이름: 끼서폿 – KY SOPHOT

학번: 12180152 분반: 논리회로-003

# 8 Bits ALU Report

## Background

Using only full adders we can build an ALU responsible for most mathematical calculations.

This project is to build an 8 bits ALU with the same functionality of 4 bits ALU 74LS382 IC.

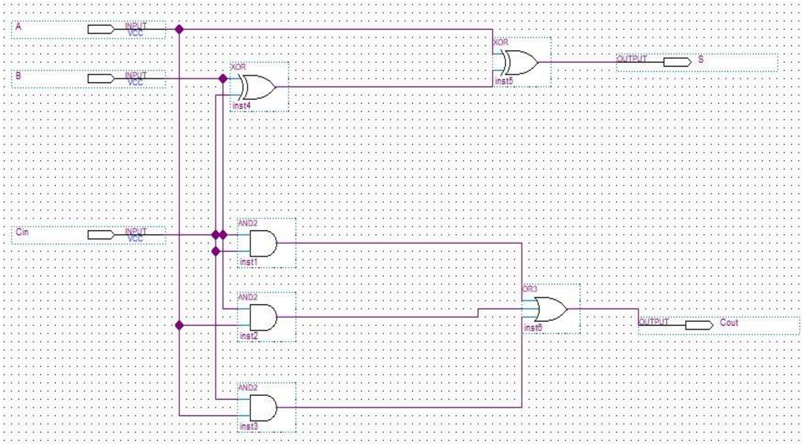
This report will look at:

- Full Adder (in folder "FA")
- 1 Bit ALU (in folder "test")
- 8 Bits ALU (in folder "EightBitALU")
- Testing results (in folder "Captures")

## Full Adder

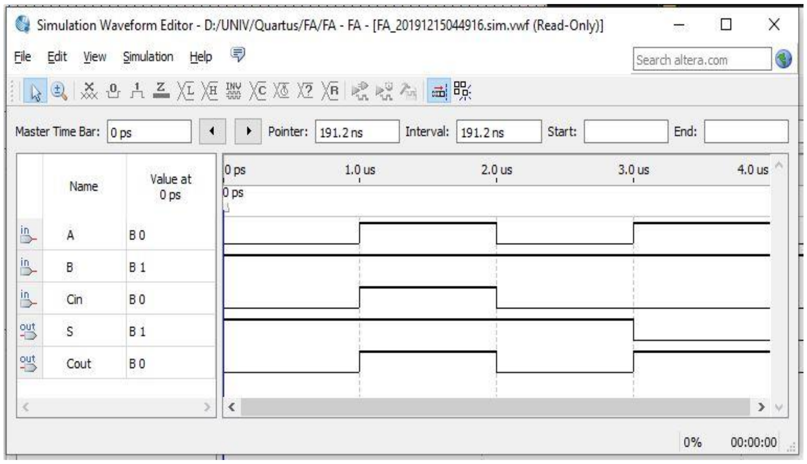
Figure (a) is a full adder that take 3 input bits, A, B, and Carry In, and add the three bits.

It produces Sum and Carry out outputs.



*(a) Full Adder Gate [FA]*

Figure (b) shows the correct result of FA with different inputs.



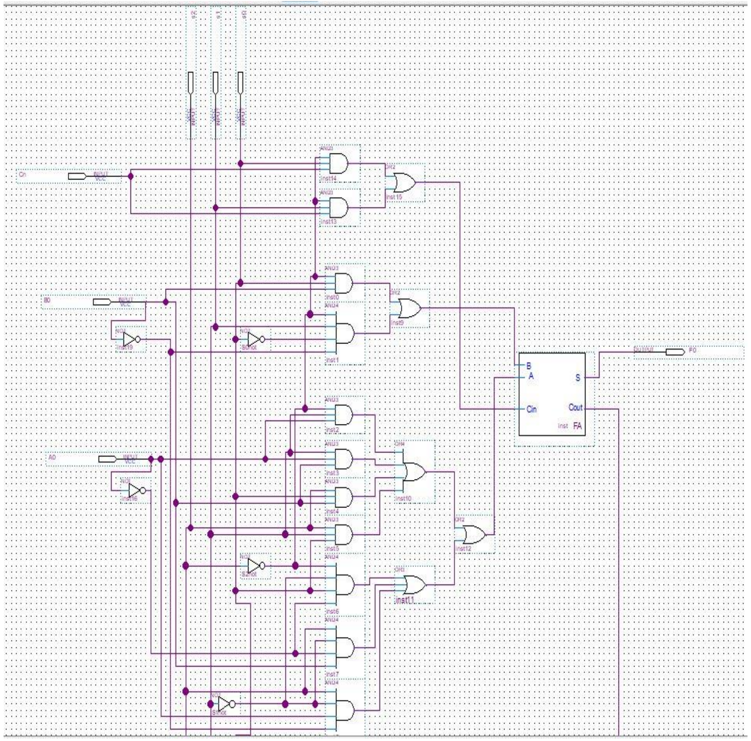
Simulation Waveform Editor - D:/UNIV/Quartus/FA/FA - FA - [FA_20191215044916.sim.vwf (Read-Only)]

Master Time Bar: 0 ps Pointer: 191.2 ns Interval: 191.2 ns Start: End:

| Name | Value at 0 ps |
|---|---|
| A | B 0 |
| B | B 1 |
| Cin | B 0 |
| S | B 1 |
| Cout | B 0 |

