INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

EFFICIENT MULTIOUTPUT CARRY LOOK-AHEAD ADDERS

B. Venkata Sreecharan[1], C. Venkata Sudhakar[2]

[1]*M.TECH (VLSI DESIGN) Deportment of ECE SVNE, Tirupathi, boyapati.brothers@gmail.com*
[2] *ASSISTANT PROFESSOR Deportment of ECE SVNE, Tirupathi, sudhakar.chowdam@gmail.com*

Abstract: - Addition is the fundamental operation for any VLSI processors or Digital Signal Processing (DSP). In this paper we present an efficient implementation of a 16-bit Manchester carry chain (MCC) adder using an enhanced multiple output domino logic. In adder circuits the main drawback is propagation delay and to overcome this drawback using domino logic. In this paper 4-bit, 8-bit and 16-bit adders are been designed and power, delay and area are measured using TANNER tool, and then compared with the conventional adder. The experimental results reveal that the proposed adders achieve delay, power and area reductions for Multi bit addition.

Keywords: Carry look-ahead adder, domino logic, Manchester carry chain, High Performance, delay, TANNER tool.

I. Introduction

In VLSI the one of the basic components are adders for an ALU and there are 'n' number of adders each with their own advantages and disadvantages. When two numbers are to be added and if each of them is of 'n' bits then we can add them in two different ways serial and parallel. High-speed adder architectures include the Ripple-Carry Adder (RCA), Carry-Select Adders (CSLA), Carry-Skip Adders (CSA), Carry Look-Ahead (CLA) adders [1], Conditional sum adders and combinations of these high speed adders based on the carry look-ahead adders principle.

Ripple-Carry Adder: A Ripple-Carry Adder is a digital circuit that produces the arithmetic sum of two binary numbers. It can be constructed with full adders connected in cascaded with the carry output from each full adder connected to the carry input of the next full adder in the chain. Fig.1 shows the interconnection of four full adder (FA) circuits to provide a 4-bit ripple carry adder. Notice from Fig.1 that the input is from the right side because the first cell traditionally represents the least significant bit (LSB). Bits a_0 and b_0 represent the least significant bits of the numbers to be added. The sum output is represented by the bits s_0 to s_3.

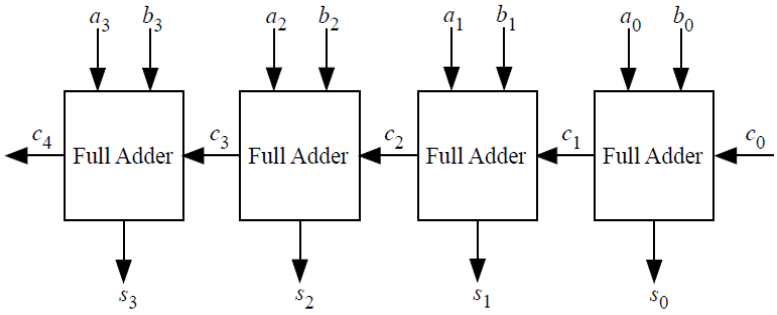

Fig.1 4-bit full adder

Carry-Select Adder: 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 Adder) 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.

Carry-Skip Adder: A Carry-Skip Adder (also known as a Carry-Bypass Adder) is an adder implementation that improves on the delay of a Ripple-Carry Adder with little effort compared to other adders. The improvement of the worst-case delay is achieved by using several carry-skip adders to form a block-carry-skip adder.

Carry Look-Ahead Adder: To reduce the computation time, engineers devised faster ways to add two binary numbers by using Carry Look-Ahead Adders. They work by creating two signals (P and G) for each bit position, based on whether a carry is propagated through from a less significant bit position (at least one input is a '1') a carry is generated in that bit position (both inputs are '1') or if a carry is killed in that bit position (both inputs are '0'). In most cases, P is simply the sum output of a half-adder and G is the carry output of the same adder. After P and G are generated the carries for every bit position are created. Some advanced Carry Look-Ahead architectures are the Manchester carry chain, Brent-Kung adder, and the Kogge-Stone adder.

