Implementation Of 24-Bit Delay And Area Efficient Enhanced Equal Segment Adders

Roopali Shinde[1], Prameela Kumari N[2]
[1,2]School of Electronics and Communication Engineering, Reva University,Bangalore,India-560064
[1]rupaalii.s@gmail.com, [2]prameela.n@reva.edu.in

Abstract

The proposed work presents the enhancement of the general architecture of an n-bit approximate adder done using the carry predictor unit. The simulation is carried out using the Xilinx ISE 14.7 environment. The aim of the proposed work is to determine the carry block output. The paper also presents the comparative analysis of ESA based 24-bit Ripple carry adder, Kogge stone adder and carry increment adder with emphasis on area, delay and power consumption based on ESA structure with and without overlapping.

Keywords: *Full Adder, 24-Bit, Ripple carry adder, Kogge stone adder, Gate Delay, PDP (Power Delay Product), Parallel Adders*

1 Introduction:

Devices such as Microprocessors, microcontrollers and DSP processors have a very complex logic circuits, one of the fundamental logic block which interms of device functioning is adder circuits. These circuits that is, adder blocks are also mainly important to carry out functions like multiplication and divisions. Thus one of the important factors in carrying out the operations as mentioned earlier including addition, is time, delay and power consumption.

There are some applications where the need for the result just needs to be approximate, and those results are sufficient.

Approximate computing is one such computation technique that actually returns with a probably incorrect results but are sufficient for some applications. For example, situations such as search engines, where the results are not exact but are acceptable. In similar way, sometimes dropping of few frames in the video applications which are normally undetected because of perceptual limited abilities of humans. The process of approximate computing is also mainly based on the observation which actually states that in many of the scenarios, even though performing the exact computation needs a large amount of resources, but allowing the use of bounded approximations can also provide un appropriate gains to performance and energy, all at the same time achieving the acceptable result accuracy.

Thus here an approximate adder structure is developed, and enhanced by carry predictor unit. And using the intelligent 24- bit full adder structure, parallel adders like Ripple carry adder, Kogge stone adder and carry increment adder are simulated and worst case gate delay, time duration and power is measured and compared with the existing technologies.

The paper is further organized with section II giving the insight into the previous work, section III gives details regarding the several different strategies of approximate computing, while section – IV explains the different adder circuits implemented, Section – V discusses the experimental results and finally section – VI outlines the proposed work with conclusion.

2 Review Of Previous Work:

It is always necessary to review the work of the researchers done previously to deduce out the problem statement and solution for it. This leads to the development of device which overcomes the problems posed by the earlier ones.

The demands of the available resources have by far exceeded the computational applications like adder circuits, processors and storage demands of the modern systems as applications being used on large scale like financial analysis, social media, and scientific computing, have started to acquire eminence. As the number of processors in use is growing only at the rate of times, where as it is said and mostly awaited that in the coming decade the amount of data and information which will be controlled by worldwide data centers to grow by 50 times compared to the growth of processors [1].

The energy consumption of the data centers located in US has been increasing rapidly, as the facts indicate, the consumption is awaited to grow from 61 billion kilo watt-hour(kWh) during 2006 [2] to an estimated 91 billion kilo watt-hour(kWh) in 2013 and finally to 140 billion kilo watt-hour(kWh) in 2020 [3].

The problems that awaits the computing industry in coming future is that the performance demands will be out pacing the growth of in the resource budgets and thus leading to over provisioning of various resources which might not be able to solve the problems. Hence a guaranteed solution for this is the use of approximate computing and storage, as it is based on the instinctive observations, but when performing high level accurate computation or maintaining high level service requires high amount of resources, which leads to use of selective approximation technique and also violation of some specification which may lead to the inaccuracy in gained efficiency. For instance, 50x energy economization can be gained by using categorization of accuracy loss of just percent [4]

A reverse kinematics application can be accelerated by up to 26x when it compared to the GPU being executed, when a neural approximation approach is used [5].

For intelligently having the tradeoff for execution, storage and results of precision for the performance or sometimes energy acquired, approximate computing borrows the use of error tolerant code regions in various applications. In a nutshell the approximate computing explores the wide gap between the accuracy level which are needed by the applications or the users and those made available by the computing devices for gaining the diverse optimizations. Hence the approximate computing has the ability to support a wide range of the application and device modelling, for example, multimedia, scientific computing, data analytics, ML technology, Signal processing to name a few. But though AC is promising but not final solution or remedy. Thus effective Approximate computing needs a precise selection of approximate code or data along with approximate strategy, as inconsistent approximation can lead to unsatisfactory quality loss [6-9].