*(b) FA Waveform*

- **Build a 1 Bit ALU**

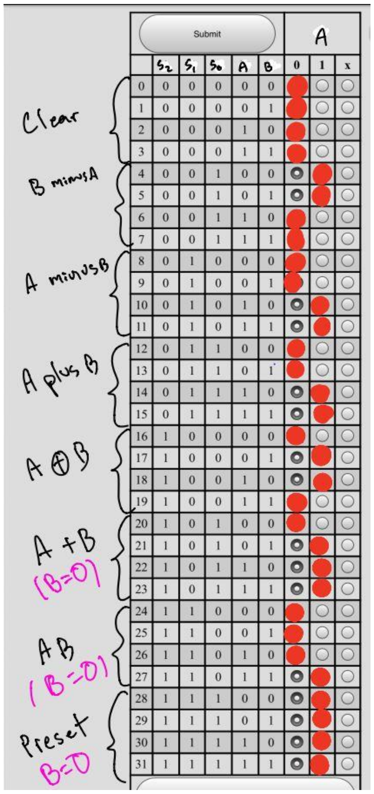


*(c) 1 Bit ALU*

Figure (c) shows a 1 bit ALU, takes 3 bit selection inputs S[2..0], A0, B0, and Cn. It gives output of F0 and Cout.

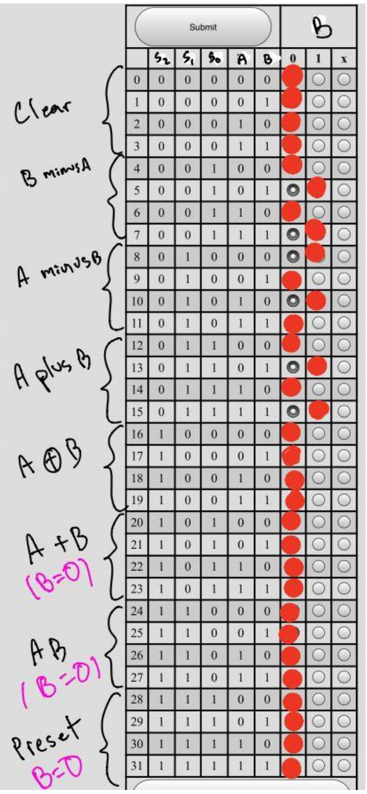
The ALU is built using K-Map of inputs S[2..0], A0, B0 give outputs to A and B of FA. FA's Cn is calculated form K-Map of inputs S[2..0] and Cn.

Note: A0, B0, Cn, and S[2..0] is input from users. While FA's A, B and Cn is changed accordingly to S[2..0].



| | $S_2$ | $S_1$ | $S_0$ | A | B | 0 | 1 | x |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | ● | ○ | ○ |
| 1 | 0 | 0 | 0 | 0 | 1 | ● | ○ | ○ |
| 2 | 0 | 0 | 0 | 1 | 0 | ● | ○ | ○ |
| 3 | 0 | 0 | 0 | 1 | 1 | ● | ○ | ○ |
| 4 | 0 | 0 | 1 | 0 | 0 | ○ | ● | ○ |
| 5 | 0 | 0 | 1 | 0 | 1 | ○ | ● | ○ |
| 6 | 0 | 0 | 1 | 1 | 0 | ● | ○ | ○ |
| 7 | 0 | 0 | 1 | 1 | 1 | ● | ○ | ○ |
| 8 | 0 | 1 | 0 | 0 | 0 | ● | ○ | ○ |
| 9 | 0 | 1 | 0 | 0 | 1 | ● | ○ | ○ |
| 10 | 0 | 1 | 0 | 1 | 0 | ○ | ● | ○ |
| 11 | 0 | 1 | 0 | 1 | 1 | ○ | ● | ○ |
| 12 | 0 | 1 | 1 | 0 | 0 | ● | ○ | ○ |
| 13 | 0 | 1 | 1 | 0 | 1 | ● | ○ | ○ |
| 14 | 0 | 1 | 1 | 1 | 0 | ○ | ● | ○ |
| 15 | 0 | 1 | 1 | 1 | 1 | ○ | ● | ○ |
| 16 | 1 | 0 | 0 | 0 | 0 | ● | ○ | ○ |
| 17 | 1 | 0 | 0 | 0 | 1 | ○ | ● | ○ |
| 18 | 1 | 0 | 0 | 1 | 0 | ○ | ● | ○ |
| 19 | 1 | 0 | 0 | 1 | 1 | ● | ○ | ○ |
| 20 | 1 | 0 | 1 | 0 | 0 | ● | ○ | ○ |
| 21 | 1 | 0 | 1 | 0 | 1 | ○ | ● | ○ |
| 22 | 1 | 0 | 1 | 1 | 0 | ○ | ● | ○ |
| 23 | 1 | 0 | 1 | 1 | 1 | ○ | ● | ○ |
| 24 | 1 | 1 | 0 | 0 | 0 | ● | ○ | ○ |
| 25 | 1 | 1 | 0 | 0 | 1 | ● | ○ | ○ |
| 26 | 1 | 1 | 0 | 1 | 0 | ● | ○ | ○ |
| 27 | 1 | 1 | 0 | 1 | 1 | ○ | ● | ○ |
| 28 | 1 | 1 | 1 | 0 | 0 | ○ | ● | ○ |
| 29 | 1 | 1 | 1 | 0 | 1 | ○ | ● | ○ |
| 30 | 1 | 1 | 1 | 1 | 0 | ○ | ● | ○ |
| 31 | 1 | 1 | 1 | 1 | 1 | ○ | ● | ○ |

