DESIGN AND IMPLEMENTATION OF 64 BIT PARALLEL SELF-TIME ADDER (PASTA)

M.Haritha[1] **M.Pavitra**[2]
haritharedd417@gmail.com[1]
[1]PG Scholar, Dept of ECE, PBR Visvodaya Institute of Technology & Science, Kavali, AP, India
[2]Associate Professor, Dept of ECE, PBR Visvodaya Institute of Technology & Science, Kavali, AP, India

ABSTRACT: Adders being core building blocks in different VLSI circuits like microprocessors, ALU's etc. performance of adder circuit highly affects the overall capability of the system. In this paper we will present the design and performance of Parallel Self-Timed Adder. It is based on a recursive formulation for performing multibit binary addition. The operation is parallel for those bits that do not need any carry chain propagation. Simulation and Synthesis of the proposed paper are obtained by using Xilinx ISE 13.2. Results show that the proposed paper yields good results when compared with the existing Carry select adder.

Keywords: CSA, PASTA.

I. INTRODUCTION

Addition is the most common and often used arithmetic operation in microprocessor, digital signal processor, especially digital computers. Also, it serves as a building block for synthesis all other arithmetic operations. Thus performance of any circuit is mainly determined by speed of adder circuit. Circuits may be classified as synchronous or asynchronous. Synchronous circuits are based on clock pulse whereas an asynchronous circuit, or self-timed circuit, is not governed by a clock circuit or global clock instead, they often use signals that indicate completion of operations. Such a system tends to have better noise and electromagnetic compatibility properties than synchronous systems due to the absence of a global clock reference. Asynchronous operation by itself does not imply low power, but often suggests low power opportunities based on the observation that asynchronous circuits consume power only when it is active. The synchronous adders perform slowly due to its incremental nature of operation and therefore it is not recommended for fast and parallel adders. The basic building block of combinational digital adders is a single bit adder. The simplest single bit adder is a half adder (HA). The full adders (FA) are single bit adders with the carry input and output. The full adders are basically made of two half adders in terms of area, interconnection and time complexity .This paper proposes the design of parallel self timed adder (PASTA). The design of PASTA is regular and uses half adders along with multiplexers with minimum interconnection requirement. The interconnection and area requirement is linear which makes it feasible to fabricate in a VLSI chip. The design operates in a parallel manner for those bits that do not require any carry propagation. It is self timed, which means that as soon as the addition is done, it will signal the completion of addition thereby overcoming the clocking limitations.

The rest of the paper is organized as the following. In section II CSA is explained. In Section III PASTA is designed, in Section IV Comparison results of the PASTA and CSA is discussed and finally ends with a conclusion in Section V.

II. Description of the Related Art

Multiplexer are one of the basic circuits of digital arithmetic. The speed at which a multiplexer can deliver the product of two binary numbers becomes critical in certain applications where repetitive multiplications are required. Applications requiring repetitive multiplications include various digital signal processing functions, such as Finite Impulse Response (FIR) filters, and 3D rendering. Such applications require both high throughput and fast response time. The design of multiplexer employed in these applications can have a significant effect on overall application performance.

Since multiplication is essentially repeated addition, it stands to reason that digital multiplexer rely heavily on adder circuits. Commonly used adder circuits include the half-adder, the full-adder, and the carry-look ahead adder. The half adder takes two 1-bit inputs, and returns two outputs, a sum bit and a carry bit. A full adder returns the same outputs, but it has an extra input, known as a carry-in. The carry-in input is configured to receive a carry-out bit from an addition of lower-order bits. Because of the carry-in, full-adders can be cascaded to allow the addition of numbers larger than one bit. An adder formed by cascading several full adders is known as a ripple carry adder.

One problem with ripple carry adders is the fact that a carry generated at the lowest order bit position must be propagated through each subsequent bit position in a sequential manner. Such propagation adds a significant amount of time to the addition process. One solution to this problem is the carry-look ahead adder (CLA). In a CLA, the carry in bit is presented to each bit position in the adder, and is combined with the operand bits to either generate or propagate a carry. Therefore, the carry-in bit is not required to propagate through multiple stages sequentially as in a ripple carry adder. The CLA will require extra circuitry over a ripple carry adder. However, since the carry is not required to ripple through each stage sequentially, it can perform additions at a significantly greater speed.

