ISSN: 2321-2152

IJMECE

International Journal of modern electronics and communication engineering

E-Mail
editor.ijmece@gmail.com
editor@ijmece.com

www.ijmece.com

ISSN 2321-2152
www.ijmece.com
Vol 13, Issue 1, 2025

CONCURRENT ERROR DETECTABLE CARRY SELECT ADDER WITH EASY TESTABILITY OF 16-BITS

[1]Mr. P.Suresh Kumar, [2]D.Akhila, [3]V.Bindhu, [4]M.Gayathri
[1](Associate Professor), ECE. Bhojreddy engineering college for women
[234]B.tech Scholar, ECE. Bhojreddy engineering college for women

ABSTRACT

As the process shrink of integrated circuits advances and the integration density increases, reliability of integrated circuits in field becomes an issue. For critical systems such as server class computers and embedded systems, concurrent detection of errors is important. Addition is the most basic arithmetic operation, and a lot of adders are used in VLSIs. Reliable adders are important for realizing reliable systems. In these previously proposed concurrent error detectable adders, any erroneous output caused by a single fault can be detected. However, an erroneous output caused by multiple faults is not guaranteed to be detected. If a second fault has occurred before the first fault is found through detection of an erroneous output, the concurrent error detectability can be lost. For reliable adders, it is crucial that the first fault is noticed before a second fault occurs. Effective self-repairing can be achieved if the fault along with its exact location can be determined. In this work, a self repairing hybrid adder is proposed with fault localization. It uses the advantages of ripple carry adder and carry-select adder to reduce the delay and area overhead.

Keywords: FPGA, VLSI, FFT, DSP, FIR.

INTRODUCTION

As the scale of integration keeps growing, more and more sophisticated signal processing systems are being implemented on a VLSI chip. These signal processing applications not only demand great computation capacity but also consume considerable amount of energy. While performance and Area remain to be the two major design tolls, power consumption has become a critical concern in today's VLSI system design[]. The need for low-power VLSI system arises from two main forces. First, with the steady growth of operating frequency and processing capacity per chip, large currents have to be delivered and the heat due to large power consumption must be removed by proper cooling techniques. Second, battery life in portable electronic devices is limited. Low power design directly leads to prolonged operation time in these portable devices.

Addition usually impacts widely the overall performance of digital systems and a crucial arithmetic function. In electronic applications adders are most widely used. Applications where these are used are multipliers, DSP to execute various algorithms like FFT, FIR and IIR. Wherever concept of multiplication comes adders come in to the picture. As we know millions

ISSN 2321-2152
www.ijmece.com
Vol 13, Issue 1, 2025

of instructions per second are performed in microprocessors. So, speed of operation is the most important constraint to be considered while designing multipliers. Due to device portability miniaturization of device should be high and power consumption should be low. Devices like Mobile, Laptops etc. require more battery backup.

So, a VLSI designer has to optimize these three parameters in a design. These constraints are very difficult to achieve so depending on demand or application some compromise between constraints has to be made. Ripple carry adders exhibits the most compact design but the slowest in speed. Whereas carry look ahead is the fastest one but consumes more area. Carry select adders act as a compromise between the two adders. In 2002, a new concept of hybrid adders is presented to speed up addition process by Wang et al. that gives hybrid carry look-ahead/carry select adders design. In 2008, low power multipliers based on new hybrid full adders is presented.

DESIGN of area- and power-efficient high-speed data path logic systems are one of the most substantial areas of research in VLSI system design. In digital adders, the speed of addition is limited by the time required to propagate a carry through the adder. The sum for each bit position in an elementary adder is generated sequentially only after the previous bit position has been summed and a carry propagated into the next position.

The CSLA is used in many computational systems to alleviate the problem of carry propagation delay by independently generating multiple carries and then select a carry to generate the sum.

LITERATURE SURVEY

In most of the digital circuits arithmetic addition is important and elementary operation that is encountered in the critical-path [1]. Adders influence the system performance [2]. Adder plays crucial role when integrated with hardware. The analysis time of addition is directly proportional to number of operands. The propagation of carry affects the operational speed of the device. There are different types of adders reported in the literature. Different types of adders are 1) single-bit, 2) multi-bit, 3) Tree-based structures. Half-adder and Full-adder are of single-bit type. Ripple Carry adder (RCA), Carry Increment adder (CIA), and Carry save adder (CSA), Carry Skip adders (CSKA) and Carry Look-ahead adder (CLA) comes under Multi-bit adder category. Tree-based structures include Han-Carlson, Ladner-Fischner, Brent-kung, Sklansky and Kogge-stone adders.