*(d) FA's A Truth Table*



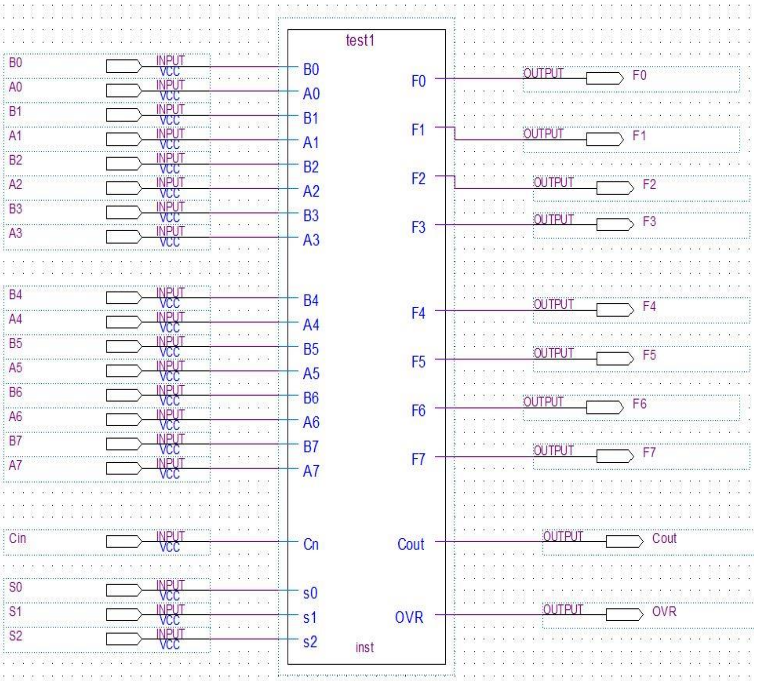
| | $S_2$ | $S_1$ | $S_0$ | A | B | 0 | 1 | x |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | ● | ○ | ○ |
| 1 | 0 | 0 | 0 | 0 | 1 | ● | ○ | ○ |
| 2 | 0 | 0 | 0 | 1 | 0 | ● | ○ | ○ |
| 3 | 0 | 0 | 0 | 1 | 1 | ● | ○ | ○ |
| 4 | 0 | 0 | 1 | 0 | 0 | ● | ○ | ○ |
| 5 | 0 | 0 | 1 | 0 | 1 | ○ | ● | ○ |
| 6 | 0 | 0 | 1 | 1 | 0 | ● | ○ | ○ |
| 7 | 0 | 0 | 1 | 1 | 1 | ○ | ● | ○ |
| 8 | 0 | 1 | 0 | 0 | 0 | ○ | ● | ○ |
| 9 | 0 | 1 | 0 | 0 | 1 | ● | ○ | ○ |
| 10 | 0 | 1 | 0 | 1 | 0 | ○ | ● | ○ |
| 11 | 0 | 1 | 0 | 1 | 1 | ● | ○ | ○ |
| 12 | 0 | 1 | 1 | 0 | 0 | ● | ○ | ○ |
| 13 | 0 | 1 | 1 | 0 | 1 | ○ | ● | ○ |
| 14 | 0 | 1 | 1 | 1 | 0 | ● | ○ | ○ |
| 15 | 0 | 1 | 1 | 1 | 1 | ○ | ● | ○ |
| 16 | 1 | 0 | 0 | 0 | 0 | ● | ○ | ○ |
| 17 | 1 | 0 | 0 | 0 | 1 | ● | ○ | ○ |
| 18 | 1 | 0 | 0 | 1 | 0 | ● | ○ | ○ |
| 19 | 1 | 0 | 0 | 1 | 1 | ● | ○ | ○ |
| 20 | 1 | 0 | 1 | 0 | 0 | ● | ○ | ○ |
| 21 | 1 | 0 | 1 | 0 | 1 | ● | ○ | ○ |
| 22 | 1 | 0 | 1 | 1 | 0 | ● | ○ | ○ |
| 23 | 1 | 0 | 1 | 1 | 1 | ● | ○ | ○ |
| 24 | 1 | 1 | 0 | 0 | 0 | ● | ○ | ○ |
| 25 | 1 | 1 | 0 | 0 | 1 | ● | ○ | ○ |
| 26 | 1 | 1 | 0 | 1 | 0 | ● | ○ | ○ |
| 27 | 1 | 1 | 0 | 1 | 1 | ● | ○ | ○ |
| 28 | 1 | 1 | 1 | 0 | 0 | ● | ○ | ○ |
| 29 | 1 | 1 | 1 | 0 | 1 | ● | ○ | ○ |
| 30 | 1 | 1 | 1 | 1 | 0 | ● | ○ | ○ |
| 31 | 1 | 1 | 1 | 1 | 1 | ● | ○ | ○ |

*(e) FA's B Truth Table*

Figures (d) and (e) are the Truth table to find Full Adder's A and B by applying K-Map rules.

- ## 8 Bits ALU

Having eight 1 Bit ALUs and connect Carry out of previous ALU to Carry In of next ALU, we can create an 8 Bits ALU shown in figure (f).

