CSE306 (Computer Architecture Sessional)

A Report On

4-bit ALU Simulation

Contributor Student ID:

1705068

1705069

1705070

1705071

1705074

Section: B

Subsection: B1

Group NO: **3**

Introduction:

Arithmetic Logic Unit or ALU is the digital function that implements the microoperations stored in processor registers. The control routes the source information from registers to the inputs of the ALU to perform the microoperations. ALU receives information from registers and performs operations specified by the control. The result of the microoperations are also transferred to a destination register.

In this experiment we have implemented a 4bit ALU using four cascaded 1bit full adders. We have also implemented four status bits as mentioned below:

Carry(C): Set to 1 if the output carry is 1, 0 otherwise.

Sign(S): Set to 1 if the highest order bit of the result is 1, 0 otherwise.

Zero(Z): If all the bits of the result is 0, only then Z is set to 1.

Overflow(V): If the input carry and the output carry are different, then V is set to 1. In other words, V is the result of the XOR operation of carry in and carry out.

Problem Specification:

We are to design a 4 bit-ALU Circuit as well as show the effects of various operations on flags as per the rules of Assembly Language.

Required Flags:

– Carry (C)

– Sign (S)

– Overflow (V)

– Zero (Z)

Assigned Instructions:

Inputs:

A (4-bit)

B (4 bit)

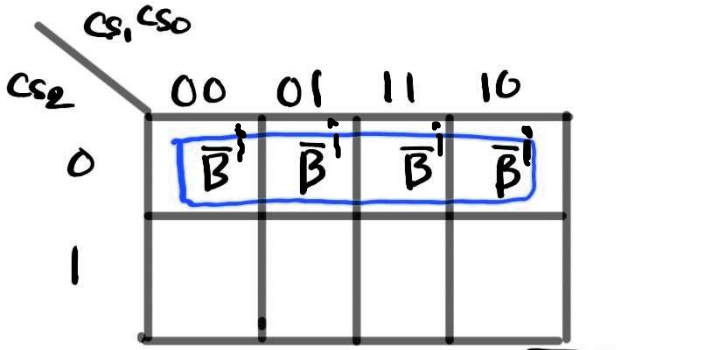
cs1, cs2, cs0 (Selection bits)

cs2	cs1	cs0(cin)	Functions	Form
0	0	0	Subtract with borrow	**A + ¬*B***
0	0	1	Subtract	**A + ¬*B*+ 1**
0	1	x	AND	**A ∧ B**
1	0	0	Transfer A	**A**
1	0	1	Increment A	**A + 1**
1	1	x	OR	**A ∨ B**

Truth Table

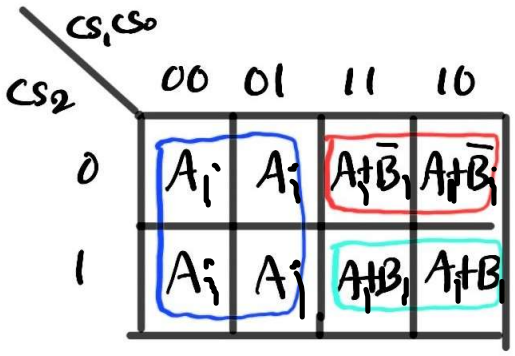
cs2	cs1	cs0(cin)	Function	F	Xi	Yi	Zi
0	0	0	Subtract with borrow	**A + ¬*B***	A	¬B	0
0	0	1	Subtract	**A + ¬*B*+ 1**	A	¬B	1
0	1	x	AND	**A ∧ B**	A + ¬B	¬B	0
1	0	0	Transfer A	**A**	A	0	0
1	0	1	Increment A	**A + 1**	A	0	1
1	1	x	OR	**A ∨ B**	A + B	0	0

K-map for Yi:



$\therefore Y_i = \overline{cs_2}\,\overline{B_i} = \overline{(cs_2 + B_i)}$

K-map for X:



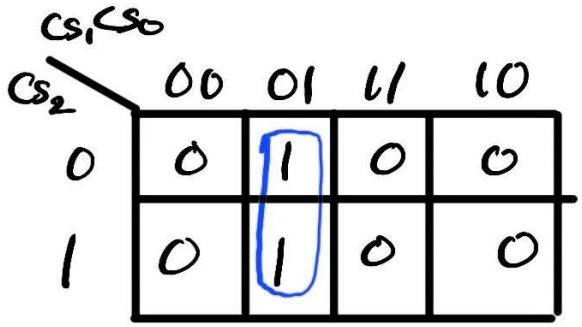
$$X_i = \overline{CS_1}A_i + \overline{CS_2}CS_1(A_i+\overline{B_i}) + CS_2CS_1(A_i+B_i)$$

