Faculty of Engineering and Technology

Electrical and Computer Engineering Department

DIGITAL ELECTRONICS AND COMPUTER ORGANIZATION LABORATORY

ENCS2110

Experiment No. 8

Half-Adder, Full-Adder Implementation Using Quartus

Prepared By: **Ghaith Haj-Ali** → **1220612**

Partners: **1. Abd Al-Rahman Sia'rah** → **1230219**

2. Zaid Qamhieh → **1230069**

Instructor: **Nasser Ismail** T.A: **Raneem Al-Qaddi**

Section: **3** Date: **21-Sep.2025**

Abstract

In this experiment, Half Adder and Full Adder circuits were designed and tested using Verilog in the Quartus environment. The Half Adder was first implemented to add two single-bit inputs, generating the Sum and Carry outputs, which illustrated the basic principle of binary addition.

The Full Adder was then constructed by combining two Half Adders and an OR gate, allowing the circuit to handle three inputs: two data bits and a carry input. This configuration produced both a Sum and a Carry output, making the Full Adder suitable for multi-bit addition when cascaded.

The operation of both circuits was verified using simulation waveforms, logic symbols, and block diagrams, which confirmed their correctness under all input conditions.

In addition, a multiplexer was implemented alongside the Full Adder to demonstrate controlled output selection. By applying control signals, the multiplexer allowed specific outputs to be chosen, and its functionality was also validated through simulation.

Overall, the experiment covered the complete design and verification cycle, showing how adders and multiplexers form essential building blocks in digital systems.

Table of Contents

List Of Tables

Table of Figures:

Theory

Half-Adder:

A Half Adder is a basic digital circuit designed to perform the addition of two binary inputs. It accepts inputs A and B, generating the Sum through an XOR gate and the Carry through an AND gate. While it is effective for simple one-bit addition, it cannot manage carry propagation between stages, and therefore, it is mainly used as a fundamental component in constructing Full Adders.

Figure 1: Half-Adder Block Diagram

- A Half Adder can be implemented using only NAND gates. Since NAND is a universal gate, both the XOR function (for the Sum) and the AND function (for the Carry) can be derived from NAND gate combinations. For example, in a Half Adder with inputs A and B, a set of NAND gates is arranged to form the XOR output, while another NAND configuration produces the AND output. This design demonstrates that a complete Half Adder can be realized using just a single type of logic gate.

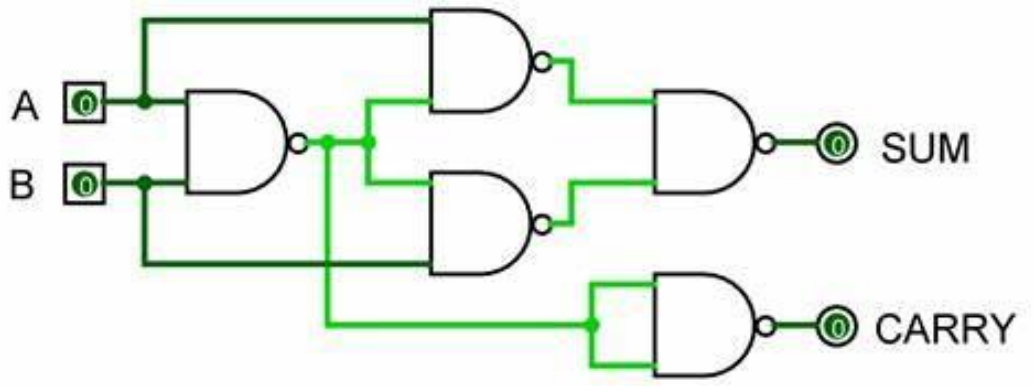

Figure 2: Half-Adder Implementation Using Basic Gates

Half-Adder Truth Table:

Truth Table			
Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Table 1: Half-Adder Truth Table

Full-Adder:

A Full Adder is a digital circuit designed to perform the addition of three binary inputs: A, B, and a Carry-In (Cin) from a previous stage. It generates the Sum output using XOR logic and produces the Carry-Out (Cout) through a combination of AND and OR operations. Unlike the Half Adder, the Full Adder supports carry propagation, which makes it suitable for multi-bit binary addition and forms the basis for more advanced arithmetic circuits.

