip (PSoC). The third method i.e. algorithmic truncation is a software level based solution, this solution is usually compared with hardware level approximation as shown in [3]. Algorithmic truncation usually refers to methods of removing a fixed number of LSBs from the output or input. Another method of algorithmic adaptation would be to skip certain steps in the algorithm in return for reduced accuracy and power savings. Hardware approximation refers to reduction in the size of critical path by reducing the number of transistors in the path.

The primary purpose of this paper is to give a brief overview of progress made in Approximate Units in the past few years and discuss their implementation and effects on real world applications including the effect of variability on smaller truncated circuits.

II. PROCESS VARIATION AND ITS EFFECTS

Process Variation in today's technology both in interconnect side of computer systems as well as chip manufacturing process. On 45nm and above chips process variation wasn't a major problem due to the fact that the percentage error was relatively small. But when technology was scaled down to 22nm, the percentage error increased. For example if at 45nm the variation caused due to process variation is ±1nm then the percentage error in this case is 2% but when the variation is the same 1nm for a 22nm process then the percentage error is 4.5%. This percentage error is more than 2 times the percentage error for 45nm even though the absolute value is still 1nm.

The parts of the transistor which are affected by Process Variation include the gate oxide, Random Dopant Fluctuations and Device Geometry. Gate Oxides can be as thin as 5nm, hence even 0.5nm change in thickness could cause major changes in the threshold voltages which could again effect the delay of the circuit, changes in dopant concentrations can cause lowering or increasing of the threshold voltage and Device geometry is affected by the wavelength of the LASERs used in the manufacturing process.

Fortunately even though process variations are random, they can be simulated to a great degree of accuracy using modern random process techniques. In this paper we find a range of values for threshold voltages (V_{th}) and widths (W) for the Full Adders, the values are sampled using the Monte-Carlo Simulation technique. All the random values in the range are obtained by using the Gaussian Curve as shown in Fig.1. The values obtained from HSPICE aren't truly random values but rather pseudo-random numbers. The curve becomes flatter as technology is scaled down as shown by the red curve in Fig.1. It is predicted that eventually the Curve becomes flat enough to be called a Uniform Curve.

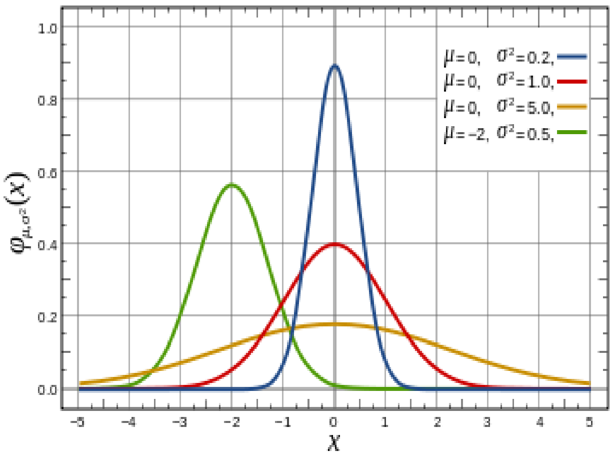


Fig.1 Gaussian Curve

III. TYPES OF APPROXIMATION

A. Voltage Overscaling

Orthodox designs take into account various redundancies to select a high V_{DD} to accommodate any errors occurring due to process variation, timing errors, aging etc. But when V_{DD} is scaled below the critical voltage, the number of errors drastically increase and hence in turn drastically degrade the output quality One way to improve signal quality in such cases is to introduce error correcting mechanisms. [4] Introduces an approach under *algorithmic noise tolerance*. This type of circuit contains two blocks namely the main computing circuit and an error correcting circuit. The main computing circuit is run on a low V_{DD} whereas the simpler error correcting circuit runs on a higher voltage. As the error correcting circuit consists of lesser number of transistors than the main computing circuit the energy savings are notable.

B. Probabilistic CMOS circuits

As CMOS technology scales down to nanometer region, hurdles due to noise poses several challenges [2]. Probability based CMOS technology looks upon noise as a resource which can be harnessed [5]. Probabilistic CMOS have been known to yield better *energy performance product* (EPP). EPP is defined as the product of energy consumed and running time and denoted by Γ. In [2] PCMOS is defined as the CMOS which is affected by thermal (ambient) noise.

The computation in such circuits use Probabilistic Algorithms to reduce the effect of noise in the output. In addition to the above qualities, PCMOS can also generate high quality random bits. These random bits are important, as they are used extensively in probabilistic algorithms. The circuits implementing PCMOS typically contain a deterministic host processor, which is used to compute most of the control intensive components of an application and a probabilistic co-processor, which is used as an accelerator.