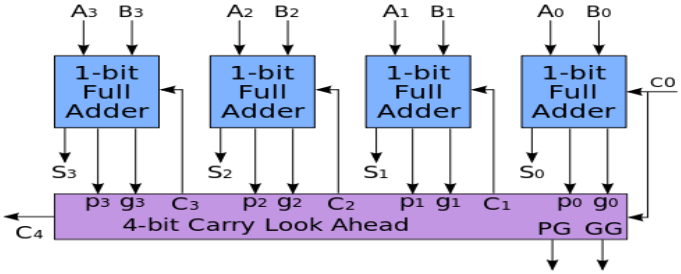

Fig.2 4-bit adder with Carry Look-Ahead

By combining multiple Carry Look-Ahead Adders even larger adders can be created. This can be used at multiple levels to make even larger adders. For example, the following adder is a 64-bit adder that uses four 16-bit CLAs with two levels of LCUs.

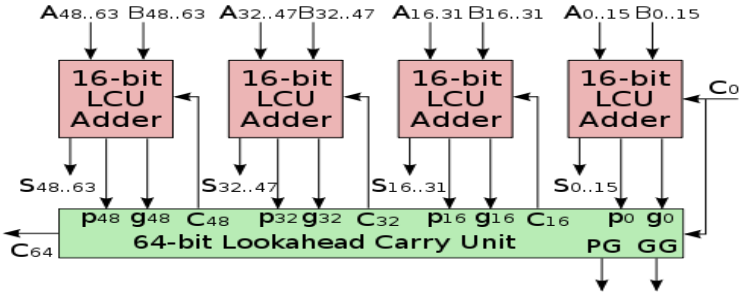

Fig.3 64-bit adder with Carry Look-Ahead

II. PRELIMINARY CONCEPTS OF DOUBLE CARRY CHAIN MCC ADDERS

MCC adders can efficiently be designed in CMOS logic. We chose the 4-bit MCC [2] [3] adder because it minimized the delay for 16-bit adder. This adder consists of three main cell types identified as Propagate, Generate and Sum bit slices. The 4-bit MCC Adder is shown in Fig. 4.

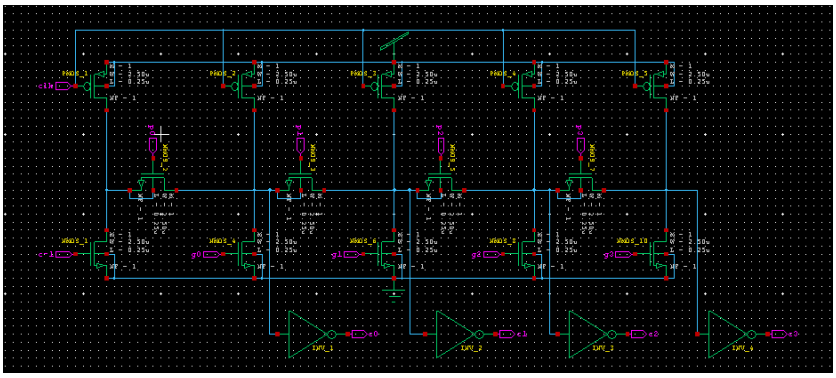

Fig.4 4-bit MCC

In this paper the carry chain has been split into even and odd carry chains. The even and odd carries of this adder are computed in parallel by two independent 4-bit carry chains. The delay for 4-bit and 8-bit will be higher when compared to the conventional adder but for 16-bit and more than that delay will be reduced drastically.

Let $A = an-1 \cdot \cdot \cdot a1\ a0$ and $B = bn-1 \cdot \cdot \cdot b1\ b0$ represent two binary numbers to be added and $S = sn-1 \cdot \cdot \cdot s1\ s0$ be their sum. In the following, the symbols ・, +, ⊕ and ~ are used to denote the AND, INCLUSIVE OR, EXCLUSIVE OR, and NOT logical operations, respectively. In binary addition, the computation of the carry signals is based on the following recursive formula:

$$ci = gi + zi \cdot ci-1 \qquad (1)$$

Where $gi = ai \cdot bi$ and zi are the carry generate and the carry propagate terms, respectively. The latter, for the case of INCLUSIVE OR adders, is defined as $zi = ti = ai + bi$, while for the case of EXCLUSIVE OR adders, it is defined as $zi = pi = ai \oplus bi$. Expanding relation (1), each carry bit ci can be expressed as

$$ci = gi + zigi-1 + zizi-1gi-2 + \cdot \cdot \cdot + zizi-1 \cdot \cdot \cdot z1g0 + zizi-1 \cdot \cdot \cdot z0c-1 \qquad (2)$$

The sum bits of the adder are defined as si = pi ⊕ci−1, where c−1 is the input carry.