Sometimes approximation in the process of control flow and memory ingress operations can lead to disastrous results like segmentation faults [10].

Finally, to make sure that the quality specifications are met, a very proper and sensitive monitoring of system output is needed. As huge losses make the output to be unacceptable and needs the execution to be repeated again with preciseness. Thus using the full potential of th approximate computing requires the answering several issues, and in recent times many techniques have been proposed to full fill this need and have been explained in the coming sections.

3 Dıfferent Strategıes For Approxımate Computıng

A. Approximate circuits

Hardware overhead can be reduced with the use of approximate adders, logical circuits and multipliers. For instance, in an approximate multi- bit adder circuits it can actually ignore the carry chains and hence allow all the sub-adders logical circuits to perform the addition operations in parallel.

B. Approximate storage

By using the truncating, the lower bit in floating point data, the data can be stored approximately, rather than using or storing the data directly. There is various other method of accepting or using less reliable memory is for example, using DRAM and eDRAM, the refresh rate can be lowered and also in SRAM, the supply voltage can be reduced. Thus finally any error detection and the correction mechanism are disabled.

C. Software-level approximation

At software level, various ways are there to carry out approximate computing. In some cases, memorization method can be applied. For achieving faster results, there is provision to skip some of the iterations of loops. And also, some tasks can also be skipped, like the one where a run time condition can be left out as they are not going to be useful. While Monte Carlo algorithms and randomized algorithms are trade corrected for guarantee of execution of time. For some application where acceleration on particular hardware is required, the computation process can be reformulated based on paradigms. For example: a neural processing unit.

D. Approximate system

This strategy is used for different subsystem of a particular system like processors, sensors, memory and communication module are synergistically approximated to get a much better system level Q-E trade off curve, as compared to the individual approximations to each of the subsystems.

4 Dıfferent Adder Cırcuıts Iplemented:

This section presents different fast adder circuits implemented for comparative analysis based on various factors like worst case gate delay, with and without overlapping and power consumption.

A. Ripple carry adder

But the cascading two full adders blocks in series, the ripple carry adder is developed. At any instant, for addition of two binary digits of ripple carry, one full adder is responsible. Here the carryout of one stage is given as input to the carry-in of the next stage. Though it is a very simple adder, but can’t be used for large bit numbers. As the bit length increase the delay also increases linearly, which is one of the drawbacks. A very serious delay in the RCA type is when the carry signal transition ripples goes through all the stages of adder segments from the least significant bit to most significant bit and is approximated as:

$$T = (n-1)\, tc + ts \qquad \text{Eq (1)}$$

Where tc is delay through carry stage of a full adder, and ts is the delay to compute the sum of the last stage.

The delay in RCA, is actually linearly proportional to n, which is the number of bits, and hence its performance is limited as n grows bigger.

Path delay: -- 2n+1 = 2*24 +1= 49 gate delays

Figure 1a and 1b below shows the – bit ripple bit carry adder. As shown below, here 4 full adders are connected in cascade with each other. CO is represented as the carry input bit and which is always zero. As the input carry CO is given or applied to two input sequences of A1, A2, A3, A4 and then B1, B2, B3, B4 there by output being represented by S1, S2, S3, S4 and finally the output carry being C4. Speeding the addition process is done by determining carry values sooner.

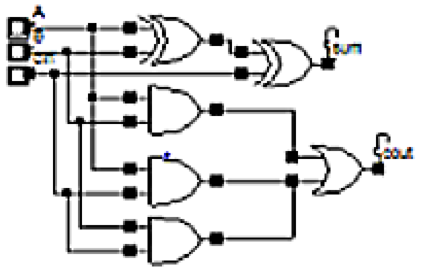


Figure 1. Full adder logical gate realization

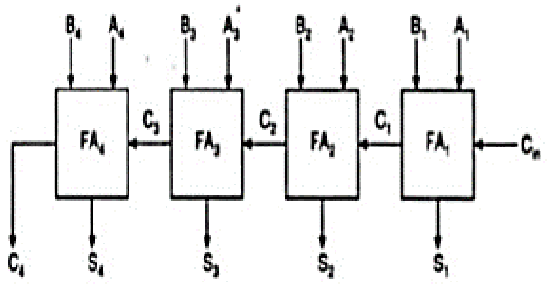


Figure 2. Four-bit ripple carry adder

I. Advantages:

1. Addition process for n-bit sequences can be performed to achieve accurate results.
2. Design is simple
3. Low power consumption
4. Compact layout there by smaller chip area.

