CSE 306
Computer Architecture Sessional

Assignment 01: 4 bit ALU Simulation
Report of Group 01 Section B1

Prepared by:
1705061 - Maksudur Rahaman Rana
1705064 - Tanvir Raihan
1705081 - Md. Kamrujjaman
1705083 - Hozifa Rahman Hamim
1705090 - Ashrafi Zannat Ankon
1305050 - Zaki Tahmeed

Introduction

ALU(Arithmetic and Logic Unit) is a part of the Central Processing Uniit(CPU) which can perform arithmetic(addition,subtraction etc.) and logic(and,or,xor etc.) operations.In this experiment, we have to design a **4 bit Arithmetic and Logic Unit(ALU)** by using combinatorial circuits and Integrated Circuits(ICs). We need to also implement the following status flags :

- Carry
- Sign
- Overflow
- Zero

These flags will change status(0 or 1) corresponding to ALU operations except for the logic operations.

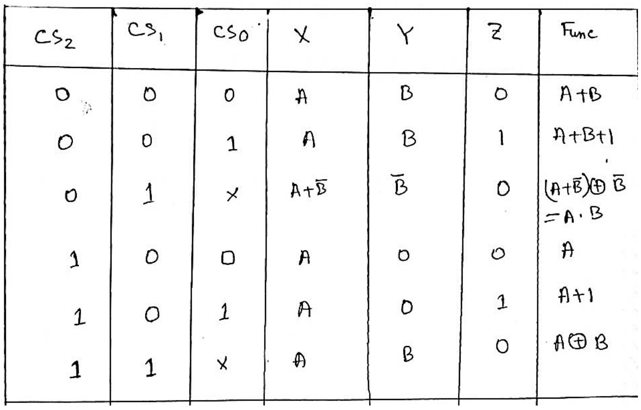
Problem Specifications

CS_2	CS_1	$CS_0(C_{in})$	Function
0	0	0	Add
0	0	1	Add with carry
0	1	x	AND
1	0	0	Transfer A
1	0	1	Increment A
1	1	x	XOR

We need to design a 4-bit ALU that performs the operations mentioned in the table above. We need to use three control signals to simulate the arithmetic and logic operations. From the table, we can observe that CS_1 works as the mode select control bit that switches the ALU operations from arithmetic to logic and vice-versa.. CS_0 is the input carry for our problem. CS_2 is the function select bit that determines which operations to perform. In this assignment, we need to simulate **Six** operations. Four Arithmetic operations:
1. Addition(F = A+B) 2. Addition with carry(F = A+B+1) 3. Transfer A(F = A) 4. Increment A(F = A+1)
and Two logical operations- **1.AND 2.XOR**. According to the instructions, two status flags - Carry and Overflow have to be cleared after the logic operations which means their value will be set to 0. The other two flags will be changed as usual.

Truth Table and K-Maps

Truth Table:



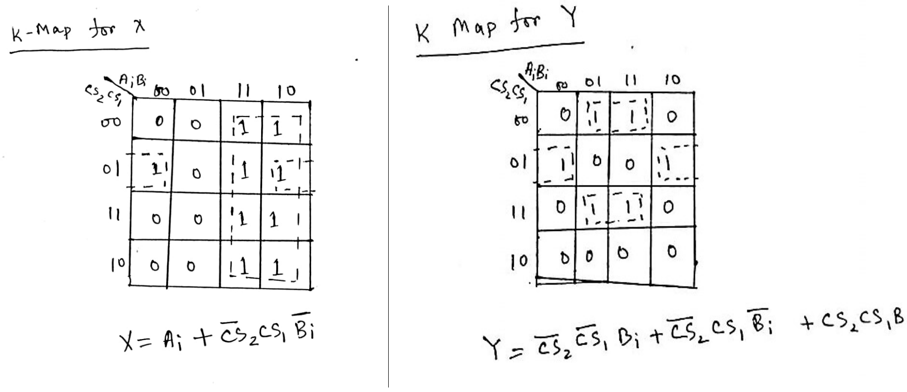
CS_2	CS_1	CS_0	X	Y	Z	Func
0	0	0	A	B	0	A+B
0	0	1	A	B	1	A+B+1
0	1	x	$A+\bar{B}$	$\bar{B}$	0	$(A+\bar{B})\oplus\bar{B}$ $=A\cdot B$
1	0	0	A	0	0	A
1	0	1	A	0	1	A+1
1	1	x	A	B	0	$A\oplus B$

Figure 1: Truth Table

Here, $(A+\bar{B})\oplus\bar{B} = \overline{A+\bar{B}}.\bar{B} + (A+\bar{B}).B = \bar{A}.B.\bar{B} + A.B + B.\bar{B} = A.B$

The figure in 4 shows the truth table for the simulation of the ALU for our assigned instructions. We can observe from the table that CS_1 functions as the mode select control bit. X,Y and Z represents the altered form of input A,B and C_in which will be given as inputs to 1-bit full adder bit by bit and Func denotes the desired function of the ALU.