Even Carry Computation:

For i = 0 and z0 = t0, from relation (1), we get that c0 = g0 + t0 ・c−1. Since the relation gi = gi・ti holds, we get that c0 = t0 ・(g0 + c−1) = t0 ・h0, where h0 = g0 + c−1 is the new carry.
From relation (2), for i = 2 and zi = pi, we get that

$$c2 = g2 + p2g1 + p2p1g0 + p2p1p0c-1$$

Since gi + pi ・gi−1 = gi + ti・gi−1 and pi = pi ・ti, we have

$$c2 = t2\ (g2 + g1 + p2p1g0 + p2p1p0c-1)$$
$$= t2\ (g2 + g1 + p2p1t0\ (g0 + c-1))$$
$$= t2 \cdot h2$$

Where

$$h2 = g2 + g1 + p2p1t0\ (g0 + c-1) \text{ is the new carry.}$$

In the same way, the new carries for i = 4, 6 are computed as

$$h4 = g4 + g3 + p4p3t2\ (g2 + g1 + p2p1t0\ (g0 + c-1))$$

h6 =g6 + g5 + p6p5t4 × (g4 + g3 + p4p3t2 (g2 + g1 + p2p1t0 (g0 + c−1)))

Then, the following equations are derived for the new carries for even values of *i*

h2 =G2 + P2G0

h4 =G4 + P4G2 + P4P2G0

h6 =G6 + P6G4 + P6P4G2 + P6P4P2G0

Odd Carry Computation:

The new carries for the odd values of *i* are computed according to the aforementioned methodology proposed for the even carries as follows

h1 =g1 + g0 + p1p0c−1

h3 =g3 + g2 + p3p2t1 (g1 + g0 + p1p0c−1)

h5 =g5+g4+p5p4t3 (g3 + g2 + p3p2t1 (g1 + g0+p1p0c−1))

h7 =g7 + g6 + p7p6t4 × (g5 + g4 + p5p4t3 × (g3 + g2 + p3p2t1 (g1 + g0 + p1p0c−1)))

While for odd values of *i*, the equations for the new carries are rewritten as follows

h1 =G1 + P1c−1

h3 =G3 + P3G1 + P3P1c−1

h5 =G5 + P5G3 + P5P3G1 + P5P3P1c−1

h7 =G7 + P7G5 + P7P5G3 + P7P5P3G1 + P7P5P3P1c−1.

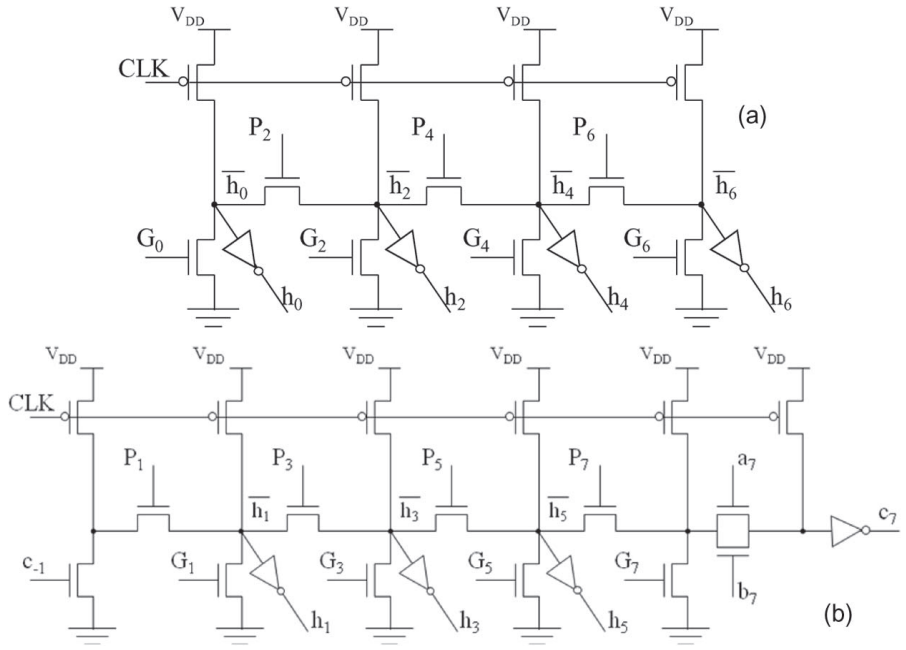

Fig.5 Proposed carries' implementation for (a) the even carry chain and (b) the odd carry chain.