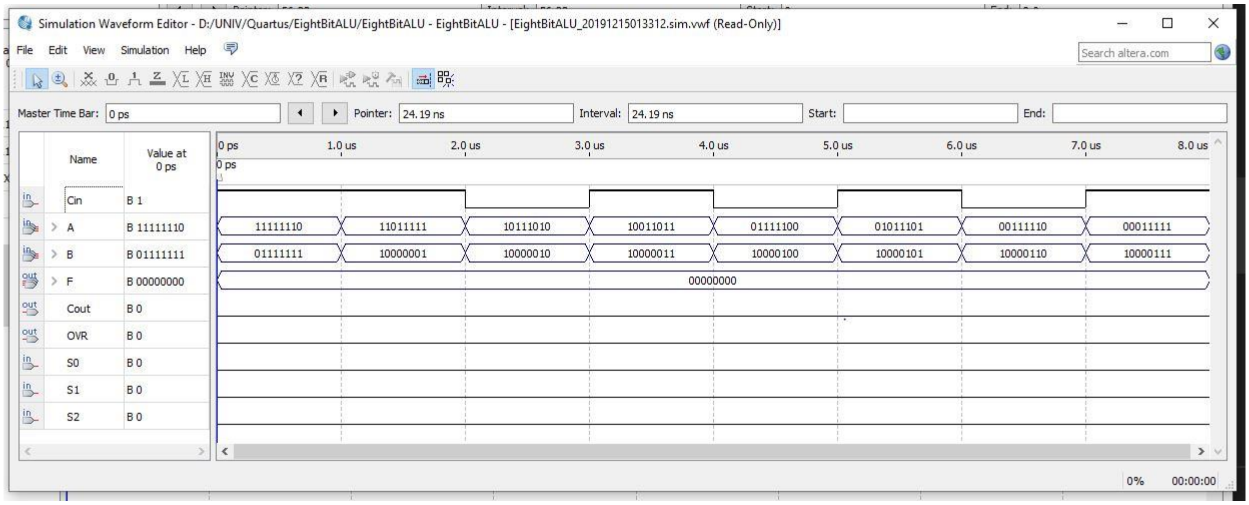


*(f) 8 Bits ALU*

- ## Testing Results

**Operation CLEAR (S2 = S1 = S0 = 0)**

Regardless of inputs the result F will always be LOW


Simulation Waveform Editor - D:/UNIV/Quartus/EightBitALU/EightBitALU - EightBitALU - [EightBitALU_20191215013312.sim.vwf (Read-Only)]
File
Edit
View
Simulation
Help
Search altera.com
0 ps
1.0 us
2.0 us
3.0 us
4.0 us
5.0 us
6.0 us
7.0 us
8.0 us
0 ps
0%
00:00:00

Master Time Bar: 0 ps Pointer: 24.19 ns Interval: 24.19 ns Start: End:

| Name | Value at 0 ps |
|---|---|
| Cin | B 1 |
| A | B 11111110 |
| B | B 01111111 |
| F | B 00000000 |
| Cout | B 0 |
| OVR | B 0 |
| S0 | B 0 |
| S1 | B 0 |
| S2 | B 0 |

A: 11111110, 11011111, 10111010, 10011011, 01111100, 01011101, 00111110, 00011111

B: 01111111, 10000001, 10000010, 10000011, 10000100, 10000101, 10000110, 10000111

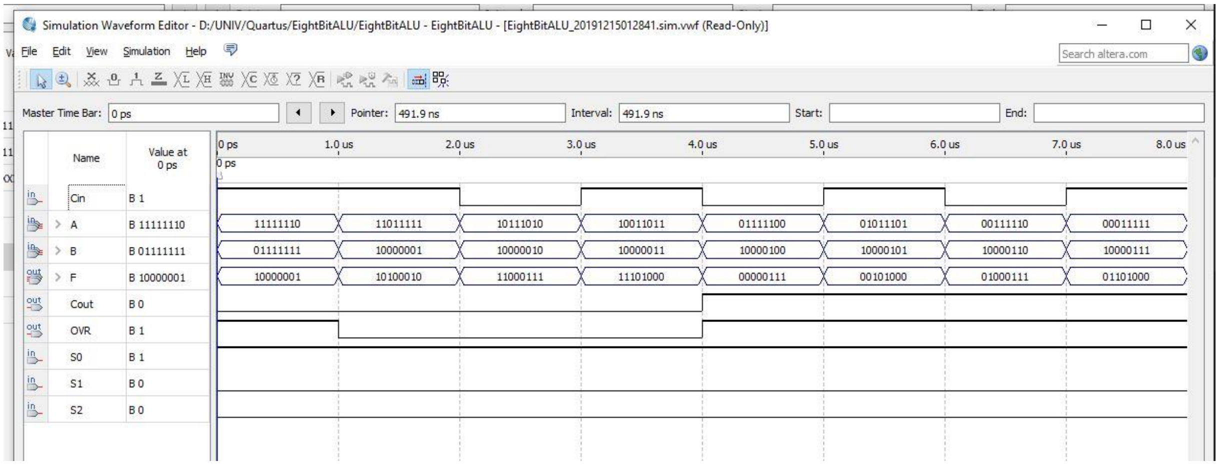
F: 00000000

*CLEAR [000]*

**Operation B minus A  (S2 = 0, S1 = 0, S0 = 1)**

At interval 1.0us – 2.0us [B = 1000 0001 = -127] minus [A = 1101 1111 = -33]. Answer is -92. The result [F = 1010 0010 = -92].

