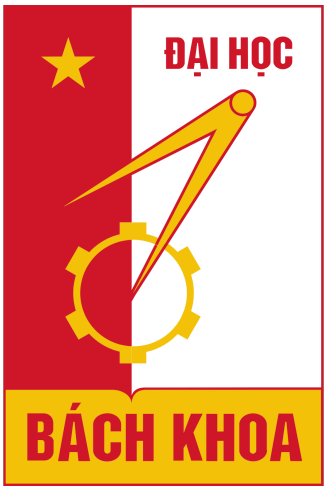
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
SCHOOL OF ELECTRICAL AND ELECTRONIC ENGINEERING



DESIGN SPECIFICATION

LAB2

32 bits ALU, 32 bits Majority Module, and 8 bits CRA Module

Student: Nguyen Phuong Linh

Ha Noi, April 2023

Contents

List of Figures

List of Tables

1 32 bits ALU

1.1 Top module

Figure 1 depicts the Top module of 32 bits ALU.

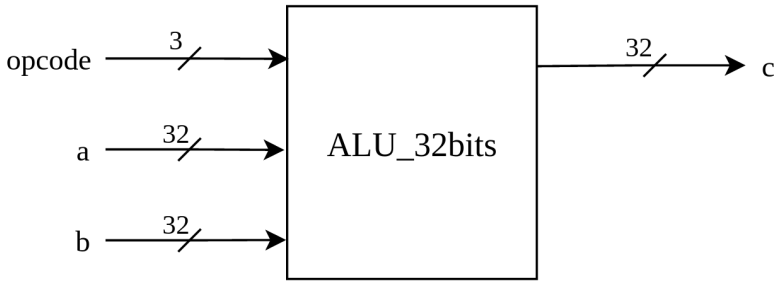


Figure 1: 32 bits ALU top module

1.2 Port description

Table 1 shows port description of 32 bits ALU.

Signal name	**Width**	**I/O**	**Description**
opcode	3	Input	Encode for 8 supported operators
a	32	Input	First operand
b	32	Input	Second operand
c	32	Output	The result of operation

Table 1: Port description of 32 bits ALU

1.3 Functional Descriptions

- Module has 2 32 bits input operands **a** and **b**, 1 **opcode** input (with width 3 encode for 8 supported operator) to determine desired operation of those 2 operands, and 1 32 bit output **c**.
- There are 8 operators that the ALU supports. Detail operators encoded by **opcode** is depicted in table 2
- If overflow occur, then ignore the overflow bit (MSB).

1.4 Timing Diagram

I use randomly generated input method to generate 1000 input in testbench file, then compare the output with the created golden_output. The result is 1000/1000 passed. Figure 2 is simulated wave form at some specific time.

opcode	Operator name	Description
000	ADD	c = a + b
001	SUB	c = a - b
010	Bitwise NOT	c = ∼ a
011	Bitwise AND	c = a & b
100	Bitwise OR	c = a | b
101	Bitwise XOR	c = a ^ b
110	Arithmetic Shift Left	c = a <<< 1
111	Arithmetic Shift Right	c = a >>> 1

Table 2: Supported operator encoded by opcode

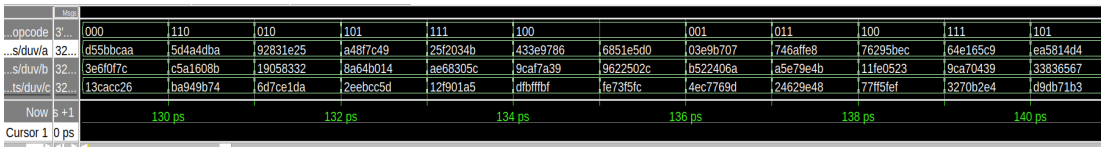


Figure 2: Timing Diagram of ALU (randomly generated input)

2 32 bits Majority Module

2.1 Top module

Figure 3 depicts the Top module of 32 bits Majority module.