PROPOSED SYSTEM

The main concept behind VSS structure is to balance the critical path delay. The hybrid variable feature is achieved by replacing the nucleus stage with the proposed technique. The hybrid structure shown in the diagram below consists of stages 1 to Q. The first stage consists of a single Ripple Carry Adder (RCA) block with half adders. The consecutive stages consist of concatenation and incrementation block, RCA and skip logic. The middle stage has large delay and

ISSN 2321-2152
www.ijmece.com
Vol 13, Issue 1, 2025

sizes, which in turn replaced by PPA to minimize the delay. In the proposed structure, the parallel prefix network uses a ling adder and Kogge Stone Adder. The generation of carry is done by Pseudo generate and propagate functions. The generation of sum and carry bits is less complex when compared to the conventional structure and outputs will be available within a minimum time.

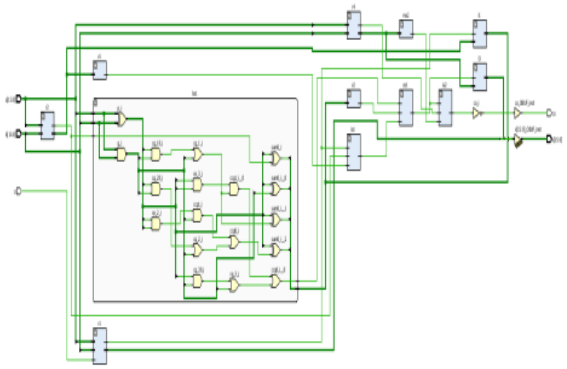


Fig. 1 Proposed Hybrid Varied Latency Carry Skip Adder

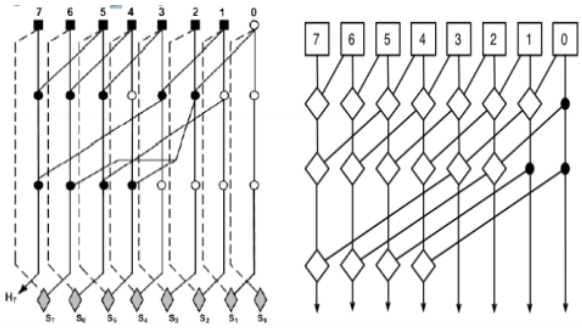


Fig. 2 Internal structure of Ling Adder and Kogge Stone

Figure 2 depicts the internal structure of 8 bit PPA based on ling equations and Kogge Stone Adder. The proposed structure is developed for 16 bit. In this method, there are three stages for ling carry and sum computation.

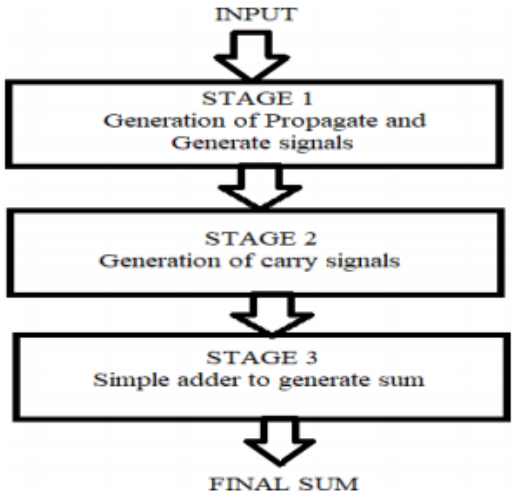


Fig. 3 PPA Carry and Sum generation

The propagate (P) and generate function (G) are compute in the stage 1 as depicted in the figure 3. In the next stage Ling Parallel prefix architecture is responsible for computation of longest carry P (8:1) and G (8:1) which is the product of the input bits and the intermediate signals.

