PRATHYUSHA N, et al, International Journal of Computers, Electrical and Advanced Communication Engineering [IJCEACE]™ Thomson Reuters Research ID: D-1150-2018, Volume 1, Issue 13, PP: 65 - 79, JAN - JUL' 2018.

UT MULTIPLIER DEVELOPED BY USING HIGH SPEED CI-CSKA

NARREDDY PRATHYUSHA 1*, K BALA 2*

1. *M.Tech Student, Dept of ECE, Srinivasa Institute of Technology and Science, Kadapa, AP.*
2. *Assoc.Prof, Dept of ECE, Srinivasa Institute of Technology and Science, Kadapa, AP.*

ABSTRACT A carry skip adder (CSKA) structure has the high speed and very low power consumption. The speed of the structure is achieved by concatenation of all the blocks. The incrementation blocks are used to improve the efficiency of the carry skip adder structure. In existing method multiplexer logic is used, the proposed structure uses the AND-OR-Invert (AOI) and OR-AND-Invert (OAI) for the skip logic. The carry skip adder structure is realized with both fixed stage size and variable stage size where the delay is reduced, and speed is improved. A hybrid variable latency extension lowers the power consumption without affecting the speed of the circuit. The results are obtained using XILINX and it gives 42% and 37% improvements in the delay and energy of the structures. In addition to this structure, the power–delay product was low among all the structures, while having its energy–delay product was almost same as that of the conventional structure. Simulations on the proposed structure by using hybrid variable latency CSKA reduces the power consumption compared with the previous works and it produces a high speed.

Keywords— Carry Skip Adder (CSKA), AOI, OAI, energy efficient, high performance, hybrid variable latency adders

INTRODUCTION

The most basic branch of Mathematics is Arithmetic. From the Greek word arithmos, the name Arithmetic is derived. To achieve the solutions during the day to day works like counting to the advancement calculations in the field of business, science and engineering arithmetic are fundamental component. Accordingly, researchers are trying to implement for a faster and efficient Arithmetic structures in computers. The major operations in the arithmetic are Addition, multiplication, subtraction and division. To realize the procedure of scaling one quantity by another, vital arithmetic

operation is multiplication. Talking about today's engineering world, multiplier will have diverse applications because of the significance of the multiplication operations. Multiplier used in lot of Digital Signal Processing (DSP) function for example Convolution, Fast Fourier Transform, filtering. Microprocessors contain Arithmetic Logic Unit (ALU) which also uses multiplier for different arithmetic operations. Since multiplication is vital operation, it is essential for a multiplier to be fast and competent and improvement of a speed and competent multiplier has been a subject of interest over decades.

Minimizing the delay for the digital systems involves several techniques which mean choosing the optimum algorithm for the situation, this being the top level of design, then the circuit styles the topology and last the technology used to complete the design of digital circuits. Depending up on the components present, diverse of multiplier architectures are formed. In different applications diverse of multiplier architectures are used. Egyptian, Greek, Babylonian, Indus Valley and Chinese civilizations implement diverse of multiplication methods. In before days of Computers, different operations are used to perform multiplication. They are successive of addition, subtraction and shift. The trade off is exist in multiplier architecture in terms of delay, circuit complexity, area occupied on chip and power consumption. The delay involved in the multiplication algorithms which are performing in DSP applications, mainly depends on latency and throughput. Latency is the actual delay of computing a function. Latency is defined as the delay between the inputs to the device is become stable and the final result available on outputs. Throughput is defined as the measure of several multiplications can be executed in a known period of time. Multiplier is a high delay block. So, it is necessary to decrease the delay by means of various optimization methods.

