International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 07, July-2019

ASIC Implementation of High Speed and Low Power ALU

Mr. Madhukar G
Student,
Dept ECE, Dr. AIT
Bengaluru

Mrs. Akalpitha L Kulkarni
Associate Professor
Dept ECE, Dr.AIT
Bengaluru

Dr. J S Baligar
Associate Professor
Dept ECE, Dr.AIT
Bengaluru

***Abstract* ---In this paper, we have presented a design of arithmetic and logic unit for low power application with high speed. In this we have designed an ALU which consists of four arithmetic and four logic operations. Arithmetic part of the ALU is designed with the architectures which have less power dissipation and high speed. The modules are written in Verilog HDL. Xilinx ISE 14.7 software is used for simulation. For synthesis and physical design cadence innovus software is used.**

Key words ---Low power, High speed, Cadence innovus.

I. INTRODUCTION

Arithmetic and logic unit is the necessary part of any central processor which perform arithmetic and logic function required for processor and it is controlled by control unit. Designing of ALU have high scope ever because performance of any processor depends on its ALU efficiency. Arithmetic and logic unit is multifunctional unit, performs one of few arithmetic and logic functions with two operand.

ALU, the most complex building block in the Microprocessor, performs different arithmetic and logic functions and consumes high power in microprocessors. So, it is desirable to design power efficient ALU meeting the performance requirement of applications. Now days the portable devices market growing rapidly. These portable devices uses battery and battery is packed with certain amount of power. As long as the device uses less power we achieve longer battery life for device performance. This is one of the main reason to turn the designing world into low power devices. Although exes leakage of power in the circuit causes circuit damage in VLSI circuit. In this design we trying to decrease power consumption of ALU.

The key parameters for the performance measure of any processor is speed. As ALU is the main block of processor the speed of processor is largely dependent on ALU speed. Any embedded processors working in real time required to be working very fast. Some real time embedded system processors the late result cause in the functional wrong result. So, it is desirable to design high speed ALU.

II. DESIGN

Arithmetic and Logic Unit (ALU) works as a data processing unit which is an important part in the central process unit (CPU) of any computer architecture. ALU is a multi-functional circuit that performs one of a few possible functions on two operands and which depends on the control inputs. In this paper ALU is designed that performs 4 Logical Operations and 4 Arithmetic operations. Different ALU functions are selected according to the selection lines of the MUX listed in the Table 1.

Select line	Operation
000	Addition
001	Subtraction
010	Multiplication
011	Division
100	AND
101	OR
110	XOR
111	NOT

Table 1 : Selection of operation.

In this arithmetic and logic unit design various arithmetic circuits are tested and selected the best one in regard of its operation speed and power consumption. This ALU is built using Brent kung adder and subtractor, Booth multiplier and restoring division algorithm.

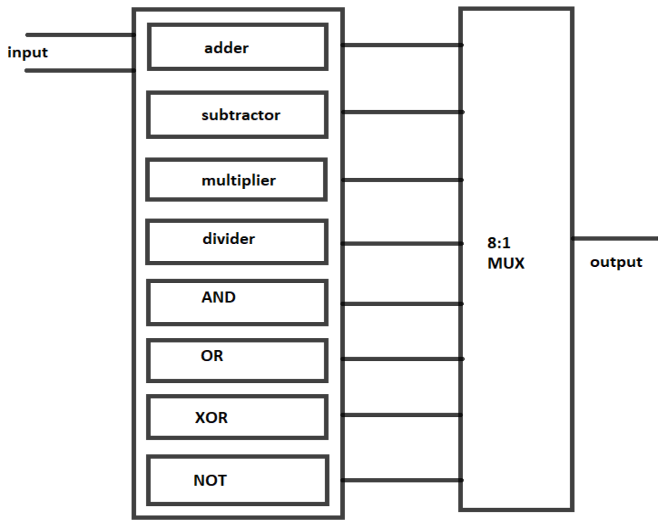

Fig 1: Block diagram of ALU.

1. *Adder and Subtractor:*

Addition and subtraction are frequently used in ALU. By using low power architecture for these operation we can achieve low power ALU. For this we used Brent kung adder technique. Brent kung adder is one type of parallel prefix adder. Parallel Prefix adder are the ones widely used in Digital Design. This is primarily because of the flexibility in designing. Power consumption of Brent Kung adder is very less because it requires less circuitry. Parallel prefix adder operation we can categorize the addition operation in three different stages.

i. First stage is the Pre-processing stage where we obtain the Group Generate and Group Propagate signals.
ii. Second stage is the Carry generation stage where we generate the carry using the Group Generate and Group Propagate signals.
iii. Third and Final stage is where we obtain the Sum bit using the Carry bit and the Propagate signal.