CARRY SELECT ADDER

DESIGN of area- and power-efficient high-speed data path logic systems are one of the most substantial areas of research in VLSI system design. In digital adders, the speed of addition is limited by the time required to propagate a carry through the adder. The sum for each bit position in an elementary adder is generated sequentially only after the previous bit position has been summed and a carry propagated in to the next position .The CSLA is used in many computational

ISSN 2321-2152
www.ijmece.com
Vol 13, Issue 1, 2025

systems to alleviate the problem of carry propagation delay by independently generating multiple carries and then select a carry to generate the sum. However ,the CSLA is not area efficient because it uses multiple pairs of Ripple Carry Adders (RCA) to generate partial sum and carry by considering carry input cin=0 and cin=1, then the final sum and carry are selected by the multiplexers (mux).The basic idea of this work is to use Binary to Excess-1 Converter(BEC) instead of RCA with cin=1 in the regular CSLA to achieve lower area and power consumption The main advantage of this BEC logic comes from the lesser number of logic gates than the n-bitFull Adder (FA) structure.

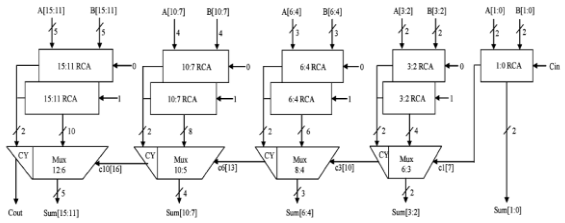


Fig.4 Regular 16-b SQRT CSLA.

In the proposed SQRT CSLA, the transistor count is trade-off with the speed in order to achieve lower power delay product. Thus the proposed SQRT CSLA using CBL is better than all the other designed adders. Fig. 9 shows the Block diagram of Proposed SQRT CSLA.

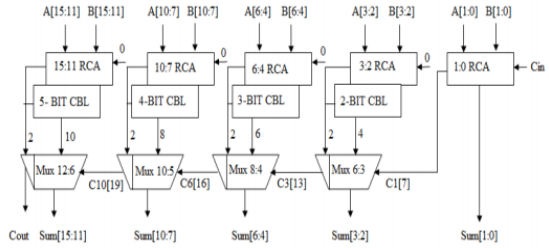


Fig. 5 16-Bit Proposed SQRT CSLA using Common Boolean Logic.

MULTIPLEXER

In electronics, a multiplexer (or MUX) is a device that selects one of several analog or digital input signals and forwards the selected input into a single line.[1] A multiplexer of 2n inputs has n select lines, which are used to select which input line to send to the output.[2] Multiplexers are mainly used to increase the amount of data that can be sent over the network within a certain amount of time and bandwidth.[1] A multiplexer is also called a data selector. They are used in CCTV, and almost every business that has CCTV fitted, will own one of these.

An electronic multiplexer makes it possible for several signals to share one device or resource, for example one A/D converter or one communication line, instead of having one device per input signal.

On the other hand, a demultiplexer (or demux) is a device taking a single input signal and selecting one of many data-output-lines, which is connected to the single input. A multiplexer is often used with a complementary demultiplexer on the receiving end.[1]

ISSN 2321-2152
www.ijmece.com
Vol 13, Issue 1, 2025

An electronic multiplexer can be considered as a multiple-input, single-output switch, and a demultiplexer as a single-input, multiple-output switch.[3] The schematic symbol for a multiplexer is an isosceles trapezoid with the longer parallel side containing the input pins and the short parallel side containing the output pin.[4] The schematic on the right shows a 2-to-1 multiplexer on the left and an equivalent switch on the right. The wire connects the desired input to the output.

CONCURRENT ERROR DETECTABLE ADDER WITH EASY TESTABILITY

We propose a concurrent error detectable adder with easy testability. In Section 3.1, we show a design of the adder and a configuration of its adder blocks. In Section 3.2 and Section 3.3, we prove concurrent error detectability and easy testability of the adder. In Section 3.4, we discuss hardware overhead of the adder.

3.1 Configuration

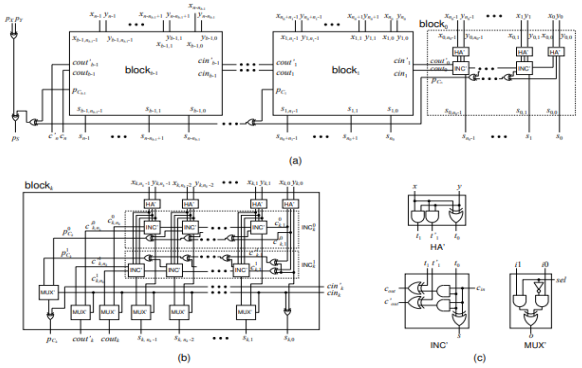


Fig. 6. Concurrent error detectable adder with easy testability (a), the design of blockk ($k \geq 1$) for the adder (b), and the gate-level designs of HA', INC' and MUX' (c).