Adders are a vital building block in arithmetic and logic units (ALUs), therefore speed of the processor depends on the performance of the adder. One may select any kind of adder among similar adder structures/families for optimizing the speed. There are several adder families each having unique variety of delays, power consumptions and area usages. Ripple carry adder (RCA), carry skip adder (CSKA), Carry increment adder (CIA), carry select adder (CSLA), and parallel prefix adders

(PPAs) are examples for the different adders. The structure of the Ripple Carry Adder (RCA) is extremely simple, it takes less silicon area and it uses low power but it has a worst critical path delay. The CSLA is significantly superior in terms of speed, power utilization and area usages over RCA. One of the PPA structure is carry look-ahead adders, exploit direct parallel prefix structures which produces carry as fast as probable. Presently different parallel prefix adder algorithms are existing. Each algorithm forms the different adder structure and performance is not unique for all adder structures. The Kogge–Stone adder (KSA) is also parallel prefix adder. This is one of the fastest structures but occupies more area, results more power consumption. The complexity of the PPA structure is very high when compared with the other adder structures. The efficient adder in terms of area, power consumption and delay is the CSKA. The propagation delay of this structure is smaller than the propagation delay of the RCA but the power consumption and area are similar for both of the structures. The power-delay product (PDP) of the CSKA structure is smaller when evaluated with the CSLA and PPA structures. In addition, due to the small number of transistors in the CSKA structure, it is advantageous in moderately short wiring lengths with a normal and easy layout.

In this work, based on the properties of the Carry Skip Adder (CSKA) structure, we have concentrated on lower its delay by applying some techniques to the structure of the CSKA. The modification in the CSKA structure increases the speed of the CSKA. The contributions in the proposed adder can be summarized as follows. To achieve the speed enhancement CSKA can be developed with both concatenation and the incrementation (CI-CSKA) schemes. CSKA uses AOI/OAI compound gates which are simpler carry skip logics. Finally describes the construction of an efficient CI-CSKA structure based on analytically expressions presented for the critical path delay. Next, some fundamental multiplication algorithms are given and then the Indian Vedic Mathematics algorithms are discussed.

The multiplication of two words each n bit length requires n^2 multiplications. To challenge the above multiplication process, "Urdhva Tiryakbhyam Sutra" or "Vertically and Crosswise" algorithm is used. This algorithm is used to advance the digital multiplier structure. This appears relatively

just like the general array multiplier architecture. The multiplication of a larger number (NxN, of each N bit length) with the aid of breaking it into smaller numbers of size (N/2=n, say) is demonstrated by the above sutra and the same process is repeated to obtain smaller numbers (n/2 each) until where each multiplicand size of (2x2), consequently, simplifying the entire multiplication process. The Vedic Multiplier designed with proposed adder works with the high speed and produces less delay over the Vedic Multiplier designed with conventional adder.

PROBLEM OF STATEMENT

Modern consumer electronics make extensive use of Digital Signal Processing(DSP) offering customized accelerators for the domains of multimedia,Communications etc.Traditional DSP purposes participate in a tremendous variety of arithmetic operations as their implementation is situated on computationally intensive kernals,harking back tp Fast Fourier Transform(FFT).As predicted, the performance of DSP techniques is inherently affected by picks on their design regarding the allocation and the structure of arithmetic models.Modern-day be taught persuits with in the discipline of arithmetic optimization have shown that the design of arithmetic add-ons combining operations which share knowledge,can outcome in efficiency enhancements. In an Conventional Carry Skip Adder the Full Adder performs the operation one after the other.In this stage the second Full Adder waits for the output of first Full Adder.This process will continues for all the stages.By this times taking process is more.This is the main disadvantage of this Conventional Carry Skip Adder.By this disadvantage we have proposed an Adder CI-CSKA.In this we have used 4-bit RCA.In this proposed Adder CI-CSKA the carry input will be given for each stage.So, we no need to wait until the first stage output comes to the second stage input.So, the opration performs simultaneously.By this we can reduce the time delay.When compared to both CSKA and CI-CSKA time delay is the main disadvantage.In this proposed CI-CSKA we have used concatenation and Incrementation blocks,AOI and OAI gates to get the exact results.We also notice alternative schemes utilizing Full Adders and Half Adders in the constructing blocks.

