MANTECH Publications

Journal of Research in VLSI Design Tools and Technology

Volume 7 Issue 1, January-April 2022

VHDL-Based 4-Bit Low Power Arithmetic Logic Unit Design

Supriya Rathode[1], Lalita Kumari[2]

Assistant Professor[1], Student[2]

Department of ECE

NIT Jamshedpur, Jharkhand, India

Corresponding Author's E-mail: *lalitakumari2222@gmail.com[1]*

Abstract

We will learn how to utilise VHDL to develop a low power ALU in this paper. Following Moore's law, developments in VLSI technology have allowed for a three-year doubling in component density on a silicon chip. As the basic function of a transistor has improved, designs may now be realised with fewer transistors and fewer interconnections. Many integrated circuits with traditional CMOS technology that use multi-input floating gate MOSFETs have been disclosed in the literature. The proposed ALU architecture is more efficient as a consequence of the use of modern VLSI technology.

Keywords: *VHDL, VLSI, ALU, CMOS, 4-BIT LOW POWER*

INTRODUCTION

An arithmetic logic unit (ALU) is a digital circuit in computing that performs arithmetic and logical operations. The ALU is a fundamental building component of a computer's central processing unit (CPU), and even the most basic microprocessors have one for reasons such as timer maintenance. Modern CPUs and graphics processing units (GPUs) include processors that house highly powerful and complicated ALUs; a single component may have many ALUs. In 1945, mathematician John von Neumann suggested the ALU concept in a study on the foundations of a new computer called the EDVAC. ALU research is still an essential aspect of computer science, and is classified under Arithmetic and logic structures in the ACM Computing Classification System.

A. System Operations

A basic arithmetic logic unit can do AND, OR, XOR, and addition. The majority of ALUs can execute the following operations: Arithmetic operations on integers (addition, subtraction, and sometimes multiplication and division, though this is more expensive). Operations on bits in logic (AND, NOT, OR, XOR). Operations involving bit-shifting (shifting or rotating a word by a specified number of bits to the left or right, with or without sign extension). Shifts are equivalent to multiplications by 2 and divisions by 2.

B. System Inputs & Outputs

The data to be operated on (called operands) and a control unit code indicating which operation to perform are the inputs to the ALU. Its output is the computation's outcome. In many configurations, the ALU additionally accepts or creates a set of condition codes from or to a status register as inputs or outputs. These codes are used to identify situations such as carry-in or carry-out, overflow, divide-by-zero, and so on. The core of a CPU in a computer is the arithmetic logic unit (ALU). The basic unit of an ALU is the adder cell. The limits that the adder must meet include area, power, and speed requirements. Traditional adders include ripple-carry adders, carry-look ahead adders, carry-skip adders, and Manchester carry chain adders. The carry chain dominates the delay in an adder. Carry chain analysis must take into account transistor and wire delays. The ripple carry adder is an n-bit adder made up of complete adders. Figure 1 depicts a four-bit ripple carry adder. Carry-ahead adders compute carry propagate and create first, then SUM and CARRY from these. It enables carry to be calculated in each bit.

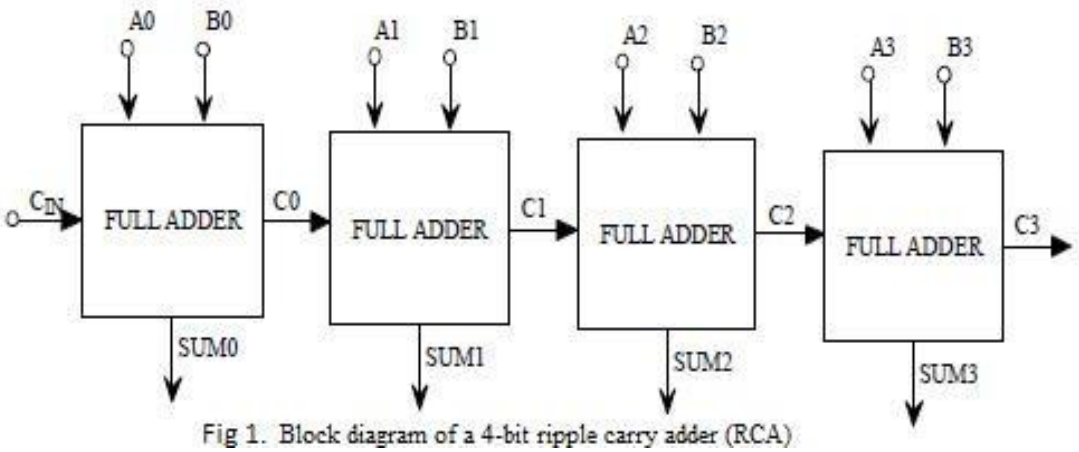

Fig 1. Block diagram of a 4-bit ripple carry adder (RCA)

A 4-bit carry-look ahead adder is shown in Figure 2. Carry-look ahead unit necessitates complex wiring between adders and look ahead unit, as values from the look ahead unit must be transmitted back to the adder. With numerous layers of look ahead, the layout becomes more complicated.

