# High-Speed and Energy-Efficient MAC design using Vedic Multiplier and Carry Skip Adder

**Krutika Kashinath Soman[1], D. Praveen Kumar[2]**

[1]M.Tech Student, Dept. of Electronics and Communication Engineering, MREC, Hyderabad, Telangana, India
[2] Associate Professor, Dept. of Electronics and Communication Engineering, MREC, Hyderabad, Telangana, India

**Abstract -** *In this paper, we present a High-speed energy efficient Multiply and Accumulator(MAC) structure using the Vedic multiplier and various carry-skip adders. In this paper, first, we examine the area and delays of three types of Carry Skip Adder designs. The first design is conventional CSKA using MUX. Second CSKA structure the by applying concatenation and incrementation techniques to provide better efficiency than that of conventional CSKA (Conv-CSKA) structure. Also, instead of using multiplexer logic, the CI-CSKA makes use of AND-OR-Invert (AOI) and OR-AND-Invert (OAI) compound gates for the skip logic (CI-CSKA). The third CSKA design which implements variable stage size(VSS) CSKA and with the 8-bit parallel prefix adder (PPA) to reduce the delay. These CSKA adders are used in MAC implementations along with the 16x16 Vedic multiplier and the results are studied. Verilog HDL is used for designing the circuits. The Xilinx Xpower tool is used to study the power and thus the energy efficiency is also calculated. The synthesis and simulation results are obtained using Xilinx ISE 14.7.*

***Key Words***: MAC units, Vedic Multiplier, Carry skip Adders.

## 1. INTRODUCTION

Due to the rapid growth of portable electronic systems like a laptop, calculator, mobile etc., the low power devices have become very important in today's world. Low power and high-throughput circuitry design are playing the challenging role of VLSI designer. For real-time signal processing, high-speed and energy efficient MAC unit is necessary to achieve high performance in DSP systems. The functioning of MAC unit involves performing multiplication and accumulation processes repeatedly and provide accurate results. It also consists of reset and clock signals to control its operation. Many researchers have been focusing on the design of advance MAC unit architectures. The previous MAC output and the current output will be added in the Adder section. The MAC unit consists of a multiplier unit, an adder unit and the results are stored in an accumulate unit. MAC units find their applications in multiple digital devices such as microprocessors, DSPs, logical units etc. as the speed of the overall system is based on the speed of MAC unit. The applications of MAC unit are in Nonlinear Computation like Discrete Cosine or wavelet Transform (DCT), FFT/IFFT. Since they are basically executed by insistent application of multiplication and addition, the entire speed and performance can be computed by the speed of the addition and multiplication taking place in the system.

The delay and Critical delay are contributed majorly due to the multiplication process and the propagation delay is caused due to parallel adders used in Adder stage. The speed of MAC units is highly affected by multiplication process. Thus, there is a dire need to reduce the delay in multiplication process. Designing of MAC unit that caters to both delay and energy issues has, henceforth, become the need of the hour. Vedic Mathematics was formulated by Swami Bharati Krishna Tirthaji Maharaja from the ancient Indian scriptures (Vedas) after extensive research on the Vedas. Vedic Mathematics is built on the sixteen principles or „sutras‟. Thus, integration of multiplication with Vedic Mathematics can lead to wonders in various domains of engineering, such as Digital Signal Processing.

## 2 Prior work

As the focus of this paper is on the MAC structure with Vedic multiplier and hybrid -CSKA design, first the basic related work to this MAC are reviewed.

### 2.1 Multiplier Unit

Multiplication function is carried out in three steps as discussed further. The three stages are partial product Generation (PPG), partial product addition(PPA), and final conventional addition. Speed improvement in MAC can be achieved by reducing the delay in creation of partial products. MAC design using conventional multiplier is replaced by Vedic multiplier using Urdhava Triyagbhayam sutra [1]. Multiplication is the fundamental operation of MAC unit. The major issues in the design of a multiplier unit are its area, speed, latency and power consumption and dissipation. To avoid them we implement fast multipliers in most of the DSP and networking applications. Thus, the critical path delay of multiplier should be reduced for improving the performance of the complete MAC system.