Parallel array multiplexer are a commonly used multiplier circuit in systems where increased performance is required. In one type of parallel array multiplexer, the first step performed is the formation of a bit-product matrix. A bit-product matrix is simply an array of bit-products formed by multiplication of the individual bits of the two numbers being multiplied, a multiplicand and a multiplier. Formation of a bit-product matrix may become complicated in certain situations, such as multiplying signed numbers.

III. DESIGN OF CSA

CARRY SELECT ADDER (CSA):

The carry select adder comes in the category of conditional sum adder. Conditional sum adder works on some condition. Sum and carry are calculated by assuming input carry as 1 and 0 prior the input carry comes. When actual carry input arrives, the actual calculated values of sum and carry are selected using a multiplexer. The conventional carry select adder consists of k/2 bit adder for the lower half of the bits i.e. least significant bits and for the upper half i.e. most significant bits (MSB's) two k/ bit adders. In MSB adders one adder assumes carry input as one for performing addition and another assumes carry input as zero. The carry out calculated from the last stage i.e. least significant bit stage is used to select the actual calculated values of output carry and sum. The selection is done by using a multiplexer.

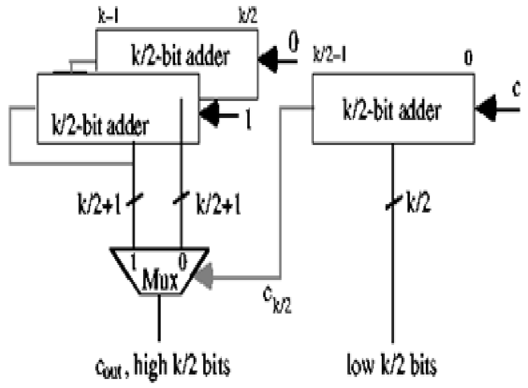


Fig.1: Carry select adder

In electronics, a carry-select adder is a particular way to implement an adder, which is a logic element that computes the $(n+1)$-bit sum of two n-bit numbers. The carry-select adder is simple but rather fast, having a gate level depth of $O(\sqrt{n})$.

The carry-select adder generally consists of two ripple carry adders and a multiplexer. Adding two n-bit numbers with a carry-select adder is done with two adders (therefore two ripple carry adders) in order to perform the calculation twice, one time with the assumption of the carry being zero and the other assuming one. After the two results are calculated, the correct sum, as well as the correct carry, is then selected with the multiplexer once the correct carry is known.

The number of bits in each carry select block can be uniform, or variable. In the uniform case, the optimal delay occurs for a block size of $\lfloor\sqrt{n}\rfloor$. When variable, the block size should have a delay, from addition inputs A and B to the carry out, equal to that of the multiplexer chain leading into it, so that the carry out is calculated just in time. The $O(\sqrt{n})$ delay is derived from uniform sizing, where the ideal number of full-adder elements per block is equal to the square root of the number of bits being added, since that will yield an equal number of MUX delays.

Problem in the system - This technique of dividing adder into stages increases the area utilization but addition operation fastens.

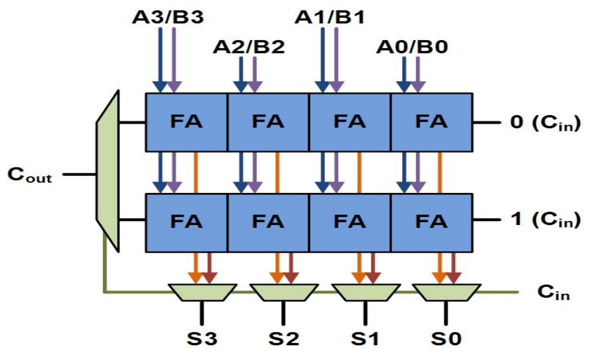


Fig.2: Block diagram of Carry Select Adder (CSA)

PIPELINED PARALLEL ADDER:

Pipelining a plan intends to embed registers into each phase of the structure. In this way, if a structure has K-stages, K registers must be embedded from a contribution to a yield. One enlist will be included for each phase of the circuit.