Adders

In digital computers, microprocessors and digital signal processors addition perform a vital role since it is frequently used operation. In all digital IC designs, the addition is one of the most important and frequent operations. Instruction sets for DSP's and general purpose processors perform at least one type of addition. Other instructions such as multiplication and subtraction employ addition in their operations, and their hardware structure is similar if not identical to addition hardware. In electronics, an adder or summer is a digital circuit that can be used for addition of binary numbers. Adders are also used in different parts of processors for different operations to estimate addresses and table indices.

Adders are essential in digital systems because of their significant use in other basic digital operations such as multiplication, division and subtraction. Therefore, improving performance of the digital adder would greatly advance the execution of digital operations inside a circuit composed of such blocks. When looking at other attributes of a chip, such as power or area, the designer will find that the hardware for addition will be a great contributor to these areas. It is therefore beneficial to choose the adder based on requirements to implement in a design because of the many factors it affects the overall chip. Here we begin with the basic building blocks used for addition, then go through the Ripple Carry Adder algorithm, given its advantages and disadvantages.

Carry Skip Adder (CSKA)

From examination of the RCA, the limiting factor for speed in the Ripple Carry Adder is the propagation of the carry (cout) bit. This problem is overcome in the Carry Skip Adder (CSKA, also known as the Carry Bypass Adder).

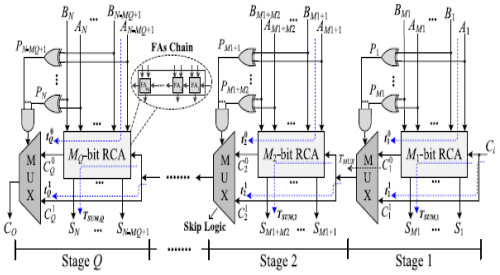

Fig 3:Structure of the Carry Skip Adder (CSKA)

The structure of an N-bit CSKA is shown in the figure 3 and it is constructed with the RCA blocks. Every stage comprises of chain of Full Adders and convey skip rationale. For a RCA that contains N fell FAs, the most exceedingly bad spread postponement of the summation of two fold numbers A

and B of N-bit length has a place with the situation where every one of the FAs are in the engendering mode. This implies the most pessimistic scenario delay has a place with the situation where

$$P_i = A_i \oplus B_i = 1 \quad \text{for } i = 1, \ldots, N \tag{12}$$

Where Pi is the propagation signal related to Ai and Bi. This demonstrates the deferral of the Ripple Carry Adder is directly identified with N. For the situation, where a gathering of fell Full Adders are in the spread mode, the yield convey of the structure is equivalent to the convey input. In the CSKA, the convey skip rationale identifies this condition, and makes the convey prepared for the following stage without sitting tight for the operation of the FA chain to be finished. The skip function is performed using the gates and the multiplexer shown in the figure 3. Based on this explanation, the N number of FAs of the CSKA is grouped in Q stages. Each stage contains an RCA block and size of the block is denoted with the Mj FAs (j = 1, . . . , Q) and a skip logic. In every stage, the inputs of the multiplexer are the carry input of the corresponding RCA block and the carry output of its RCA block. Besides, the selector signal of the multiplexer is the product of the propagation signals (P) of the corresponding stage.

2-to-1 Multiplexer

The logic symbol of the 2-to-1 multiplexer is shown in the figure 3.1.In electronics, a multiplexer (or mux) is a device and this device selects one of several analog or digital input signals and forwards this selected input signal into a single line. A multiplexer of 2^n inputs has n select lines. The select lines decide which input line to send to the output. The multiplexer is also known as data selector. Multiplexers are mainly used to enhance the amount of data that can be sent over the network within a certain amount of time and enhance the bandwidth. Boolean functions of multiple variables can also be implemented with the multiplexers. An electronic multiplexer makes it feasible for multiple signals to share single device or resource, for example one A/D converter or one communication line, as a substitute of having one device per input signal.