II. Disadvantage:

1. Longer delays caused due to the propagation of carry from LSB to MSB
2. Highest delay for n-bit adder = 2n+1—24*2 which is equal to 49 gates delay

III. Applications:

1. Are used for addition of n-bit input sequences

2. Applicable for DSP and microprocessors.

B. Kogge Stone adder

A parallel prefix form of carry look – ahead adder is explained here, which is also known as Kogge stone adder. Carry signal is generated in O time, that is Log2 N time. It is mostly used in industry operations and is considered as one of the fastest adder design. As the computing is done in parallel operations, the carries are generated very fast. The operation speed is also very high because of low depth of node and operations are carried out in parallel. One of the important factors is that the output of the adder depends on the previous inputs. Normally the Kogge stone parallel prefix adder constructed using a three-stage process. The carry generation block shown in Figure 2 below.

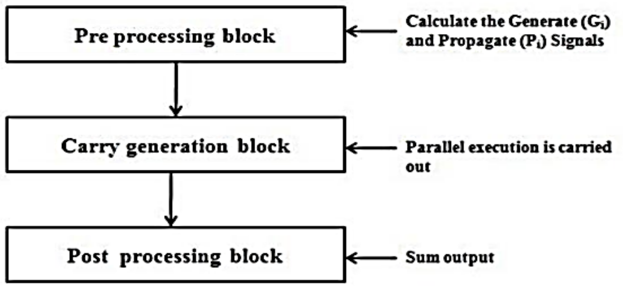

Pre processing block
Calculate the Generate (G_i) and Propagate (P_i) Signals
Carry generation block
Parallel execution is carried out
Post processing block
Sum output

Figure:General Architecture of Parallel Prefix Adder

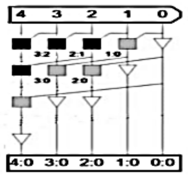


Figure 3. Carry Generation Block

I. Pre-Processing Block

Here the block is early or starting process block of the parallel prefix adder, and mainly used for the generation of the sent signal and the signal which is generated is computed using the inputs by using the equations shown below.

Pi = Ai XOR Bi (1)

Gi = Ai AND Bi (2)

II. Carry Generation Block

This block is most dominant block in the entire adder design. The block is made up of 2 components which are block cell and gray cell. Here the block cell is used for the generation of the signal and this signal which is sent then is used for the next stage of calculation. Whereas the grey cell is used for the production of generated signal which is needed for the calculation of sum in the next stage.

III. Post Processing Block

The post processing block is the final stage of the adder circuit. here the final output is sum and carry. And

Si = Pi

IV. Advantages

1. Fast operation

2. Low depth high node count gives more area

V. Disadvantages

1. Large area required for implementation and complex circuit routing because of fan out interconnections.

VI. Applications

1. Is used in high speed applications

C. Carry Increment adder

A standard CIA is made up of RCA and incremental circuitry. Half adders in ripple carry chain with a proper sequential order is used to design the incremental circuit. Addition is performed by separating all the number of bits into groups of 4-bit and then the operation is carried out by using several 4-bit RCA. Rather than computing two partial sum from every group and finally selecting the correct one, only one sum is calculated and incremented if required based on input carry. Hence a second adder and multiplexers can be replaced by a very small incremental circuit and the advanced resultant architecture is known as carry incremental adder (CIA). This can be explained as follows, usually an 8-bit CIA is made up of two 4- bit RCA and first block of RCA adds the first 4-bits which results in a 4-bit partial sum and carry output. The below Figure 3 shows the block diagram of an 8-bit CIA.

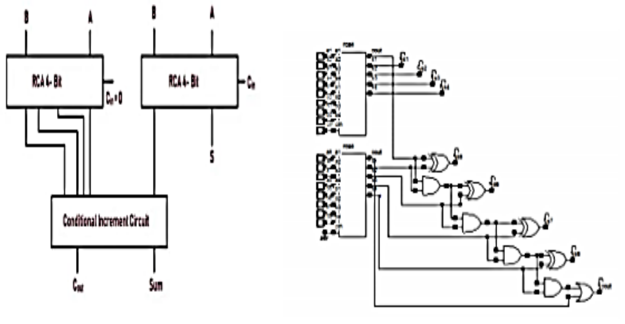


Figure 4. Block diagram of CIA and CIA incremental circuit Using XOR & AND.

