|| Volume 6 || Issue 1 || January 2021 || ISSN (Online) 2456-0774
INTERNATIONAL JOURNAL OF ADVANCE SCIENTIFIC RESEARCH
AND ENGINEERING TRENDS

RECURSIVE APPROACH TO THE DESIGN OF PARALLEL SELF TIMED ADDER

B.Srividya[1], Y.David Solomon Raju[2]

[1]PG Scholar, Dept of Electronics and communication engineering, Holymary Institute Of Technology And Science, Bogaram(V), Keesara (M),Hyderabad -501 301

[2]Associate Professor, Head of Dept, Electronics and communication engineering, Holymary Institute Of Technology And Science, Bogaram(V), Keesara (M),Hyderabad -501 301

---------- ***----------

Abstract:- As technology scales down into the lower nanometer values power, delay, area and frequency becomes important parameters for the analysis and design of any circuits. As technology scales down into the less nanometer values power, delay, area and frequency becomes important parameters for the analysis and design of any circuits. In this paper, an asynchronous parallel self timed adder based on a recursive formulation for performing multi bit binary addition is being implemented. A practical implementation is provided along with a completion detection unit. The implementation is regular and does not have any practical limitations of high fanouts. A high fan-in gate is required though but this is unavoidable for asynchronous logic and is managed by connecting the transistors in parallel. Simulations have been performed using industry standard toolkits that verify the practicality and superiority of the proposed approach over existing asynchronous adders.

Keywords: *Half adder, Array multiplier, Booth's Multiplier, Vedic Multiplier, Vedic Mathematics.*

---------- ***----------

I INTRODUCTION

Adder is the fundamental logic block in the design of more complex logic circuits such as microprocessors, microcontrollers and DSP processors. The adder block is also responsible for other arithmetic operations such as multiplication and division. Hence the time consumed for addition greatly affects the performance and efficiency of the entire logic block. Addition forms the basis for many processing operations, from counting to multiplication to filtering. As a result, adder circuits are of great interest to digital system designers. Adders and Subtractors are important components in the applications like Digital Signal Processing (DSP) architectures. For signal processing, digital full-adder and full-subtraction are the basic logic circuits which can find applications in digital computing and packet labels processing. Addition is the most basic arithmetic operation; and adder is the most fundamental arithmetic component of the processor. The rapid increase in the number of transistors on chips has enabled a dramatic increase in the performance of computing systems. A majority of the present-day digital systems are clock based or synchronous, which assume that signals are binary and time is discrete. In general, synchronous systems comprise a number of subsystems that change from one state to another depending on a global clock signal, with flip-flops (registers) being used to store the different states of the subsystems. A conventional synchronous system is portrayed by figure 1.1. The state updates within the registers are carried out on the rising edge (positive edge) or falling edge (negative edge) of the global clock – single edge triggering. The state of the global clock permits either data loading or data storage. Since the overall clock utilization is only 50% for single edge triggered systems, double edge triggered flip-flops were subsequently proposed in the literature with the motive of increasing the system throughput as data can be loaded on both the rising and falling clock edges and data is retained when the clock signal does not toggle.

|| Volume 6 || Issue 1 || January 2021 || ISSN (Online) 2456-0774
INTERNATIONAL JOURNAL OF ADVANCE SCIENTIFIC RESEARCH
AND ENGINEERING TRENDS

