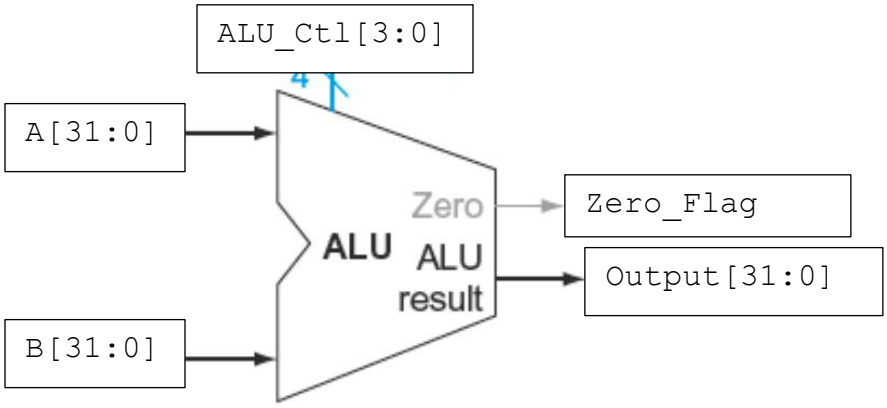
CECS 440
ALU and ALU Control modules

OBJECTIVES:

- Model an **ALU** (Arithmetic Logic Unit) in Verilog
- Verify correct operation of ALU using a Test Fixture in Simulation
- Model the **ALU_Control** module in Verilog
- Verify correct operation of ALU_Control using a Test Fixture in Simulation
- Verify correct operation of ALU controlled by ALU_Control in Verilog simulation

ACTIVITY 1

Model an ALU (Arithmetic Logic Unit) in Verilog (**ALU.v**):



The ALU has:

- Two 32-bit data inputs (designate these as A and B)
- One 4-bit ALU_Ctl which is an operation selection control input
- One 32-bit data Output
- One 1-bit Zero result indicator (Zero_Flag = 1 if Output == 0; Zero_Flag = 0 otherwise)

The ALU is defined by the following table:

ALU_Ctl	Function for Output Assignment	Functional Description
0000	A & B	AND
0001	A | B	OR
0010	A + B	Add
0110	A – B	Subtract
0111	if(A < B))Output = 0x00000001 else Output = 0x00000000	Set on less than
1100	~(A | B)	NOR

Be sure to include a default condition in your case statement where Output will be 32'hXXXXXXXX and Zero_Flag will be 1'bX

Your ALU module source code will be contained in your Lab Submission.

ACTIVITY 2

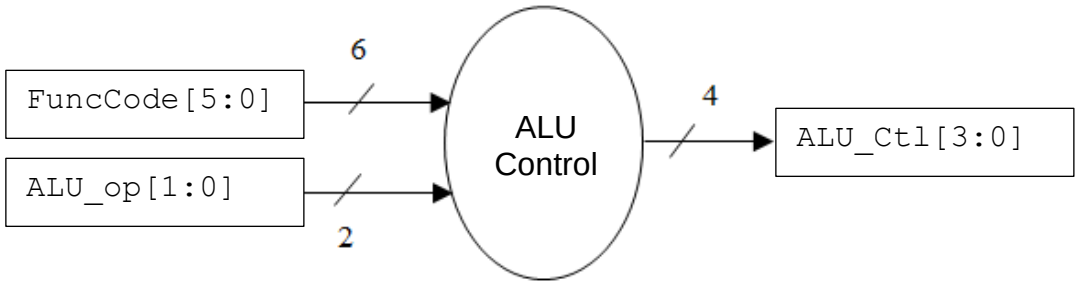
Verify correct operation of ALU using a Test Fixture in Simulation:

Write a simulation that will perform the following series of tests:

- Leave all inputs as 0 initially for the first 10 time units to check if the Zero flag works
- For each of the following test cases:
  - Use the lower 8 digits of your Student ID in hex as the 32-bit value for input A
  - Use the value 0x12345678 as the value for input B
  - Test each valid ALU_Ctl input value to ensure each operates correctly
- For your last test case, provide an invalid ALU_Operation input value i.e. 4'b1111 to ensure the default condition produces garbage outputs as intended.
- DISPLAY SIMULATION VALUES IN HEX FOR EASE OF VIEWING
- *Take a screenshot or multiple screenshots that clearly show the correct test results as they will be contained in your Lab submission.*

ACTIVITY 3

Model a combinatorial ALU_Control module in Verilog (**ALU_Control.v**):



The ALU_Control module is defined by the following table:

Inputs			Output		Instruction Type
FuncCode	**ALU_Op**		**ALU_Ctl**	**Operation**	
100000	10		0010	Add	R-Type
100010	10		0110	Subtract	
100100	10		0000	AND	
100101	10		0001	OR	
101010	10		0111	SLT	
100111	10		1100	NOR	
XXXXXX	00		0010	Address Calculation	I-Type (load/store)
XXXXXX	X1		0110	Equality Comparison	I-Type (branch equal)

Be sure to include a default condition in your case statement.

Your ALU_Control module source code will be contained in your Lab Submission.

CECS 440
ALU and ALU Control modules

ACTIVITY 4

Verify correct operation of ALU_Control using a Test Fixture in Simulation:

- The first test case will have input values set to 0 to test the invalid input scenario
- The following test cases will test each combination of valid inputs once
- *Take a screenshot or multiple screenshots that clearly show the correct test results as they will be contained in your Lab submission.*

ACTIVITY 5

Verify correct operation of ALU controlled by ALU_Control in Verilog simulation:

- Write a single Verilog test fixture that instantiates and interconnects both the ALU created in Activity 1 and the ALU_Control created in Activity 3
- Write a simulation that will test each operation once, for each test case:
  - **Use** the lower order 8 digits **Student ID** as a **8 digit hex** value for **input A**
  - **Use** your **hex valued Student ID + 1** as the value for **input B**
- *Take a screenshot or multiple screenshots that clearly show the correct test results as they will be contained in your Lab submission.*