Thus, first 4-bit of sum of CIA is directly obtained from first block of RCA. And the carry output of first RCA block is given as input to the Cin of incremental circuit. Incremental circuit consists of Half Adders (HA)/ Full adders. Hence, the partial sum obtained from the second RCA block is given to incremental circuit.

I. Advantages

1. Has best delay performance which is very important factor in very high speed operations.

II. Applications

1. VLSI applications for low power

2. Used in parallel computing

5 Experımental Results:

This section presents the simulation results of the above presented adder and comparative analysis based on with and without overlap bits.

A. Ripple carry adder

The proposed simulation has been carried out in Xilinx ISE 14.7 software. The results are presented below for 24-bit RCA type adder without overlap. In a 24-bit [ripple carry] adder, there are 24 full adders, so the critical path (worst case) delay is 21 * 2(for carry propagation) + 3(for sum) = 45 gate delays.

The Figure 4 below shows the waveform of delays for the 24- bit RCA type adder.

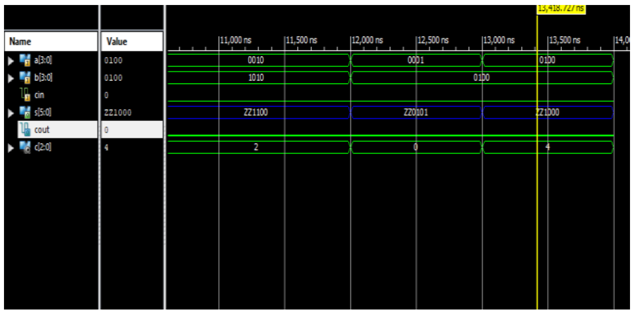


Figure 5. output waveform of RCA type 24 – bit adder

The results are supported with schematic development of proposed type in Figure 5 below.

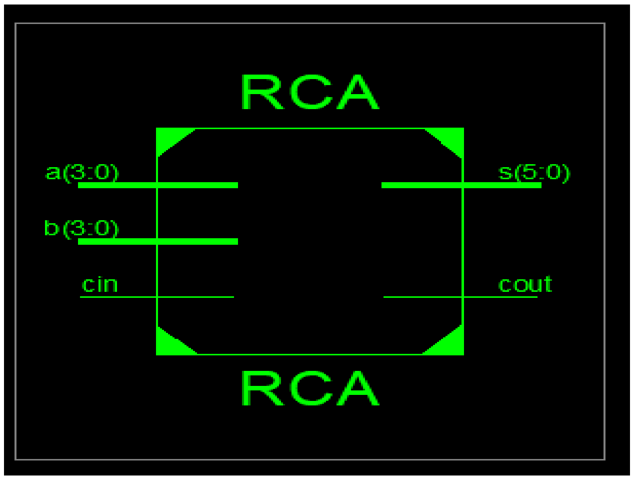


Figure 6. Schematic of 24 – bit RCA.

The schematic shows the inputs A, B and Cin, and outputs sum and carry.

The below Figure 6 shows the technological view of the implemented 24 – bit RCA type adder.

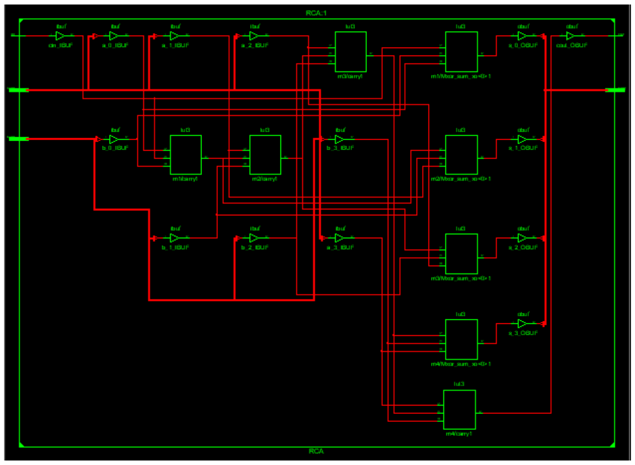
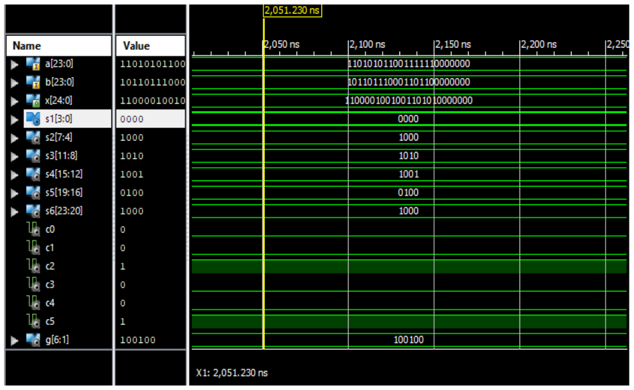