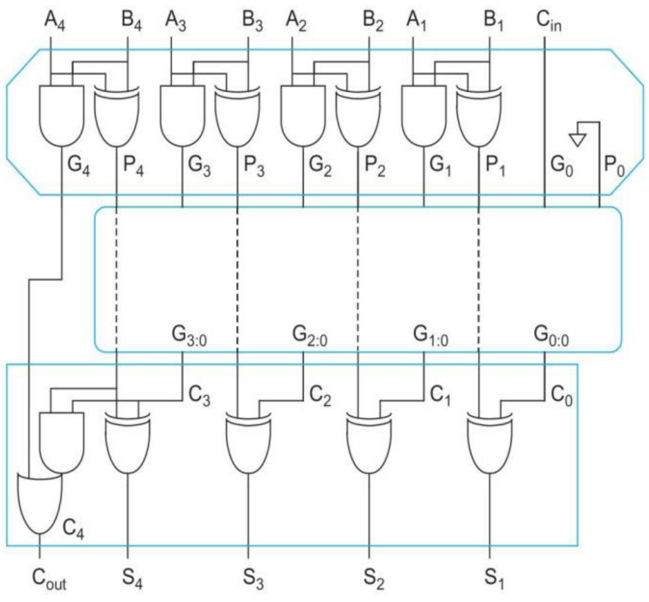

Fig 2: Parallel prefix stages

Equations for propagate and generate signals:
Gi:j = Gi:k + Pi:k Gk-1:j
Pi:j = Pi:k Pk-1:j
The basic case is,
Gi:i = Gi = Ai Bi
Pi:i = Pi = Ai ^ Bi
The group generates a carry if the upper (i.e. the more significant) or the lower portion generates
and the upper portion propagates the carry. If both the upper and lower portions propagate the
carry, the group propagates.
The input carry is also equally important. Take the input carry Cin as C0 = Cin and CN = Cout.
The Generate and Propagate signals for the bit 0 will be
G0:0 = Cin
P0:0 = 0.
The carry into the ith bit is the carry-out of (i-1)th bit and Ci-1 = Gi-1:0 .

The Sum for the ith bit is computed as

Si = Pi ^ Gi-1:0

2. *Multiplier:*

Multiplication is the most using operation in any digital processing. To achieve high speed and low power ALU multiplier should be very efficient in terms of speed and power consumption. In this design we used Booth multiplier. Comparatively Booth multiplier power consumption is lesser than Array and Wallace. Although Wallace multiplier has more speed than Booth multiplier, it uses larger area and its power consumption is high.

Multiplication requires repeated addition partial product it is very time consuming.
Computers are faster in shifting bits than addition, by knowing this Booth developed Booth algorithm for multiplication. Booth algorithm reduces the number of addition that takes place during multiplication.

Booth's Algorithm:

1) Assume product.
   Product: {0000, Multiplier, 0}

2) Now, take the two least significant bits of the product and depending on the value proceed with one the following:
   a) 00: No changes to product.
   b) 01: Add the multiplicand to the left side of the product.
   c) 10: Subtract the multiplicand from the left side of the product.
   d) 11: No changes to product.
3) Right shift the product by 1 bit.
4) Repeat the process x number of times, where x is the number of bits in the multiplicand.

3. *Division:*

Division is very complex arithmetic operation in an ALU. It requires larger area and power, therefore some cheap ALU designed for small processors exclude the division operation. Improving of division algorithm definitely increase the performance of the ALU. In this design we used Restoring algorithm for division. Restoring algorithm is very simple algorithm and consumes less power.

Restoring algorithms use subtractive methods to calculate quotients one digit per iteration. Divisor is subtracted from left bits of dividend. If the result of subtraction is negative than then restore the dividend value and pad 0 to quotient. If the subtraction result is positive then take it as the new dividend and pad 1 to quotient. At the end the dividend value is the remainder.

III. RESULTS AND DISCUSION:

The above defined high speed and low power ALU design is first check for the functional correctness. The design of high speed and low power ALU defined in this paper is simulated using the Cadence Innovus NC simulator. Simulation result shows that the designed ALU working functionally correct.