At interval 4.0us – 5.0us [B = 1000 0100 = -124] minus [A = 0111 1100 = 124]. Answer is -248 which clearly will cause Overflow. And because MSB of A will be flipped to 1 to add with MSB of B causing Cout HIGH. Also, F is not correct because Cin is not set to HIGH.



Simulation Waveform Editor - D:/UNIV/Quartus/EightBitALU/EightBitALU - EightBitALU - [EightBitALU_20191215012841.sim.vwf (Read-Only)]

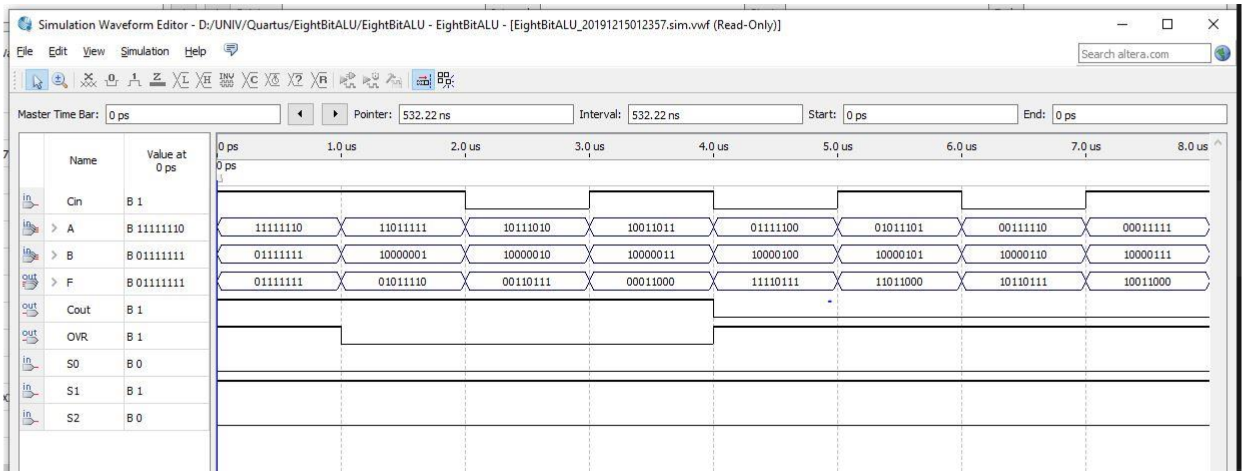
Master Time Bar: 0 ps  Pointer: 491.9 ns  Interval: 491.9 ns  Start:  End:

| Name | Value at 0 ps | 0 ps | 1.0 us | 2.0 us | 3.0 us | 4.0 us | 5.0 us | 6.0 us | 7.0 us |
|---|---|---|---|---|---|---|---|---|---|
| Cin | B 1 | | | | | | | | |
| A | B 11111110 | 11111110 | 11011111 | 10111010 | 10011011 | 01111100 | 01011101 | 00111110 | 00011111 |
| B | B 01111111 | 01111111 | 10000001 | 10000010 | 10000011 | 10000100 | 10000101 | 10000110 | 10000111 |
| F | B 10000001 | 10000001 | 10100010 | 11000111 | 11101000 | 00000111 | 00101000 | 01000111 | 01101000 |
| Cout | B 0 | | | | | | | | |
| OVR | B 1 | | | | | | | | |
| S0 | B 1 | | | | | | | | |
| S1 | B 0 | | | | | | | | |
| S2 | B 0 | | | | | | | | |

*B minus A [ 001]*

**Operation A minus B  (S2 = 0, S1 = 1, S0 = 0)**

It calculated similarly to B minus A



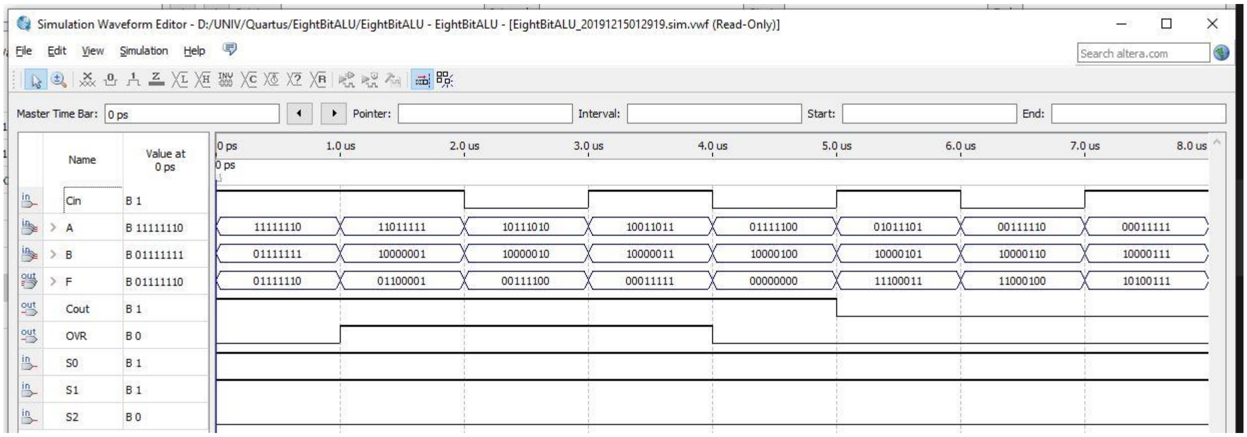
Simulation Waveform Editor - D:/UNIV/Quartus/EightBitALU/EightBitALU - EightBitALU - [EightBitALU_20191215012357.sim.vwf (Read-Only)]