Figure 7. Technological view

B. Kogge stone adder

The results for the Kogge stone adder is presented here, Figure 7 shows the output waveform of the KSA.



```
Name        Value
a[23:0]     11010101100     110101011001111110000000
b[23:0]     10110111000     101101110001101100000000
x[24:0]     11000010010     1100001001001101010000000
s1[3:0]     0000            0000
s2[7:4]     1000            1000
s3[11:8]    1010            1010
s4[15:12]   1001            1001
s5[19:16]   0100            0100
s6[23:20]   1000            1000
c0          0
c1          0
c2          1
c3          0
c4          0
c5          1
g[6:1]      100100          100100
X1: 2,051.230 ns
```

Figure 8. Output waveform of 24- bit KSA

The schematic of the KSA is shown below in Figure 8. With A and B as input and X as output.

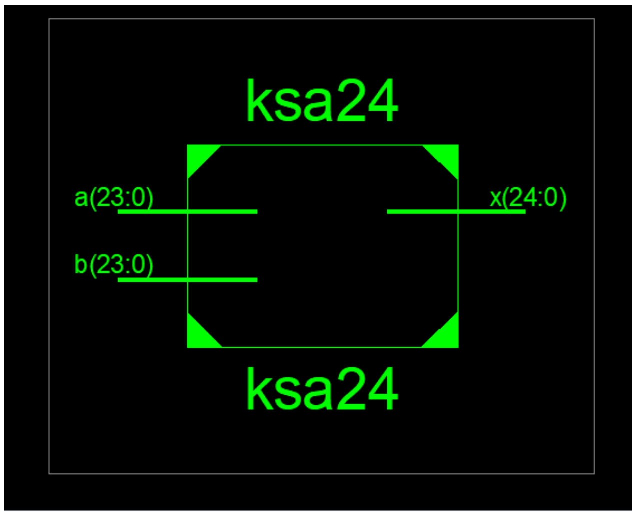


Figure 9. Schematic of KSA 24 – bit adder

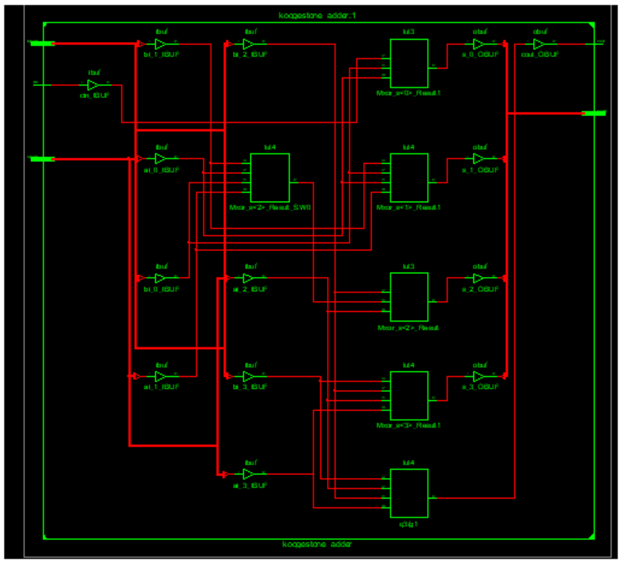


Figure 10. technological view of KSA

The above figure shows the technological buildup of 24-bit KSA without overlap.

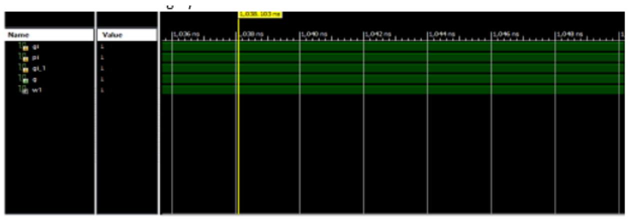


Figure 11. output wave form of Grey cell block

The carry generation block made up of two important cells, grey and black cell. The above Figure 10 shows the output waveform of the grey cell.