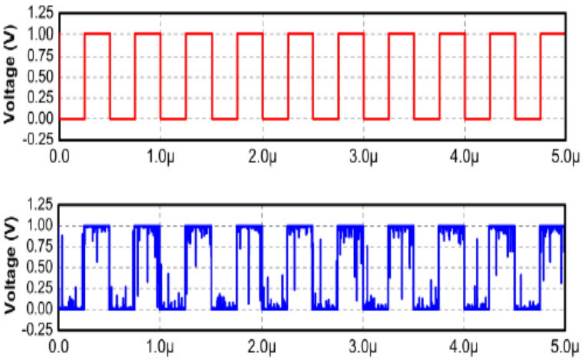


Fig 2. (a) Output Waveform of a conventional CMOS and (b) Output Waveform of Probabilistic CMOS [6]

C. Hardware Approximation

The process of removing transistors from the critical path or anywhere else to reduce the path length resulting in reduced accuracy of the circuit can be termed as Hardware Approximation. In this paper we review four papers which provide in-depth description of various implementations and their performance when they are substituted for their accurate counterparts in computation units as shown in papers [3] and [7].

Hardware Approximation can be segregated into various methods depending on the technique used to gain imprecision in the output. Some of the techniques are truncation, voltage overscaling etc. The main technique focused on in this paper is truncation, we measure the power consumption of various adders developed in [3]. Then we attempt to simulate the effects of process variation and determine the delay of the critical path of the circuit to put into perspective the position of Approximate Circuits in terms of performance.

IV. ERROR MEASUREMENT TECHNIQUES

A number of methods have been proposed to quantify the errors occurring in approximate circuits, some methods are similar to Hamming Distance. In this paper we review three types of error measurement techniques as mentioned in [8].

A. Error Distance

In [8] a new measurement technique is developed to measure precision in approximate circuit. This method is termed as Error Distance. Error Distance is the arithmetic distance between two binary values i.e. the arithmetic distance between the predicted/required value and the obtained/output value. For example, let us consider a Full Adder, if the exact output of the Sum is “100101”, but the circuit gives an output of “100100”, then the *error distance* between the two values is said to be 1. But if the output was “101111 then the error distance would be 10. It can be seen that the value of error distance (ED) increases or decreases depending on the position of the incorrect bit. [8] defines ED with the formula

$$ED(\mathrm{a,b}) = |\mathrm{a}-\mathrm{b}| = \left|\sum_i \mathrm{a}[i]*2^{\mathrm{i}} - \sum_j \mathrm{b[j]}*2^{\mathrm{j}}\right|$$

For non-deterministic application, the output is probabilistic and usually follows a set distribution for a given input a_i. In this case ED is defined as the weighted average of all possible outputs to the nominal output. Assume that for a given input, the output has a nominal value b, but it can take any value given in a set of vectors *bj* (1<j< r); the ED of the output is then given by

$$d_i = \sum_j \mathrm{ED}(\mathrm{b_j}, \mathrm{b}) * \mathrm{p_j} \qquad (2)$$

Where,

p_j is the output probability of *bj* (1<j< r)

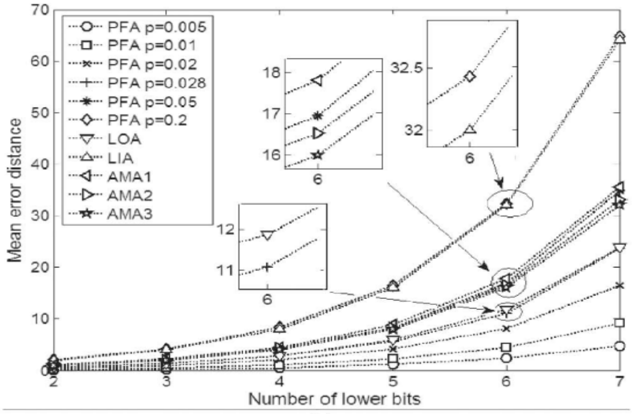


Fig. 4. Normalized error distance versus the number of lower bits

B. Mean Error Distance

Mean Error Distance is used in cases where the inputs are probabilistic and thus each input occurs only at a particular probability. The *mean error distance* (MED) of a circuit is defined as the mean value of the EDs of all possible outputs for each input [8]. Assume that the input is a set of vectors a_i and that each vector occurs with a probability of q_i [8]. Then, the MED of the circuit is given by

$$d_m = \sum_i \mathrm{d_i} * \mathrm{q_i} \qquad (3)$$

Where,

d_i is the ED of outputs for input a_i which can be computed by (1)