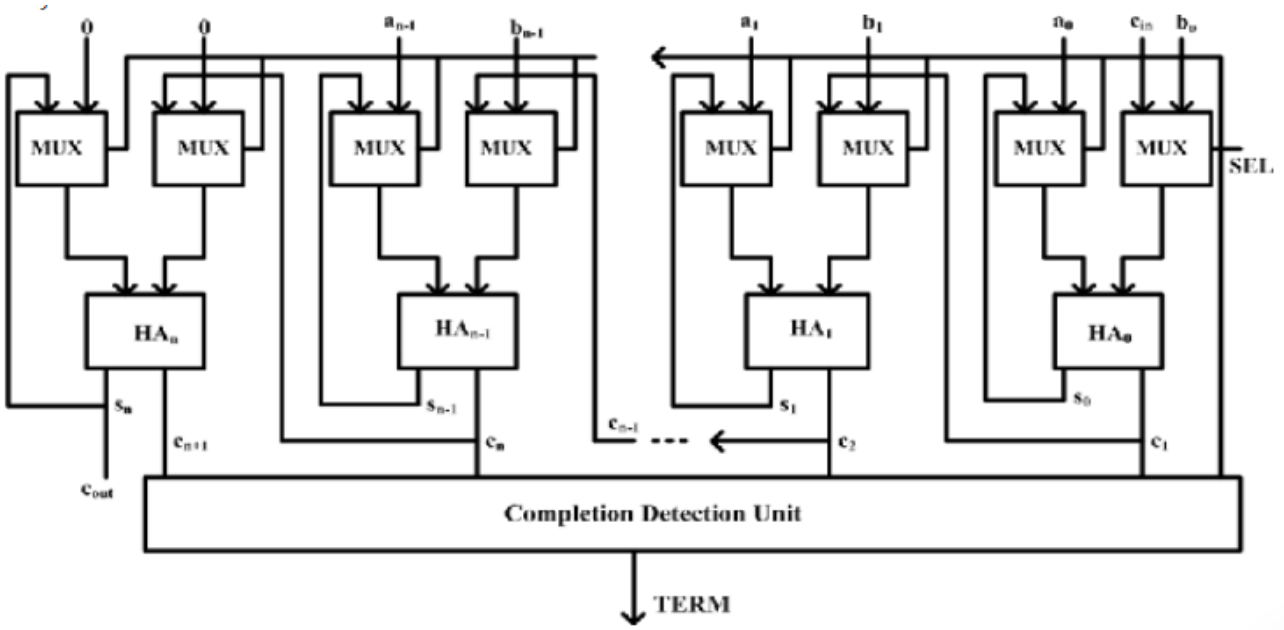

Fig 1: A typical synchronous system stage

II LITERATURE SURVEY

The proposed circuit manages automatic single-rail pipelining of the carry inputs separated by propagation and inertial delays of the gates in the circuit path. Thus, it is effectively a single-rail wave-pipelined approach and quite different from conventional pipelined adders using dual-rail encoding to implicitly represent the pipelining of carry signals. Parallel single-rail self-timed adder is based on a recursive formulation for performing multi-bit binary addition. The operation is parallel for those bits that do not need any carry chain propagation. Thus, the design attains logarithmic performance over random operand conditions without any special speedup circuitry or look-ahead schema. A practical implementation is provided along with a completion detection unit. The implementation is regular and does not have any practical limitations of high fan-outs. A high fan-in gate is required though but this is unavoidable for asynchronous logic and is managed by connecting the transistors in parallel. The adder first accepts two input operands to perform half additions for each bit. The general architecture of the adder is shown in Fig.1. The selection input for two-input multiplexers corresponds to the Req handshake signal and will be a single 0 to 1 transition denoted by SEL. It will initially select the actual operands during SEL = 0 and will switch to feedback/carry paths for subsequent iterations using SEL = 1. The feedback path from the HAs enables the multiple iterations to continue until the completion when all carry signals will assume zero values. (1) Andrew Karanicolas et al (1993) presented a 15-b 1-M sample/s digitally self calibrated pipeline analog-to-digital converter (ADC). The advantages compared to flash or successive approximation techniques because potentially high resolution and high speed can be achieved at the same time. (2) Derek Wong et al (1993) proposed wave pipelining is a technique for pipelining digital systems that can increase the clock frequency of practical circuits without increasing the number of storage elements. Using this design techniques to minimize the impact of delay dependencies. (3) Furber et al (1996) investigated Standard micro pipelines use simple two-phase control circuits. Level sensitive latches are employed on AMULET1, so two- to four-phase converters are required in each latch controller. Investigation has been carried out into four-phase micro pipeline 41 control circuits to avoid the overheads; this has thrown up several design issues relating to cost, performance and safety. Four-phase control circuits offer higher performance and lower cost than two-phase circuits for micro pipelines which use conventional level-sensitive data latches. (4) Stephen Furber et al (1996) designed four-phase control circuits offer higher performance and lower cost than twophase circuits for micro pipelines which use conventional level-sensitive data latches.