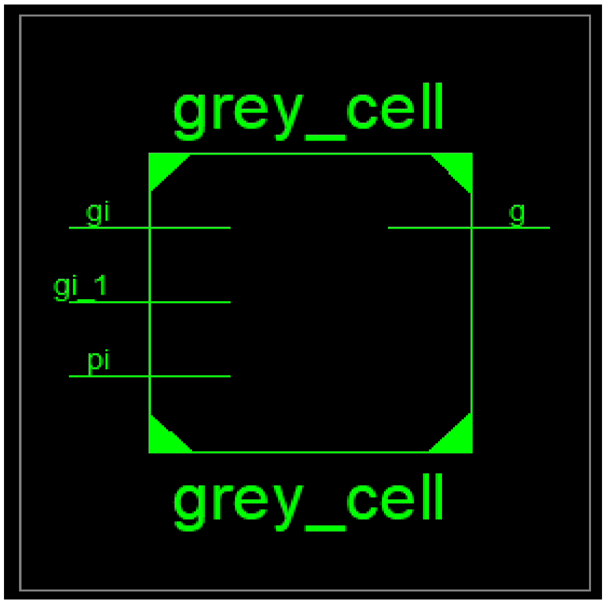


Figure 12. Schematic of grey cell

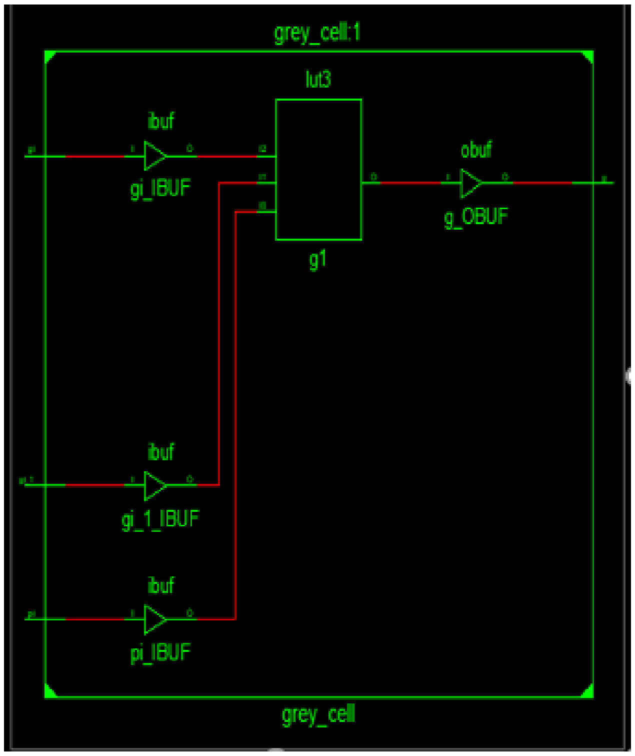


Figure 13. Technological view of grey cell

Figure 11 and 12 show the schematic and technological view of the grey cell respectively.

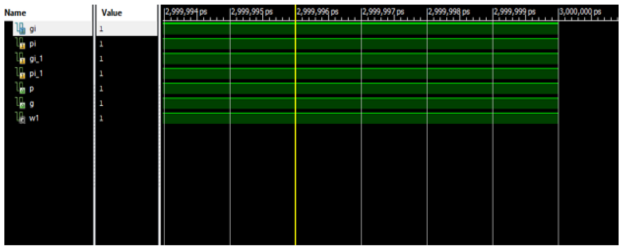


Figure 14. Output waveform of Black cell

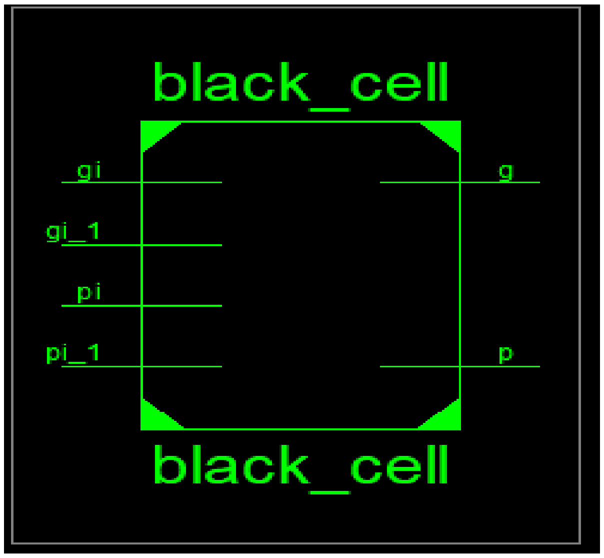


Figure 15. Schematic of Black cell

The other block which makes up the carry generation block, is the black cell, Figure 13, 14 and 15 represent the output waveform, schematic and technological view of the black cell.