Master Time Bar: 0 ps  Pointer: 532.22 ns  Interval: 532.22 ns  Start: 0 ps  End: 0 ps

| Name | Value at 0 ps | 0 ps | 1.0 us | 2.0 us | 3.0 us | 4.0 us | 5.0 us | 6.0 us | 7.0 us |
|---|---|---|---|---|---|---|---|---|---|
| Cin | B 1 | | | | | | | | |
| A | B 11111110 | 11111110 | 11011111 | 10111010 | 10011011 | 01111100 | 01011101 | 00111110 | 00011111 |
| B | B 01111111 | 01111111 | 10000001 | 10000010 | 10000011 | 10000100 | 10000101 | 10000110 | 10000111 |
| F | B 01111111 | 01111111 | 01011110 | 00110111 | 00011000 | 11110111 | 11011000 | 10110111 | 10011000 |
| Cout | B 1 | | | | | | | | |
| OVR | B 1 | | | | | | | | |
| S0 | B 0 | | | | | | | | |
| S1 | B 1 | | | | | | | | |
| S2 | B 0 | | | | | | | | |

*A minus B [010]*

## Operation A plus B (S2 = 0, S1 = 1, S0 = 1)



Simulation Waveform Editor - D:/UNIV/Quartus/EightBitALU/EightBitALU - EightBitALU - [EightBitALU_20191215012919.sim.vwf (Read-Only)]

Master Time Bar: 0 ps Pointer: Interval: Start: End:

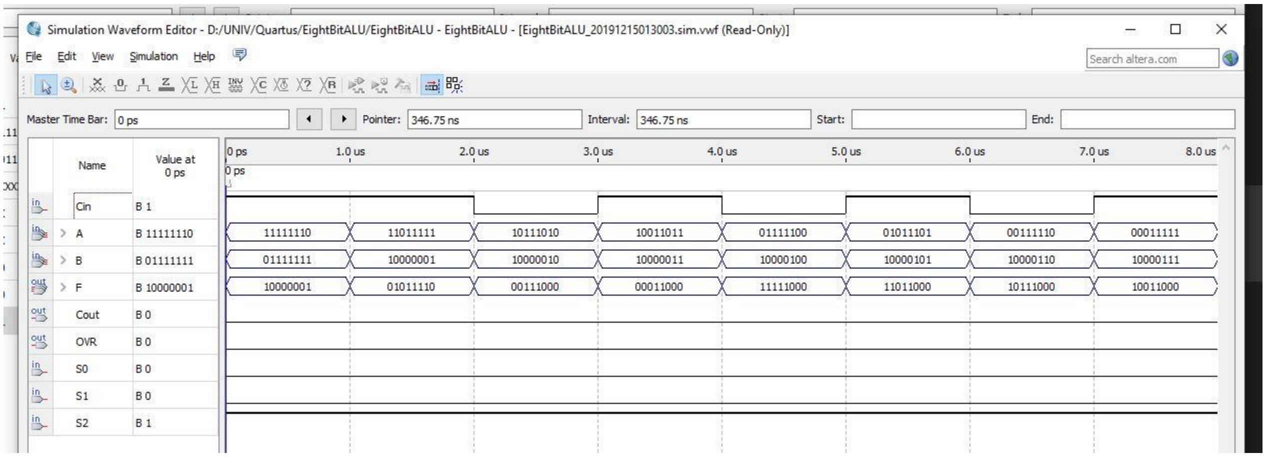
| Name | Value at 0 ps | 0 ps | 1.0 us | 2.0 us | 3.0 us | 4.0 us | 5.0 us | 6.0 us | 7.0 us |
|---|---|---|---|---|---|---|---|---|---|
| Cin | B 1 | | | | | | | | |
| A | B 11111110 | 11111110 | 11011111 | 10111010 | 10011011 | 01111100 | 01011101 | 00111110 | 00011111 |
| B | B 01111111 | 01111111 | 10000001 | 10000010 | 10000011 | 10000100 | 10000101 | 10000110 | 10000111 |
| F | B 01111110 | 01111110 | 01100001 | 00111100 | 00011111 | 00000000 | 11100011 | 11000100 | 10100111 |
| Cout | B 1 | | | | | | | | |
| OVR | B 0 | | | | | | | | |
| S0 | B 1 | | | | | | | | |
| S1 | B 1 | | | | | | | | |
| S2 | B 0 | | | | | | | | |

*A plus B [011]*

## Operation A xor B (S2 = 1, S1 = 0, S0 = 0)

Overflow and Cout are set to LOW and Cin will not affect the calculation.