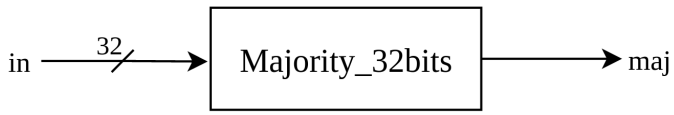


Figure 3: 32 bits Majority top module

2.2 Port description

Table 3 shows port description of 32 bits Majority module.

2.3 Functional Description

- 32 bits input signal is feed to **in**.
- 1 bit output **maj** represent the majority bit of input signal
  - If number of bits 1 in input signal **in** is greater than 16, **maj = 1**.

Signal name	Width	I/O	Description
in	32	Input	32 bits input
maj	1	Output	1 bit majority output

Table 3: Port description of 32 bits Majority module

– Otherwise, if number of bits 1 in input signal **in** is less than or equal to 16, **maj = 0**.

2.4 Timing Diagram

To verification the module, I use directly generated input method to generate inputs in testbench file at different time, then compare the output with the created golden_output. Figure 4 is simulated wave form at some specific time.

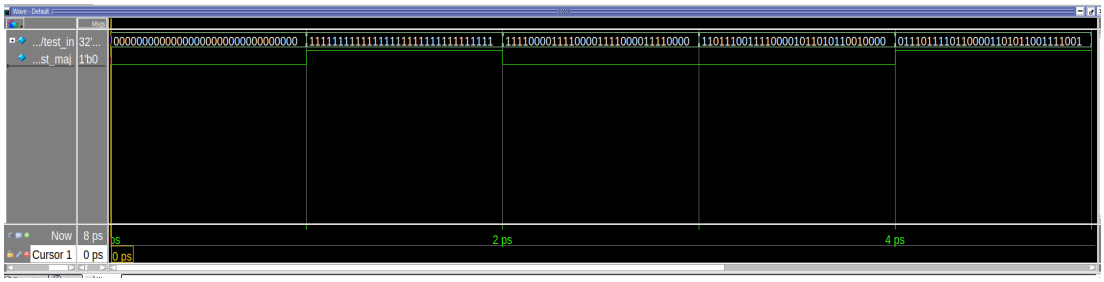


Figure 4: Timing Diagram of Majority module (directly generated input)

3 8 bits CRA

3.1 Top module

Figure 5 depicts the Top module of 8 bits Carry-Ripple Adder.

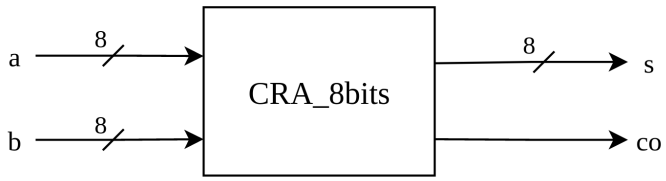


Figure 5: 8 bits Carry-Ripple Adder top module

3.2 Port description

Table 4 shows port description of 8 bits Carry-Ripple Adder.

Signal name	Width	I/O	Description
a	8	Input	First operand
b	8	Input	Second operand
s	8	Output	Sum of 2 operands
co	1	Output	Carry out signal

Table 4: Port description of 8 bits Carry-Ripple Adder

3.3 Functional Description

- Two 8-bit operands are feed to **a** and **b** respectively.
- **c** is 8-bit result of a + b operation (**c = a + b**).
- **co** represents the carry out bit of the a + b operation.

3.4 Timing Diagram

To verification the module, I use directly full-test input method to generate all possible inputs in testbench file, then compare the output with the created golden_output. Result is 65536/65536 passed. Figure 6 is simulated wave form at some specific time.

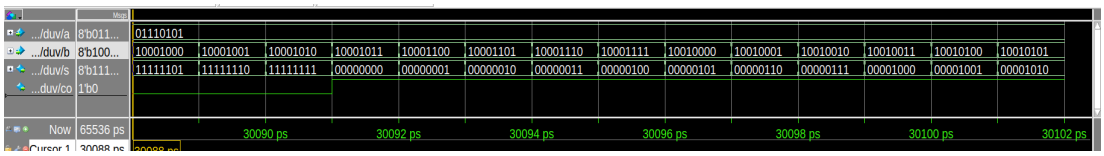


Figure 6: Timing Diagram of Majority module (full-test generated input)