Mean Error Distance can be calculated by using equation (3). Assume a NOT gate, the probability of the input to the NOT gate being 1 is 0.7 and if the output of the NOT gate is 1 even if the input is 1. In this case the MED is given by ED multiplied by the probability (0.7) which is equal to 0.7 as the ED is 1 in this case. In case of multiple outputs. The ED is individually calculated and then multiplied with individual probabilities and after the values are calculated, all the values are summed with each other to obtain the final value. Fig 3 shows that the expected MED escalates with an increase in the number of lower bits, to reduce this effect we introduce the next concept. The curves in Fig 3 show the Mean Error Distances of various adders used in the measurement process. The figure clearly shows that accuracy wise, in the MED method, LIA and PFA at p=0.2 are similar and also, AMA1, AMA2, AMA3 and PFA at p=0.05 are similar. Here, p refers to the probability

C. Normalized Error Distance

As seen in fig 3, MED increases with increase in the number of lower bits. It is therefore unfair to use MED to compare between two adders of with different lower bits as the maximum value of error that can be effectively reached has also changed [8]. To overcome this limitation *normalized error distance* (NED) is used. NED is defined as

Fig. 3. Mean error distance versus the number of lower bits in a 32 bit adder

$$d_n = \frac{d_m}{D}$$

Where,

d_m is the MED and D is the maximum value of the error. The maximum value is usually 2^n for n lower bits [8].

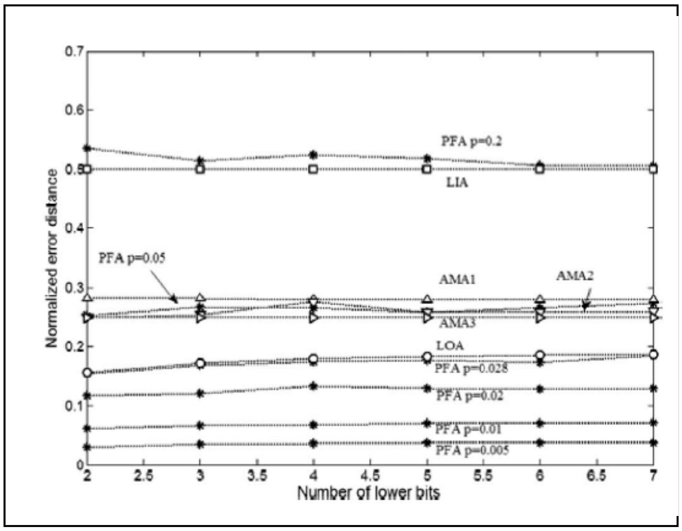


Fig. 4 shows the NEDs of various adder implementations in [8]. The NEDs show little change and only fluctuate in a small interval when different lower bits are compared.

V. Approximate circuits and their implementation

In this section we discuss the various approximations used in [3] and [9]. In [9] a standard 10 transistor adder is used to create three different types of Approximations, in this paper the correctness of the output is measured using ED. In [3] a mirror adder is used and 5 new approximations are obtained, [3] implements this in DSP blocks and measures its functionality are named as XA, AXA1, AXA2 and AXA3, where XA is the accurate adder and AXA1, AXA2 and AXA3 are the approximations obtained from XA. The adders of paper [3] are named as MA, AMA1, AMA2 and AMA3, where MA is the accurate adder from which other approximations are obtained

A. *Accurate XOR/XNOR based Adders (XA)* [9]

The proposed approximate XOR/XNOR-based adder 1 (i.e., AXA1) is based on the 10-transistor full adder in [11], while AXA2 and AXA3 are based on the accurate design in [12]. As shown in Fig. 5, the adder in [12] is based on four-transistor (4T) XNOR gates; the total number of transistors in this adder is 10. *X*, *Y* and C_{in} are inputs; *I* is an internal signal.

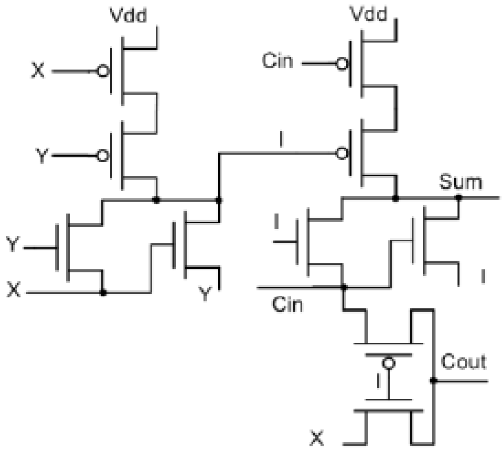


Fig 5. Accurate Full Adder with 10 transistors [12]

B. *Approximate XOR based Adder 1 (AXA1)* [9]

Fig. 5 shows the first approximate adder. In this design, the XOR operation is achieved by an inverter and two pass transistors connected to *X* and *Y* respectively. Both *Sum* and C_{out} are accurate for 4 out of the total 8 input combinations. The total error distance achieved with this design is 4. The transistor count for this design is 8. The functions of *Sum* and C_{out} are given by:

$$Sum = C_{in},$$
$$C_{out} = \overline{(X \oplus Y)C_{in} + \bar{X}\bar{Y}}. \qquad (5)$$