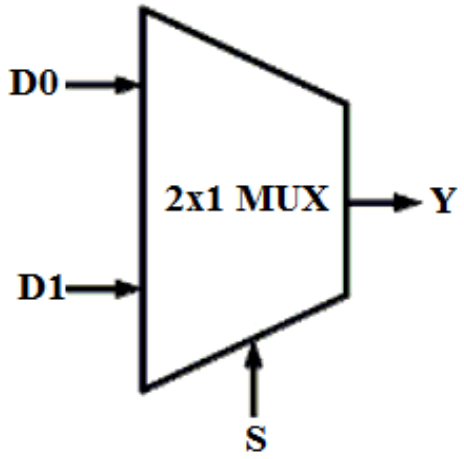

Figure 3.1: 2-to-1 Multiplexer logic symbol

An electronic multiplexer can be treated as a multiple-input, single-output switch. The schematic symbol for a multiplexer is an isosceles trapezoid. This symbol contains two parallel sides of different lengths. The longer side receives the input pins and the shorter side containing the output pin. The schematic shows a logic symbol of 2-to-1 multiplexer and it has two inputs D0, D1 and single output Y. The wire S (select line) connects the desired input to the output. The internal structure of the 2-to-1 multiplexer is shown in the figure 3.2.

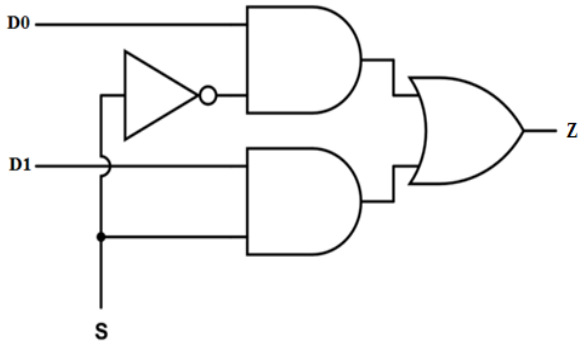

Figure 3.2: Internal structure of the 2-to-1 Multiplexer

Table 3 shows that when S=0 then Z = D0 but when S= 1 then Z = D1.

Table 4: Truth table of 2-to-1 Multiplexer

S	D_1	D_0	Z
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

In digital circuit design, the selector wires are considered as a digital value. In the case of a 2-to-1 multiplexer, if the selector signal is a logic value of 0 then D0 is connected to the output and if the selector signal is a logic value of 1 then D1 is

connected to the output. A straight forward realization of this 2-to-1 multiplexer constructed with 2 AND gates, an OR gate, and a NOT gate which is shown in the figure 3.2. In larger multiplexers, number of selector pins is identical to [$\log_2$ (n)] where n is the number of inputs. A 2-to-1 multiplexer has a Boolean equation where D0 and D1are the two inputs, S is the selector input, and Z is the output is as follows.

$$Z = (D0 \cdot \overline{S}) + (D1 \cdot S) \quad (13)$$

Bit Carry Skip Adder

The limiting factor for speed in the RCA overcomes by Carry Skip Adder. The CSKA addresses this issue by looking at groups of input bits and determines if this group has a carryout or not. This is accomplished by creating a group propagate signal ($p_{CSKAgroup}$) to establish whether the group carryin ($carryin_{CSKAgroup}$) will propagate across the group to the carryout ($carryout_{CSKAgroup}$). To investigate the operation of the whole CSKA, an N-bit adder can be divided into N/M groups, where M is the number of bits per group. Each group contains a 2-to-1 multiplexer logic to calculate the carryout ($carryout_{CSKAgroup}$). The select line for the mux is simply the $p_{CSKAgroup}$ signal, and it chooses between $carryin_{CSKAgroup}$ or $cout_4$.

