# Design and Comparative Analysis of Conventional Adder and Pipelined Adder

**Anjana Bhardwaj, Swati Gupta**

***Abstract--Adding two binary numbers is a basic operation in binany electronic processing system. Pipelining digital systems has been shown to provide significant performance gains over non-pipelined systems and remains a standard in microprocessor design. The desire for increased performance has seen a push for pipelines. Pipelining is considered to be a good technique for increasing the circuit speed. In this paper, 4-bit conventional adder and 4-bit pipelined adder has been implemented using Cadence virtuoso tool and simulation was performed using the generic 0.18 µm CMOS Technology at 5V. For comparison purposes, various parameters such as delay time, rise time and fall time has been compared which shows that pipelined adders are more efficient in terms of speed, power and throughput.***

***Keywords- Pipelining, Full Adder, Binary Adder.***

## I. INTRODUCTION

Addition is one of the basic arithmetic operations in binary arithmetic by adding two binary numbers. An adder should be designed to meet the requirements of many modern devices: small chip area and high circuit speed. Since high speed computer arithmetic units such as adders, multipliers and fast dividers that dominate the power dissipation and device complexity is dramatically increasing. Now-a-days, low power design has come to the forefront in addition to the two traditional issues mentioned above. Especially, the adder is critical in the aim to reduce overall power consumption since they are used for implementation of multipliers. Over the years, there has been an increased growth in wireless electronics and distributed computer architectures. This has pushed the need for developing innovative designs for realizing fast multi-bit adders. To increase the frequency of operation, pipelining is considered. Careful optimization of adders and other data path circuits will grow in importance as methods to reduce power while maintaining or improving performance are sought. There exist numerous adder implementations each with good attributes and some drawbacks. As the number of input bits increases, so does the delay associated with the computation of the carries. The desire to reduce the delays associated with carry propagation has resulted in novel adder architectures and implementations [5], [6], [8].

## II. CONVENTIONAL ADDER

Most of the VLSI applications, such as digital signal processing image and video processing, and microprocessors, extensively use arithmetic operations. Addition, subtraction, multiplication, and multiply and accumulate (MAC) are examples of the most commonly used operations.

**Manuscript published on 30 December 2013.**
***** Correspondence Author (s)

**Anjana Bhardwaj,** Asst. Professor, Dept. of ECE ABES Engineering College, Ghaziabad, India.
**Swati Gupta,** M.Tech Scholar, Dept. of ECE Amity University, Noida, India.

The 1-bit full-adder cell is the building block of all these modules. The sum and carry output are given by

$$C_{out}= AB + AC_{in} + BC_{in}$$
$$Sum=A \oplus B \oplus C_{in}$$

The Binary Adder is made up from standard AND & Ex-OR gates and allow us to “add” together single bit binary numbers, a and b to produce two outputs, the SUM of the addition and a CARRY called the Carry-out, ($C_{out}$) bit. One of the main uses for the Binary Adder is in arithmetic and counting circuits.

The schematic of conventional 1-bit full adder is shown in figure1and its truth table and logical diagram is shown in figure2 &3.

It is possible to create a logical circuit using multiple full adders to add *N*-bit numbers. Each full adder inputs a $C_{in}$, which is the $C_{out}$ of the previous adder. This kind of adder is called a ripple-carry adder, since each carry bit "ripples" to the next full adder. Note that the first (and only the first) full adder may be replaced by a half adder.

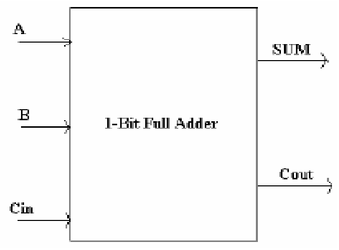

Fig. 1. Schematic of 1-bit Full Adder

| A | B | $C_{in}$ | SUM | $C_{out}$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

Fig. 2. Truth Table of 1-bit full adder

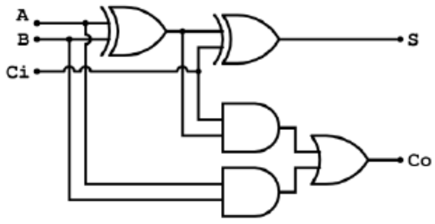

Fig. 3. Logical Diagram for 1-bit full adder

One main disadvantage of "cascading" together 1-bit binary adders to add large binary numbers is that if inputs A and B change, the sum at its output will not be valid until any carry-input has "rippled" through every full adder in the chain. Consequently, there will be a finite delay before the output of an adder responds to a change in its inputs resulting in the accumulated delay especially in large multi-bit binary adders becoming prohibitively large. This delay is called Propagation delay.