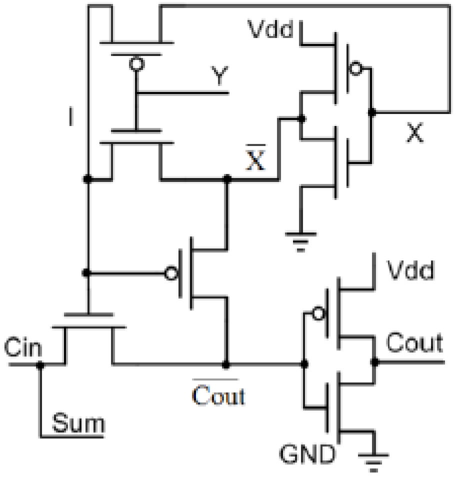


Fig. 6. Approximate XOR-based Adder 1 (AXA1) [9]

C. *Approximate XNOR Adder 2 (AXA2)* [9]

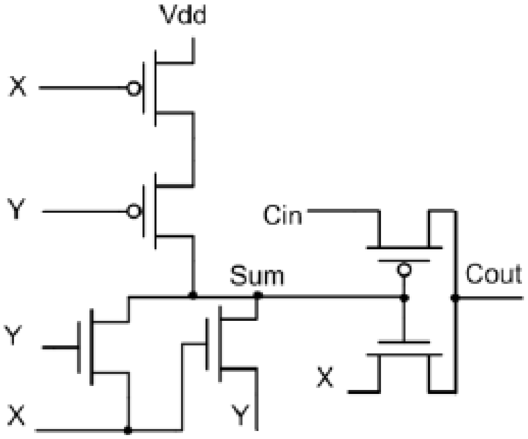


Fig. 7. Approximate XNOR Adder 2 [9]

The design in Fig. 7 implements an approximate adder with 6 transistors; it consists of a 4-transistor XNOR gate and a pass transistor block. *Sum* is accurate for 4 out of the 8 input combinations, while C_{out} is accurate for all input combinations. The total error distance for this design is also 4. The functions of *Sum* and C_{out} are given by:

$$Sum = \overline{(X \oplus Y)},$$
$$C_{out} = (X \oplus Y)C_{in} + XY. \qquad (6)$$

D. *Approximate XNOR Adder 3 [9]*

This design in Fig. 8 is an extension of AXA2; it uses 2 more transistors in a pass transistor configuration for a better accuracy of *Sum*. In total, there are 8 transistors, 4 of which are utilized in the XNOR gate. *Sum* is accurate for 6 out of the total 8 input combinations, while C_{out} is accurate for all possible configurations; the total error distance achieved by this design is 2. [9]

$$Sum = \overline{(X \oplus Y)}C_{in},$$
$$C_{out} = (X \oplus Y)C_{in} + XY. \tag{7}$$

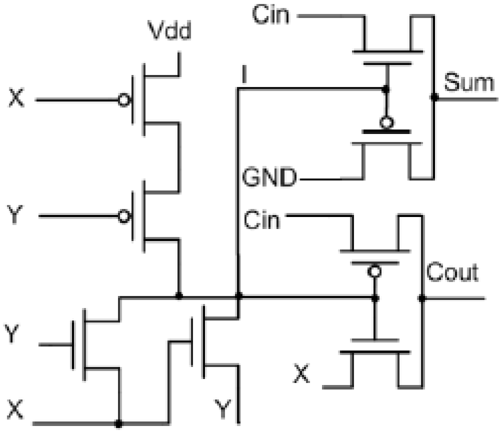


Fig.8 Approximate XNOR Adder 3

E. Accurate Mirror Adder (MA) [3]

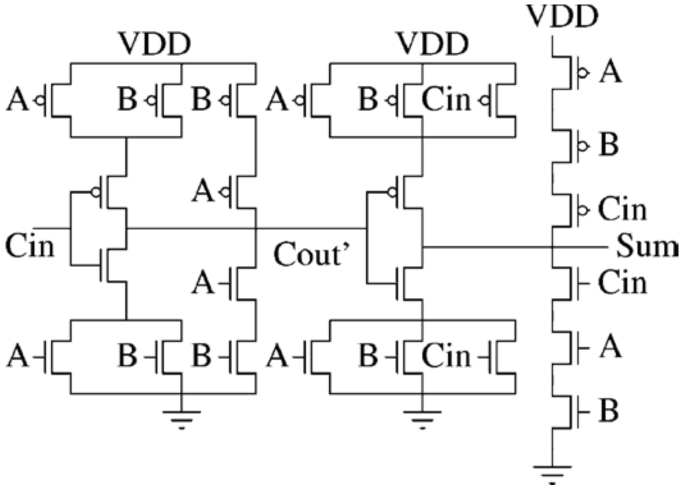


Fig.9. Accurate Mirror Adder [3]

Fig. 9 shows the transistor-level schematic of a conventional MA [13], which is a popular way of implementing a Full Adder (FA). It consists of a total of 24 transistors. Since this implementation is not based on complementary CMOS logic, it provides a good opportunity to design an approximate version with removal of selected transistors [3].