Figure 3.3, which shows the hardware for a group of 4 bits (M=4) in the CSKA. There are four full adders cascaded together and each FA creates a carryout (cout) and a sum. The propagate signal from each FA comes at extra hardware cost since it is calculated by using the xor logic. For the $carryout_{CSKAgroup}$ to equal $carryin_{CSKAgroup}$, all of the individual propagates must be asserted (equations (14) and (15)). If this is true then $carryin_{CSKAgroup}$ skips past the group of full adders and equals the $carryout_{CSKAgroup}$. For the case where $P_{CSKAgroup}$ is 0, at least one of the propagate signals is 0. This implies that either a delete or generate occurred in the group. A delete signal simply means that the carryout for the group is 0 in spite of the carryin, and a generate signal means that the carryout is 1 regardless of the carryin. This is beneficial since it implies that the carryout for the group is not dependent on the carryin. The additional hardware required for the group carryout is shown in the figure 3.3 is accomplished with a xor logic for each FA, 4-input AND gate and a 2-to-1 multiplexer.

In general, an M-input AND gate and a 2-to-1 mux are required for a group of bits, including the logic to calculate the sum bits.

$P_{CSKAgroup}=p_o.p_1.p_2.p_3$ (14)

$carryout_{CSKAgroup}$ = carryinCSKAgroup $.P_{CSKAgroup}$ (15)

In examining the critical path for the CSKA, we are mostly concerned whether the carryin can be propagated ("skipped") across a group or not. Assuming all input bits approach into the adder at the same time, each group can calculate the group propagate signal (mux select line) simultaneously. Every mux then knows which signal to pass as the carryout of the group. There are two cases to regard as the multiplexer select line has been determined. In the first case, $carryin_{CSKAgroup}$ will propagate to the carryout only when the $p_{CSKAgroup}$=1 and the carryout is dependent on the carryin. The second case is that the carryout signal of the most significant adder will become the group carryout which means $p_{CSKAgroup}$=0. The carryout is independent of the carryin.

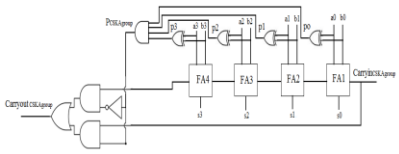

Figure 3.3: Internal structure of the 4-bit CSKA

-Bit Carry Skip Adder

The structure of a 16-bit CSKA is shown in the figure 3.4. In addition to the chain of FA in each stage, carry skip logic is placed. If we isolate a particular group, the second case carryout (cout4) always takes longer because the carryout signal must be calculated through logic. The first case ($carryin_{CSKAgroup}$) requires only a wire through which the signal propagates. Looking at the whole structural design, however, this second case is part of the critical path for only the first CSKA group. Since the second case is not dependent on the group carryin, every group in the CSKA produces carryout in parallel. If a group needs its carryin ($p_{CSKAgroup}$=1), then it must stay until it arrives after being calculated from a previous group. In the worst case, a carryout have to be calculated in the first group, and every group afterwards needs to propagate this carryout. When the last group receives this propagated signal, then it can

calculate its sum bits. Figure 3.4 shows a 16-bit CSKA with 4-bit groups and Figure 3.5 shows a darkened line indicating the critical path of the signals in the 16-bit CSKA.

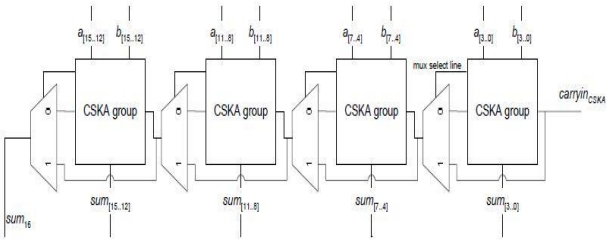

Figure 3.4: A 16-bit Carry Skip Adder =16, M=4

Carry Skip Adder Delay

If we consider a 16-bit CSKA with 4-bit groups, with each group containing a 4-bit Ripple Carry Adder to obtain final summation, then the worst case propagation delay throughout this adder is expressed in equation (16).