The basic operational block in MAC unit is the multiplier that determines the critical path and the delay. The (log2N + 1) partial products are produced by 2N-1 cross products of different widths for N*N. The partial products are generated by Urdhava sutra is by Criss-Cross Method [2]. The maximum number of bits in partial products will lead to the Critical path. The MAC unit with area efficiency at high-speed processing with reasonable power consumption, for computation of squares, is studied in [3]. The speed enhancement is achieved in a Vedic multiplier by reducing the number of computations thus reducing the overall complexity of the complete system. Thus, computations decrease with Vedic multiplications by removing the shift operation performed in a traditional multiplier. Thus, Vedic multiplier provides multiple advantages.

### 2.2 Addition Block

The existing architecture uses an RCA and Conventional multiplier unit as shown in Fig.1.

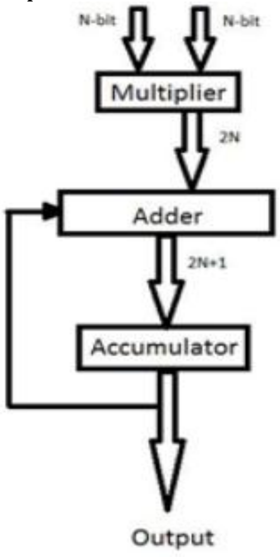

Fig -1: Existing MAC unit

The proposed architecture uses the combination of Vedic multiplier and Proposed VSS-CSKA with Brent Kung Adder for implementing high speed and low power MAC.

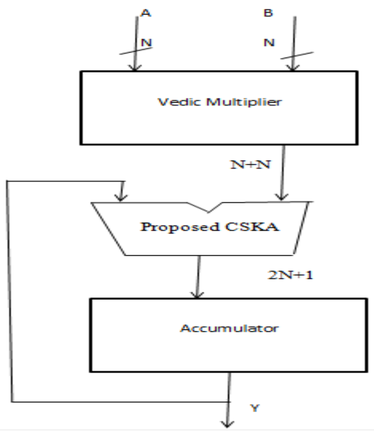

Fig -2: Proposed MAC unit

### 2.3 Accumulator

Accumulator though a basic and common method plays a vital role in many of VLSI and applications. The main purpose of an accumulator is used to add the multiplied numbers to the previous accumulator value. Vedic Multiplier has been used to strengthen the action of the MAC Unit.

## 3. PROPOSED MAC STRUCTURE

The design of MAC architecture consists of 3 sub designs.

- Design of 16×16-bit Vedic multiplier.
- Design of adder using different types of CSKA logic.
- Design of accumulator which integrates both multiplier and adder stages.

### 3.1 Vedic Multiplier

A new technique is known as "Urdhva-Tiryakbhyam–Vertically and crosswise." is used in the place or a normal multiplier. Vedic multiplier finds its applications not only in decimal multiplication but also for the binary type. The process mainly consists of the generation of partial products in parallel and then the addition is performed simultaneously [4]. This algorithm can be used for 2x2, 4x4, 8x8....N×N bit multiplications. The Vedic multiplier does not depend upon the processor clock frequency as their sums and carries are calculated in parallel. Thus, there is no need to increase the clock frequency. If the clock frequency is increased the power dissipation also increases considerably. Hence by using this Vedic multiplier technique, we can reduce the power dissipation. By using the Vedic multiplier, the area and delay are reduced significantly when compared to that of other multipliers. This is because the shift operation can be ignored and the product is calculated in single step resulting in an area and delay efficient systems.