To begin with, let perceive how to go from a combinatorial plan into a pipelined structure. Fig.3 demonstrates a combinatorial circuit made out of three hubs. By embeddings an enlist for each phase from a contribution to a yield, the pipelined plan of fig is acquired.

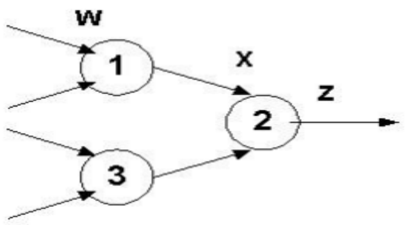


Fig.3: Combinatorial design of Pipelined parallel adder

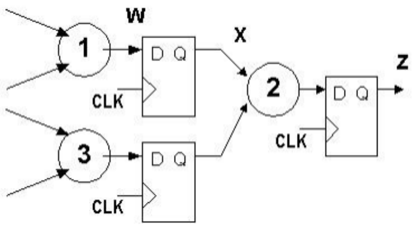


Fig 3: Pipelined design

Pipelining a design will increase its throughput. The trade-off of this improvement is the use of registers and latency is shown in below fig 4.

As a combinatorial structure gets obfuscated, additional registers must be added to keep the transitional computational results inside a comparable clock cycles. If pipelining is to be useful, regardless, we ought to be looked with the need to play out a long game plan of essentially similar endeavors. Additionally, these three must be accessible:

- The basic limit is on and on executed.

- The basic limit must be unmistakable into free stages having unimportant cover with each other.

- The stages must be of practically identical multifaceted nature.

Parallel adders respect these contemplations. Thus we should change over a parallel snake into a pipelined parallel snake. Survey the 4-bit parallel snake.

The extra multifaceted nature of such a pipelined snake fulfills if long progressions of numbers are being incorporated is showed up in underneath fig.4.

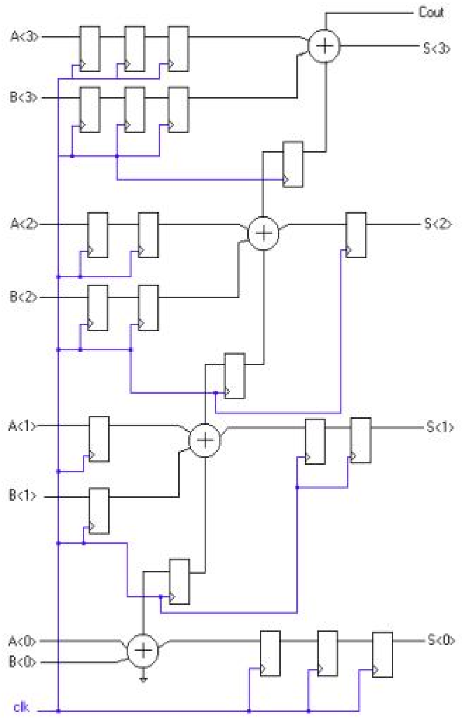


Fig 4: Pipelined parallel adder

IV. PROPOSED WORK

DESIGN OF PARALLEL SELF-TIMED ADDER: ARCHITECTURE OF PARALLEL SELF-TIMED ADDER:

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.

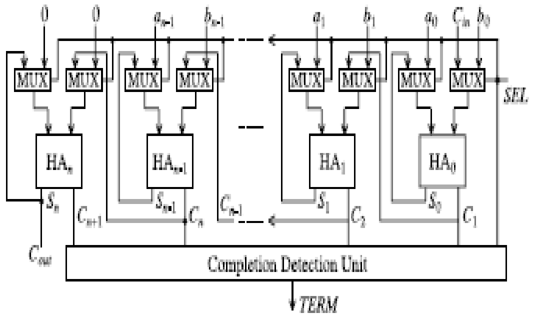


Fig 5: General block diagram of parallel self timed adder (PASTA)

Architecture of PASTA

The general architecture of the adder is shown in Fig.5. 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 State Diagrams

In Fig.6, 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* represent carry out and sum values, respectively, from the *i*th 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 cannot appear.