Figure 1depicts the use of a 4bit carry-skip adder and skip module. The skip module decides whether it may simply transmit a carry in (CIN) the next four bits for addition or if it must wait until the carry out (C3) propagates via the design's last complete adder. The skip module, in essence, may make the carry in (CIN) appear to skip through the four complete adders. A precharged carry chain containing P and G signals is used by the Manchester carry chain adder. Signal Pi is the XOR of input bits Ai and Bi, whereas signal Gi is the NAND of the same bits. The Propagate signal links neighbouring carry bits, while the Generate signal discharges the carry bit.

When all of the input bits are "0," Gi is HIGH, and the carry out node is discharged. When one of the input bits is "1," Pi is HIGH, and carry out comes after carry in. When both bits are "1," both Gi and Pi are LOW, and the carry out node is isolated from the carry in and ground nodes. Because the node has been pre-charged to HIGH, the carry out stays HIGH. Aside from the whole adder architecture, each of the adder combinations may or may not require extra logic. Table 2.1 indicates how many more gates and transistors are needed for each adder arrangement. The ripple carry adder is recommended in terms of area efficiency. Our ALU was developed with ripple carry in mind due to the short layout area and low number of interconnections. However, the worst-case delay time is longer when compared to other adders.

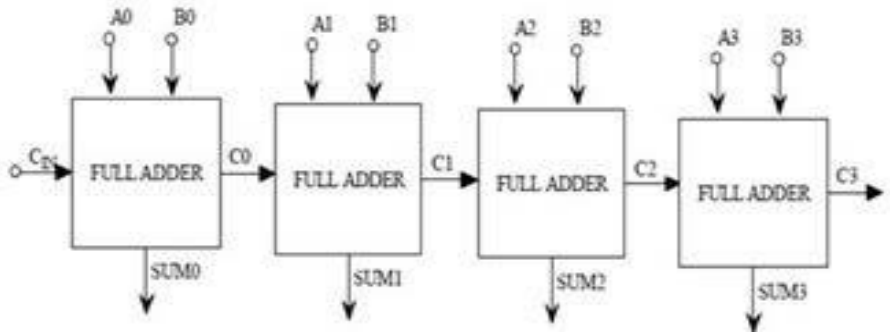

Figure 2.Block diagram of a 4-bit carry-look ahead adder (CLA).