$F_i = A_i \text{ xor } B_i$



Simulation Waveform Editor - D:/UNIV/Quartus/EightBitALU/EightBitALU - EightBitALU - [EightBitALU_20191215013003.sim.vwf (Read-Only)]

Master Time Bar: 0 ps Pointer: 346.75 ns Interval: 346.75 ns Start: End:

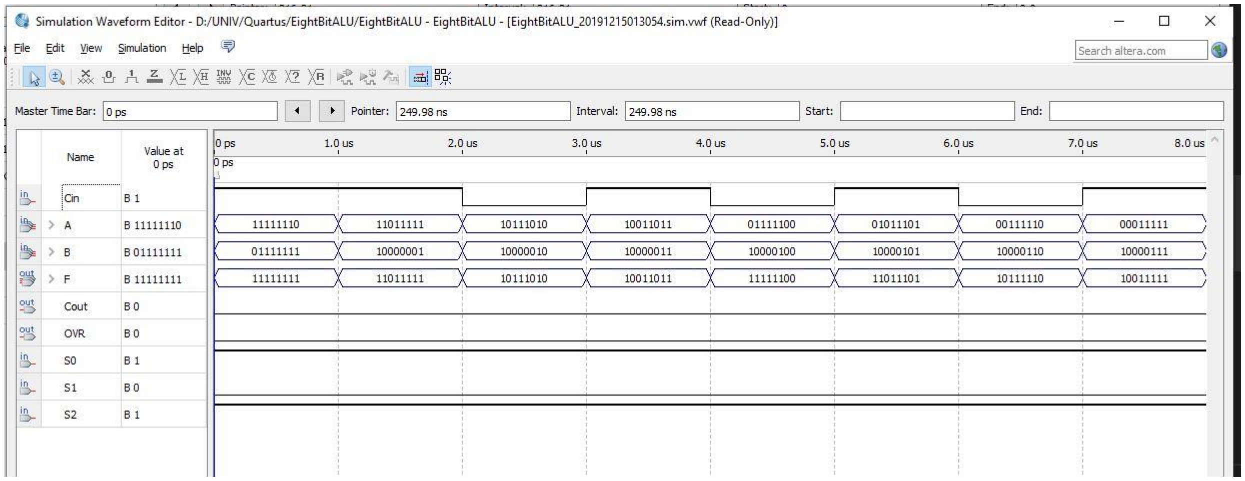
| Name | Value at 0 ps | 0 ps | 1.0 us | 2.0 us | 3.0 us | 4.0 us | 5.0 us | 6.0 us | 7.0 us |
|---|---|---|---|---|---|---|---|---|---|
| Cin | B 1 | | | | | | | | |
| A | B 11111110 | 11111110 | 11011111 | 10111010 | 10011011 | 01111100 | 01011101 | 00111110 | 00011111 |
| B | B 01111111 | 01111111 | 10000001 | 10000010 | 10000011 | 10000100 | 10000101 | 10000110 | 10000111 |
| F | B 10000001 | 10000001 | 01011110 | 00111000 | 00011000 | 11111000 | 11011000 | 10111000 | 10011000 |
| Cout | B 0 | | | | | | | | |
| OVR | B 0 | | | | | | | | |
| S0 | B 0 | | | | | | | | |
| S1 | B 0 | | | | | | | | |
| S2 | B 1 | | | | | | | | |

*A xor B [100]*

### Operation (OR) A+B   (S2 = 1, S1 = 0, S0 = 1)

Overflow and Cout are set to LOW and Cin will not affect the calculation.

$F_i = A_i$ or $B_i$



Simulation Waveform Editor - D:/UNIV/Quartus/EightBitALU/EightBitALU - EightBitALU - [EightBitALU_20191215013054.sim.vwf (Read-Only)]

Master Time Bar: 0 ps   Pointer: 249.98 ns   Interval: 249.98 ns   Start:   End:

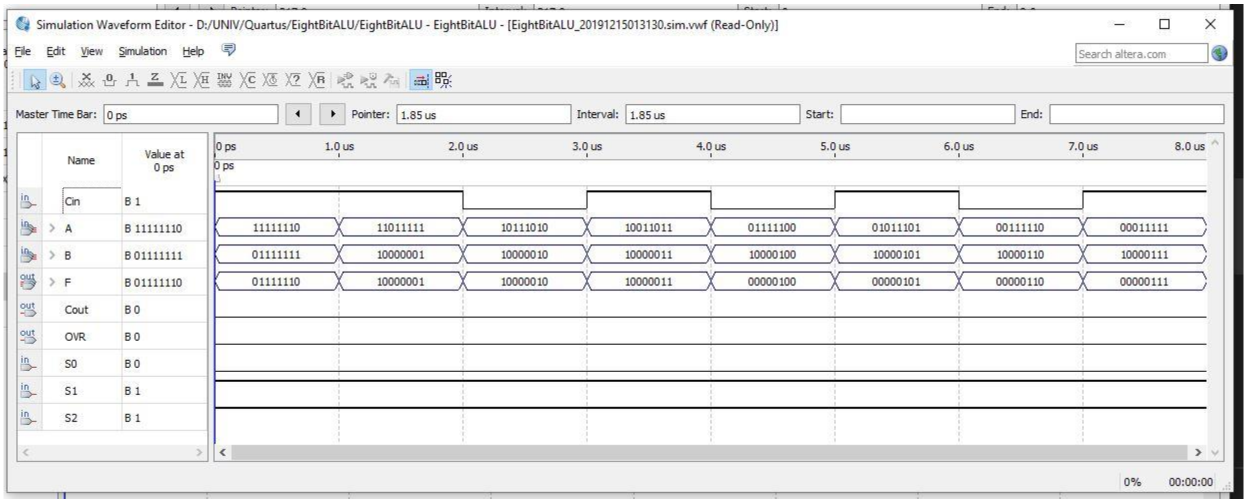
| Name | Value at 0 ps | 0 ps | 1.0 us | 2.0 us | 3.0 us | 4.0 us | 5.0 us | 6.0 us | 7.0 us |
|---|---|---|---|---|---|---|---|---|---|
| Cin | B 1 | | | | | | | | |
| A | B 11111110 | 11111110 | 11011111 | 10111010 | 10011011 | 01111100 | 01011101 | 00111110 | 00011111 |
| B | B 01111111 | 01111111 | 10000001 | 10000010 | 10000011 | 10000100 | 10000101 | 10000110 | 10000111 |
| F | B 11111111 | 11111111 | 11011111 | 10111010 | 10011011 | 11111100 | 11011101 | 10111110 | 10011111 |
| Cout | B 0 | | | | | | | | |
| OVR | B 0 | | | | | | | | |
| S0 | B 1 | | | | | | | | |
| S1 | B 0 | | | | | | | | |
| S2 | B 1 | | | | | | | | |