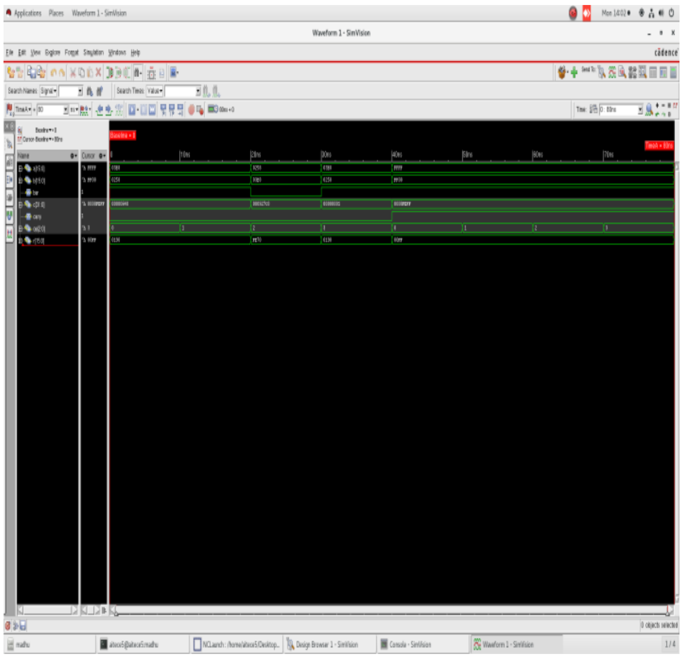
Fig 3: simulation result in cadence

The schematic for the high speed and low power ALU design is generated using the cadence innovus tool. At this stage tool generated the power, area and timing report.

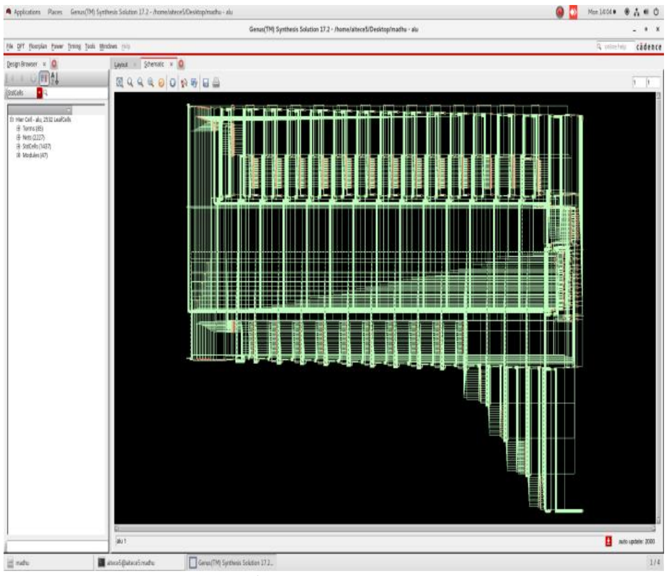
Fig 4:schematic of high speed and low power ALU

Leakage power (in mw)	Dynamic power (in mw)	Total power (in mw)	Area (μm^2)
0.086075	1.1009	1.18701	16664.667

Table 2: Power and area report.

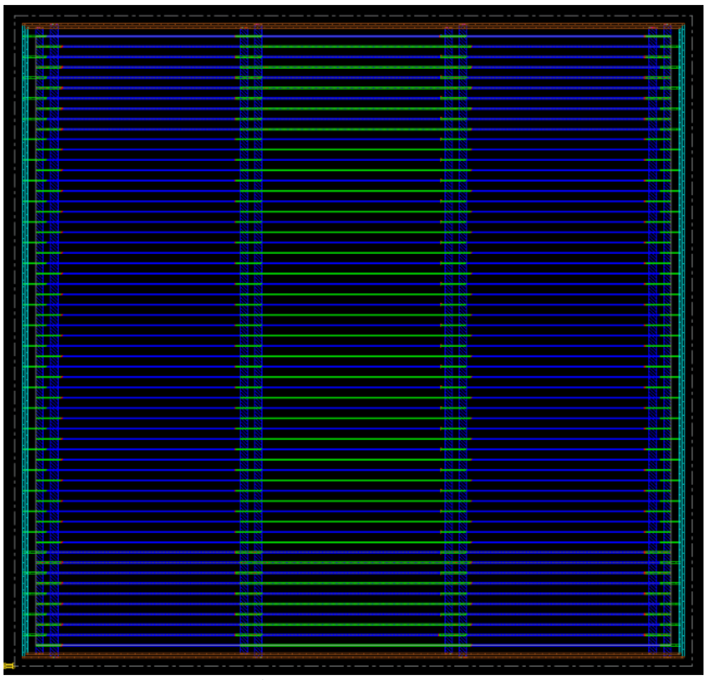
Fig 5:Partitioning of high speed and low power ALU.