During the iterative phase (SEL = 1), the feedback path through multiplexer block is activated. The carry transitions (*Ci*) 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

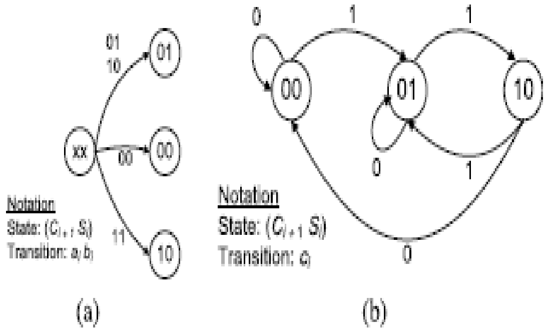


Fig 6: State diagram of PASTA (a) initial phase (b) iterative phase

2 Recursive Formula for Binary Addition

Let $S_i{}^j$ and $C\,ji$+1 denote the sum and carry, respectively, for ith bit at the j th iteration. The initial condition (j = 0) for addition is formulated as follows:

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

$$C_{i+1}^0 = a_i b_i .$$

The jth iteration for the recursive addition is formulated by

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

$$C_{i+1}^j = S_i^{j-1} C_i^{j-1}, \quad 0 \le i \le n.$$

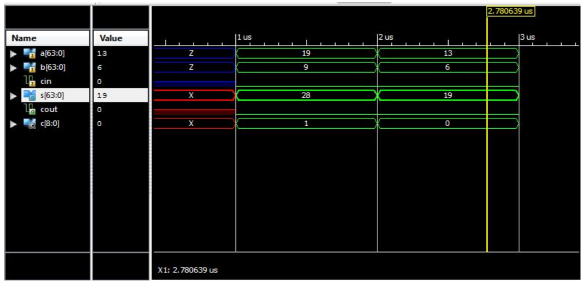
The recursion is terminated at kth iteration when the following condition is met:

$$C_n^k + C_{n-1}^k + \cdots + C_1^k = 0, \quad 0 \le k \le n.$$

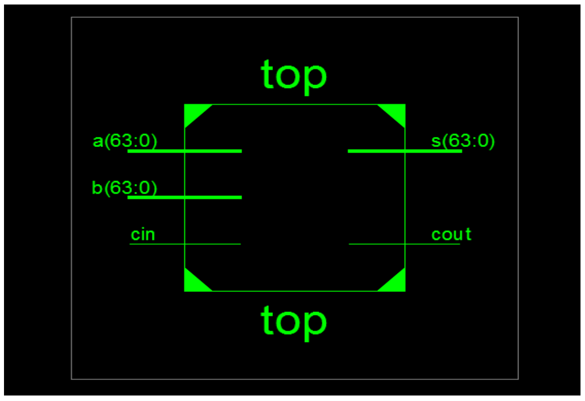
V. RESULTS

Existing 64-bit Carry Select Adder.

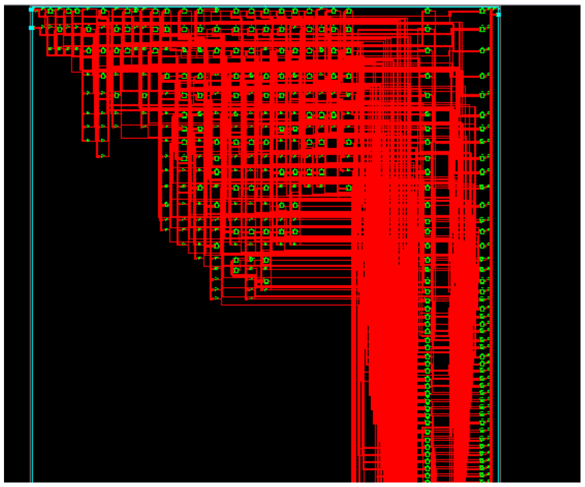
Simulation.



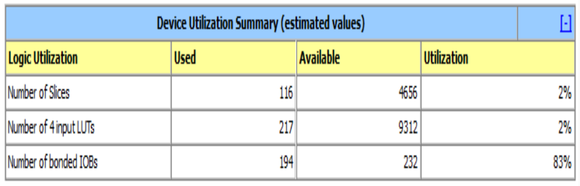
RTL Schematic of 64-bit Carry Select Adder.



Technology Schematic of 64-bit Carry Select Adder.