The design process has been described for each of these circuits and some optimizations presented which simplify the circuits.These circuits will be incorporated into the next version of the AMULET asynchronous implementation of the ARM microprocessor, and are expected to make a significant contribution to its performance and power-efficiency. (5) Kenneth Yun et al (1996) designed edge-triggered D flip- flops (DETDFF) for data storage in place of traditional transmission gate latches for high-performance asynchronous pipeline circuits to achieve high throughput This higher throughput for the DETDFF design is due to latching the data on both edges of the latch control, removing the need of a reset phase and simplifying the control structures. The DETDFF design has very low delay overhead with or without processing logic. The main disadvantage of the pseudo-static DETDFFs are larger area and heavily loaded latch control lines. (6) Kearney et al (1997) designed balancing pipelines with significant variation in response time, as is the case with this design, can be difficult. Development of a stochastic performance analysis tool that can help further optimize the design by considering synchronization point locations and delay distributions.

III PROPOSED SYSTEM

The adder first accepts two input operands to perform half additions for each bit. Subsequently, it iterates using earlier generated carry and sums to perform half-additions repeatedly until all carry bits are consumed and settled at zero level. A. Architecture of PASTA The general architecture of the adder is shown in Fig. 1. The selection input for two-input multiplexers corresponds to the Req handshake signal and will be a single 0 to 1 transition denoted by SEL. It will initially select the actual operands during SEL = 0 and will switch to feedback/carry paths for subsequent iterations using SEL = 1. The feedback path from the HAs enables the multiple iterations to continue until the completion when all carry signals will assume zero values. B. State Diagrams In Fig. 2, two state diagrams are drawn for the initial phase and the iterative phase of the proposed architecture. Each state is represented by (Ci+1 Si) pair where Ci+1, Si represents carry out and sum values, respectively, from the ith bit adder block. During the initial phase, the circuit merely works as a combinational HA operating in fundamental mode. It is apparent that due to the use of HAs instead of FAs, state (11) cannot appear. During the iterative phase (SEL = 1), the feedback path through multiplexer block is activated. The carry transitions (C i) are allowed as many times as needed to complete the recursion. From the definition of fundamental mode circuits, the present design cannot be considered as a fundamental mode circuit as the input–outputs will go through several transitions before producing the final output. It is not a Muller circuit working outside the fundamental mode either as internally, several transitions will take place, as shown in the state diagram. This is analogous to cyclic sequential circuits where gate delays are utilized to separate individual states. C. Recursive Formula for Binary Addition Let and denote the sum and carry, respectively, for ith bit at the jth iteration. The initial condition (j = 0) for addition is formulated as follows:

$$s_i^0 = a_i \oplus b_i$$

$$c_{i+1}^0 = a_i b_i \quad (1)$$

The j_{th} iteration for the recursive addition is formulated by,

$$S_i^j = S_i^{j-1`} \oplus C_i^{j-1}, 0 \le i < n \quad (2)$$

$$C_{i+1}^k = S_i^{j-1} C_i^{j-1}, \quad 0 \le i \le n \quad (3)$$

The recursion is terminated at kth iteration when the following condition is met: The recursion is terminated at kth iteration when the following condition is met: + …. + = 0, (4) Now, the correctness of the recursive formulation is inductively proved as follows. Theorem 1: The recursive formulation will produce correct sum for any number of bits and will terminate within a finite time. Proof: We prove the correctness of the algorithm by induction on the required number of iterations for completing the addition (meeting the terminating condition). Basic: Consider the operand choices for which no carry propagation is required, i.e. = 0 for ∀ i, i ∈ [0,...n]. The proposed formulation will produce the correct result by a single-bit computation time and terminate instantly. Induction: Assume that 0 for some ith bit at kth iteration. Let l be such a bit for which = 1. We show that it will be successfully transmitted to next higher bit in the (k + 1) th iteration. As shown in the state diagram, the kth iteration of lth bit state ( ) and (l + 1) th bit state ( ) could be in any of (0, 0), (0, 1), or (1, 0) states. As = 1, it implies that = 0. Hence, , for any input condition between 0 to l bits. We now consider the (l + l) th bit state ( )) for kth iteration. It could also be in any of (0, 0), (0, 1), or (1, 0) states. In (k+1)

th iteration, the (0, 0) and (1, 0) states from the kth iteration will correctly produce output of (0, 1) following (2) and (3). For (0, 1) state, the carry successfully propagates through this bit level following. Thus, all the single-bit adders will successfully kill or propagate the carries until all carries are zero fulfilling the terminating condition. The mathematical form presented above is valid under the condition that the iterations progress synchronously for all bit levels and the required input and outputs for a specific iteration will also be in synchrony with the progress of one iteration. In the next section, we present an implementation of the proposed architecture which is subsequently verified using simulations.