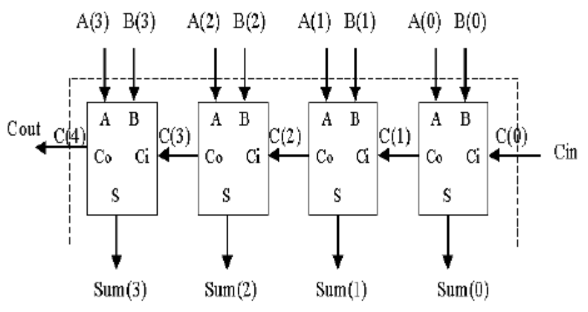

Fig. 4. 4-bit full adder

## III. PIPELINED ADDERS

Pipelining is a technique used for increasing the throughput of the system. Pipelining is done by splitting a task into several subtasks and inserting registers between these subtasks. There will be an increase in area due to the presence of these intermediate registers.Pipelining is a method to get higher throughput without significantly increasing hardware cost or latency.

The following definitions are necessary to understand pipelining.

1. Throughput = (no. of usable outputs) / (unit time).
2. Latency = Time delay from valid inputs provided until outputs valid.
3. Cost = Area + Power.

The objective is to increase the throughput and reduce the latency and cost.

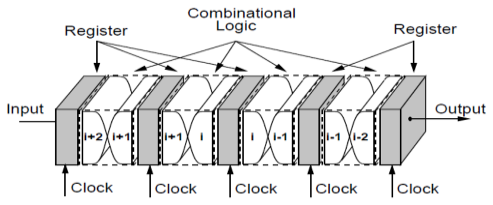

Fig. 5. Conventional pipelining

In figure5, n=4, thus the circuit should be clocked four times faster than a non-partitioned counterpart i.e., a circuit with no internal synchronization elements and where every new computation is started only after the previous one has completed. In general, reciprocal of the maximum stage delay establishes the maximum pipeline clock-rate. If the logic is further partitioned into shorter stages and additional synchronization elements are inserted between stages, still higher pipeline clock-rates can be achieved. For example, each stage in four stage pipeline shown in figure5 might be spilt into two new stages, each having half the original delay, and the clock-rate would ideally double.

The pipeline gets high throughput by applying new inputs to the first stage while the second stage works on intermediate results from the first stage. The pipeline gives maximum throughput when the clock period, T, is as small as possible. If the delay of the original adder, $t_d$ , is split between the delay of the first stage of the adder, $t_{d1}$, and the delay of the second stage, $t_{d2}$, the

$T \geq \max(t_{d1} , t_{d2}) + T_{reg}$

Where $t_{d1} + t_{d2} = t_d$

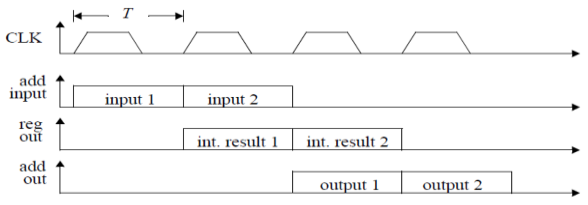

Fig. 6. Two stage pipeline

The minimum value for T is attained when

i. $t_{d1} = t_{d2} = t_d /2$

ii. $T= T_{min} = t_d /2 + T_{reg}$

Therefore,

Max Throughput = $1/T_{min} = 1/(t_d/2 + T_{reg}) = 2/( t_d + 2T_{reg})$

Latency = $2T_{min} = 2( t_d /2 + T_{reg} ) = t_d + 2T_{reg}$

If $T_{reg} << T$, then pipeline performance is as good as multiple parallel hardware, and if cost of pipeline reg << *cost of adder*, then cost of pipeline is as good as single adder.

Generalization to n Stages--The adder example shows that pipeline efficiency is limited by

1. Register delay
2. Inability to evenly divide delay between pipeline stages.

If we pipeline a circuit with original delay, $t_d$, we find that the clock period cannot be shorter than

$$T = t_d /n + \Delta T$$

where $\Delta T$ represents the additional delay from the registers and the inability to divide up the stages evenly, and n is the number of pipeline stages.

Throughput = $1/(t_d / n + T_{reg}) = n / (t_d + nT_{reg})$

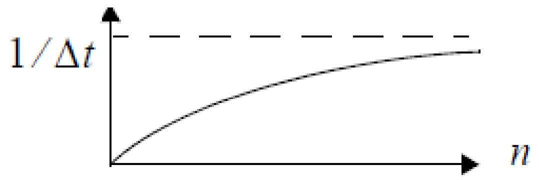

Fig. 7. ThroughputVs n

Latency = $n (t_d/n + T_{reg}) = t_d + nT_{reg}$

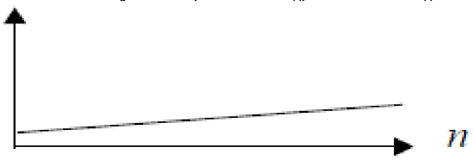

Fig. 8. Latency Vs n

Cost = n [(unpipelined cost / n) + register cost]
Cost = unpipelined cost + n (register cost)