From the above mentioned equations, it is evident that the groups of even and odd new carries can be computed in parallel by different carry chains in multi-output domino CMOS logic. The new generate and propagate signals Gi and Pi can be easily proven to be mutually exclusive, avoiding false node discharges. Between the new and the conventional carries, ci−1 = ti−1 · hi−1 holds; therefore, the sum bits are computed as si= pi ⊕ (ti−1 · hi−1). According to [4], the computation of the sum bits can be performed as follows:

$$s_i = h_{i-1} \cdot p_i + h_{i-1} \cdot (p_i \oplus t_{i-1})$$

for i > 0, while s0 = p0 ⊕c−1

The above Relation can be implemented using a 2 → 1 multiplexer that selects either pi or pi ⊕ ti−1 according to the value of hi−1. Taking into account that an XOR gate introduces equal delay with a 2 → 1 multiplexer and both terms pi and pi ⊕ ti−1 are computed faster than hi, then no extra delay is introduced by the use of the proposed carries for the computation of the sum bits according to above equation.

For the implementation of the sum signals, the domino chain is terminated, and static CMOS technology is used for the pi ⊕ ti−1 gate and the final 2 → 1 multiplexer. An efficient static CMOS implementation of the 2→1 multiplexer is used for Sum bit implementation.

III. PROPOSED WORK

To evaluate the speed performance of the proposed design over the conventional 4-bit, 8-bit, 16-bit, 32-bit and 64-bit adders have been designed according to the even and odd carry chain principle respectively. The conventional 8-bit, 16-bit, 32-bit and 64-bit MCC adders are designed by cascading two, four, eight and sixteen 4-bit MCC adder modules, respectively. The proposed 16-bit, 32-bit and 64-bit MCC adders are designed by cascading two and four of the proposed 8-bit MCC adder modules, respectively.

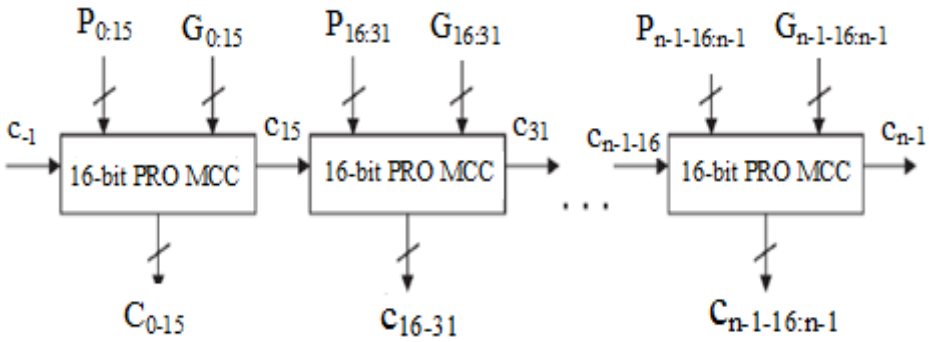

Fig.6 Proposed 16-bit MCC adder module

In this paper we proposed a new independent carry technique which will not require the previous carry bit to generate the current carry. While it will generate all the carry bits parallel. The proposed Carry Equations for 4-bit are

C(0) = e(0) AND p(0)
C(1) = (e(1) AND p(1) OR p(1) AND g(0)) or k(1)
C(2) = (e(2) AND p(2) OR p(2) AND g(1)) or k(2)
C(3) = (e(3) AND p(3) OR p(3) AND g(2)) or k(3)

INTERMEDIATE TERMS

e(i) = a(i) XNOR b(i)
k(i) = p(i) and …….. p(0)

PROPAGATE AND GENERATE TERMS

p(i) = a(i) OR b(i)
g(i) = a(i) AND b(i)

SUM BIT IMPLEMENTATION

S(i)= a(i) xor b(i) xor c(i-1)

IV. Simulation Results

The analysis of 16-bit Manchester carry chain (MCC) adder using an enhanced multiple output domino logic is targeted and verified on TANNER tool. The constraints taken to considerations are power, delay and area. The strategies are to minimizing the power, delay and area.

The Proposed system is simulated and verified using TANNER tool. The below simulation results show the outputs of the conventional delay and proposed day.