F. Approximation 1 (AMA1) [3]

In order to get an approximate MA with fewer transistors, we start to remove transistors from the conventional schematic one by one. However, we cannot do this in an arbitrary fashion. We need to make sure that any input combination of *A*, *B* and C_{in} does not result in short circuits or open circuits in the simplified schematic. Another important criterion is that the resulting simplification should introduce minimal errors in the FA truth table. A judicious selection of transistors to be removed (ensuring no open or short circuits) results in a schematic shown in Fig. 10, which we call approximation 1. Clearly, this schematic has eight fewer transistors compared to the conventional MA schematic [3].

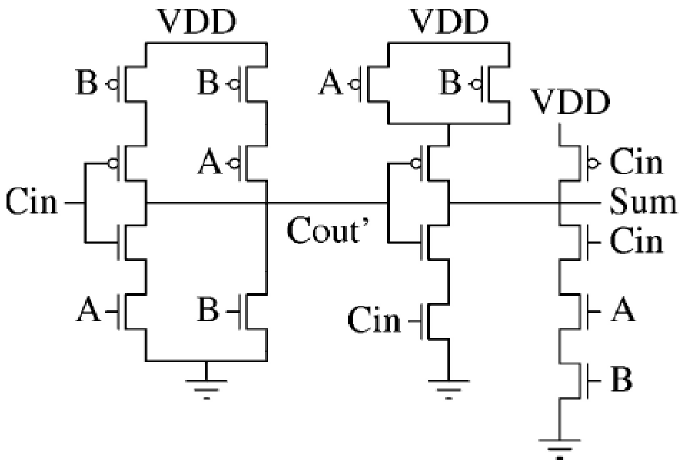


Fig. 10 Approximation 1 [3]

G. Approximation 2 (AMA2) [3]

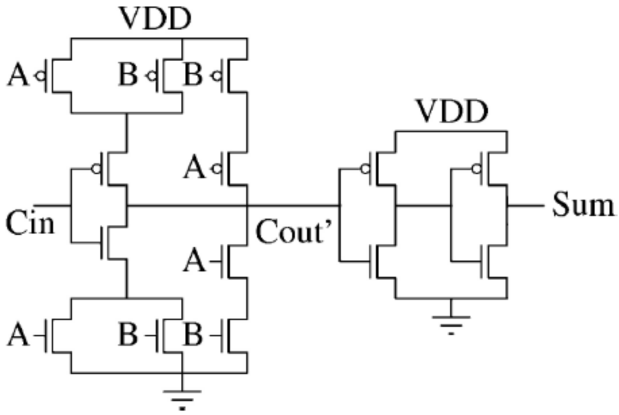


Fig. 11 Approximation 2 [3]

The truth table of an FA shows that Sum= C_{out} for six out of eight cases, except for the input combinations $A = 0, B = 0, C_{in} = 0$ and $A = 1, B = 1, C_{in} = 1$.

Now, in the conventional MA, C_{out} is computed in the first stage. Thus, an easy way to get a simplified schematic is toset Sum= C_{out}. However, we introduce a buffer stage after C_{out} (See Fig. 11) to implement the same functionality. The reason for this can be explained as follows. If we set Sum= C_{out} as it is in the conventional MA, the total capacitance at the Sum node would be a combination of four source–drain diffusion and two gate capacitances. This is a considerable increase compared to the conventional case or approximation 1. Such a design would lead to a delay penalty in cases where two or more multi-bit approximate adders are connected in series, which is very common in DSP applications [3].

H. *Approximation 3 (AMA 3)* [3]

Further simplification can be obtained by combining approximations 1 and 2. Note that this introduces one error in *C*out and three errors in Sum. The corresponding simplified schematic is shown in Fig. 12 [3].

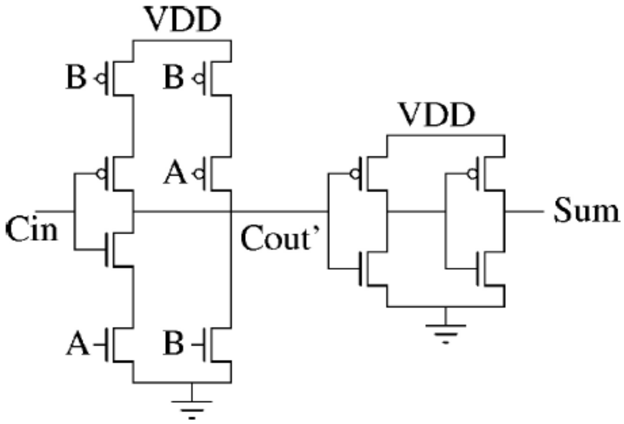


Fig. 12 Approximation 3 [3]