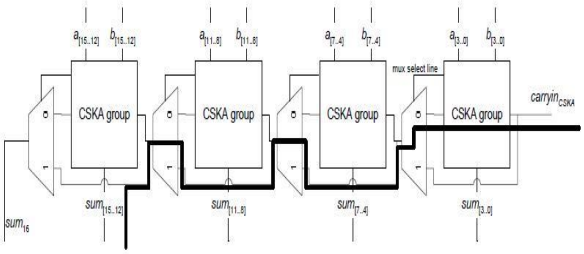

Figure 3.5: Critical path through 16-bit CSKA

In this equation, $t_{RCAcarry}$ and t_{RCAsum} are the delays to compute the carryout and sum signals of an RCA, respectively. Each CSKA group has 4 bits, so the delay through the primary group has 4 RCA carryout delays. The carryout obtained from the first group potentially propagates through 3 multiplexers, where one multiplexer delay is considered as $t_{muxdelay}$. Finally, when the carryout signal reaches the final group, the sum for this final group can be calculated. This is represented by the final two components of the equation (16)

$$t_{CSKA16} = 4*t_{RCAcarry} + 3*t_{muxdelay} + 3*t_{RCAcarry} + t_{RCAsum} \quad (16)$$

For equation (16), there are primarily some assumptions about the delay through the circuit, we assume in the first CSKA group that the group propagates signal is calculated before the carryout of the most significant adder. Thus, the multiplexer for this first group is waiting for the carryout. For the final CSKA group, we assume that sum15 takes longer time to be calculated than for sum16 to be calculated. Because carryin for final group of the CSKA is known, the delay for sum16 is the delay of the mux; for sum15 it is a delay of $3*t_{RCAcarry} + t_{RCAsum}$ (carry will be 3 times

ripples through the adder before the last sum bit can be calculated). For an N-bit CSKA, the critical path equation is expressed in Equation (17). M represents the number of bits in each group. There are N/M groups in the adder, and every multiplexer in this group except for the last one is in the critical path. As in equation (16), equation (17) assumes that each group contains a ripple carry adder.

$$t_{CSKAN} = M*t_{RCAcarry} + (N/M - 1)*t_{muxdelay} + (M - 1)*t_{RCAcarry} + t_{RCAsum} \quad (17)$$

From a VLSI design perspective, this adder shows enhanced speedup over a RCA without much area increase. The additional hardware comes from the 2-to-1 mux and group propagates logic in each group. One drawback to this structure is that CSKA delay is still linearly dependent on the width of the adder, therefore adders with large size where speed is important, the delay may be intolerable. Also, there is a long wire in between the groups that $carryout_{CSKAgroup}$ will propagated through this long wire. This long wire path starts at the carryout of the first CSKA group and ends at the carryin to the last CSKA group. This signal also needs to travel through (N/M -1) multiplexers, and these will introduce long delays and signal degradation if pass gate multiplexers are used. If buffers are essential in between these groups to reproduce the signal, then the critical path is lengthened. An example of a worst case delay input pattern is as follows. For a 16-bit CSKA with 4-bit groups is where the input operands are 1111111111111000 and 0000000000001000. This produces a carryout in the first group that skips through the second and third groups and reaches the final stage of the CSKA. In the final stage of the CSKA, carryin ripples through to the final sum bit (sum15). To determine the optimal speed for this adder structure, one needs to find the delay through a multiplexer and the carryout delay of a FA. It is one of these two delays that will rule the delay of the whole CSKA. For short adders ($\leq$ 16 bits), the $t_{carryout}$ of a FA will probably dominate delay, and for long adders the long wire that skips through stages and multiplexers will probably dominate the delay.

Proposed CSKA Structure

The Conventional CSKA introduced with the concatenation and the incrementation schemes. So it is known as CI-CSKA and it provide to use the simpler carry skip logic.

Here conventional structure uses 2-to-1 multiplexer logic replaces AOI/OAI compound gates. The number of transistors present in the gates is fewer, then gates, area, delay, lesser and consume slighter power in comparison of the 2-to-1 multiplexer. The CI-CSKA arrangement is appeared in the figure 4.