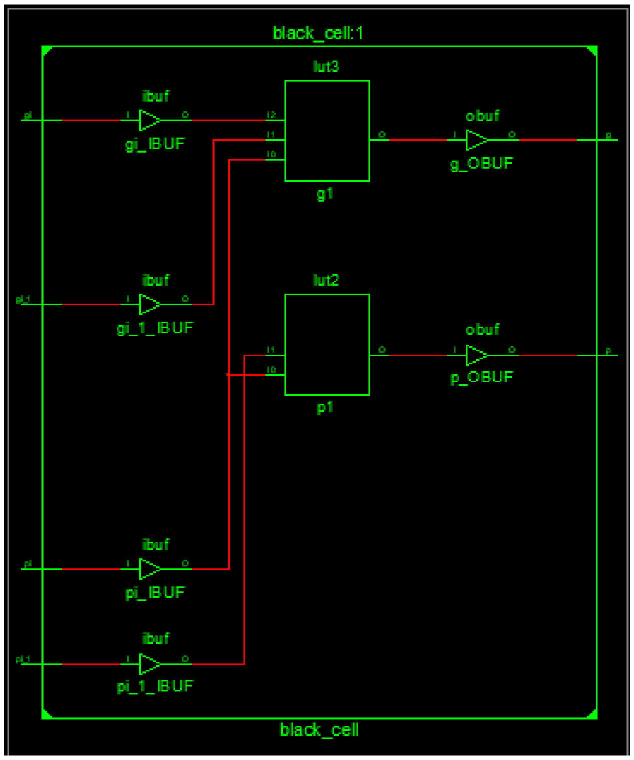
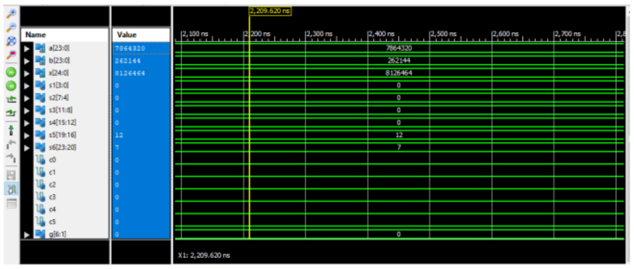


Figure 16. Technological view of black cell

C. Carry increment adder

A 24 – bit CIA includes 6 RCA with 4 bit each. Here the Figure 16 shows the output wave form for the

24-bit Carry increment adder.



Name	Value
a[23:0]	7864320
b[23:0]	262144
x[24:0]	8126464
s1[3:0]	0
s2[7:4]	0
s3[11:8]	0
s4[15:12]	0
s5[19:16]	12
s6[23:20]	7
c0	0
c1	0
c2	0
c3	0
c4	0
c5	0
g[6:1]	0

Figure 17. Output waveform of CIA

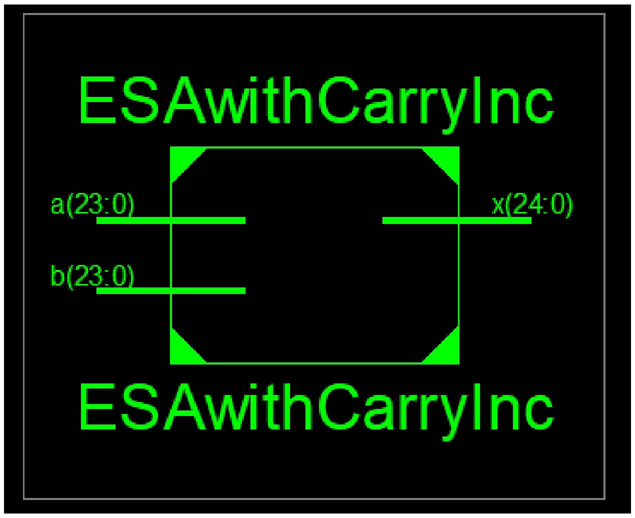


Figure 18. Schematic of CIA 24 – bit adder.

The above Figure shows the schematic of a 24-bit adder circuit. With A and B as inputs and X as output.

The technological view of the 24 – bit carry increment adder is shown in below Figure. 18.