IV SIMULATION RESULTS AND DISCUSSION

Technology Schematic

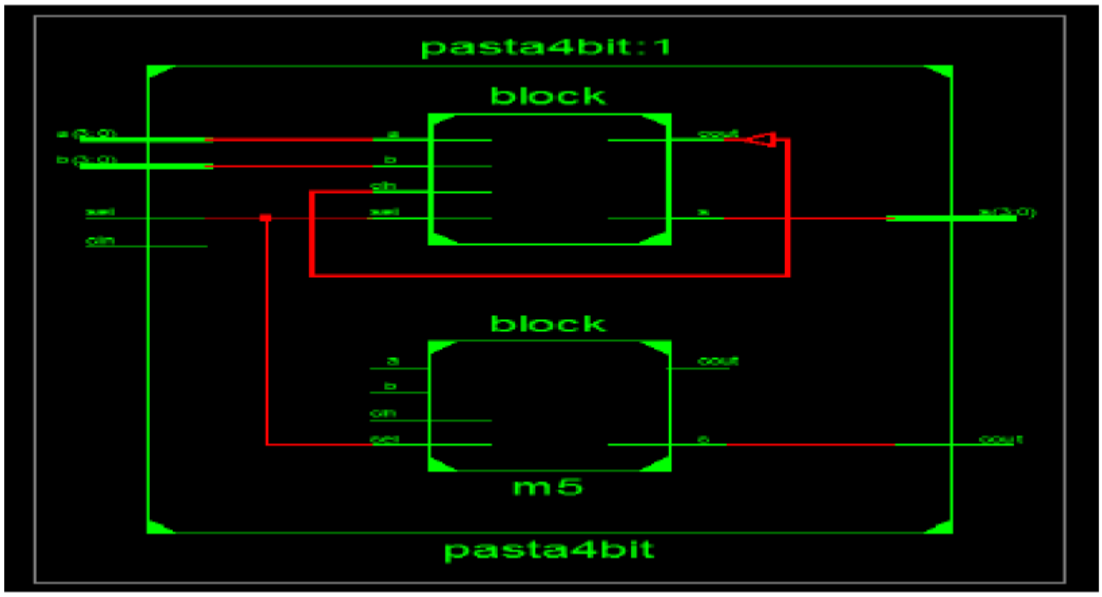

Fig 2. RTL Schematic

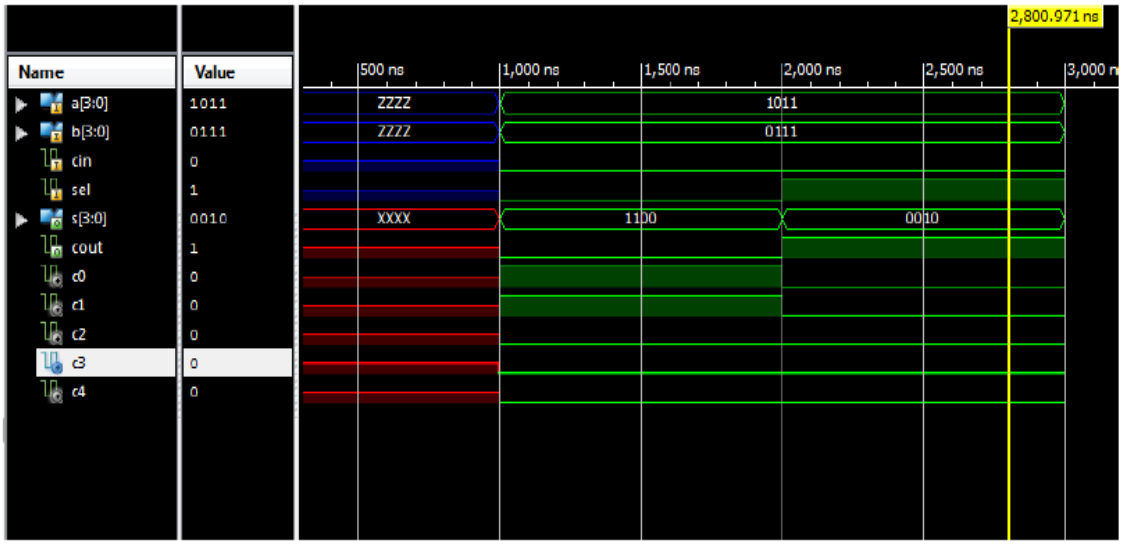

Fig.3: Simulation results

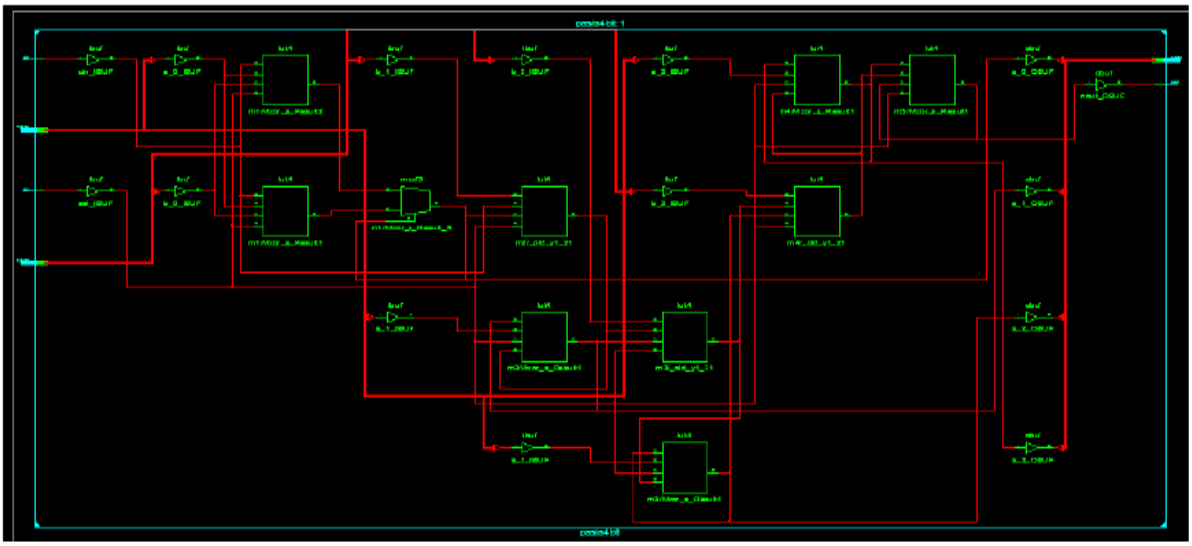

Fig. 4. Simulation results

Device Utilization Summary (estimated values)			[-]
Logic Utilization	Used	Available	Utilization
Number of Slices	5	4656	0%
Number of 4 input LUTs	9	9312	0%
Number of bonded IOBs	15	232	6%

Device utilization summary

```
Timing Summary:
---------------
Speed Grade: -5

   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 14.289ns
```

Timing Summary

V CONCLUSION

This brief presents an efficient implementation of a Modified PASTA. Initially, the theoretical foundation for a single-rail wave pipelined adder is established. Subsequently, the architectural design and CMOS implementations are presented. The design is compared with the Radix Approach which achieves a very simple n-bit adder that is area, power consumption wise much more efficient than the previous self timed adder. Moreover, the circuit works in a parallel manner for independent carry chains, and thus achieves logarithmic average time performance over random input values. The completion detection unit for the proposed adder is also practical and efficient. Simulation results are used to verify the advantages of the modified self timed adder.

