Design and Simulation of Energy Efficient Full Adder for Systolic Array

Pratibhadevi Tapashetti, A. S Umesh, Ashalatha Kulshrestha

Abstract— Full adder is an essential component for the design and development of all types of processors viz. digital signal processors (DSP), microprocessors, Microcontrollers, ARM processors etc. Full adder is the basic building block for all arithmetic and logical operations. For the speed improvement the systolic array using the full adders is involved in almost all the processors. Adders are the core elements of complex arithmetic operations like addition, subtraction, multiplication, division, exponentiation etc. In most of these systems adder lies in the critical path that affects the overall speed of the system. So enhancing the performance of the 1-bit full adder cell is a significant goal. The present study proposes an efficient full adder cell design and simulation using the simulation software Edvin XP which considerably increases the speed.

***Index Terms*— Auto Sequencing Memory(ASM),Central processing Units(CPU),Data Processing Units(DPU).**

I. INTRODUCTION

In computer architecture, a systolic array[1] is a pipe network arrangement of processing units called cells. It is a specialized form of parallel computing, where cells compute data and store it independently of each other. Full adder is the most widely used block to perform all the necessary operations of this Systolic array.

II. SYSTOLIC ARRAY

The systolic array can be defined as "Imagine n simple processors arranged in a row or an array and connected in such a manner that each processor may exchange information with only its neighbors
to the right and left. The processors at either end of the row are used for input and output. Such a machine constitutes the simplest example of a systolic array" and "Systolic Arrays are regular arrays of simple finite state machines, where each finite state machine in the array is identical .A systolic algorithm relies on data from different directions arriving at cells in the array at regular intervals and being combined". By pipelining, processing may proceed concurrently with input and output, and consequently overall execution time is minimized. Pipelining plus multiprocessing at each stage of a pipeline should lead to the best-possible performance.
The advantages of systolic array are

1. it has multiple cells networked together to form an array.
2. Faster speed due to register to register transfer of data.
3. Data is not destroyed until it has been completely used.
4. All cells run off of a central clock.
5. Host Data Entry – All cells are I/O capable.
6. Easily extend the architecture to many more processors.
7. Capable of supporting SIMD organizations for vector operations and MIMD for non homogeneous parallelism.
8. Allow extremely high throughput with multidimensional arrays.

At the same time Systolic array is complicated in software and hardware. Expensive in comparison to uni processor systems, although much faster. They are highly specialized for particular applications.

2.1 Description of systolic array architecture

A systolic array is composed of matrix-like rows of data processing units called cells. Data processing units DPUs are similar to central processing units (CPU)s, except for the usual lack of a program counter[1], since operation is transport-triggered, means by the arrival of a data object. Each cell shares the information with its neighbours immediately after processing. The systolic array is often rectangular where data flows across the array between neighbour DPUs, often with different data flowing in different directions. The data streams entering and leaving the ports of the array are generated by auto-sequencing memory units, ASMs. Each ASM includes a data counter. In embedded systems a data stream may also be input from and/or output to an external source.

Manuscript received on 21th Dec, 2011.

Mrs. Pratibhadevi Tapashetti is with Kruti Institute of Technology and Engineering, Raipur, India (Email: pratibhat@yahoo.com)

Dr. A S Umesh is with Kruti Institute of Technology and Engineering , Raipur, India. (Email:umeshasp@yahoo.co.in.)

Dr. Ashalatha Kulshrestha is with Kankeshwari Devi IT, Jamnagar, Gujarat, India (Email:amy.sonpalyahoo.com)