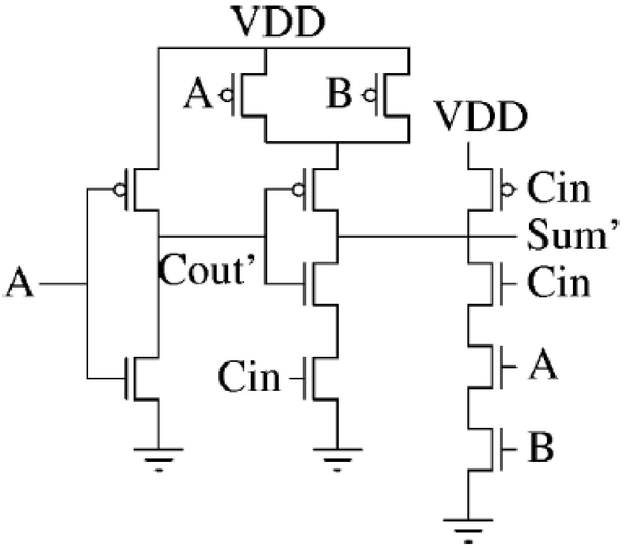


Fig. 13 Approximation 4 [3]

I. *Approximation 4 (AMA4)*

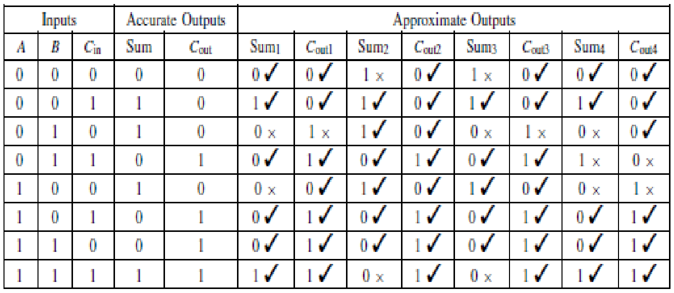
Careful observation of the FA truth table in Fig.14 shows that *C*out = *A* for six out of eight cases. Similarly, *C*out = *B* for six out of eight cases. Since *A* and *B* are interchangeable, we consider *C*out = *A*. Thus [3] proposes a fourth approximation where it just uses an inverter with input *A* to calculate *C*out and Sum is calculated similar to approximation 1. This introduces two errors in *C*out and three errors in Sum, as shown in Fig. 13. The corresponding simplified schematic is shown in Fig. 12. In all these approximations, *C*out is calculated by using an inverter with *C*out as input.

VI. RESULTS

A. *DSP blocks with Approximate Adders*

Since most DSP algorithms used in multimedia systems have inherent error-resiliency, these occasional errors might not manifest as an appreciable reduction in the final output quality. Multimedia DSP algorithms mostly consist of additions and multiplications, which use adders as basic building blocks. The results are shown in fig 14 for various number of bits in image compression algorithm.

TRUTH TABLE FOR CONVENTIONAL FA AND APPROXIMATIONS 1–4



Inputs			Accurate Outputs		Approximate Outputs							
A	*B*	*C*in	Sum	*C*out	Sum1	*C*out1	Sum2	*C*out2	Sum3	*C*out3	Sum4	*C*out4
0	0	0	0	0	0 ✓	0 ✓	1 ×	0 ✓	1 ×	0 ✓	0 ✓	0 ✓
0	0	1	1	0	1 ✓	0 ✓	1 ✓	0 ✓	1 ✓	0 ✓	1 ✓	0 ✓
0	1	0	1	0	0 ×	1 ×	1 ✓	0 ✓	0 ×	1 ×	0 ×	0 ✓
0	1	1	0	1	0 ✓	1 ✓	0 ✓	1 ✓	0 ✓	1 ✓	1 ×	0 ×
1	0	0	1	0	0 ×	0 ✓	1 ✓	0 ✓	1 ✓	0 ✓	0 ×	1 ×
1	0	1	0	1	0 ✓	1 ✓	0 ✓	1 ✓	0 ✓	1 ✓	0 ✓	1 ✓
1	1	0	0	1	0 ✓	1 ✓	0 ✓	1 ✓	0 ✓	1 ✓	0 ✓	1 ✓
1	1	1	1	1	1 ✓	1 ✓	0 ×	1 ✓	0 ×	1 ✓	1 ✓	1 ✓

Fig. 14. Truth Table of Conventional Full Adder and Approximations [3]