#### 3.1.1 2×2 Vedic Multiplier Block

To explain this method let us consider 2 numbers with 2 bits each and the numbers are A and B where A=a0a1 and B=b0b1 as shown in the below line diagram. First, the least significant bit (LSB) bit of final product (vertical) is calculated by taking the product of two least significant bit (LSB) bits of A and B is a0b0. In the Second step, the products are calculated in a crosswise manner such that the least significant bit (LSB) of the first input is multiplied with the MSB of the second input and vice-versa. The output generated from each multiplication yields two bits containing a sum bit which is used in the result and carry bit, which is forwarded as Carry Input ($C_{in}$) for next bits. Thus, the Output Carry forms the fourth bit and the result is obtained.

s0 = a0*b0 (1)
c1s1 = a1*b0 + a0*b1 (2)
c2 s2 = c1 + a1*b1 (3)

The obtained result is given as c2 s2 s1 s0. A 2×2 Vedic multiplier block is implemented by using two half adders and four two input and gates as shown in below Figure 3.

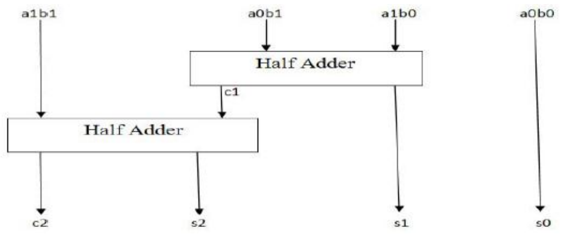

Fig -3: Block Diagram of 2×2 Vedic Multiplier

### 3.1.2 4x4 Vedic Multiplier Block

In this section, now we will discuss 4x4 bit Vedic multiplier. For explaining this multiplier let us consider two four bit numbers are A and B such that the individual bits can be represented as the A3A2A1A0 and B3B2B1B0. The procedure for multiplication can be explained in terms of line diagram shown in Fig-4.

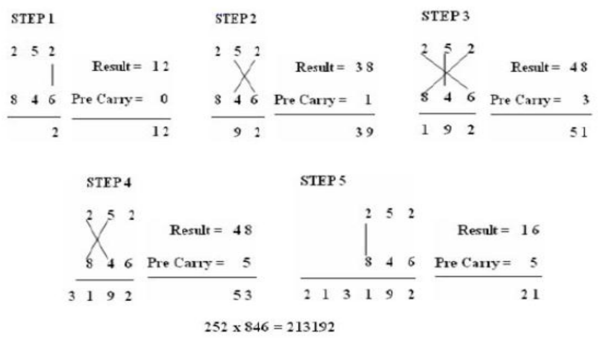

Fig -4: Multiplication of two decimal numbers

The final output can be obtained as the C6S6S5S4S3S2S1S0. In Vedic multiplier, the LSB bit S0 is obtained easily without any shift operations by multiplying the LSBs of the multiplier and the multiplicand as the partial products are calculated in parallel and thus resulting in the reduction of delay as an increase in the number of bits. Once all the steps are performed the output values of Sum and Carry is calculated. The same process is repeated in which the previous stage carry propagates to the next stage.

S0 = A0B0 (4)
C1S1 = A1B0 + A0B1 (5)
C2S2 = C1 + A0B2 + A2B0 + A1B1 (6)
C3S3 = C2 + A0B3 + A3B0 + A1B2 + A2B1 (7)
C4S4 = C3 + A1B3 + A3B1 + A2B2 (8)
C5S5 = C4 + A3B2 + A2B3 (9)
C6S6 = C5 + A3B3 (10)

For higher understanding, observe the diagrams for 4x4 as shown below figure nine and among the block diagram 4x4 entirely there are four 2x2 Vedic multiplier modules, and 3 ripple carry adders that are of 4-bit size are used the four-bit ripple carry adders are used for addition of two four bits and likewise totally four are use at intermediate stages 3 of multiplier. The output carry generated from this RCA unit is rippled to the next ripple carry adder the other two inputs are zero. The arrangement of the ripple carry adders are shown in below block diagram which can reduces the computational time such that the delay can be decrease.

The Ripple Carry Adders are shown in below block diagram which can reduce the computational time such that the delay can be decreased.

The mathematical illustration of the Vedic multiplier and its hardware structure is as shown in Fig. 5. The product of two 8-bit inputs numbers using 4-bit multiplier is as discussed below.