Fig. 2(a) shows a design of the proposed adder. In addition to operands X and Y , it receives their parities pX and pY , i.e., the XORed value of X and the XORed value of Y . In addition to sum output S, it produces the predicted parity pS of the sum output and two carry outputs cn and c'_n . We restrict the block length nk to be even. Any erroneous output of the adder caused by a fault modeled as a single stuck-at fault can be detected by comparing the predicted parity pS with the parity of sum output S and comparing cn and c'_n . One inconsistency in the two input pairs, i.e., a pair of X and pX or a pair of Y and pY , can also be detected. Each addition block except block0 has two carry inputs. Each addition block has two carry outputs, and produces parity pCk of carry signals which has the same value as $c_{k,nk-1} \oplus \cdots \oplus c_{k,1} \oplus cin_k$ in case of correct operation, where $c_{k,nk-1}, \cdots, c_{k,1}$ are carry bits generated during the addition of Xk, Yk, and cink. Addition block blockk ($k \geq 1$) of the proposed adder is shown in Fig. 2(b). The addition block receives two operands Xk and Yk, and produces sum result Sk. In addition to those inputs and output, it receives two carry inputs cink and cin'_k , and produces parity of carries pCk and two carry outputs coutk and $cout'_k$. coutk and $cout'_k$ are connected to cink+1 and cin'_{k+1}, respectively. Fig. 2(c) shows gate-level designs of HA', INC', and MUX'.

Both of HA' and INC' have two carry outputs to obtain concurrent error detectability. One carry bit is used for addition, and the other is used for parity prediction. HA' and INC' are designed so

ISSN 2321-2152
www.ijmece.com
Vol 13, Issue 1, 2025

that effects of a single stuck-at fault in an INC' or its ascendant HA's never appear in both its sum output and one of its carry signals simultaneously because such a faulty operation generates an erroneous sum result and a predicted parity consistent with the erroneous sum result. In MUX' and INC', we use XOR gates to improve testability. Because XOR gates prevent masking of effects of a fault, effects of a fault can be observed easily. In each addition block, an XOR gate is placed for every bit position of INC0 k and INC1 k except the most significant position and the least significant position. For parity output pCk , two intermediate candidates are calculated in the two rows of INCs. One is p 0 Ck which is calculated as c ′0 k,nk−1 ⊕ · · · ⊕ c ′0 k,1 in INC0 k , and the other is p 1 Ck which is calculated as c ′1 k,nk−1 ⊕ · · · ⊕ c ′1 k,1 in INC1 k . The leftmost MUX' selects one of them according to the value of carry input cink, and the XOR gate at the output of the MUX' produces the parity of the carry bits including cin′ k . With the obtained pCk , the adder calculates predicted parity pS of the sum as (pX ⊕ pY) ⊕ (pCb−1 ⊕ · · · ⊕ pC0) with XOR gates because si is equal to xi ⊕ yi ⊕ ci in the correct operation.

3.2 Concurrent Error Detectability

Effects of a single stuck-at fault in an INC' or its ascendant HA' never appear in both its sum output and one of its two carry signals simultaneously as described in Section 3.1. Therefore, effect of a fault in an INC' or its ascendant HA' appears on either (1) one of the two carry signals of the INC', (2) the sum signal, or (3) all the three signals (two carries and the sum). In any case, any erroneous result caused by a fault can be detected by comparing the predicted parity pS with the parity of S and comparing cn and c ′ n . In case (1), one of the two carry signals of INC' is used for addition, and the other is used for parity prediction. When the carry signal for parity prediction is erroneous, only onebit of carry bits for the parity prediction is affected and the sum result is correct. Thus, erroneous results caused by the fault can be detected. When the carry signal for addition is erroneous, we let ck,j be the carry that is affected by the fault occurred at an INC'. Then, the error affecting ck,j induces also an error at the sum signal sk,j , and if it is not propagated in any subsequent positions, the error is detected because the carry signal affected by the fault is not used for parity prediction. On the other hand, if the erroneous value of ck,j is propagated at q subsequent carry signals ck,j+1, · · · , ck,j+q, they will also induce errors in the q sum signals sk,j+1, · · · , sk,j+q. Thus, the q+ 1 sum signals sk,j , sk,j+1, · · · , sk,j+q will be erroneous. Here, as the logic generating the carry signals c ′ k,j+1, · · · , c′ k,j+q is identical to the logic generating the carry signals ck,j+1, · · · , ck,j+q, and they have both the same carry inputs, i.e. the carry signals ck,j , ck,j+1, · · · , ck,j+q−1, the q carry signals c ′ k,j+1, · · · , c′ k,j+q will be also erroneous. Therefore, q + 1 sum signals sk,j , sk,j+1, · · · , sk,j+q will be erroneous, and q carry signals c ′ k,j+1, · · · , c′ k,j+q will be erroneous. Thus, the predicted parity (pX ⊕ pY) ⊕ (pCb−1 ⊕ · · · ⊕ pC0) will be computed by means of q erroneous signals, while the parity of the sum signals will be computed by means