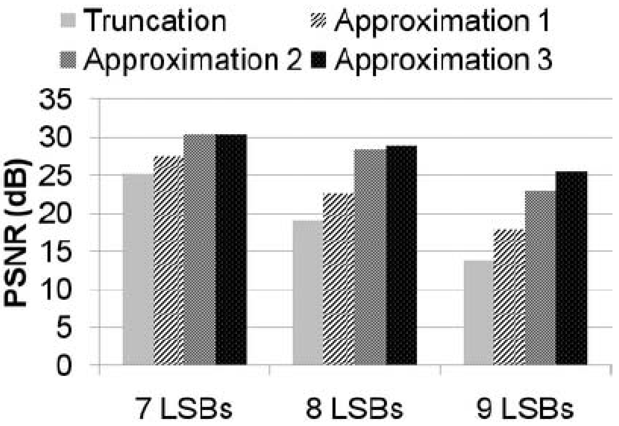


Fig. 15 Output Quality for different techniques [3]

Table 1 shows the comparison between different methods and implementation of variable accuracy and fixed approximations. It can be seen from the comparison table that [3] stands out due to its extensive testing on DSPs on both Image based and Video based algorithms, also the testing is done based on benchmarks and PSNR hence making it relevant to real world application. But [3] does not implement variable configuration Adders which could give us an idea about the tradeoffs necessary to obtain on-demand accuracy and the overhead costs. For example if in certain cases where it is necessary to have high accuracy and some area constraints are present, more than 90% of running time and the area penalty of variable accuracy adder is greater than 35%, then it would be wiser to opt for a fully accurate adder rather than a variable accuracy adder.

Table 1: Comparison Table for Various Methodologies

Research Paper	**Adder Type**	**Accuracy**	**Type of Inputs**	**Method of Testing**
Yang [9]	10 Transistor Adder	Fixed Approximate	Test Vectors	Truth Table
Roy [3]	Mirror Adder	Fixed Approximate	Image and Video	Benchmarks
Raha [12]	DMFA Adder	Accuracy Configurable	Video	Benchmarks
Kahng [13]	ACA Adder	Accuracy Configurable	Image	Benchmarks
Ye [14]	CLA Adder	Accuracy Configurable	Image	Benchmarks

B. *Effect of Process Variation on smaller Approximate Circuits*

Under normal circumstances we would expect, smaller and shorter critical paths to yield smaller delays, this might hold true for transistors which use 45nm or greater channel lengths. But in case of smaller channel lengths the effect of process variations increase drastically as the range tends to lean more towards uniform distribution than Gaussian distribution.

In this paper we simulate the effects of process variation on 22nm Full Adders shown in Fig 9, Fig 10, Fig 11 and Fig 12. We simulate the effects of Process variation by using Monte Carlo simulations on HSPICE by Synopsys. Both the widths of the transistors and the Threshold Voltage are setup to be simulated at different values in this process. By using this method we were able to get realistic results of effects of Process Variation on a 22nm transistor based on Accurate and Approximate Adders. 50 iterations of Monte Carlo Simulations are performed, which gives greater than 80% statistical probability of all the components working correctly.

Most of the Approximate Adders we simulated showed lesser power usage, but a significantly higher delay than the Accurate Adder. In certain cases the delays were up to 1nS greater than that of the Accurate Adder as shown on Table 2. Table 2 is based on a nominal Voltage V_{dd} at 0.8V and using the worst case analysis i.e. when all the inputs are 1.

Table 2: Comparison of Delays of Various Adders

Adder	**Sum Delay (nS)**	**C_{out} Delay (nS)**	**Power Consumption (nW)**
Accurate Adder	2.906	4.1722	540.89
Approximate Mirror Adder 1	6.6693	3.6521	290.11
Approximate Mirror Adder 2	4.1674	4.1665	245.7520
Approximate Mirror Adder 3	3.8051	3.5389	242.1245
Approximate Mirror Adder 3	8.1484	4.2049	239.9926

From Table 1, we can see that the delays of Accurate Adder and Approximate Adders are identical. This erases most of the gains of Approximate Adders. Although the power consumption is low, the delays are on the higher side. These circuits were also tested at various voltages ranging between 0.75V and 0.65V. The results for those simulations are given in Table 3. As the voltages are scaled the delays seem to get worse.

Table 3: Comparison of Delay vs the Scaled V_{dd} for C_{out}

Delay	**Vdd at 0.65V**	**Vdd at 0.7V**	**Vdd at 0.75V**
Accurate Adder	4.1608nS	4.168nS	4.1738nS
Approximate Mirror Adder 1	4.0779nS	3.8672nS	3.6463nS
Approximate Mirror Adder 2	4.1605nS	4.1632nS	4.7106nS
Approximate Mirror Adder 3	5.8247nS	3.8034nS	3.546nS
Approximate Mirror Adder 4	4.205nS	4.205nS	4.205nS

From Table 3 we can see that as the voltage is scaled down the Accurate Adder seems to be working better than Approximate Circuits. This may be due to process variation affecting approximate circuits far greater than the Accurate Circuit due to the number of transistors present in either circuit. The Accurate circuit could calculate the Sum faster than all the other circuits in all cases.16bit Adder for various Approximations wasn’t simulated due to the long time required to simulate the circuit.