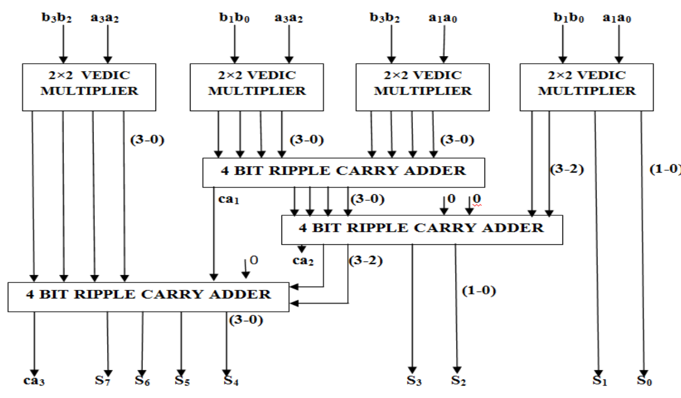

Fig -5 Structure of 4x4 bit Vedic Multiplier

Let us denote the High and Lower bits of inputs A and B as AHAL and BHBL where each corresponds to 4 bits of an 8-bit number. In Vedic multiplier, we are using the Urdhava Tiryakbhyam (vertically and crosswire) method to find the product of inputs A and B.

AH AL
BH BL

The product terms are formed as below

(AH * BH) (AH * BL + BH * AL) (AL * BL).

Thus, four 4-bit multipliers are required and two adders will be required to add the partial products and thus 4-bit intermediate carry generated. Since the product of a 4 x 4 multiplier is 8 bits, in each step the 4 LSB bits belong to the product and the remaining 4 bits will be carried to the next step. The same process is continued for 3 steps in this example.

Similarly, the 16-bit multiplier would require 4 8 x 8 multipliers and three 16 bit adders with 8-bit carry. Thus, we see that the multiplier is highly modular in nature. Thus, proving regularity and scalability. The structure of a 16×16 Vedic multiplier is shown in Fig. 6. This is developed using the 8×8 Vedic multiplier and the design is implemented using Verilog HDL in Xilinx ISE.

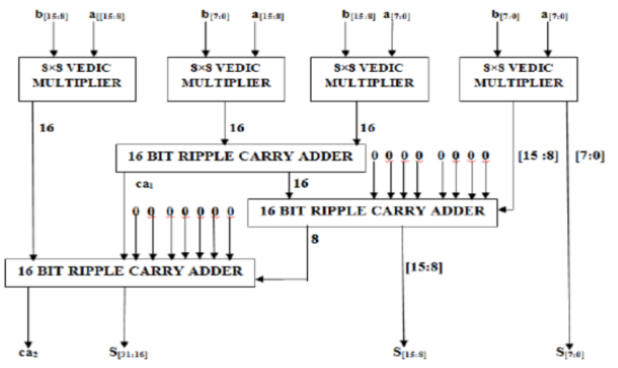

Fig -6 Structure of 16x16 Vedic Multiplier

## 4 Design of Various Carry-Skip Adders

1) Designing a CSKA structure using the conventional CSKA (Conv-CSKA) structure.

2) Designing a modified CSKA structure by combining the concatenation and the incrementation schemes to the conventional CSKA (Conv-CSKA) structure for increasing the speed and reducing the area of the adder. This modification enables us to use easier carry skip logic implementation based on the AOI/OAI compound gates which were obtained using multiplexer in Conv-CSKA.

3) Designing a hybrid variable latency CSKA structure based on the extension of the suggested CSKA, by replacing some of the middle stages in its structure with a Kogge-Stone PPA, which is modified in ].

### 4.1 Conv-CSKA structure

The conventional structure of the CSKA consists of stages containing a chain of full adders (FAs) (RCA block) and 2:1 multiplexer (carry skip logic) as shown in Fig-7. The RCA blocks are connected to each other through 2:1 multiplexers, which can be placed into one or more level structures [5]. The CSKA configuration (i.e., the number of the FAs per stage) has a significant impact on the speed of this type of adders. Multiple types of research have been performed to find the optimal number of Full Adders. The techniques presented in [6] make use of VSSs to lessen the delay of adders based on a single level carry skip logic. These techniques, however, cause area and power increase considerably and less regular layout.