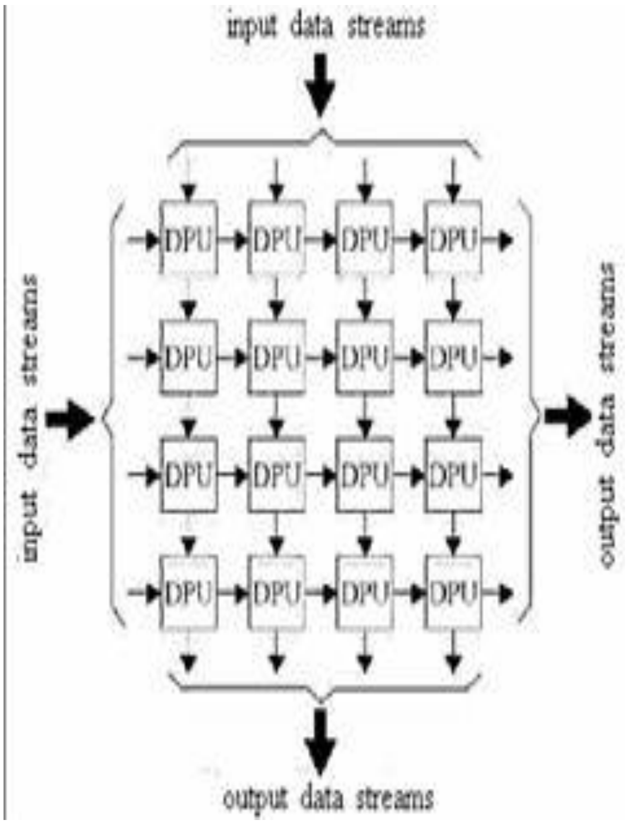


Fig. 1 Architecture of systolic array.

An example of a systolic algorithm might be designed for matrix multiplication. One matrix is fed in a row at a time from the top of the array and is passed down the array, the other matrix is fed in a column at a time from the left hand side of the array and passes from left to right. Dummy values are then passed in until each processor has seen one whole row and one whole column. At this point the result of the multiplication is stored in the array and can now be output a row or a column at a time, flowing down or across the array.
Systolic arrays are arrays of DPUs[3] which are connected to a small number of nearest neighbour DPUs in a mesh-like topology. DPUs perform a sequence of operations on data that flows between them. Because the traditional systolic array synthesis methods have been practiced by algebraic algorithms, only uniform arrays with only linear pipes can be obtained, so that the architectures are the same in all DPUs. The consequence is that only applications with regular data dependencies can be implemented on classical systolic arrays. Like SIMD machines, clocked systolic arrays compute in "lock-step" with each processor undertaking alternate compute and communicate phases. But systolic arrays with asynchronous handshake between DPUs are called *wavefront arrays*. One well known systolic array is Carnegie Mellon University's iwrap processor, which has been manufactured by Intel. An iWarp system has a linear array processor connected by data buses going in both directions.

A linear systolic array in which the processors are arranged in pairs where one multiplies its input by *x* and passes the result to the right and the next adds a_j and passes the result to the right

2.2 Super Systolic Array

The super systolic array is a generalization of the systolic array. Because the classical synthesis methods, yielding only uniform DPU arrays permitting only linear pipes, systolic arrays could be used only to implement applications with regular data dependencies. By using simulated annealing instead Rainer Kress has introduced the generalized systolic array which can be called as super systolic array. Its application is not restricted to applications with regular data dependencies. The Kress Array[4] is the reconfigurable version of the super systolic array. Because of the wide applicability of the super systolic array its reconfigurability makes sense that Kress Array having been pioneered by Rainer Kress for reconfigurable computing.

2.3 Bit Level systolic array and Matrix Multication.

A bit level systolic multiplier which multiplies input data by a weighting coefficient. A bit-level systolic adder which computes intermediate summation result and a multiple-number systolic adder which performs the addition of *M* parallel available numbers. These units have the advantages of systolic array, such as local communication, pipelining, and modular structure. Since the most complex cell in the design is a single-bit full adder, these structures can be easily implemented on MOS technology. Besides, the delay time of the slowest cell is quite small therefore throughput rate is much higher than other solutions
A digital data processor for matrix and matrix multiplication includes a systolic array of nearest neighbor connected gated full adders. The adders are arranged to multiply two input data bits and to add their product to an input cumulative sum bit and a carry bit from a lower order bit computation. The result and input data bits are output to respective neighboring cells, a new carry bit being recirculated for later addition to a higher order bit computation. Column elements of one matrix and row elements of the other are input to either side of the array bit-serially and least significant bit leading for mutual counter propagation there through with a cumulative time delay between input of adjacent columns or rows. Bit level matrix interactions for product matrix computation occur at individual cells. Pairs of intercalcuated adder trees are connected switchable to the array to accumulate bit level contributions to product matrix elements

III. ADDER

In electronics, an adder or summer[6] is a digital circuit that performs addition of numbers. In many computers and other kinds of processors, adders are used not only in the arithmetic logic units but also in other parts of the processor, where they are used to calculate addresses, table indices, and similar.
Although adders can be constructed for many numerical representations, such as binary coded decimal or excess-3 the most common adders operate on binary numbers. In cases where two's complement or ones' complement is being used to represent negative numbers then it is trivial to modify an adder into an adder–subtractor. Other signed number representations require a more complex adder.
A half adder adds two one-bit binary numbers *A* and *B*. It has two outputs, sum *S* and carryout *C* the final sum is $2C + S$. The simplest half-adder design incorporates an XOR gate for *S* and an AND gate for *C*.