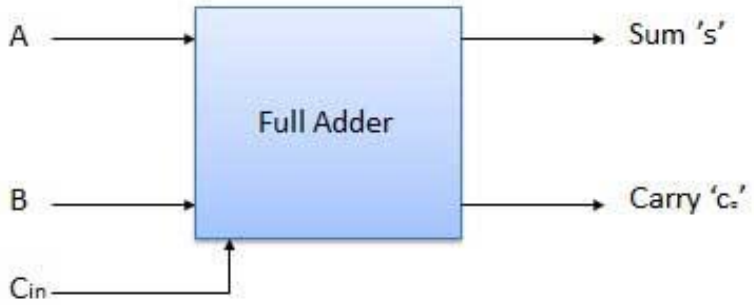

Figure 3: Full-Adder Block Diagram

- A Full Adder can be implemented using only NAND gates. Since NAND is a universal gate, the required functions—XOR for the Sum and AND-OR combinations for the

Carry-Out (Cout)—can all be derived using NAND gate networks. For inputs A, B, and Cin, a group of NAND gates is arranged to generate the XOR-based Sum output, while additional NAND configurations replicate the AND and OR logic needed to produce the Carry-Out. This approach shows that an entire Full Adder circuit, capable of handling three inputs and supporting carry propagation, can be constructed using only NAND gates.

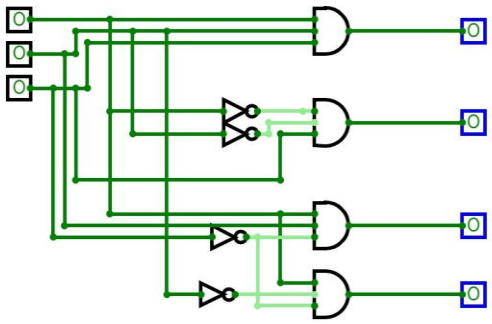

Figure 4: Full-Adder Implementation Using Basic Gates

Full-Adder Truth Table:

Inputs			Outputs	
A	B	C_{in}	C_{out}	S
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

Table 2: Full-Adder Truth Table

Multiplexer:

A Multiplexer (MUX) is a combinational circuit that allows one input to be chosen from multiple data inputs and directed to a single output line. The specific input passed to the output is determined by the values of the **selection lines**, which act as control signals. By transmitting only the selected input and ignoring the rest, the multiplexer provides an efficient method of managing and transferring data within digital systems.