Alioto and Palumbo [6] proposed a simple strategy for the design of a single-level CSKA. This method is based on the Variable Stage Size technique where the near-optimal number of the Full Adders are determined based on the skip time (delay of the multiplexer), and the ripple time which is the time taken by the carry to ripple to the next stage. The goal of this method is to reduce the critical path delay by considering a non-integer ratio of the skip time to the ripple time on contrary to most of the previous work. In the prior works, the focus was mainly on the speed whereas the power consumption and area in CSKA were neglected. Also, the multiplexer delay and critical path delay were not improved considerably despite a reduction in the overall delay.

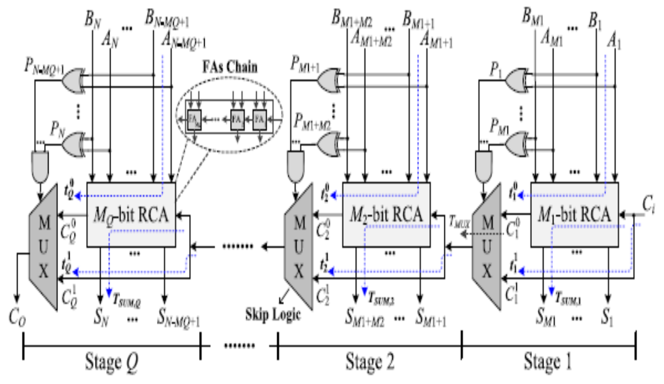

Fig -7 Conventional Carry Skip Adder

### 4.2 Structure of modified CI-CSKA

The structure is based on implementing the concatenation and the incrementation techniques combinedly [7] along with the traditional-CSKA structure, and hence, it is abbreviated as CI-CSKA. Thus, by replacing the multiplexer with AOI/OAI logic the skip logic can be easily implemented (Fig. 8). These gates consist of a lesser number of transistors and have a lower delay, area, and lesser power consumption compared with that of the 2:1 multiplexer [8]. Because of the use of use of AOI and OAI, the carry which is propagating through this skip logics becomes complemented. Thus, at the output of the skip logic in even stages, the complemented carry is generated. The structure has a significant lower propagation delay with a little smaller area compared with that of the conventional one. It is to be noted that though the power consumption of the AOI (or OAI) gate is lesser than that of a multiplexer, power consumption of the CI-CSKA is slightly higher than that of the conventional one. This is a result of an increase in the number of the gates causing higher wiring capacitance, especially in the noncritical paths.

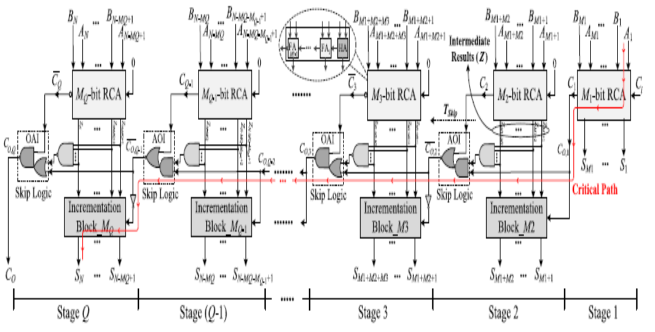
Fig -8 Modified CI-CSKA Structure

Both the compound gates - AOI and OAI are used in the skip logic because of the presence of these inverting functions of these gates in standard cell libraries. This way, the need for an inverter gate can be avoided in turn avoiding the further increase in power consumption and delay. As shown in Fig. 5, if the skip logic starts with an AOI the next skip logic should use OAI compound gate. In addition, it is important to mention that the use of the skipping structure in the CI-CSKA structure increases the delay of the critical path significantly. This is due to the fact that, in the CI-CSKA, the skip logic formed by use of AOI or OAI compound gates is unable to bypass the zero-carry input until the zero-carry input propagates from the previous RCA block. To solve the above-mentioned problem, in the proposed hybrid CSKA structure, we an RCA block with a Cin of 0 is used by using the concatenation method. The output of previous stages is calculated in parallel as the RCA block need not wait for the COUT of the previous stage.