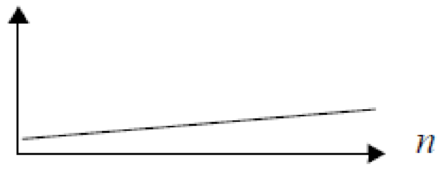

Fig. 9. Cost Vs n

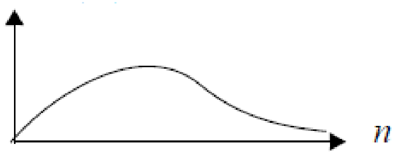

Fig. 10. Throughput / Cost Vs n

This simple model is intended to show that:

1. A moderate amount of pipelining can be used to increase throughput without significantly increasing cost or latency.
2. Pipelining cannot be used to achieve arbitrary high levels of throughput.
3. There is an optimum number of pipeline stages beyond which pipelining is not economical.

Here, we reference a couple of conventionally pipelined (CP) adders and highlight the significant issues the design address. Pipelining improves throughput at the expense of latency, however, once the pipe is filled we can expect one data item per unit of time. Some of the conventional pipelined adder designs that have been reported include one that uses overlapped clocks in an effort to eliminate sources of overhead [5]. The 4-bit carry propagate adder employs a series of three registers to equalize the delays in adding the four bits and hasa three cycle latency [4]. Time borrowing is performed to shorten the critical path and the adder design has been realized in 0.18μm technology. A number of pipeline registers are introduced and in a similar fashion as in [4] several of these registers are inserted in series to equalize data arrival times at adder units. The gain in speed is achieved by clocking sub-circuits faster than would be possible with a ripple carry adder. The conventional pipelined adder architectures achieve path delay equalization by inserting registers. In the instances where these registers are used for delay equalization no logic is used between the set of registers, the output of one register connects directly to the input of the next. Figure 2.4 shows the register placement. Introducing several registers increases the clocking overhead and skew.

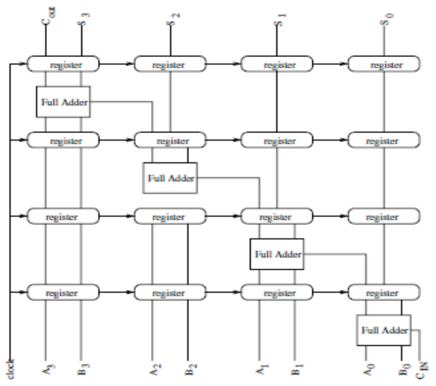

Fig. 11. Conventional pipelined adder architecture

## IV. RESULTS

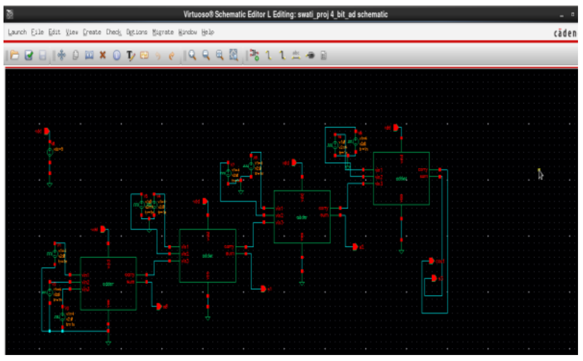
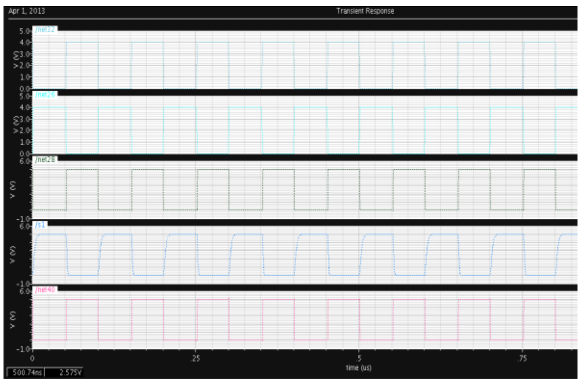