*A + B [101]*

### Operation (AND) AB   (S2 = 1, S1 = 1, S0 = 0)

Overflow and Cout are set to LOW and Cin will not affect the calculation.

$F_i = A_i$ and $B_i$



Simulation Waveform Editor - D:/UNIV/Quartus/EightBitALU/EightBitALU - EightBitALU - [EightBitALU_20191215013130.sim.vwf (Read-Only)]

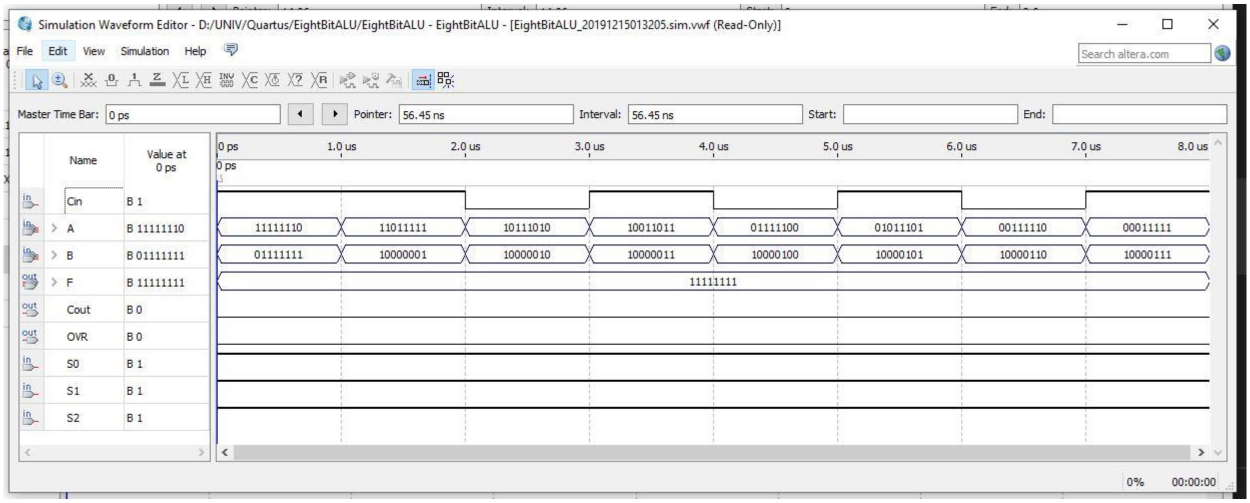
Master Time Bar: 0 ps   Pointer: 1.85 us   Interval: 1.85 us   Start:   End:

| Name | Value at 0 ps | 0 ps | 1.0 us | 2.0 us | 3.0 us | 4.0 us | 5.0 us | 6.0 us | 7.0 us |
|---|---|---|---|---|---|---|---|---|---|
| Cin | B 1 | | | | | | | | |
| A | B 11111110 | 11111110 | 11011111 | 10111010 | 10011011 | 01111100 | 01011101 | 00111110 | 00011111 |
| B | B 01111111 | 01111111 | 10000001 | 10000010 | 10000011 | 10000100 | 10000101 | 10000110 | 10000111 |
| F | B 01111110 | 01111110 | 10000001 | 10000010 | 10000011 | 00000100 | 00000101 | 00000110 | 00000111 |
| Cout | B 0 | | | | | | | | |
| OVR | B 0 | | | | | | | | |
| S0 | B 0 | | | | | | | | |
| S1 | B 1 | | | | | | | | |
| S2 | B 1 | | | | | | | | |

*AB [110]*

## Operation PRESET (S2 = S1 = S0 = 1)

Regardless of inputs the result F will always be HIGH.



Simulation Waveform Editor - D:/UNIV/Quartus/EightBitALU/EightBitALU - EightBitALU - [EightBitALU_20191215013205.sim.vwf (Read-Only)]

Master Time Bar: 0 ps

Pointer: 56.45 ns

Interval: 56.45 ns

Start:

End:

| Name | Value at 0 ps |
|---|---|
| Cin | B 1 |
| A | B 11111110 |
| B | B 01111111 |
| F | B 11111111 |
| Cout | B 0 |
| OVR | B 0 |
| S0 | B 1 |
| S1 | B 1 |
| S2 | B 1 |

```
A  11111110  11011111  10111010  10011011  01111100  01011101  00111110  00011111
B  01111111  10000001  10000010  10000011  10000100  10000101  10000110  10000111
F  11111111
```

*PRESET [111]*