## 4.3 Hybrid Variable Latency CSKA Structure

The principle idea behind using VSS CSKA structures is based on reducing the critical path delay considerably when compared to that of the Fixes Stage Size structure of CSKA. This doesn't allow us to use the slick time for further supply voltage scaling. To provide variable latency feature, we have replaced some of the middle stages of nucleus stage with parallel prefix adder. As the Conv-CSKA structure has lower speed than that of the variable Stage Size structure we are not focusing on the conventional structure. The hybrid variable latency CSKA structure is as shown in Fig. 9 where a Mp-bit modified PPA is used for the pth stage, called as nucleus stage. The nucleus stage, which has the largest size and exhibits the greatest delay among all the stages, is present in both SLP1 and SLP2. Thus, replacing the nucleus stage by the PPA results in a reduced delay of the longest off-critical paths. Hence it can be concluded that the use of the fast PPA helps in increasing the available slack time in the variable latency structure. Since the input bits of the PPA blocks are also used in the predictor block, this block becomes parts of both SLP1 and SLP2 structures.

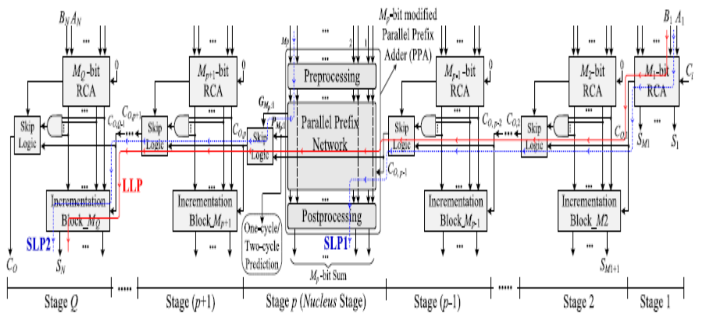
Fig -9 Hybrid variable stage size CI-CSKA

In the proposed hybrid structure, the Brent-Kung adder is used in prefix as the nucleus stage (Fig. 9). The most significant advantage of this adder compared with other prefix adders is that in this structure, by using forward paths, the longest carry is calculated sooner when compared with other or intermediate carries, which are computed only by backward paths. Also, the fan-out of this adder is less than other parallel adders, while its wiring length is smaller [9]. An additional advantage of using this type of PPA is that it has a simple and regular layout.

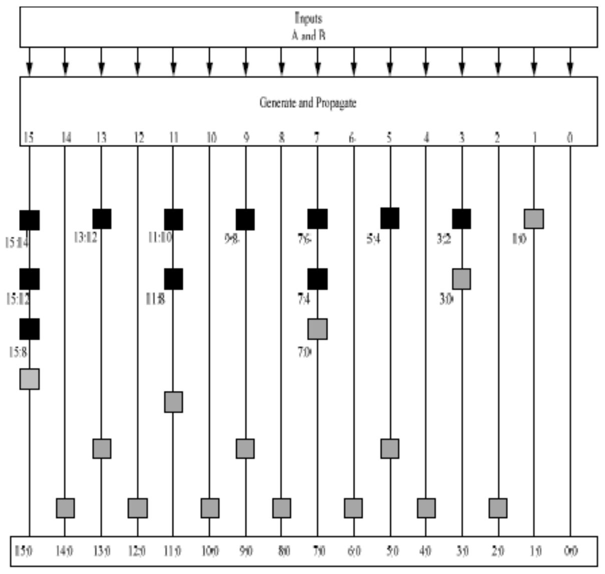

Fig -10 16 bit Brent Kung adder

In the pre-processing level, the propagate signals (Pi) and generate signals (Gi) for the inputs are determined. In the next level, using Brent–Kung parallel prefix network, the longest carry (i.e., G8:1) of the prefix along with P8:1 is calculated, Here, P8:1 is the product of the all propagate signals of the inputs. These are usually calculated sooner than other intermediate signals in this network. The largest propagate signal, P8:1 is used in the skip logic to determine if the $C_{OUT}$ of the previous stage (i.e., $CO_{p-1}$)

