Assignment On
4-bit ALU Simulation

CSE-306

Computer Architecture Sessional

Section – A2

Group -5

Roll No:

1605049

1605050

1605052

1605053

1605054

Introduction:

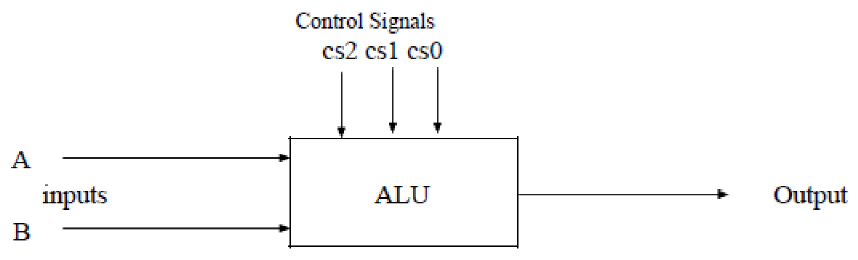
Arithmetic logic unit(ALU) is a multipurpose arithmetic and logical function. In its core an ALU is a combination of parallel adders with some other gates. There are selection bits which control the mode (arithmetic or logical) and action of ALU. Depending on number of selection bits, there is a upper limit of actions of any ALU. Generally with n selection bits we can do 2^n different but interrelated actions.

Problem Specification with Assigned Instruction:

We have to design an ALU for the following table.

C_{in}			Functions For
CS_2	CS_1	CS_0	Group 5
0	0	0	Transfer A
0	1	0	Subtract with borrow
1	0	0	Increment A
1	1	0	Subtract
X	0	1	OR
X	1	1	AND

We have to design an ALU with following general design outline.



Our design also has to include the following:

1. Carry Flag
2. Sign Flag
3. Zero Flag
4. Overflow Flag

Arithmetic Part:

When CS_0 is 0, we carry out the arithmetic functions.

Function Table:

CS_2	CS_0	F
0	0	A
0	1	$A+\overline{B}$
1	0	A+1
1	1	$A+\overline{B}+1$

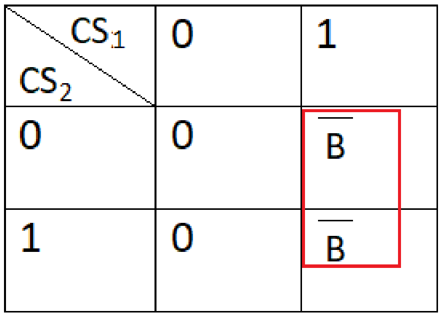
For X we put X=A

We need to find out Y for Arithmetic functions.

Truth Table:

CS_2	CS_1	Y
0	0	All 0s
0	1	$\overline{B}$
1	0	All 0s
1	1	$\overline{B}$

K-Map:



CS_2 \ CS_1	0	1
0	0	$\boxed{\overline{B}}$
1	0	$\boxed{\overline{B}}$

From K-Map

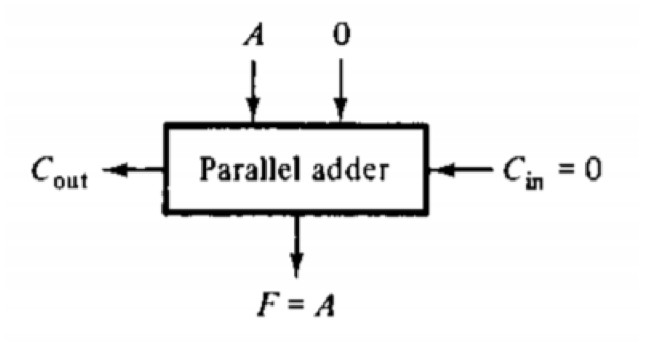
$Y = CS_1 \ \overline{B}$

And $C_{\text{in } i+1} = C_{\text{out } i} \ \overline{CS_0}$,

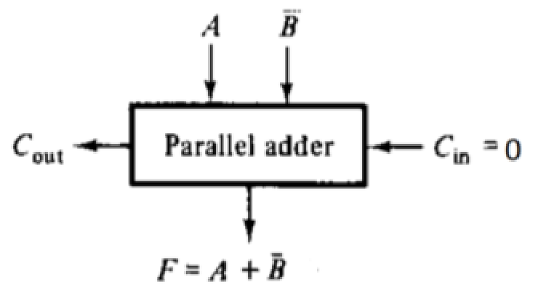
where $C_{\text{in } 0} = CS_2 \ \overline{CS_0}$

Block Diagram:

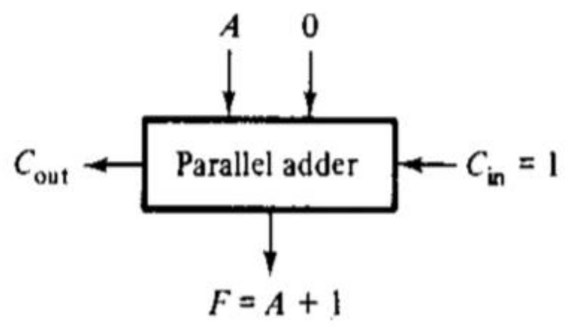
(a) Transfer A



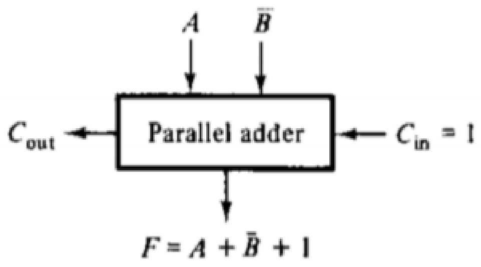
(b) Subtract with Borrow



(c) Increment A



(d) Subtract



Logical Part:

Here $CS_0 = 1$ and $CS_2 = X$

Function Table:

CS_1	Function
0	Bitwise logical OR
1	Bitwise logical AND

Derivation:

Here, we'll change X such that it meets our required functions. Comparing this with our arithmetic part we see that, when CS_1 is 0 $Y = 0$ and when $CS_1 = 1$, $Y = \overline{B}$.

When $CS_1=0$, $F = A$. However we need F to be $A + B$. We can use selection bits to sort this problem out. If we make $X = A + CS_0\ \overline{CS_1}\ B$, when $CS_0 = 1$ and $CS_1 = 0$ we get $X = A + B$.

Similarly, when $CS_1 = 1$, we get $F = A \oplus \overline{B}$. We need to convert this into AB. Let $X = A + k$.