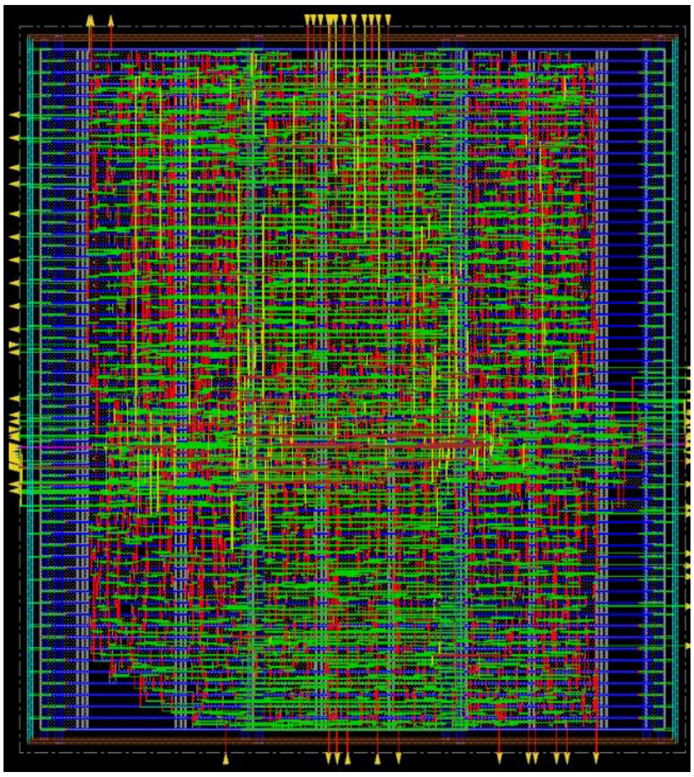

Fig 6:final layout of the high speed and low power ALU

IV. CONCLUSION

This work shows the ASIC implementation of high speed and low power ALU. The ALU was designed for 16 bit integer data with four logic and four arithmetic operations. The ALU was designed using the arithmetic circuits which have less power consumption and high speed. Hence the speed of the ALU was increased with low power consumption.

The proposed work can be expanded to support floating point operations and trigonometric functions. And also this ALU can be expanded to perform on the higher bits of data.

REFERENCES

[1] Rajit Ram Singh, Asish Tiwari, Vinay Kumar Singh, Geetam S Tomar, "VHDL environment for floating point arithmetic logic unit design and simulation" International Conference on Communication Systems and Network Technologies, 2011.

[2] Bishwajeet Pandey, Jyotsana Yadav, Manisha Pattanaik, "IO Standard Based Energy Efficient ALU Design and Implementation on 28nm FPGA" on 2013 Annual IEEE India Conference (INDICON)

[3] Nidhi Gaur, Anu Mehra, Deepika Kamboj and Devyani Tyagi, "A Novel Implementation of 32 bit Extended ALU Architecture at 28nm FPGA"

[4] Rohita Watpade and Prasanna Palsodkar, "BSD Adder for Floating Point Arithmetic: A Review" in International Conference on Communication and Signal Processing, April 6-8, 2017, India.

[5] Chaithra T G, Pradeep kumar S K and Nandini K, "ASIC Realization and Performance Evaluation of 64×64 Bit High speed Multiplier in CMOS 45nm using Wallace Tree" in 2017 2nd IEEE International Conference On Recent Trends in Electronics Information & Communication Technology (RTEICT), May 19-20, 2017, India.

[6] Savita Patil, D.V.Manjunatha and Divya Kiran, "Design of Speed and Power Efficient Multipliers Using Vedic Mathematics with VLSI Implementation" in 2014 International Conference on Advances in Electronics, Computers and Communications (ICAECC).

[7] Harpreet Kaur, "Performance analysis of various multiplication and division algorithms for large numbers"

[8] Savita Nair, Ajit Saraf "A review paper on comparison of multipliers based on performance parameters", International conference on advances in science and technology (ICAST - 2014).