should be skipped or be considered in the processing. This signal is also exploited as the predictor signal in the variable latency adder. It should be noted that since these operations are performed in parallel with other stages, the delay is reduced considerably. In the case, where P8:1 is one, $CO_{p-1}$ should skip this stage forecasting that some critical paths are triggered. On the other hand, when P8:1 is 0, $CO_p$ is equal to the largest Generate signal which is G8:1. In addition, the critical path will not be activated in this case. Once the parallel prefix network completes its calculations the intermediate carries $CO_{p-1}$, are calculated Lastly, in the third level i.e. in postprocessing level, the final output is calculated. This implementation is based on the idea of concatenation and incrementation concepts used in the CI-CSKA discussed in B. In addition, like the CI-CSKA structure, the initial point of SPL1 is the first input bit of the first stage, and the final point of SPL2 is the last bit of the Output Sum of the incrementation block of the stage Q. The steps for defining the sizes of the stages in the hybrid variable latency CSKA structure are like the ones discussed in the previous Section. Since the PPA structure is more efficient when its size is equal to an integer power of two, we should select a larger size for the nucleus stage [9]. This implies that the third step discussed in that section is modified. The larger number of bits, compared with that of the nucleus stage in the CI-CSKA structure, leads to the reduction in the number of stages as well as smaller delays for SLP1 and SLP2. Thus, the slack time increases further.

## 5 Accumulator

The accumulator is designed to store cumulative addition of MAC unit, it is a group of registers which are designed for this. The reset pin which is used for resetting is also used in an accumulator in which when the reset value is high the content of the accumulator becomes zero and when the reset is not equal to zero, the accumulator starts accumulating the summation. The output of the Vedic multiplier and the previous content of the accumulator are provided as inputs to Accumulator.

## 6 Simulations

In this section, first, we will see the RTL Schematics of the designed MAC unit using the Vedic multiplier and various CSKA implementation and their simulation results. Then we can compare the delays and area utilized by the proposed structures. The Vedic multiplier considered here are the 16x16-bit Vedic multiplier and the adder designed here are 32-bit adders. These were designed using Verilog HDL and simulated using Xilinx ISE 14.4. The RTL schematics of the MAC unit and all the submodules like Vedic multiplier and all the CSKA structures are given below.