Half adders cannot be used compositely given their incapacity for a carry-in bit.

IV. FULL ADDER DESIGN

A full adder for simplification the single bit full adder will be considered from which the device can be scaled to multiple bits. A one-bit full adder is a device with three single bit binary inputs -A, B, Cin and two single bit binary outputs Sum, C-out. Having both carry in and carry out capabilities, the full adder is highly scalable and found in many cascaded circuit implementations. The basic logic functions of the full adder can be summarized in the truth table. From the truth table it can be seen that the full adder can be trivially constructed with two half adders. The full adder can also be decomposed into the following logical relationships.

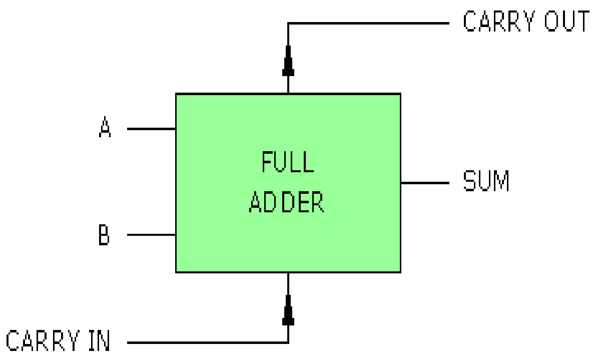


Fig .2 Full adder block diagram

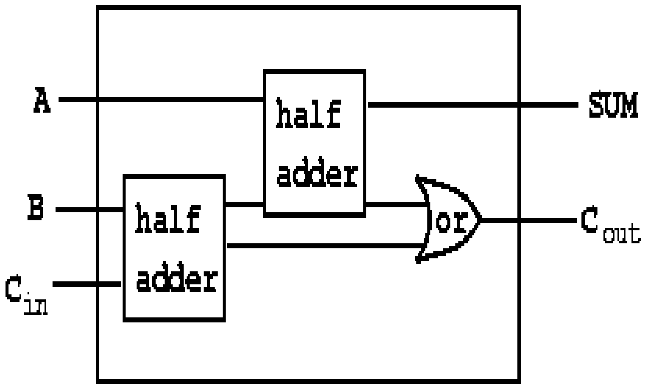


Fig. 3 Full adder using 2 half adder and an OR gate.