Fig.12. Schematic and waveform of 4-bit Conventional Full Adder

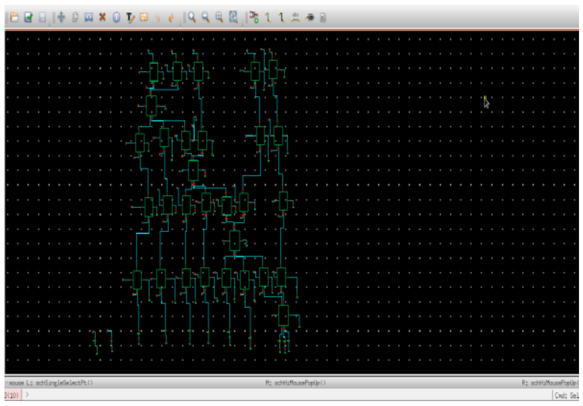
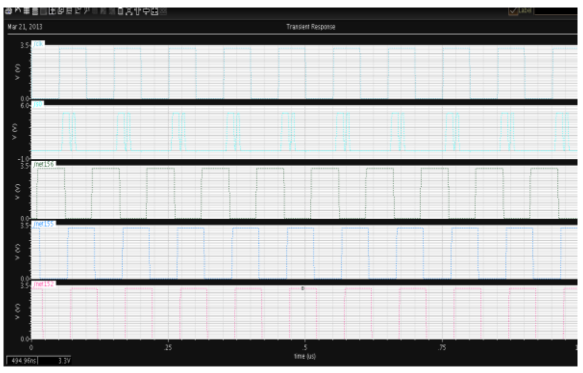

Fig. 13. Schematic and waveform of 4-bit Pipelined Full Adder

The adder was implemented using Cadence Virtuoso tool and simulation was performed using the generic 0.18 µm Technology at 5V. For comparison purposes, we selected two adders, i.e., 4-bit conventional adder and 4-bit pipelined adder. The simulation was carried out and the results are shown in the below table (Table I).

Supply Voltage = 5V
Technology = 180nm
Clock Pulse = 3.3V

Table I: Simulation result

| 4-Bit Full Adder | Delay (ns) | Power (µW) | Power Delay Product (PDP) | Rise Time (ns) | Fall Time (ns) |
|---|---|---|---|---|---|
| Pipelined Adder | 58.204 | 1.04 | 60.533 | 2.864 | 2.013 |
| Conventio-nal Adder | 63.186 | 1.23 | 77.718 | 3.248 | 2.386 |

## V. CONCLUSION & FUTURE SCOPE

In this project, 4-bit conventional adder circuit and 4-bit pipelined adder circuit has been successfully implemented and simulated using Cadence Virtuoso tool and various parameters are recorded. Finally, the recorded parameters are compared and concludes that pipelined adder is the efficient adder in speed and throughput.

Future work will involve reducing the speed, area and power dissipation further. The achievement of higher clock rates in conventional pipelined systems comes at a cost. Synchronization element occupy silicon area, dissipate power, and add overhad to the system clock, i.e., setup time, register (or latch) delay and uncontrolled clock-skew. The overhead added to the clock becomes a speed-limiting factor as the system is partitioned into very short stages and limits the maximum speed-up that can be attained. Thus, wave-pipelining offers a clock-rate maximization model without partitioning the system into a succession of short stages.

## REFERENCES

[1] Vishal D. Jaiswal, Saroj V. Bakale ,Sonal S. Bhopale, ”Implementation And Comparatively Analysis Of Low Power Adder Circuit”, International Journal of Advanced Technology & Engineering Research (IJATER) , Vol. 2, NO. 2, M ay 2013, pp. 2250-3536 .

[2] `N. M. Chore, and R. N. Mandavgane, “A Survey of Low Power High Speed 1 Bit Full Adder”, Proceeding of the 12th International Conference on Networking, VLSI and Signal Processing, pp. 302-307, 2010.

[3] James Levy, JabulaniNyathi, and Jos´e Delgado ,”High-Performance Parallel Addition Using Hybrid Wave-Pipelining”,2005 IEEE Journal of electronic circuit designs,vol. 40, NO. 2, May 2005, pp. 7803-9197.

[4] V. Sukumar, D. Pan, K. Buck, H. Hess, H. Li, D. Cox and M. M. Mojarradi, “Design of a pipelined Adder Using Skew Tolerant Domino Logic in a 0.35 _m TSMC Process,” *2004* IEEE Workshop on Microelectronics and Electron Devices, September 2004, pp. 55-59.

[5] C-H. Huang, J-S.Wang, C. Yeh and C-J.Fang, “The CMOS Carry-Forward Adders,” IEEE Journal of Solid-State Circuits, Vol. 39, NO. 2, February 2004, pp. 327-336.

[6] Y. Kim and L-S Kim, “64-bit carry-select adder with reduced area,” Electronics Letters,Vol. 37, Issue 10, May 10, 2001, pp. 614-615.

[7] N. West and K. Eshraghian, “Principles of CMOS VLSI Design: A System Perspective”, 2nd ed., Addison Wesley, NY. 1992, pp. 513-536.

[8] P. K. Chan, M. D. F. Schlag, C. D. Thomborson, and V. G. Oklobdzija,“Delay Optimization of Carry-Skip Adders and Block Carry-Lookahead Adders,” 10th IEEE Proceedings on Computer Arithmetic, June 26-28, 1991, pp.154-164.