ISSN 2321-2152
www.ijmece.com
Vol 13, Issue 1, 2025

of q+1 erroneous signals. Therefore, as the number of the erroneous signals used in these two parity computations differ by 1, these two parities will be different and thus the error will be detected. In case (2), a bit of the sum result is inverted and the parity of the sum result is different from the parity of the correct one. On the other hand, the predicted parity is calculated correctly because all the carry bits used for the prediction are correct. The effect of a fault is detected by comparing the predicted parity with the parity of the sum result. In case (3), all of the carry bits and the sum bit from the INC' are incorrect. Because both of the two carry bits are incorrect, there is no inconsistency among the obtained sum bits and carry bits used for parity prediction in the upper positions than the position of the faulty INC'. In the lower positions of the INC', though carry bits for parity prediction are correct, the sum bit from the INC' is inverted. Therefore, parity of the sum result is different from the predicted parity. In the adder, each adder block has two carry inputs cink and cin′ k . If there is an error on cink then sk,0 will be erroneous. Furthermore the error on cink can also induce errors on several other sum outputs (let us say at q sum outputs). Also, similarly to the arguments given in case (1), it will also create errors at q carry signals c ′ k,i. Thus, there will be q + 1 erroneous sum outputs and q erroneous carry signals c ′ k,i, and based to the same arguments as those given in case (1) these errors will also be detected. The output of an XOR or a MUX' affects only one sum or carry bit or the predicted parity. Therefore, the effect of a fault in them is detected by comparing the predicted parity with the parity and comparing two carry outputs of the adder. Note that inconsistency of one of the input operands and its parity input causes an incorrect result of the predicted parity because the parity input is used for the parity prediction. Therefore, it is also possible to detect inconsistency of X and pX or inconsistency of Y and pY when there are no faults in the adder. The proposed adder is suitable for systems using paritybased error detection as shown in Fig. 3. The parity-based error detection of arithmetic circuits was used in real designs [21], e.g. in [22]. Parities fed from a memory or a register file are used as pX and pY for operands X and Y , and the predicted parity obtained by the proposed adder is used for the parity bit of the result. Any erroneous output of the adder is detected by observing the parity checker of the system and comparing two carry outputs cn and c ′ n .

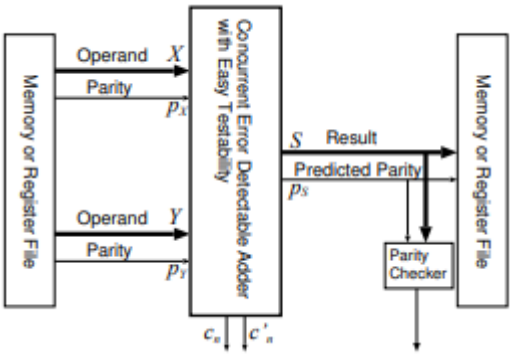


Fig. 7. Example of a datapath circuit with the proposed adder in a system using parity-based error detection