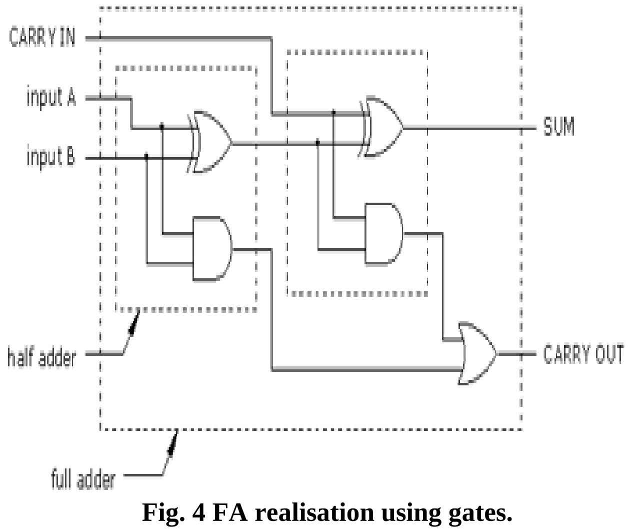


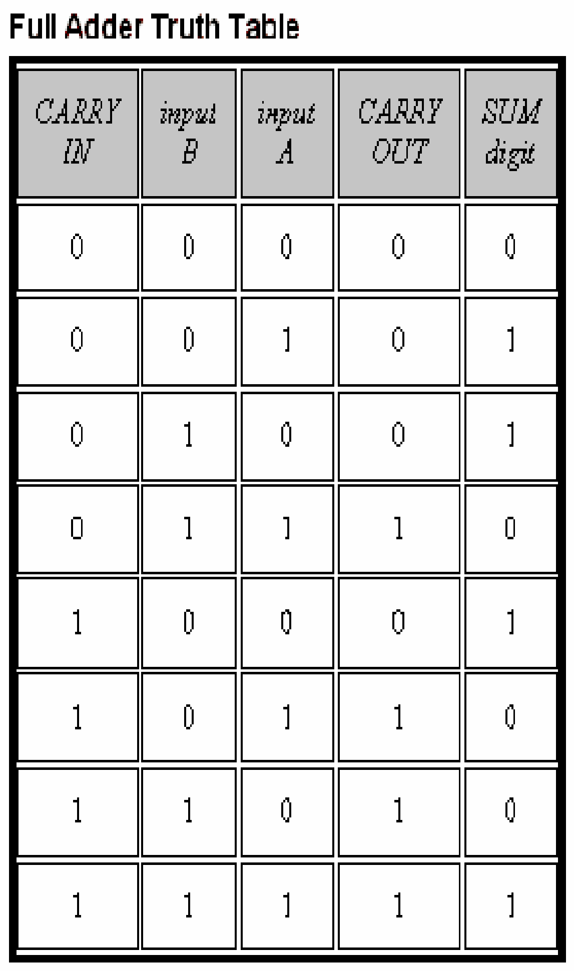
Fig. 4 FA realisation using gates.

FA Logical Expressions are as below.

$$S = \bar{A}\bar{B}C_{IN} + \bar{A}B\bar{C}_{IN} + A\bar{B}\bar{C}_{IN} + ABC_{IN}$$

$$S = A \oplus [B \oplus C_{IN}]$$

$$C_{OUT} = BC_{IN} + AC_{IN} + AB$$



Full Adder Truth Table

CARRY IN	input B	input A	CARRY OUT	SUM digit
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

V. SIMULATION RESULTS

The VHDL code for the full adder is given below and is simulated using Edvin XP simulator .The waveform of the full adder is also represented.

VHDL code for Full Adder.

```
library ieee;
use ieee.std_logic_1164.ALL;    --- can be different
dependent on tool used.
use ieee.std_logic_unsigned.ALL;  -- can be different
dependent on tool used.
entity fa is
   port (a        : in  std_logic;
         b        : in  std_logic;
         c_in      : in  std_logic;
         sum       : out std_logic; -- sum out of X+Y
         c_out     : out std_logic
-- carry out
        );
```

```
end fa;

architecture rtl of fa is          --  Define internal signals
  signal sum_low : std_logic;
  signal c_low   : std_logic;
  signal c_high  : std_logic;

                       --  Define the entity of the half adder to
instansiate


component ha  -- "ha" must be same name as used in the
entity for the file
   port (X         : in  std_logic;
         Y         : in  std_logic;
         Z         : out std_logic; -- sum out of X+Y
         C         : out std_logic  -- carry out
        );
end component;   --------- end of entity for ha ----------


begin
ha_low : ha
   port map (
   --  ha-side     fa-side
         X    =>    a,
         Y    =>    b,
         Z    =>    sum_low,
         C    =>    c_low
    );          --------- end of port map for "ha_low" ----------

ha_high : ha

port map (
   --  ha-side     fa-side
         X    =>    sum_low,
         Y    =>    c_in,
         Z    =>    sum,
         C    =>    c_high
    );          --------- end of port map for "ha_high" ----------

   c_out <= (c_low OR c_high);

end rtl;
```