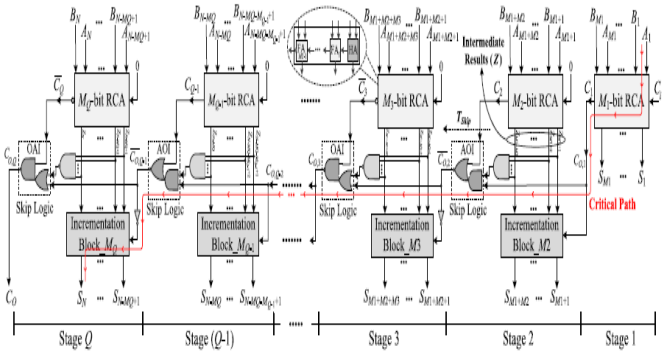

Figure 4: CI-CSKA Structure

In the proposed adder arrangement, through the skip logics the carry is propagated and complement of the carry is obtained as an output carry from the skip logic. Therefore, the skip logic output at even positioned stages, carry is produced which is complemented. The proposed adder has a significant lower propagation delay and to some extent area is smaller in comparison with those of the conventional adder. The compound gates (AOI or OAI) consumes smaller power than that of the multiplexer but the power consumption of the proposed adder is slightly higher than the power consumption of existing CSKA. This is because number of gates increase in the proposed adder. This is intern, results a higher wiring capacitance (in the noncritical paths).

Now, the construction of the proposed adder CI-CSKA described more in detail. The inputs of the adder represented as A and B of each N bit length. The CI-CSKA constructed in Q number of stages. Each stage of proposed adder is constructed with an RCA block and stage size of each block is denoted with M_j (j = 1,. . ., Q). The carry input of the first RCA block in this proposed structure is C_i and remaining all RCA blocks carry input is zero (concatenation of the RCA blocks). Therefore, all the RCA blocks perform their tasks simultaneously. In this construction, when the initial block computes the output sum bits (i.e., SM1, . . . , S1), and C1, for the specified input bits, the remaining blocks also simultaneously evaluate the intermediate results [i.e., {ZK j+Mj , . . . , ZK j+2, ZK j+1} for K j= $\sum_{r=1}^{j-1} Mr$ (j=2,….,Q)], and Cj signals also. The first stage of the proposed adder is only one block which is Ripple Carry adder.

The incrementation block and RCA blocks are used in the remaining stages 2 to Q. The incrementation block is used to calculate the final sum bits of the stage. The RCA block produces intermediate results. The incrementation block uses the intermediate results and output carry of the preceding stage to compute the final sum bits of the stage. The incrementation block is shown in the figure 4.1. A chain of half-adders are used in the incrementation block.

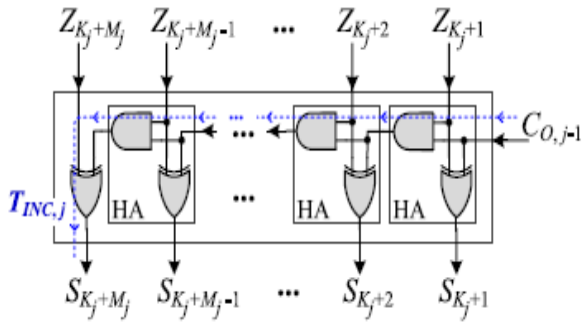

Figure 4.1: Internal structure of the j th incrementation block, $K_j = \sum_{r=1}^{j-1} Mr$ (j=2,….,Q)

In addition, to achieve the stage carry output the carry output of the incrementation block is not used because to reduce delay considerably. The carry output is determined by skip logic for the particular stage (j th stage) ($C_{O,\ j}$), based on the corresponding jth stage intermediate results and the the previous stage ($C_{O,\ j-1}$) output carry and the corresponding RCA output carry (C_j). The determination of $C_{O,j}$ the cases are evaluated as follows. When C_j is equivalent to one, $C_{O,\ j}$ is also one. In another case value of C_j is equivalent to zero and if the intermediate values product is equivalent to one (zero), then $C_{O,\ j}$ is equivalent to $C_{O,\ j-1}$ (zero). The motivation for implementing in AOI and OAI skip logics because these gates in standard cell libraries are having the inverting functions. The inverter gate produces more delay, so using the AOI and OAI skip logics use of inverter gate is not required finally delay also reduced. As represented in figure 4, AOI and OAI skip logics used alternatively, means if an AOI is used in the second stage, OAI gate is used in third stage.