ISSN 2321-2152
www.ijmece.com
Vol 13, Issue 1, 2025

Modern developments of FPGA

A recent trend has been to take the coarse-grained architectural approach a step further by combining the logic blocks and interconnects of traditional FPGAs with embedded microprocessors and related peripherals to form a complete "system on a programmable chip". This work mirrors the architecture by Ron Perlof and Hana Potash of Burroughs Advanced Systems Group which combined a reconfigurable CPU architecture on a single chip called the SB24. That work was done in 1982. Examples of such hybrid technologies can be found in the Xilinx Virtex-II PRO and Virtex-4 devices, which include one or more PowerPC processors embedded within the FPGA's logic fabric. The Atmel FPSLIC is another such deovice, which uses an AVR processor

in combination with Atmel's programmable logic architecture. The Actel SmartFusion devices incorporate an ARM architecture Cortex-M3 hard processor core (with up to 512kB of flash and 64kB of RAM) and analog peripherals such as a multi-channel ADC and DACs to their flash-based FPGA fabric.In 2010, an extensible processing platform was introduced for FPGAs that fused features of an ARM high-end microcontroller (hard-core implementations of a 32-bit processor, memory, and I/O) with an FPGA fabric to make FPGAs easier for embedded designers to use. By incorporating the ARM processor-based platform into a 28 nm FPGA family, the extensible processing platform enables system architects and embedded software developers to apply a combination of serial and parallel processing to address the challenges they face in designing today's embedded systems, which must meet ever-growing demands to perform highly complex functions. By allowing them to design in a familiar ARM environment, embedded designers can benefit from the time-to-market advantages of an FPGA platform compared to more traditional design cycles associated with ASICs. An alternate approach to using hard-macro processors is to make use of soft processor cores that are implemented within the FPGA logic. MicroBlaze and Nios II are examples of popular softcore processors.As previously mentioned, many modern FPGAs have the ability to be reprogrammed at "run time," and this is leading to the idea of reconfigurable computing or reconfigurable systems — CPUs that reconfigure themselves to suit the task at hand.Additionally, new, non-FPGA architectures are beginning to emerge. Software-configurable microprocessors such as the Stretch S5000 adopt a hybrid approach by providing an array of processor cores and FPGA-like programmable cores on the same chip.

SIMULATION RESULT

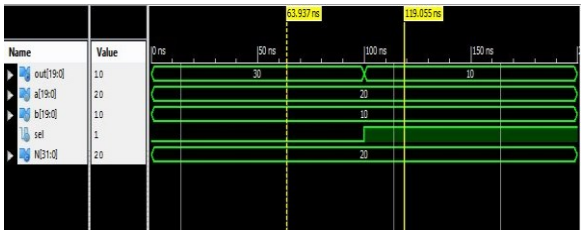
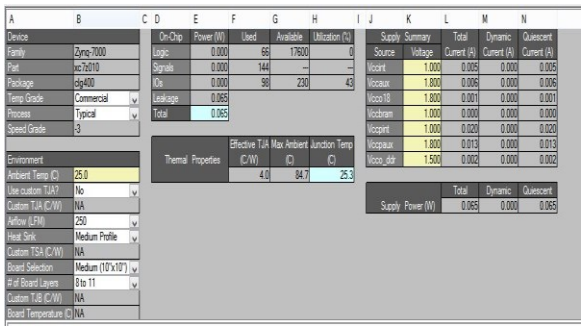


Fig. 8 Simulation result

ISSN 2321-2152
www.ijmece.com
Vol 13, Issue 1, 2025



Device		On-Chip	Power (W)	Used	Available	Utilization (%)
Family	Zynq-7000	Logic	0.000	66	17600	0
Part	xc7z010	Signals	0.000	144	--	--
Package	clg400	IOs	0.000	98	230	43
Temp Grade	Commercial	Leakage	0.065			
Process	Typical	Total	0.065			
Speed Grade	-3					

Environment	
Ambient Temp (C)	25.0
Use custom TJA?	No
Custom TJA (C/W)	NA
Airflow (LFM)	250
Heat Sink	Medium Profile
Custom TSA (C/W)	NA
Board Selection	Medium (10"x10")
# of Board Layers	8 to 11
Custom TJB (C/W)	NA
Board Temperature (C)	NA

Thermal Properties	Effective TJA (C/W)	Max Ambient (C)	Junction Temp (C)
	4.0	84.7	25.3

Supply Summary		Total	Dynamic	Quiescent
Source	Voltage	Current (A)	Current (A)	Current (A)
Vccint	1.000	0.005	0.000	0.005
Vccaux	1.800	0.006	0.000	0.006
Vcco18	1.800	0.001	0.000	0.001
Vccbram	1.000	0.000	0.000	0.000
Vccpint	1.000	0.020	0.000	0.020
Vccpaux	1.800	0.013	0.000	0.013
Vcco_ddr	1.500	0.002	0.000	0.002