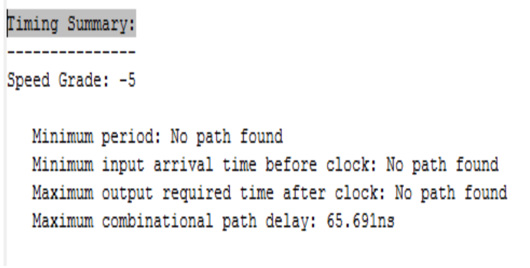


Design Summary of 64-bit Carry Select Adder.



Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	116	4656	2%
Number of 4 input LUTs	217	9312	2%
Number of bonded IOBs	194	232	83%

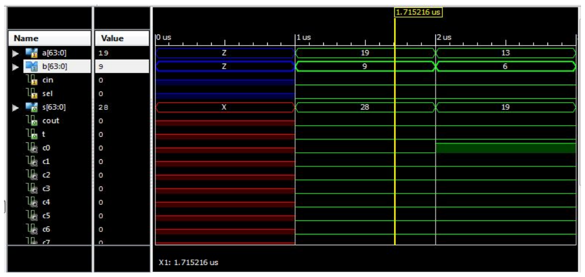
Timing Summary of 64-bit Carry Select Adder.



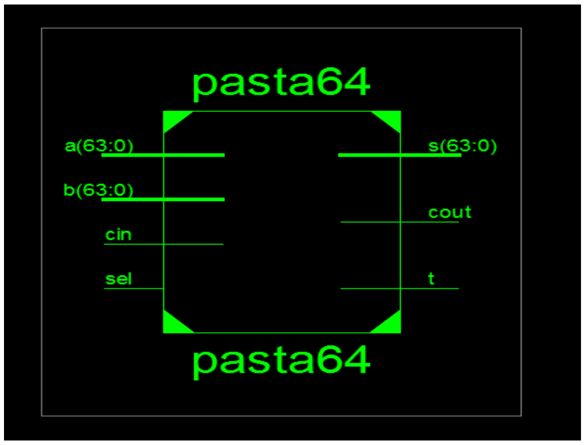
```
Timing Summary:
---------------
Speed Grade: -5

   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: 65.691ns
```

Proposed 64-bit Parallel Self-Timed Adder.

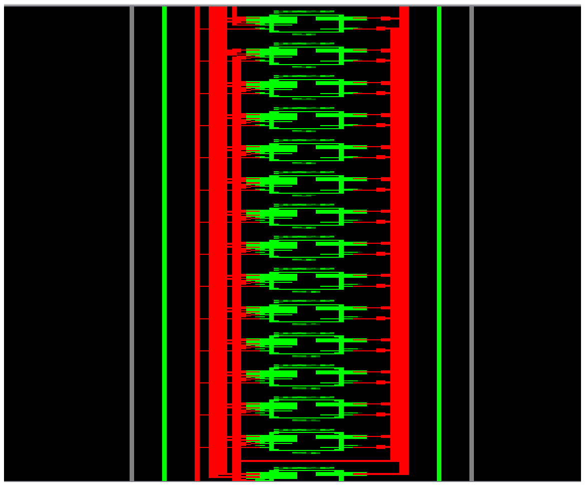
Simulation.



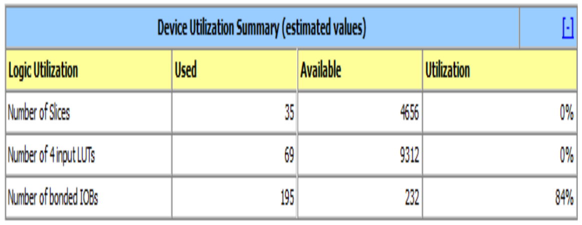
RTL Schematic of 64-bit Parallel Self-Timed Adder.



Technology Schematic of 64-bit Parallel Self-Timed Adder.

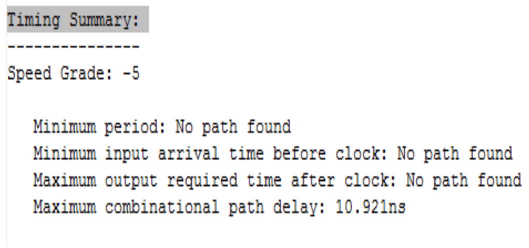


Design Summary of 64-bit Parallel Self-Timed Adder.



Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	35	4656	0%
Number of 4 input LUTs	69	9312	0%
Number of bonded IOBs	195	232	84%

Timing Summary 64-bit Parallel Self-Timed Adder.



```
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: 10.921ns
```

Comparison Table of 64-bit Carry Select Adder and 64-bit Parallel Self-Timed Adder.

DESIGN	AREA			DELAY (ns)
	SLICES	LUTS	IOBS	
CSA	116	217	194	69.691ns
PASTA	35	63	195	10.921ns

VI. CONCLUSION

In this brief we present the Modified PASTA. This achieves a very simple n-bit adder that is area, power consumption wise much more efficient than the previous self timed adder and it occupies less area and less time. 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. The PASTA is analyzed for various performance parameters. Simulation results are used to verify the advantages of the modified self timed adder.

REFERENCES

[1] D .Geer “Is It Time for Clockless Chips?”IEEE comput, vol.38.n0.3,pp,18-19,mar 2005.

[2]. Mohammed Ziaur Rahman, Lindsay Kleeman, and Mohammad Ashfak Habib, “Recursive Approach to the Design of a Parallel Self-Timed Adder

[3]. D. Geer, “Is it time for clockless chips? [Asynchronous processor chips],” IEEE Comput., vol. 38, no. 3, pp. 18–19, Mar. 2005.

[4]. J. Sparsø and S. Furber, Principles of Asynchronous Circuit Design. Boston, MA, USA: Kluwer Academic, 2001.

[5]. P. Choudhury, S. Sahoo, and M. Chakraborty, “Implementation of basic arithmetic operations using cellular automaton,” in Proc. ICIT, 2008, pp. 79–80.

[6]. M. Z. Rahman and L. Kleeman, “A delay matched approach for the design of asynchronous sequential circuits,” Dept. Comput. Syst. Technol., Univ. Malaya, Kuala Lumpur, Malaysia, Tech. Rep. 05042013, 2013.

[7]. M. D. Riedel, “Cyclic combinational circuits,” Ph.D. dissertation, Dept. Comput. Sci., California Inst. Technol., Pasadena, CA, USA, May 2004.

[8]. R. F. Tinder, Asynchronous Sequential Machine Design and Analysis: A Comprehensive Development of the Design and Analysis of Clock Independent State Machines and Systems. San Mateo, CA, USA: Morgan, 2009.

[9]. W. Liu, C. T. Gray, D. Fan, and W. J. Farlow, “A 250-MHz wave pipelined adder in 2-μm CMOS,”IEEE J. Solid-State Circuits, vol. 29, no. 9, pp. 1117–1128, Sep. 1994.

[10]. F.-C. Cheng, S. H. Unger, and M. Theobald, “Selftimed carry-lookahead adders,” IEEE Trans. Comput., vol. 49, no. 7, pp. 659–672, Jul. 2000.

[11]. S. Nowick, “Design of a low-latency asynchronous adder using speculative completion,” IEE Proc. Comput. Digital Tech., vol. 143, no. 5, pp. 301– 307, Sep. 1996.

[12]. N. Weste and D. Harris, CMOS VLSI Design: A Circuits and Systems Perspective. Reading, MA, USA: Addison-Wesley, 2005

[13]. C. Cornelius, S. Koppe, and D. Timmermann, “Dynamic circuit techniques in deep submicron technologies: Domino logic reconsidered,” in Proc. IEEE ICICDT, Feb. 2006, pp. 1–4

BIOGRAPHIES:

M. Haritha has received her B.Tech in Electronics & Communication Engineering from Visvodaya engineering College affiliated to JNTU, Ananthapur and pursuing M.Tech in Electronics&Communication Engineering (VLSI Design) from PBR Visvodaya Institute of Technology and Science affiliated to JNTU, Ananthapur.

M. PAVITRA has received her B.E in Electronics & Communication Engineering and M.Tech degree in Electronics and Communication Engineering (VLSI Design) from Andhra University in 2003 and Sathyabama Institute of Science and Technology in 2010 respectively. She is dedicated to teaching field from the last 13 years. She has guided 12 P.G and 12 U.G Students. At present she is working as Associate Professor in PBR VITS, Kavali, and Andhra Pradesh, India.