The point reveal is that the use of the AOI and OAI skip logic in the conventional CSKA results the critical path delay increases significantly. This originates a fact that, AOI and OAI skip logics unable to bypass zero input carry until to obtain zero input carry from corresponding stage RCA. To resolve this problem, RCA input carry is zero in the CI-CSKA (concatenation scheme). By using this scheme each stage of the RCA does not necessitate to stay for the output carry of the preceding stage. So all

the RCA blocks output carries are obtained in parallel.

CONCLUSION

In this work, an adder structure is proposed called CI-CSKA, which exhibits lower delay and higher speed compared with those of the conventional CSKA. The speed enhancement is achieved by modifying the structure through the concatenation and incrementation techniques. In addition, AOI and OAI compound gates are used for the carry skip logics. In the application, Vedic Multiplier is developed by using the efficient sutra named as Urdhva Tiryakbhyam. The Vedic multiplier is developed by using CI-CSKA adder results in the speed improvement and reduces the delay compared with the same multiplier developed by using CSKA.

REFERENCES

[1] R. Zimmermann, "Binary adder architectures for cell-based VLSI and their synthesis," Ph.D. dissertation, Dept. Inf. Technol. Elect. Eng., Swiss Federal Inst. Technol. (ETH), Zurich, Switzerland, 1998.

[2] J. M. Rabaey, A. Chandrakasa, and B. Nikolic, Digital Integrated Circuits: A Design Perspective, 2nd ed. Englewood Cliffs, NJ, USA: Prentice-Hall, 2003.

[3] Nan Gate 45 nm Open Cell Library. [Online]. Available: http://www.nangate.com/, accessed Dec. 2010.

[4] Y.-S. Su, D.-C. Wang, S.-C. Chang, and M. Marek-Sadowska, "Performance optimization using variable-latency design style," IEEE Trans.Very Large Scale Integr. (VLSI) Syst., vol. 19, no. 10, pp. 1874–1883, Oct. 2011.

[5] Jagadguru Swami Sri Bharati Krishna Tirthji Maharaja, "Vedic Mathematics", Motilal Banarsidas, Varanasi, India, 1986

[6] Shamim Akhter, "VHDL Implementation of Fast NXN Multiplier Based on Vedic Mathematics", Jaypee Institute of Information Technology University, Noida, 2013.07 UP, INDIA, 2007 IEEE

[7] Jagadguru Swami Sri Bharati Krishna Tirthaji Maharaja "Vedic mathematics", 1965

[8] S.S kerur "Implementation of Vedic multiplier for DSP", International journal of Technology and Engineering system, Jan 2011

[9] Umesh Akare "performance and evaluation and synthesis of Vedic multiplier", National conference on international paradigms in Engineering & Technology (NCIPET-2012), Jan 2012

[10] Pushpalata Verma, K.K Mehta" Implementation of an Efficient Multiplier based on Vedic Mathematics using EDA Tool", June 2012

AUTHOR DETAILS

NARREDDY.PRATHYUSHA received her B.Tech degree from Sri Venkateswara Institute of Science and Technology, Kadapa, India(Affiliated to JNTUA Anantapur) Department of ECE. She is pursuing M.Tech in Srinivasa Institute of Technology and Science, Ukkayapali, Kadapa, AP.

Mr.K.Bala is currently working as an Associate Professor in ECE Department, Srinivasa Institute of Technology and Science, Ukkayapalli, Kadapa, India. He received his M.Tech from Sri Kottam Tulasi Reddy Memorial College of Engineering Kondair, Mahaboobnagar, A.P, India.