### RTL Schematic & Simulation

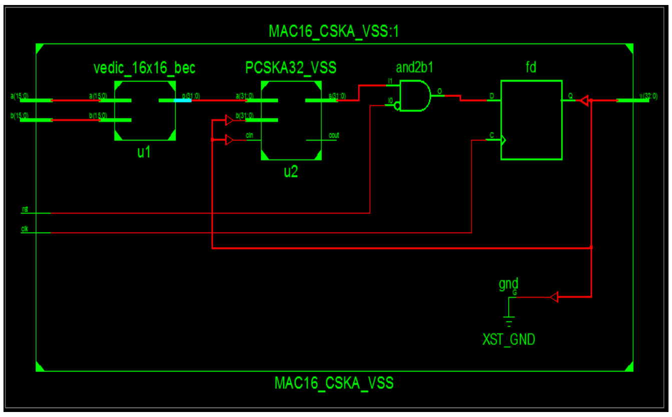

Fig.10. RTL Schematic of MAC with hybrid CSKA

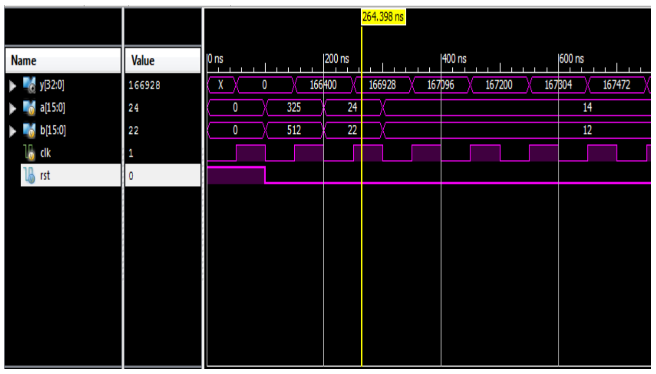

Fig. 11. Simulation of MAC unit hybrid CSKA

**Delay, Area and Energy Comparisons of CSKAs:**

The delay and its logic/route efficiency along with Number of paths and energy efficiency are noted in below Table-1.

| Design parameters | MUX | AOI-OAI |
|---|---|---|
| Delay(ns) | 54.310 | 37.004 |
| Logic/route % | 60.01% / 39.9% | 61.6% / 38.4% |
| Number of paths | 21324 | 3201 |

Table 1. Comparison of MUX and AOI-OAI implementations

The delays of various CSKAs are compared below. It is noted that the Conv-CSKA occupies less no LUTs but the delay is high.

In the CI-CSKA structure occupies a slightly higher number of LUTs but the delay is considerably less.

| Adder Design | No of slices | No of LUT's | Delay (ns) | Energy (nJ) |
|---|---|---|---|---|
| Conv-CSKA | 51 | 88 | 54.94 | 4.34 |
| CI-CSKA | 57 | 101 | 34.67 | 2.74 |
| Proposed VSS-CSKA | 55 | 98 | 21.80 | 1.72 |

Table 2. Comparison of Area, Delay, and Energy of various CSKAs

Power dissipation is calculated using the Xilinx Xpower tool. The total Energy consumption is calculated using the product of delay and Power consumption and tabulated in Table 2.

## CONCLUSION

The MAC unit designed with Vedic multiplier and different carry-skip adders are analyzed and compared for parameters like Area, and Delay Consumption. In terms of delay, MAC with CONV-CSKA and hybrid CSKA have an almost same delay. But if you consider the area perspective the hybrid CSKA occupies less number of LUTs.

## REFERENCES

[1] Avisek Sen, Partha Mitra, Debarshi Datta, “Low Power MAC Unit for DSP Processor”, International Journal of Recent Technology and Engineering (IJRTE) Vol. 1, Issue-6, January 2013, pp. 93-95.

[2] Bhavani Prasad.Y, Ganesh Chokkakula, Srikanth Reddy.P and Samhitha.N.R, Design of Low Power and High-Speed Modified Carry Select Adder for 16-bit Vedic Multiplier, ISBN No.978-1-4799-3834-6/14/$31.00©2014.

[3] Gitika Bhatia, Karanbir Singh Bhatia, Osheen Chauhan, Soumya Chourasia, Pradeep Kuma5,"An efficient MAC unit with low area consumption" IEEE INDICON,2015.

[4] Samir Palinitkar,” Verilog HDL: A Guide to Digital Design and Synthesis”.

[5] Milad Bahadori, Mehdi Kamal, Ali AfzaliKusha, and Massoud Pedram,"Hi-speed and energy efficient carry-skip adders operating under a wide range of supply voltage levels", IEEE Trans,2015.

[6] M. Alioto and G. Palumbo, "A simple strategy for the optimized design of one-level carry-skip adders," IEEE Trans. Circuits Syst. I, Fundam.Theory Appl., vol. 50, no. 1, pp. 141– 148, Jan. 2003.

[7] Maroju SaiKumar, D.Ashok Kumar, Dr.P.Samundiswary,"Design and performance of analysis of Multiply and Accumulate Unit", ICC CT 2014.

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

[9] R. P. Brent and H. T. Kung, “A regular layout for parallel adders,” IEEE Trans. Comput., vol. C-31, no. 3, pp. 260–264, Mar. 1982.