K-Maps:



K-Map for X

$CS_2CS_1 \backslash A_iB_i$	00	01	11	10
00	0	0	1	1
01	1	0	1	1
11	0	0	1	1
10	0	0	1	1

$X = A_i + \overline{CS_2}CS_1\overline{B_i}$

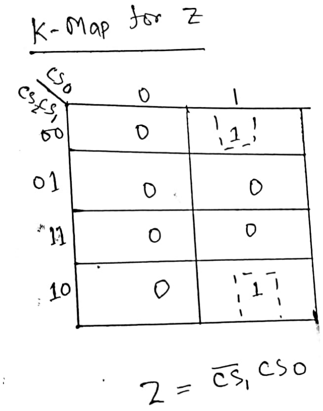
(a) K-map for X_i

K Map for Y

$CS_2CS_1 \backslash A_iB_i$	00	01	11	10
00	0	1	1	0
01	1	0	0	1
11	0	1	1	0
10	0	0	0	0

$Y = \overline{CS_2}\,\overline{CS_1}B_i + \overline{CS_2}CS_1\overline{B_i} + CS_2CS_1B_i$

(b) K-map for Y_i



K-Map for Z

CS_2CS_1 \ CS_0	0	1
00	0	1
01	0	0
11	0	0
10	0	1

$Z = \overline{CS_1}\, CS0$

Figure 2: K-map for Z_i

Here,
$X_i = A_i + \overline{CS_2}CS_1\overline{B_i}$
$Y_i = \overline{CS_2}\,\overline{CS_1}B_i + \overline{CS_2}CS_1\overline{B_i} + CS_2CS_1B_i$ and
$Z_i = \overline{CS_1}CS_0$

Block Diagram

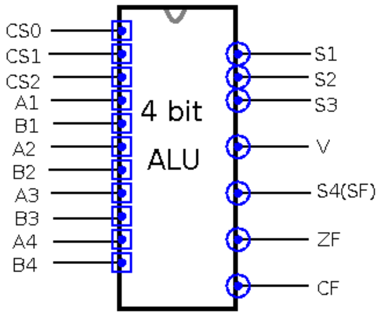


Figure 3: Block Diagram of a 4 bit ALU

The figure in 4 shows a simplified block diagram of a 4-bit ALU. Here **A** and **B** both are 4-bit numbers and Output will also be a 4-bit number.

Complete Circuit Diagram

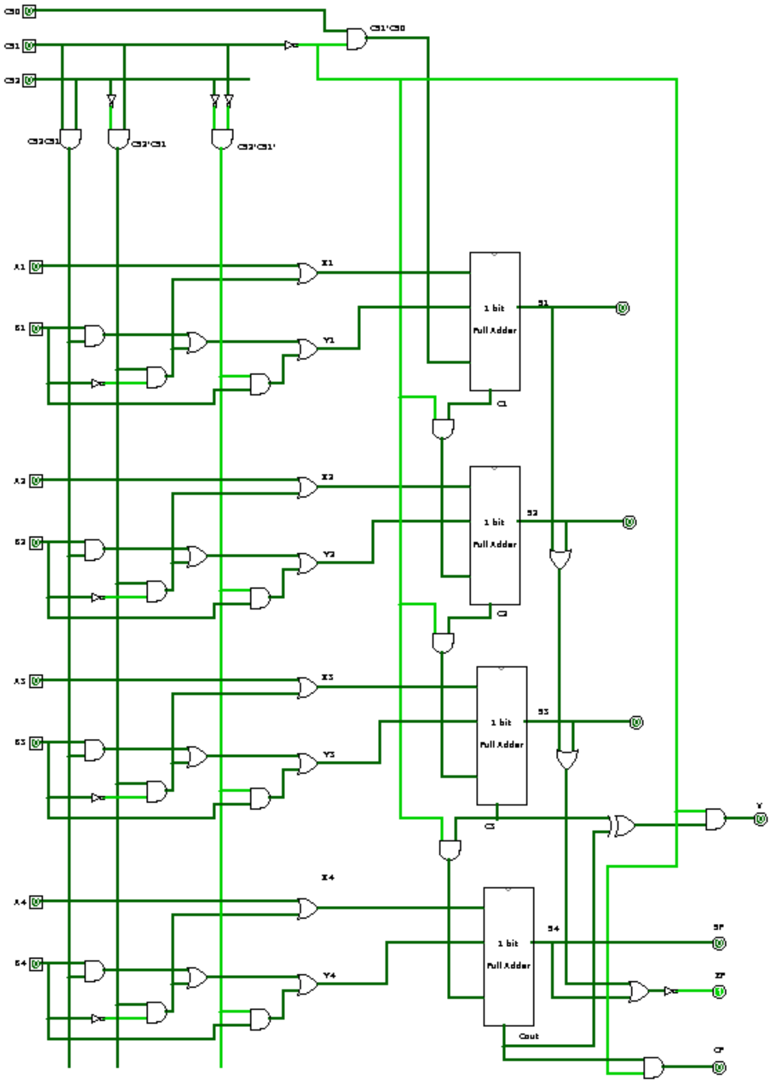


Figure 4: 4 bit ALU

The figure in 4 shows the implementation of a 4 bit ALU with **4** 1 bit full adder.

ICs with count as Charts

IC No.	Count
7432(OR)	4
7408(AND)	6
7404(NOT)	2
7486(XOR)	1

Simulator Used along with version number

Name : Logisim **Version** : 2.7.1

Discussion

The main objective of this assignment was to simulate a 4-bit ALU by implementing some arithmetic and logic operations. While designing, we had to ensure that the design will not contain unnecessary steps. K-maps were used to minimize the design. We have mainly used the basic gates - AND,OR,NOT,XOR for our design. We have also designed a 1-bit full adder which we have used multiple times in the ALU design and it made our work easier. We have also implemented the status flags that alter their status as per the operations. The whole work was divided among the group members and everyone contributed equally.