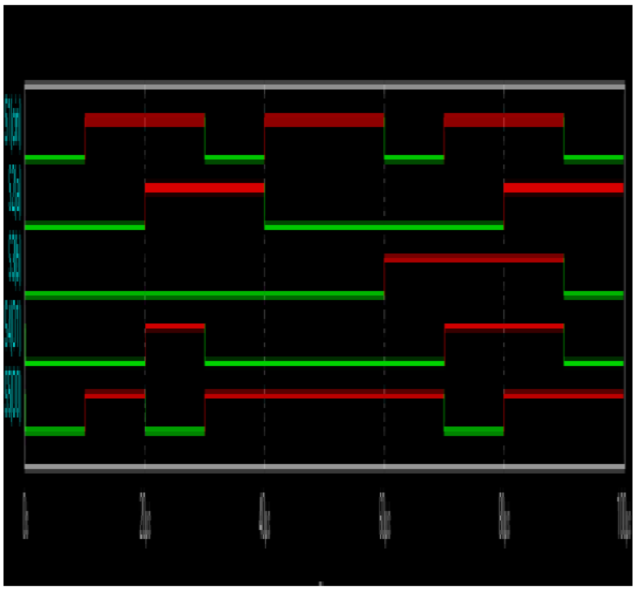


Fig.5 Simulated waveforms

VI. CONCLUSION

The Full adder is simulated and verified for the delay. Further delay can be reduced by choosing proper aspect ratio for the transistors. This full adder can be used in systolic arrays for increased throughput, speed and parallelism.

REFERENCES

1. Jonathan Break, “Systolic Arrays & Their Applications”
2. M. Kunde, H.W. Lang, M. Schimmler, H. Schmeck, H. Schröder: The Instruction Systolic Array and its Relation to Other Models of Parallel Computers. Parallel Computing 7, 25-39 (1988)
3. H.W. Lang: The Instruction Systolic Array, a Parallel Architecture for VLSI. Integration, the VLSI Journal 4, 65-74 (1986)
4. Sung Burn Pan, Seung Soo Chae and Rae-Hong Park, VLSI Architecture for Block Matching Algorithms using Systolic Array, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 6, No. 1, February 1996.
5. M. Shams, T. K. Darwish and M. A. Bayoumi, “Performance Analysis of Low-Power 1-Bit CMOS Full-Adder Cells,” IEEE Trans. on VLSI Systems, vol. 10, Feb. 2002.
6. N. Zhuang and H. Wu, “A New Design of the CMOS Full-Adder,” IEEE J. Solid-State Circuits, pp. 840- 844, May 1992.
7. Samir Palnitkar, “Verilog HDL, A Guide to Digital Design and Synthesis", Pearson Education,ISBN: 81-7758-918-0
8. M. Shams and M. A. Bayoumi, “A Novel High Performance CMOS 1-Bit Full-Adder Cell,” IEEE Trans. on Circuits and Systems II: Analog and Digital Signal Processing, pp. 478-481, May 2000.
9. Abu Shama, A. Elechouemi, S. Sayed and M. Bayoumi, “An Efficient Low Power Basic Cell for Adders,” Proc. 38th Midwest Symposium on Circuits and Systems, pp. 306-309, 1996.

AUTHORS PROFILE

Mrs. Pratibhadevi Tapashetti received M Tech degree from Visvesvaraya Technological University in VLSI and Embedded Systems and currently pursuing Ph.D from NIMS University, Rajasthan, India in the area of Wireless Communication from the Department of Electronics And Communication Engineering. She is having 15 years of experience in academics. She has published several papers at various national/International Journals/Conferences. She is a member of IETE, ISTE IAENG and CSTA. Currently she is working as Professor and Head, Department of Electronics and Tele Communication Engineering, KITE,Raipur, CG, India. Her research interests are VLSI design, Embedded systems, Wireless communications and computer networks.

Dr. A S Umesh received M E degree from Bangalore University in Computer science & Engineering and the Ph.D from Magadh University, India from the Department of Computer science and Engineering. He is having more than 18 years of experience in academics. He has published several papers at various national/International Journals/Conferences. He is a member of IEEE, ISTE, IAENG and CSTA. Currently he is working as Principal, KITE,Raipur, CG,India. His research interests are Computer Networking, Wireless communications, ad hoc networks and VOIP.

Dr.Ashalatha Kulshrestha is working as a Director Kankeshwari Devi Institute of technology Jamnagar Gujarat, India. She is B.Sc.,B.E.Sc (Telecomm.),M.E(microwave)Ph.D. With 44 years of experience in teaching as well as in industry.
She has published several papers at various national/International Journals/Conferences. She has published many articles on education in news papers. Her research interests are Wireless communications, Microwave communications, and Computer networks. She is a member of many institutional bodies.