LOGIC LAB PROJECT

8-bit ALU Design Using Verilog

Princess Sumaya University for Technology
Computer Engineering Department
Digital Logic Lab
Final Project – Fall 2018

30.12.2018

INTRODUCTION

This documentation will cover the designing process of operations in an 8-bit ALU written in Verilog programming language. An arithmetic logic unit (ALU) is a digital circuit that performs arithmetic and bit-wise logical operations on integer binary numbers.

This ALU should be able to perform arithmetic functions with two 8-bit operands, each operation will be selected through a selection register in the main ALU module, each function was written in a separate (.v) file with its own testbench.

The main ALU file takes two inputs to specify the operands & the operation to execute and outputs two values, the Result & the Status which detects overflow & determines whether the result equals to zero.

All modules were implemented using structural models only, except for the Main ALU module.

DESCRIPTION

mainALU(RegSel,Operation,Result,Status)

RegSel

The mainALU module will select 2 operands saved in a register file through (RegSel) where the upper two bits select Op1 and the lower two select Op2, as follows:

RegSel	Operand
00	A
01	B
10	C
11	D

Table(1).Operands Selection

Operation

When (operation) is 0000 the ALU operation will be One's complement for the first

operand, while when (operation) is 0001 the ALU operation will produce Operand 1 Two's complement.

When (operation) is 0010 the ALU operation will be Addition (2's complement) between operand 1 and operand 2, while when (operation) is 0011 the ALU operation will be Subtraction (2's complement) for operand 2 from operand 1.

When (operation) is 0100 the ALU operation will be a Comparator which will give the result of 00 when operand1 equals operand 2, and 01 when operand 1 is greater than operand 2, and 02 when operand1 is less than operand 2.

For the logical gates; operation 0101 will be AND gate, operation 0110 will be OR gate, operation 0111 will be XOR gate, operation 1000 will be XNOR gate.

For operation 1001 it will shift operand 1 to the right and operation 1010 will shift operand 1 to the left. While for operation 1011 the ALU operation will be unsigned addition between operand 1 and operand 2.

ALU Operations

Operation in 4-Bits	Operation In The ALU
0000	onescomplement(Result,OP1)
0001	twoscomplement(Result,OP2)
0010	twosaddition(Result,Cout,OP1,OP2)
0011	subtraction(Result,Cout,OP1,OP2)
0100	comparator(Result,OP1,OP2)
0101	andgate(Result,OP1,OP2)
0110	orgate(Result,OP1,OP2)
0111	xorgate(Result,OP1,OP2)
1000	xnorgate(Result,OP1,OP2)
1001	shiftright (Result,OP1)
1010	shiftleft(Result,OP1,OP2)
1011	addition (Result,Cout,OP1,OP2)

Table(2) Operations In The ALU

Status

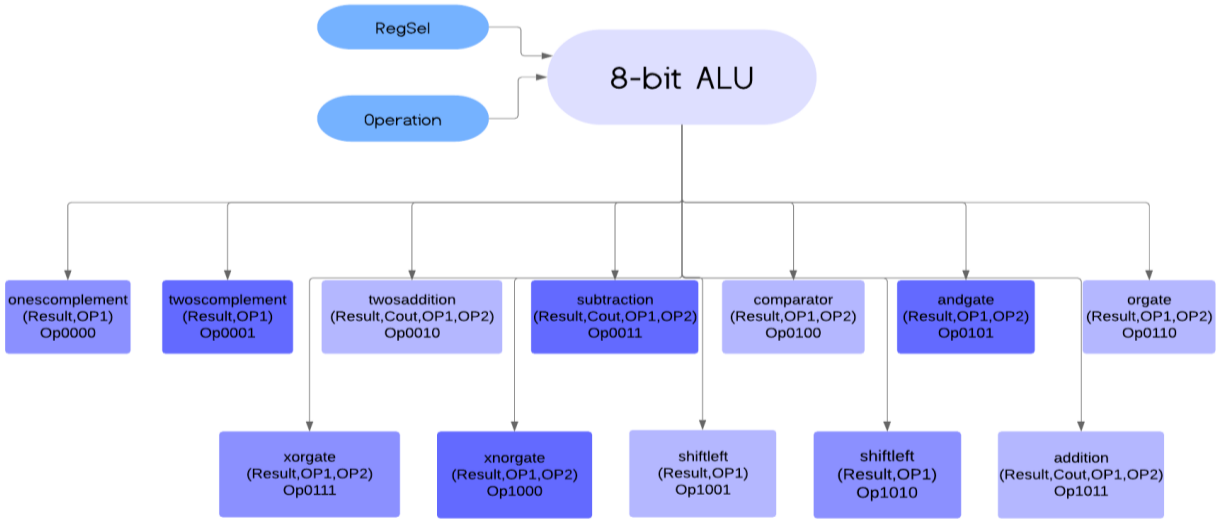
The Status output detects overflow & changes its last three bits according to the following:

Status[2]= 1 when addition operation produces carry, and 0 otherwise
Status[1]= 1 when the result register equals to zero, and 0 otherwise.
Status[0]= 1 when the subtraction operation produces overflow.

Result

The 8-bit Result holds the outcome resulting from the selected operation.

ALU Hierarchical Diagram



Figure(1):Hierarchical Representation of the ALU

CONCLUSION

This 8-bit ALU was implemented using a structural model of Verilog coding, the results were executed correctly from each operation selected, it delivered the correct results for all possible test cases, however, it can get updated with more arithmetic and logical modules to support more operations that match the processor's requirements.