C. System Block Diagram

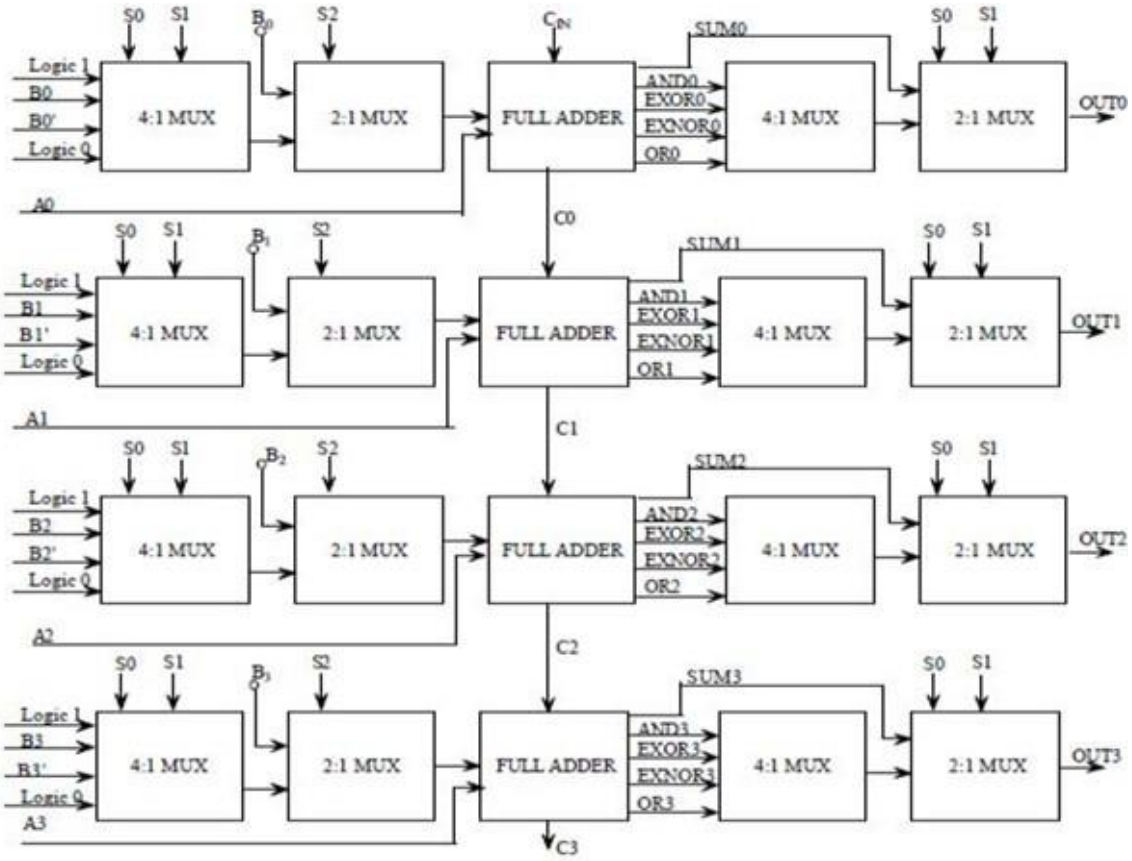

Figure 3.Block diagram of a 4-bit ALU

Figure 3 depicts the system block diagram of a 4-bit ALU. A combinational circuit that performs logic and arithmetic micro-operations on a pair of n-bit operands is known as an ALU (ex. A [3:0] and B [3:0]). An ALU's operations are controlled by a set of function select inputs. This design has a 4-bit ALU with three function select inputs: Mode M, Select S1 and S0. The mode input M allows you to choose between Logic (M=0) and Arithmetic (M=1) operations.

System Specification

The 4-bit ALU is made up of 4 to 1 and 2 to 1 multiplexers on the input and output sides, as well as a complete adder with extra logic. The entire adder is set up as a ripple carry adder. The select signals S0, S1, and S2 determine which operation is being executed. The post-layout simulates waveforms for the whole adder, displaying SUM and CARRY bits. Table 2 shows the truth table for the eight operations performed by the ALU. 1 Signal S2 is linked to logic "0" for arithmetic

operations and logic "1" for logical operations. The ultimate output is determined by the multiplexer logic on the output side of each level of the ALU. For logical operations, each stage's output is independent of the others. The carry ripples from LSB to MSB location in arithmetic operations. As a result, the output of each step is dependent on the preceding stage. Four steps of the complete adder are cascaded in ripple carry adder arrangement for the 4-bit ALU circuit. Figure 3.4 depicts the layout of the 4 bit ALU. The whole arrangement was contained within the 1.5 m pad frame. On the pad frame, connections were created for inputs, outputs, supply voltages, and ground pins.

Table 2 Truth Table of a 4-bit ALU

Table 2.1 Truth table of a 4-bit ALU

S_2	S_1	S_0	Operation performed
0	0	0	INCREMENT
0	0	1	DECREMENT
0	1	0	ADDITION
0	1	1	SUBTRATION
1	0	0	AND
1	0	1	OR
1	1	0	EXOR
1	1	1	EXNOR

Simulation Results & Analysis

Softwares such as Xilinx 11.1 and ModelSim SE 5.7 are used for synthesis and simulation, and are then utilised for developing, testing, and simulation of VHDL applications. Figure 3.4 depicts the architecture of a 4-bit ALU using Mentor Graphic 2007. The design was created using the AMI 1.5mm CMOS process. The 4-Bit ALU measures roughly 830 x 935 mm2. For post-layout extracted net lists, SPICE simulations for the 4-bit ALU were performed.

CONCLUSION

The design, optimization, and implementation of a 4-bit ALU are presented in this study. The growing need for low-power very large scale integration (VLSI) may be met at several design levels, including architectural, circuit, layout, and process technology. The simulation findings reveal an improvement in output signal latency and a reduction in waveform distortion at the output stages. Because of the significant benefits, the suggested architecture may be suited for DSP applications.

REFERENCES

1. N. Burgess. “The flagged prefix adder for dual additions”. In Proc. SPIE ASPAAI-7, volume 3461, pages 567-575, San Diego, Jul. 1998.

2. N. Quach and M. Flynn.“Design and implementation of the snap floating-point adder”.Technical

Report CSL-TR-91-501,Stanford University, Dec. 1991.

3. R.V.K. Pillai, D. Al-Khalili and A.J. Al-Khalili"Power Implications of Additions in FloatingPoint DSP-an Architectural Perspective" Concordia University, Montreal CANADA; RoyalMilitary College, Kingston, The IEEE International ConferencePages: 581-586, 1999.

4. Javier D. Bruguera and Tomas Lang "Leading one anticipation scheme for latency improvement in single data path floating point adders", Department of Electrical and Computer Engineering,Spain. Pages 125-133, 2005.

5. Ricardo Gonzalez, Benjamin M. Gordon, and Mark A. Horowitz," Supply and Threshold VoltageScaling for Low Power CMOS",IEEE JOURNAL OF SOLID-STATE CIRCUITS, VOL. 32, NO.8, AUGUST 1997.

6. E. Hokenek and R. Montoye."Leading-zero anticipator (lza) in the ibmrisc system/6000 floating-point execution unit". IBM J.Res. Develop., 34 (1): 71-77, Jan. 1990.

7. S. Kang and Y. Leblebici "CMOS Digital Integrated Circuit, Analysis and Design" (Tata McGraw-Hill, 3rd Ed, 2003).

8. M. Farmwald., "Design of High PerformanceDigital Arithmetic Units".PhD thesis, Stanford University, Aug. 1981.

9. M. Morris Mano "Digital Design" (PearsonEducation Asia. 3rd Ed, 2002).