$$= A_i + \overline{CS_2}CS_1\overline{B_i} + CS_2CS_1B_i$$

$$= A_i + CS_1(\overline{CS_2}\overline{B_i} + CS_2B_i)$$

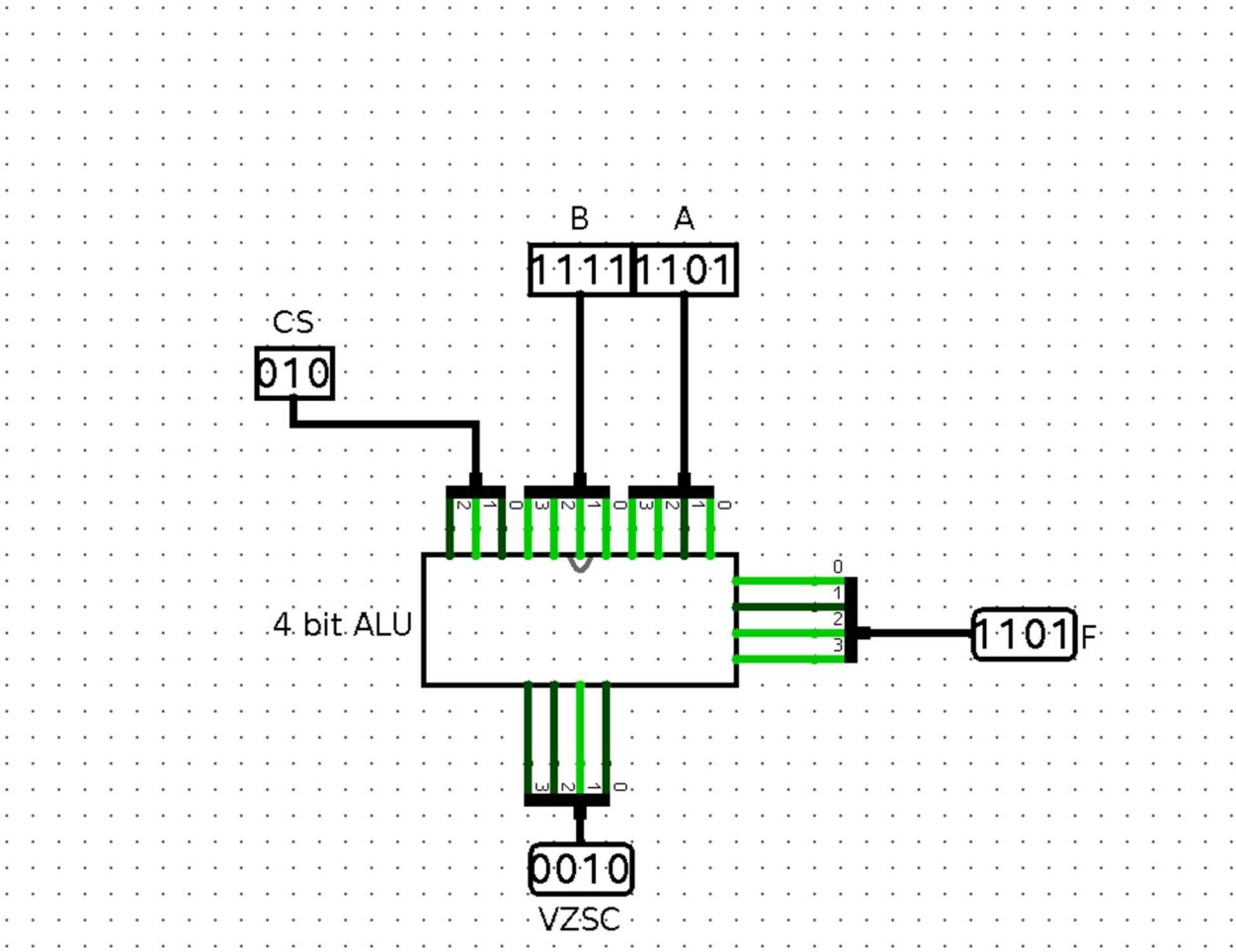
$$= A_i + CS_1(CS_2 \odot B_i)$$

K-map for Z:

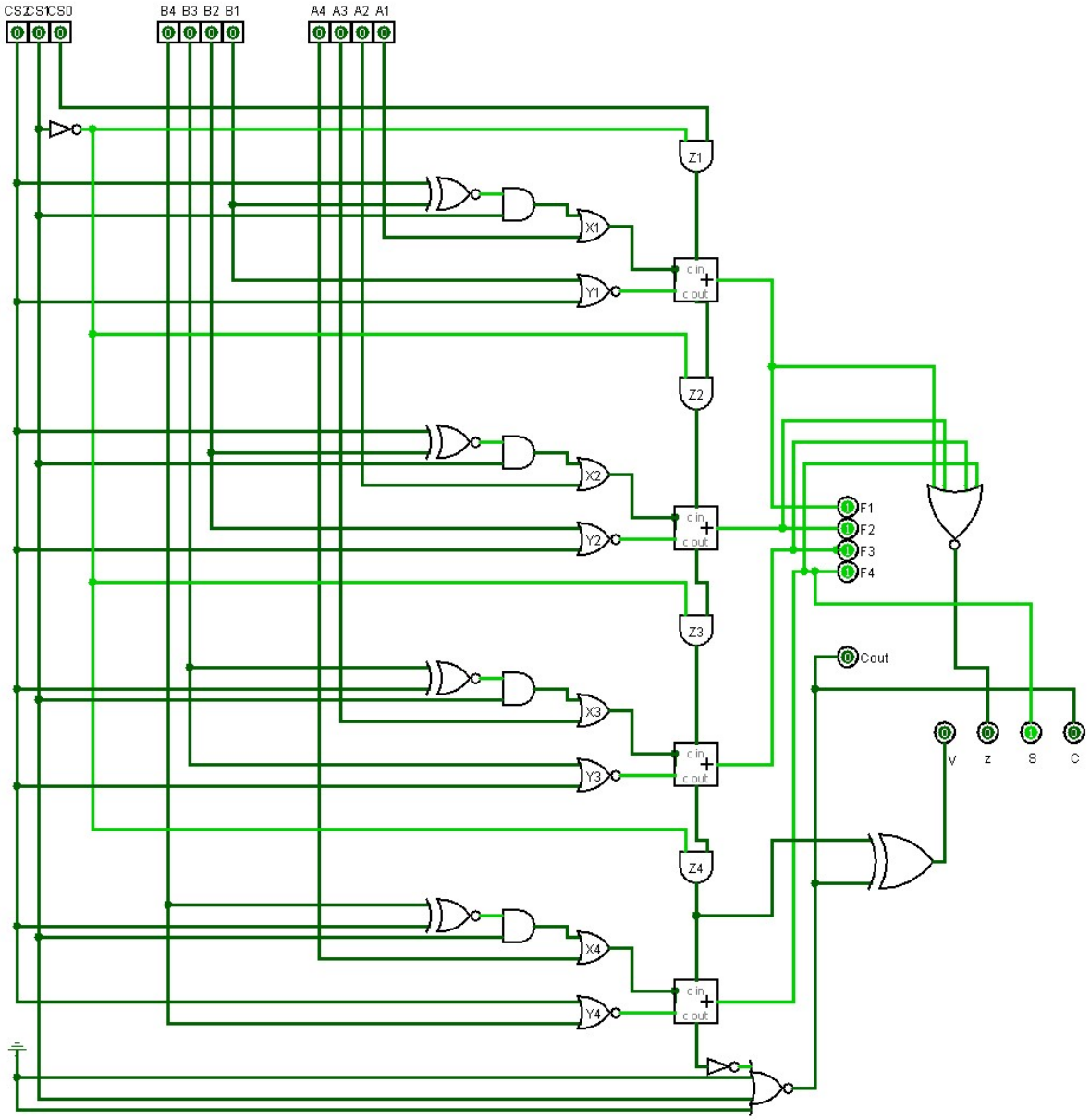


$$\therefore Z_i = \overline{CS_1}CS_0$$

Block Diagram



Circuit Diagram



IC Used with Count:

IC	IC Name	Count
7402	Quad 2-NOR Gate	1
7404	Hex Inverter	1
7408	Quad 2-AND Gate	2
7425	Dual 4-NOR Gate	1
7432	Quad 2-OR Gate	1
7480	1-Bit Full Adder	2
7486	Quad 2-Exclusive OR Gate	1
747266	Exclusive-NOR Gate	1
		Total = 10

Discussion:

In our design, first the arithmetic part was considered and a circuit was built keeping regard of each input bit of a 1bit full adder and input carry. Then for the already built circuit to support the logical information as well, the circuit was modified accordingly with the activation of Mode Select input cs1. This process finalized our design of ALU as per specification.

While implementing the specified design of the ALU, a minimum number of ICs were used. We used various types of gates such as XOR, NOT, 2 input NOR, 4 input NOR, XNOR, AND, OR and 1bit Full Adder. In order to check the status bits after the specified operations, a status register was also implemented.

During logical operations (AND, OR) implemented, as per the provided specifications, the status bits C (carry bit) and V (overflow bit) are cleared. The other two status bits S (sign bit) and Z (zero bit) provides important information about the output.