SUMMARY

In this paper, two different approaches of approximating circuits are illustrated. These two approaches are compared with various other methodologies in academia. Apart from comparing the approaches of approximating/truncating adders we have also determined the effect of voltage scaling and process variation on approximate circuits.

From the above results we can conclude that approximate circuits, may have better power savings but may not have better performance in terms of delay.

Future work may include simulating and recreating the effects obtained in larger and more complex circuits to determine large scale effects of process variation. Also, this method of testing could also be implemented in sub-22nm MOS technologies. Thereafter this circuit could be tested with real world applications and work needs to be done to find methods of reducing the effects of process variation without affecting the delay of the circuit

REFERENCES

[1] Kahng, Andrew B., et al. "Slack redistribution for graceful degradation under voltage overscaling." *Design Automation Conference (ASP-DAC), 2010 15th Asia and South Pacific*. IEEE, 2010.

[2] Chakrapani, Lakshmi N., et al. "Ultra-efficient (embedded) SOC architectures based on probabilistic CMOS (PCMOS) technology." *Proceedings of the conference on Design, automation and test in Europe: Proceedings*. European Design and Automation Association, 2006.

[3] Gupta, Vaibhav, et al. "IMPACT: imprecise adders for low-power approximate computing." *Proceedings of the 17th IEEE/ACM international symposium on Low-power electronics and design*. IEEE Press, 2011.

[4] R. Hegde and N.R. Shanbhag, "Soft digital signal processing," IEEE Trans. VLSI Systems (TVLSI), 9(6):813–823, December, 2001J. Clerk

[5] Kim, Jaeyoon, and Sandip Tiwari. "Inexact computing for ultra low-power nanometer digital circuit design." *Nanoscale Architectures (NANOARCH), 2011 IEEE/ACM International Symposium on*. IEEE, 2011.

[6] Han, Jie, and Michael Orshansky. "Approximate computing: An emerging paradigm for energy-efficient design." *Test Symposium (ETS), 2013 18th IEEE European*. IEEE, 2013.

[7] Liang, Jinghang, Jie Han, and Fabrizio Lombardi. "New metrics for the reliability of approximate and probabilistic adders." *Computers, IEEE Transactions on* 62.9 (2013): 1760-1771.

[8] Yang, Zhixi, et al. "Approximate XOR/XNOR-based adders for inexact computing." *submitted to IEEE Conf. on Nanotechnology*. 2013.

[9] H.A. Mahmoud and M.A. Bayoumi, "A 10-transistor low-power high-speed full adder cell," ISCAS'99, vol. 1, pp. 43-46, 1999.

[10] Lin, Jin-Fa, et al. "A novel high-speed and energy efficient 10-transistor full adder design." *Circuits and Systems I: Regular Papers, IEEE Transactions on* 54.5 (2007): 1050-1059.

[11] Madisetti, Vijay K., and V. Madisetti. *VLSI Digital Signal Processors*. Butterworth-Heinemann, 1995.

[12] Raha, Arnab, Hrishikesh Jayakumar, and Vijay Raghunathan. "A Power Efficient Video Encoder Using Reconfigurable Approximate Arithmetic Units." *VLSI Design and 2014 13th International Conference on Embedded Systems, 2014 27th International Conference on*. IEEE, 2014.

[13] Kahng, Andrew B., and Seokhyeong Kang. "Accuracy-configurable adder for approximate arithmetic designs." *Proceedings of the 49th Annual Design Automation Conference*. ACM, 2012.

[14] Ye, Rong, et al. "On reconfiguration-oriented approximate adder design and its application." *Proceedings of the International Conference on Computer-Aided Design*. IEEE Press, 2013.

[15] Park, Amir Yazdanbakhsh Bradley Thwaites Jongse, and Hadi Esmaeilzadeh. "Methodical Approximate Hardware Design and Reuse."

[16] Bhardwaj, Kartikeya, and Pravin S. Mane. "ACMA: Accuracy-configurable multiplier architecture for error-resilient System-on-Chip." *Reconfigurable and Communication-Centric Systems-on-Chip (ReCoSoC), 2013 8th International Workshop on*. IEEE, 2013.

[17] Strojwas, Andrzej J. "Cost effective scaling to 22nm and below technology nodes." *VLSI Technology, Systems and Applications (VLSI-TSA), 2011 International Symposium on*. IEEE, 2011.

[18] R. Ashraf and R. F. DeMara, "Scalability of Modular Redundancy for Near-Threshold Computing," Workshop on Highly-Reliable Power-Efficient Embedded Designs (HARSH 2014), Orlando, Florida, USA, February 16th, 2014.