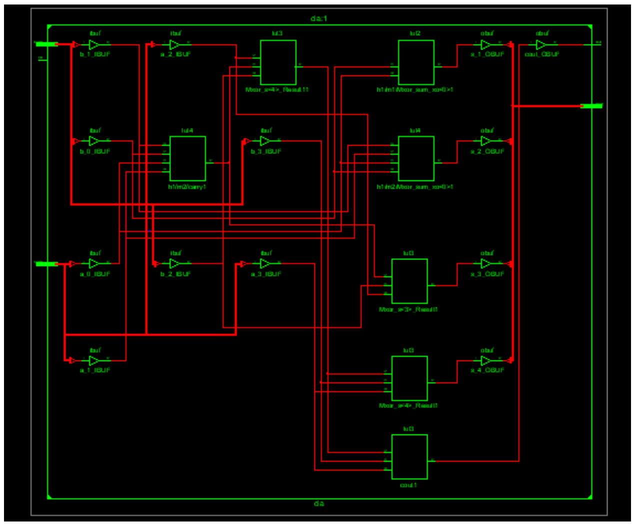


Figure 19. Technological view of Carry Increment Adder

The comparative analysis of the Ripple carry adder, Kogge stone adder and Carry Increment adder is shown in the Table 1 below.

Table 1. Comparative analysis with and without overlapping

ESA TOPOLOGY	**ADDERS**	**AREA**	**DELAY**	**Power (mw)**
Generalized ESA structure with overlapping bits	CIA	43	12.779ns	0.166
	KSA	43	6.072ns	0.166
	RCA	43	7.276ns	0.166
Proposed ESA structure without overlapping bits N\L (24/6) and carry generation block	CIA	40	5.688ns	0.166
	KSA	40	5.572ns	0.166
	RCA	40	6.264ns	0.166

6 Conclusıon:

In the proposed work, a comprehensive and detailed analysis of 24 bit - adder ESA topologies-based RCA, KSA and has been presented based on area, worst case delay and power. That is the comparative analysis is carried out with respect to above mentioned based on without and with overlapping adders. Here the simulation results have been presented as the basis of comparisons. Based on the presented analysis and results, the best adder with respect to area, worst case delay and power is the KSA-ES adder without overlapping. They are also suited for high performance applications and low power consumption circuits. While the simplest adder is the ripple carry adder without overlapping. This in a nutshell the paper presents the comparison between the ESA based RCA, KSA and CIA topologies with and without overlapping.

References

1. John Gantz and David Reinsel. 2011. Extracting Value from Chaos. http://www.emc. com/collateral/analyst-reports/idc-extracting-value-from-chaos-ar.pdf. (2011).
2. Sparsh Mittal. 2014a. Power Management Techniques for Data Centers: A Survey. Technical Report ORNL/TM-2014/381. Oak Ridge National Laboratory, USA.
3. NRDC.2013.http://www.nrdc.org/energy/data-center-efficiency-assessment.asp. (2013).
4. Vinay K Chippa, Debabrata Mohapatra, Kaushik Roy, Srimat T Chakradhar, and Anand Raghunathan. 2014. Scalable effort hardware design. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 22, 9 (2014), 2004–2016.
5. Beayna Grigorian and Glenn Reinman. 2015. Accelerating divergent applications on simd architectures using neural networks. ACM Transactions on Architecture and Code Optimization (TACO) 12, 1 (2015), 2.
6. Ashish Ranjan, Swagath Venkataramani, Xuanyao Fong, Kaushik Roy, and Anand Raghunathan. 2015. Approximate storage for energy efficient spintronic memories. In Design Automation Conference. 195.

7. John Sartori and Ravindra Kumar. 2013. Branch and data herding: Reducing control and memory divergence for error-tolerant GPU applications. IEEE Transactions on Multimedia 15, 2 (2013), 279–290.
8. Qian Zhang, Ting Wang, Ye Tian, Feng Yuan, and Qiang Xu. 2015. ApproxANN: an approximate computing frameworkfor artificial neural network. In Design, Automation & Test in Europe. 701–706.
9. SwagathVenkataramani,AnandRaghunathan,JieLiu,andMohammedShoaib.2015.Scalable-effort classifiers for energy-efficient machine learning. In Design Automation Conference. 67.
10. Yavuz Yetim, Margaret Martonosi, and Sharad Malik. 2013. Extracting useful computation from error prone processors for streaming applications. InDesign, Automation & Test in Europe Conference & Exhibition (DATE). 202–207