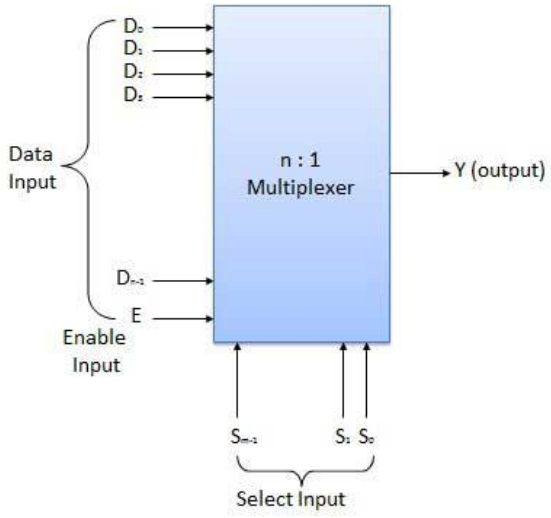

Figure 5: Multiplexer Block Diagram

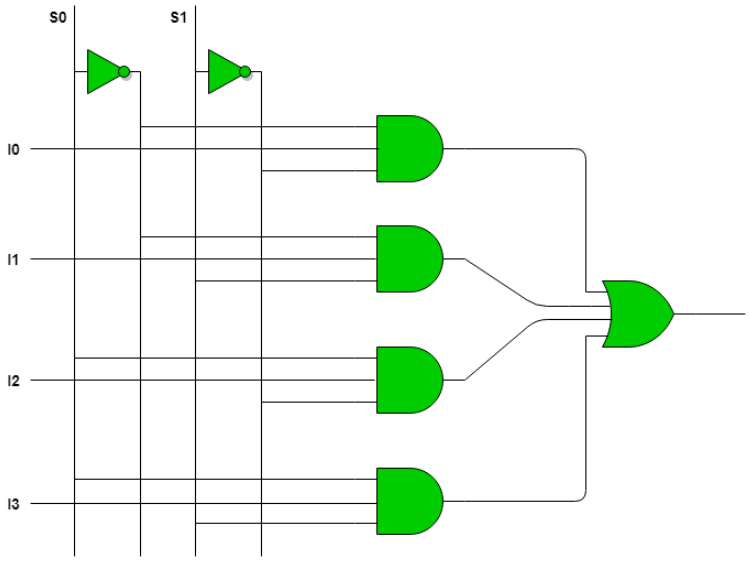

Figure 6: Multiplexer Implementation Using Basic Gates

4-1 Multiplexer Truth Table:

Select		Data				
S1	S0	A	B	C	D	Out
0	0	A	B	C	D	A
0	1	A	B	C	D	B
1	0	A	B	C	D	C
1	1	A	B	C	D	D

Table 3: 4-1 Multiplexer Truth Table

Procedure & Design

- PS: I took Some these data from the web and from my partner ‘Zaid’; Due to some issues in my Quartus on my Laptop. I also wrote the following codes by my hand.

Half-Adder

Half-Adder Verilog Code:

```
module Half-Adder (input A, B, output Sum, Cout);

assign Sum = A ^ B;

assign Cout = A & B;

endmodule
```

Half-Adder *Symbols*:

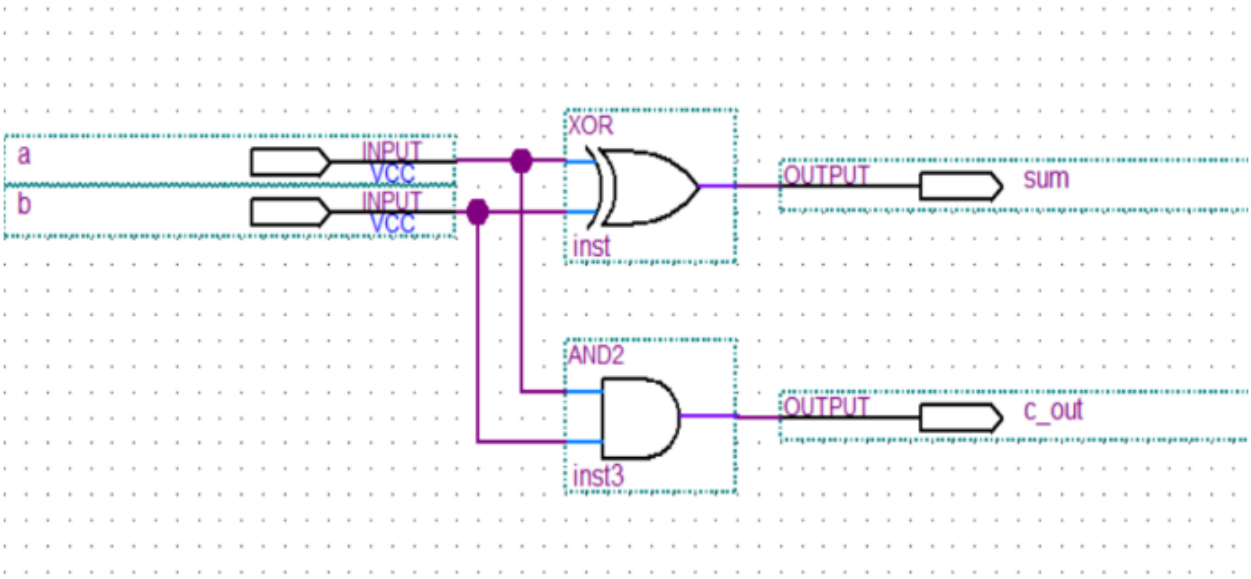

Figure 7: Half-Adder Symbols on Quartus

Half-Adder Waveform:

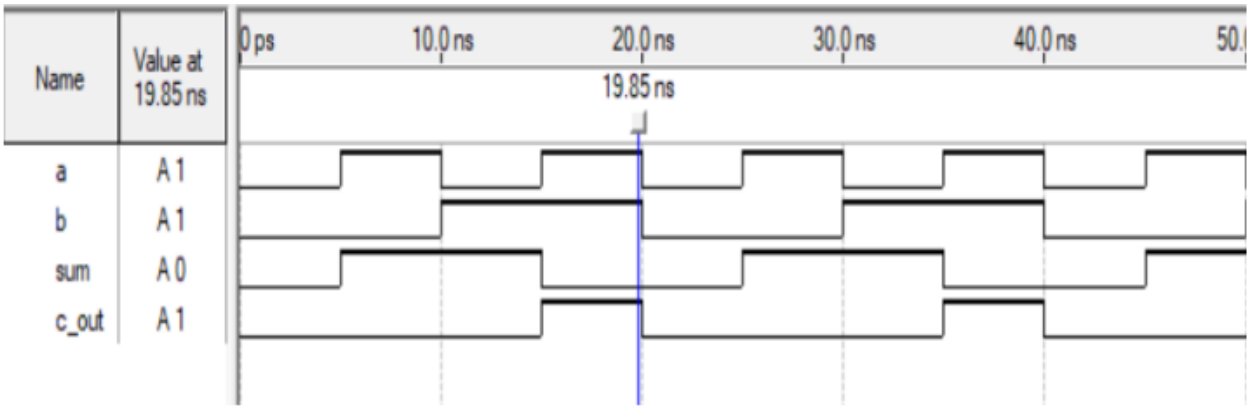

Figure 8: Half-Adder Waveform

Full-Adder

Full-Adder from Half-Adder *Symbols*:

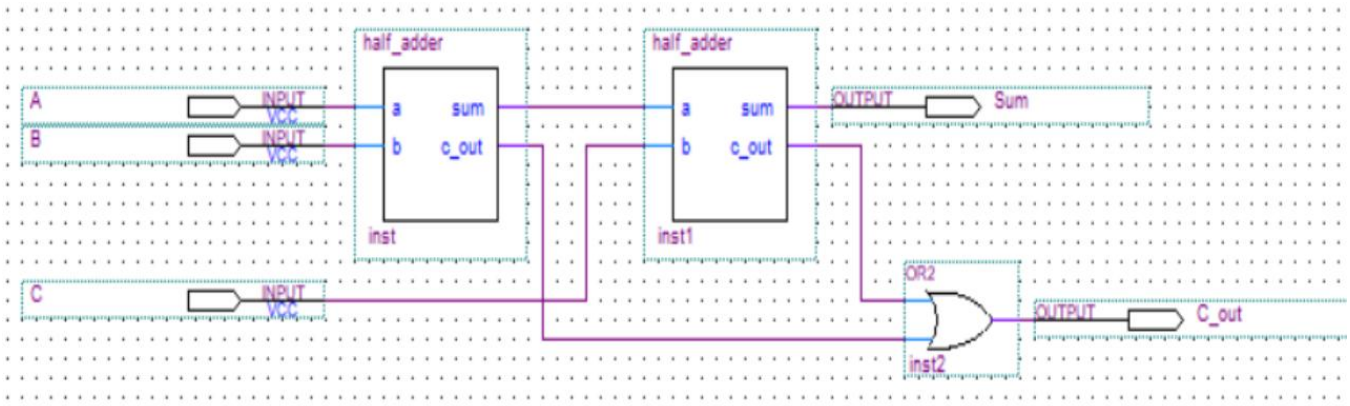

Figure 9: Full-Adder from Half-Adder Symbols

Full-Adder Waveform:

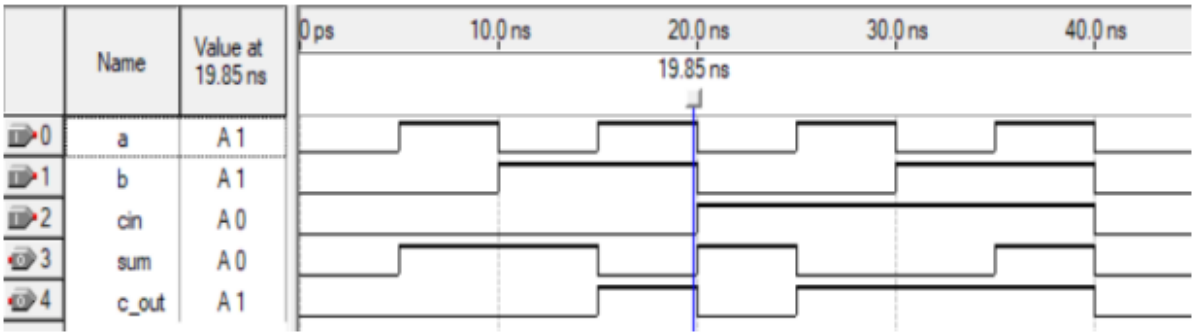

Figure 10: Full-Adder Waveform

FBGA → Full-Adder:

- ***A=0, B=0, C=0 → S=0 & C=0***

Figure 11: FBGA {A=0, B=0, C=0 → S=0 & C=0}

- ***A=1, B=0, C=0 → S=1, C=0***

Figure 12: FBGA {A=1, B=0, C=0 → S=1 & C=0}

- ***A=1, B=1, C=0 → S=0 & C=1***

Figure 13: FBGA {A=1, B=1, C=0 → S=0 & C=1}

- ***A=1, B=1, C=1 → S=1, C=1***

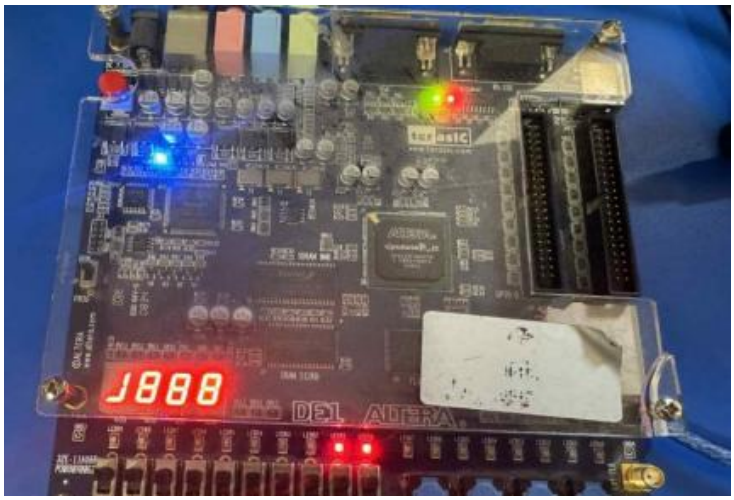

Figure 14: FBGA {A=1, B=1, C=1 → S=1 & C=1}

Multiplexer

Multiplexer Verilog Code:

```
module mux2-1 (input A, B, Se1, output Out);
assign Out = Se1 ? B : A;
endmodule
```

Multiplexer Waveform:

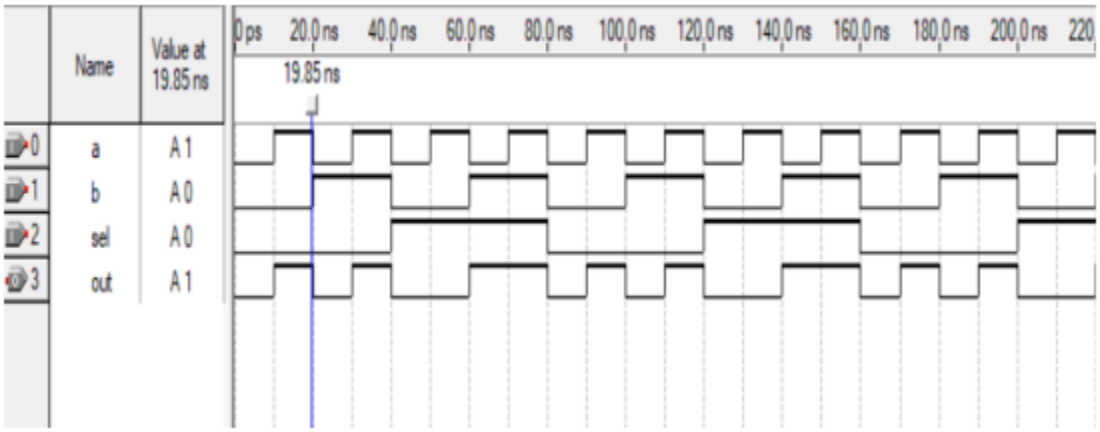

Figure 15: Multiplexer Waveform

Multiplexer from Full-Adder:

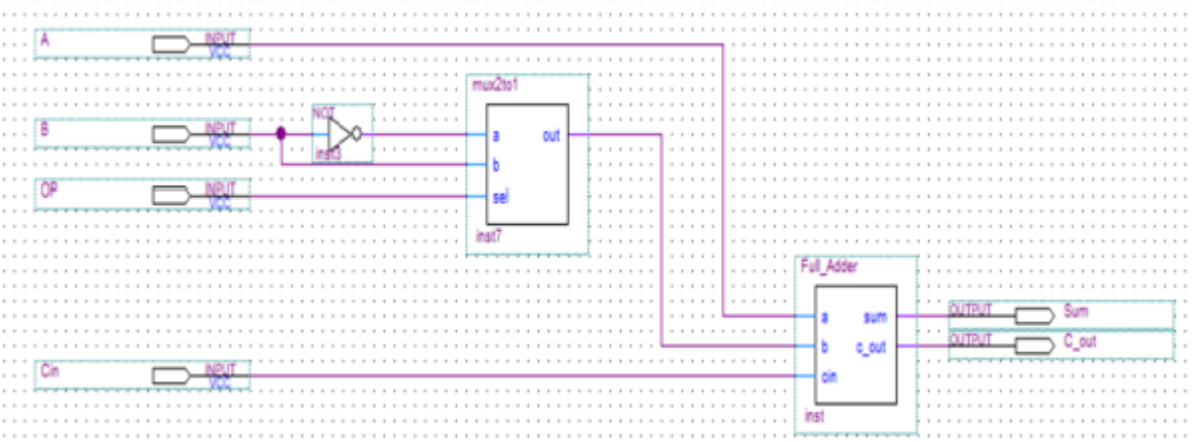

Figure 16: Multiplexer from Full-Adder.

Conclusion

During this lab, we focused on the design and implementation of Half Adders, Full Adders, and Multiplexers using Verilog, followed by testing on an FPGA board. The work provided valuable practical exposure to developing digital circuits, from writing the code to verifying correct operation in both simulated environments and physical hardware.

The activities highlighted how fundamental logic elements can be combined to form circuits capable of performing essential tasks such as binary addition and controlled data selection. By integrating multiplexers with full adders, we were able to demonstrate how outputs can be directed based on control signals. Waveform analysis was also employed to confirm accurate timing and logical behavior.

This experiment not only reinforced key concepts in digital circuit design but also strengthened our ability to apply Verilog for modeling, simulation, and hardware implementation. The skills gained here serve as a strong basis for tackling more complex digital design challenges in future projects.

References & Appendices

- Lab Manual
- Geeks for Geeks