REFERENCES

[1] “Fault Tolerant Parallel Filters Based on Error Correction Codes”, Zhen Gao, Pedro Reviriego, Wen Pan, Zhan Xu, Ming Zhao,Jing Wang, and Juan Antonio Maestro.

[2] “Area Efficient and Fault Tolerant FIR Filter”, Brajesh Kumar Gupta (nit jamshedpur), Associate prof. R. Sinha (nit jamshedpur)

[3] “Fault-Tolerant Computation Within Complex FIR Filters.”, P. Chan, G.A. Jullien, L. Imbert, V.S. Dimitrov, G.H. McGibney

[4] “Optimization of Self Checking FIR filters by means of Fault Injection Analysis”, S. Pontarelli, L. Sterpone, G.C. Cardarilli, M. Re, M. Sonza Reorda, A. Salsano, M. Violante. [5] B. Shim and N. Shanbhag, “Energy-efficient soft error-tolerant digital signal processing,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst.,vol. 14, no. 4, pp. 336–348, Apr. 2006. [6] R. W. Hamming, “Error correcting and error detecting codes,” Bell Syst. Tech. J., vol. 29, pp. 147–160, Apr. 1950.

[7] T. Hitana and A. K. Deb, “Bridging concurrent and non-concurrent error detection in FIR filters,” in Proc. Norchip Conf., 2004, pp. 75–78.