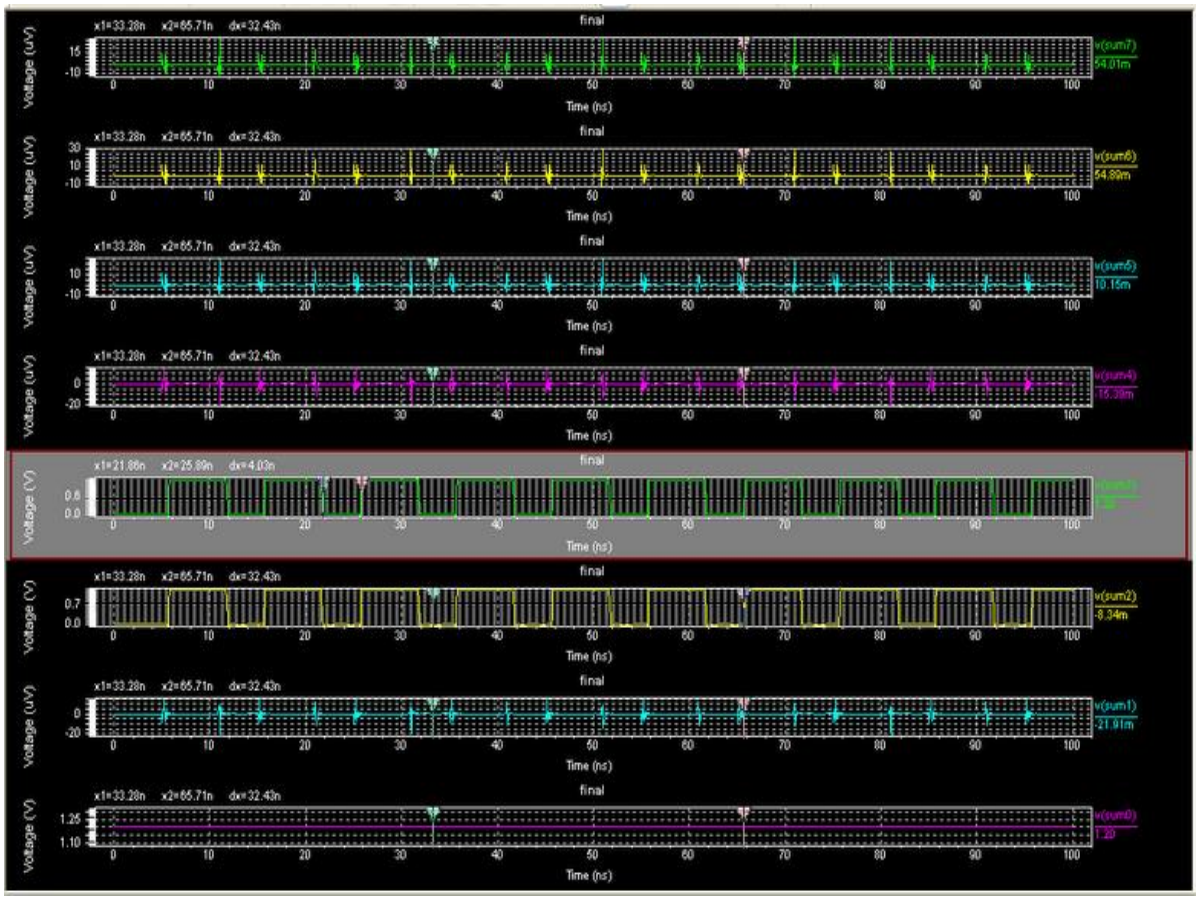

Fig.7 Simulation result for conventional delay

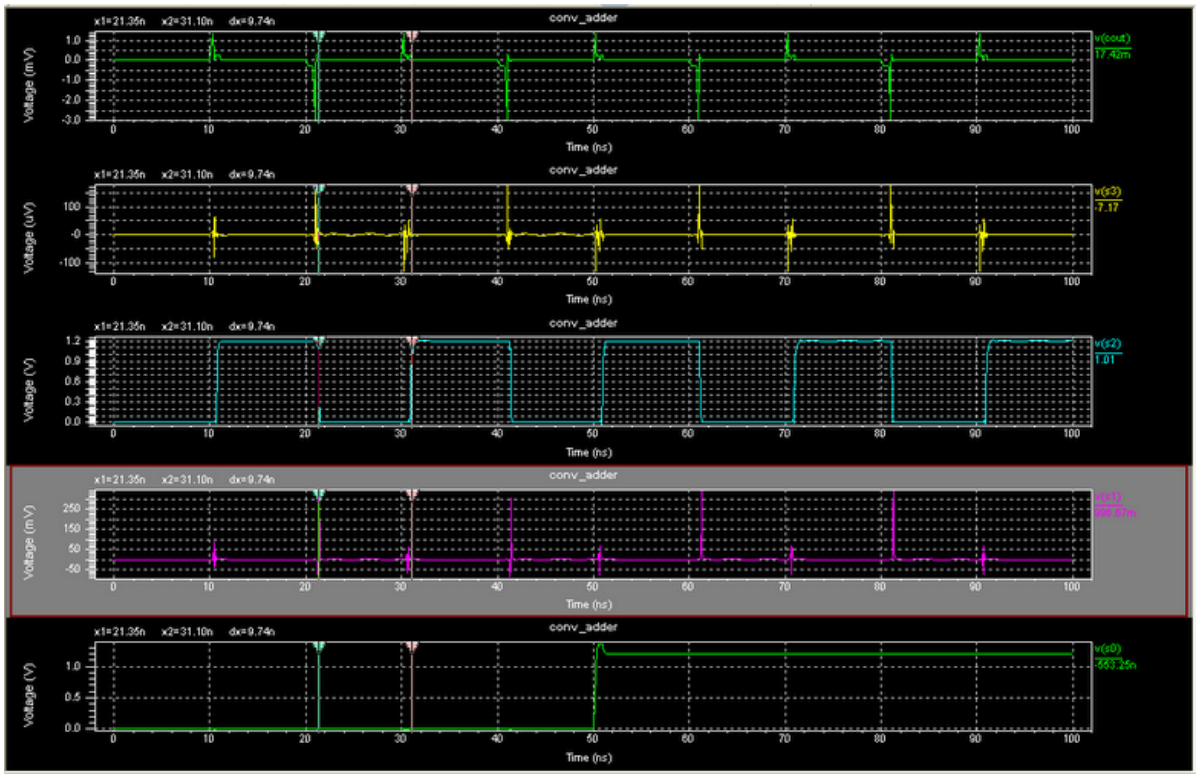

Fig.8 Simulation result for proposed delay

The synthesis result of conventional and proposed adder delay, power and area comparison is shown in the below Table I.

	Conventional			**Proposed**		
Bit type	Delay (ps)	Power (nW)	Area (μm^2)	Delay (ps)	Power (nW)	Area (μm^2)
4-bit	97.3	663.41	38	128.25	540.47	32
8-bit	120.21	1533.2	77	128.25	1125.4	64
16-bit	168.57	3765.5	175	128.25	2635.1	144

Table I. Carry look-ahead Adder Parametric Comparison

V. Conclusion

The Independent Carry technique is an efficient approach to construct Fast Carry look-ahead adders. In this paper, we presented a new independent carry technique, which does not need to wait for the previous carry to be calculated. In this way the delay has been reduced for the higher bit addition. Here TANNER tool is used to design the 16-bit Manchester carry chain (MCC) adder using an enhanced multiple output domino logic. The experimental results shows that these adder circuits gives superior performance compared to adder circuits designed using conventional domino techniques. Further, Manchester carry chain adder in 16-bit is used for increasing high speed and reduced delay in the domino circuit.

REFERENCES

[1] Efstathiou, Zaher Owda, and Yiorgos Tsiatouhas, "New High-Speed Multi-output Carry Look-Ahead adders" IEEE transactions on circuits and systems: vol. 60, no. 10, October 2013.

[2] Yukinori Ono, Member et al IEEE, ―Binary Adders of Multigate Single-electron transistors: Specific Design using pass-Transistor Logic‖, IEEE transactions on nanotechnology, vol. 1, no.2, June 2002.

[3] A. A. Amin, "Area-efficient high-speed carry chain," Electron. Lett. vol. 43, no. 23, pp. 1258–1260, Nov. 2007.

[4] Giorgosdimitrakopoulos and dimitrisnikolos, Member, IEEE ―High- Speed Parallel-Prefix VLSI Ling Adders‖ IEEE transactions on computers, vol. 54, no. 2, February 2005.

B. Venkata Sreecharan: He is currently Studying M.Tech-VLSI Sree Vidyanikethan Engineering College, Tirupati. His areas of interest are Digital System Design, VLSI Design.

C. Venkata Sudhakar: He is currently working as an Assistant Professor in ECE department of Sree Vidyanikethan Engineering College, Tirupati. His research areas of interest are Digital Design, VLSI Design.