Now $F = (A + k) \oplus \overline{B}$

$$= (A + k)B + \overline{(A + k)}\ \overline{B}$$

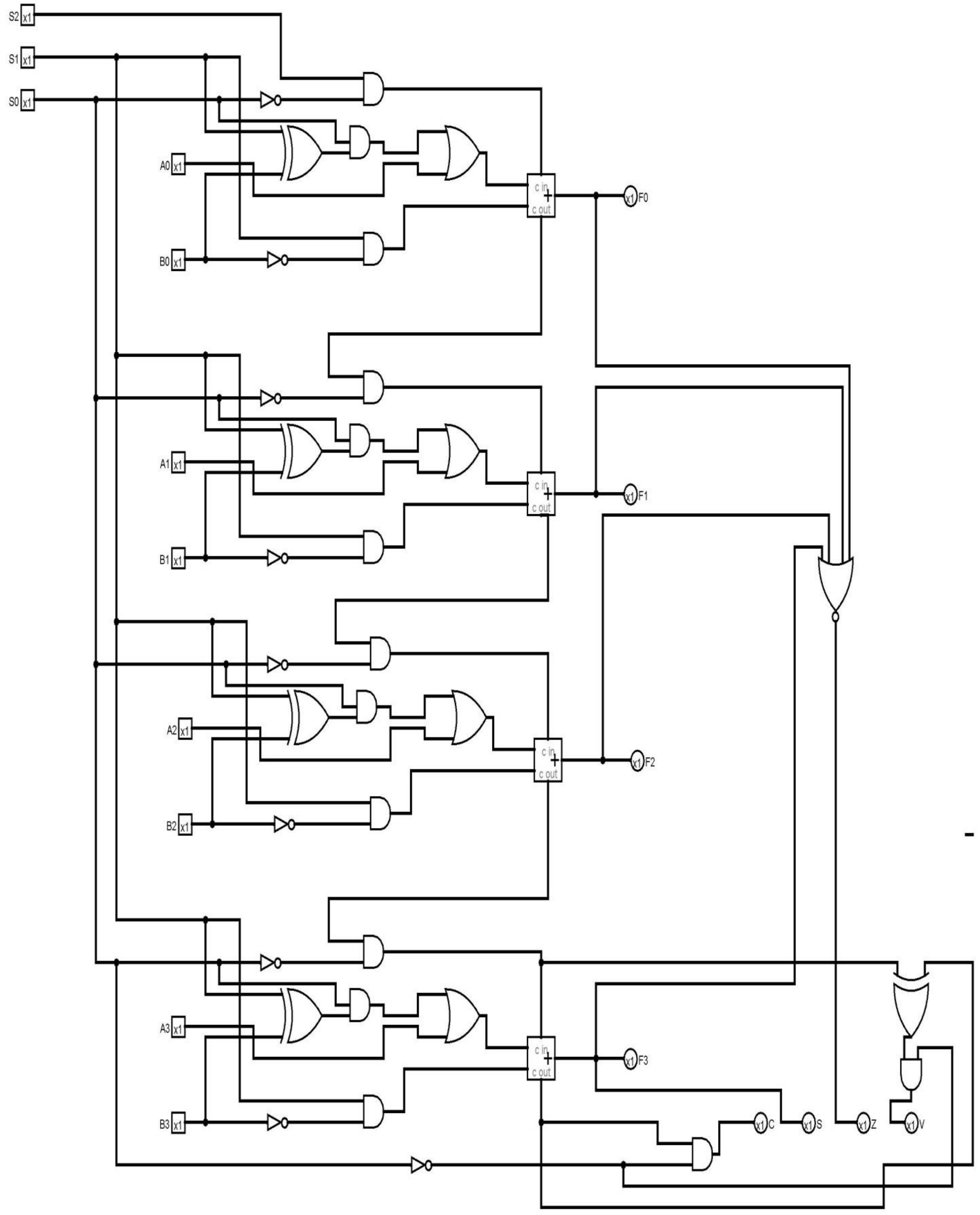
$$= AB + kB + \overline{A}\ \overline{k}\ \overline{B}$$

If $k = \overline{B}$, then F becomes AB. We again use selection bits here to select this. Thus $X = A + CS_0\,\overline{CS_1}\,B + CS_0\,CS_1\,\overline{B}$

$$= A + CS_0(\overline{CS_1}\,B + CS_1\,\overline{B})$$

$$= A + CS_0\,(CS_1 \oplus B)$$

Circuit Diagram:



-

ICs Used With Count:

IC number	Name	Count
7408	Quad 2 input AND	6
7432	Quad 2 input OR	3
7404	Hex Inverter	1
7486	Quad 2 input XOR	3

Simulator Used:

Logisim
Version : logisim-win-2.7.1
Download Link: https://sourceforge.net/projects/circuit/

Discussion:

Here, we used the concept of Full Adder with slight modification to implement the ALU. However, because of the modification to meet the assignment specification, we could not use a 4-bit full adder. Instead we had to implement 1 bit full adder by basic gates. We modified the circuit so that it gives 0 output in carry and overflow flags in case of logic part.