Supply	Power (W)	Total	Dynamic	Quiescent
		0.065	0.000	0.065

Fig. 9 Power Report

```
----------------------------------------  ------------
  IBUF:I->O             5    0.000   0.561  a_1_IBUF (a_1_IBUF)
  LUT6:I1->O            3    0.043   0.355  s1/s1/c4/Mxor_sout_xo<0>11 (s1/s1/c4/Mxo
  LUT6:I4->O            3    0.043   0.353  s1/s2/c1/cout1 (s1/s2/c_t<0>)
  LUT5:I3->O            3    0.043   0.355  s1/s2/c3/cout1 (s1/s2/c_t<2>)
  LUT6:I4->O            3    0.043   0.353  s1/generate_N_bit_Adder[8].s3/c1/cout1 (
  LUT5:I3->O            3    0.043   0.355  s1/generate_N_bit_Adder[8].s3/c3/cout1 (
  LUT6:I4->O            3    0.043   0.353  s1/generate_N_bit_Adder[12].s3/c1/cout1
  LUT5:I3->O            3    0.043   0.355  s1/generate_N_bit_Adder[12].s3/c3/cout1
  LUT6:I4->O            3    0.043   0.353  s1/generate_N_bit_Adder[16].s3/c1/cout1
  LUT5:I3->O            3    0.043   0.355  s1/generate_N_bit_Adder[16].s3/c3/cout1
  LUT6:I4->O            3    0.043   0.353  s1/generate_N_bit_Adder[20].s3/c1/cout1
  LUT5:I3->O            3    0.043   0.355  s1/generate_N_bit_Adder[20].s3/c3/cout1
  LUT6:I4->O            3    0.043   0.353  s1/generate_N_bit_Adder[24].s3/c1/cout1
  LUT5:I3->O            3    0.043   0.355  s1/generate_N_bit_Adder[24].s3/c3/cout1
  LUT6:I4->O            3    0.043   0.353  s1/s4/c1/cout1 (s1/s4/c_t<0>)
  LUT5:I3->O            2    0.043   0.433  s1/s4/c3/cout1 (s1/s4/c_t<2>)
  LUT5:I2->O            1    0.043   0.279  Mmux_out261 (out_32_OBUF)
  OBUF:I->O                  0.000          out_32_OBUF (out<32>)
----------------------------------------
Total                        6.904ns (0.688ns logic, 6.216ns route)
                                     (10.0% logic, 90.0% route)
```

DELAY REPORT

ADVANTAGES

- Low power consumption
- Less area (less complexity)
- More speed compare regular CSLA

APPLICATIONS

- Arithmetic logic units
- High Speed multiplications
- Advanced microprocessor design
- Digital signal process

CONCLUSION

In this work, low power and energy efficient 16 bit Hybrid Variable Latency Carry Skip adder is presented using two different parallel prefix algorithms. In the proposed technique the structural complexity of the adder is reduced by Ling PPA which promotes the reduction in logic level and Kogge-Stone adder which results in faster carry generation. On comparison with the simulation results obtained from Cadence RTL compiler, Kogge-Stone is found to be effective in terms of delay than Brent – Kung adder.

FUTURE SCOPE

In this paper, we propose an SARA design. It has significantly lower power/EDP than the latest previous work when comparing at the same accuracy level. In addition, SARA has considerable lower area overhead than almost all the previous works. The accuracy-power-delay efficiency is further improved by a DAR technique. We demonstrate the efficiency of our adder in the applications of multiplication circuits and DCT computing circuits for image processing.

REFERENCES

[1] Kuldeep Rawat, Tarek Darwish and Magdy Bayoumi, “A low power and reduced area Carry Select Adder”, 45th Midwest Symposium on Circuits and Systems, vol.1, pp. 467-470, March 2002.

[2] Y. Kim and L.-S. Kim, "64-bit carry-select adder with reduced area, " Electron. Lett. vol. 37, no. 10, pp. 614- 615, May 2001.

[3] J. M. Rabaey, Digtal Integrated Circuits-A Design Perspective.Upper Saddle River, NJ: Prentice-Hall,2001.

[4] Cadence, "Encounter user guide, " Version 6.2.4, March 2008.

[5] R. Priya and J. Senthil Kumar, " Enhanced area efficient architecture for 128 bit Modified CSLA", International Conference on Circuits, Power and Computing Technologies,2013.
[6] Shivani Parmar and Kirat pal Singh,"Design of high speed hybrid carry select adder",IEEE ,2012.