EEE303: DIGITAL ELECTRONICS

CLASS ASSIGNMENT: 32-bit ALU Design

L3/T1/SEC-A: Course Teacher: Upal Mahbub, Assistant Professor, Dept. of EEE, BUET.
Session: February 2013

Objective

The objective of this assignment is to design a 32-bit Arithmetic-Logic Unit (ALU) to enhance the Verilog HDL based system design concept of the class. It is expected that those who will be able to complete this assignment satisfactorily will gain in-depth knowledge on complex logical operations, Verilog HDL syntaxes and digital system design for FPGA/CPLD implementation.

Specification

The system to be designed has 5 inputs, ***Clk***, ***Reset***, ***Selector(4:0)***, ***A(31:0)*** and ***B(31:0)*** and two outputs, ***Result(31:0)*** and ***flags(3:0)***.

- ***Clk*** is the system global clock signal, and its rising edge has to trigger all the memory elements.
- ***Reset*** input signal controls a synchronous system reset, and it is active at level high ('1').
- ***Sselector*** specifies operation to be carried out by the ALU on the input signals.
- ***A*** and ***B*** store the two 32-bit input operands for the arithmetic or logic operations.
- ***Result*** output signal stores the operation result specified by the ***selector*** input.
- ***flags*** output signal contains the flag values resulting from the ALU instruction execution.
    - **flags(3):** "Negative" flag (fn). It indicates whether the resulting value of the selected operation is negative. To generate this flag, it has to be taken into account that both input operands and the output result are interpreted in two's complement. This flag has to be verified for any operation.
    - **flags(2):** “Zero” flag (fz). It indicates that the result of a given operation is zero. This flag has to be verified for any operation.
    - **flags(1):** “Carry” flag (fc). With this propagation flag arithmetic and shift instructions can be chained. It has to be verified only for arithmetic and logic shift operations. For any other an operation its value has to be '0'.
    - **flags(0):** “Overflow” flag (fo). It indicates that an overflow condition has appeared when executing a 2's complement arithmetic operation. This implies that the result is not correctly represented with the selected representation format. This flag has to be verified only when arithmetic operations are executed. For any other operation its value has to be '0'.

TABLE 1
Detailed description of the ALU operations

SERIAL	MNEMONIC	OPCODE	OPERATION
1.	NOP	00000	No operation. ***Result = A***. ***fo***, ***fc***,***fn***→0; ***fz***→1.
2.	OR	00001	***Result = A OR B***
3.	XOR	00010	***Result = A XOR B***
4.	NOR	00011	***Result = A NOR B***
5.	AND	00100	***Result = A AND B***
6.	ADD	00101	***Result = A ADD B***
7.	SUBB	00110	***Result = A SUB B***
8.	NOT	00111	***Result = NOT A***
9.	ROR	01000	The result is obtained by rotating the input ***A*** value to the right the number of positions indicated by the 3 less-significant bits of input ***B.***
10.	RCR	01001	Rotate carry right. Rotate the input ***A*** value to the right the number of positions indicated by the 3 less-significant bits of input ***B along with the carry.***
11.	ROL	01010	The result is obtained by rotating the input ***A*** value to the left the number of positions indicated by the 3 less-significant bits of input ***B.***
12.	RCL	01011	Rotate carry left. Rotate the input ***a*** value to the left the number of positions indicated by the 3 less-significant bits of input ***B along with the carry.***
13.	SLL	01100	The result is obtained by shifting the input ***A*** value to the left the number of positions indicated by the 3 less-significant bits of input ***B.*** The less significant part of the result is filled with '0' for each encoded shift.

14.	SLR	01101	Shift-right logical. The result is obtained by shifting the input ***A*** value to the right the number of positions indicated by the 3 less-significant bits of input ***B.*** The most significant part of the result is filled with '0' for each encoded shift.
15.	SRA	01110	Shift-right arithmetic. The result is obtained by shifting the input ***A*** value to the right the number of positions indicated by the 3 less-significant bits of input ***B.*** The most significant part of the result is filled with the value of input ***A*** most significant bit for each encoded shift.
16.	BITSET	01111	This instruction sets (forces to '1') the bit of input ***A*** that is located at the position encoded by the 5 less-significant bits of input ***B***.
17.	BITCLEAR	10010	This instruction resets (forces to '0') the bit of input ***A*** that is located at the position encoded by the 5 less-significant bits of input ***B***.
18.	RANDOMSET	10011	Loads the value of input ***B*** to the pseudo-random number generator.
19.	RANDOMIZE	11011	The result signal returns the current content of the pseudo-random number generator.
20.	BCD	11100	The result signal returns BCD representation of A.
21.	GRAY	11101	Produce the **A**'th Gray Code.

The following considerations have to be taken into account for the ALU proper implementation:

- The only sequential element of the design is the pseudo-random number generator.
- Inputs ***A***, ***B*** and ***Selector*** are externally registered, thus **they do not have to be loaded into any register**. These inputs are stable during a ***Clk*** clock period. For simulation purposes, input changes have to be produced with a 2 ns delay after the clock rising edge.
- All instructions are performed in a single clock cycle.
- System clock frequency is 25 MHz.
- Concerning arithmetic operations, functional subsystems can be implemented with any architectural option. However, the selected option should have the minimum area occupancy. Moreover, these subsystems must operate properly at the specified clock frequency.
- The entity input and output ports have to be named as specified.

The tasks to be done in this laboratory assignment are:

- Describe the ALU in Verilog HDL.
- Synthesize with Precision. Verify that the resulting flip-flop number from synthesis is exactly 32, that is, the corresponding to the LFSR, and make sure that no additional sequential elements (latchs) appear. Obtain the critical path area and delay results.
- Simulate with Quartus (Functional Simulation). Verify that the ALU still operates properly at the specified clock frequency.
- Use the waveform file to demonstrate the functions of the ALU. Bonus Marks will be given for hardware implementation.

Submission Instructions:

1. Must Submit the Complete Assignment within **9th Week (HARD DEADLINE).** Will deduct 10% marks for each day after 9th Week and 5% marks for each instruction left undone.
2. **Mandatory** Individual Submission.
3. **Bonus Marks** for clean and efficient design, early submission and conceptual soundness.

Contact:
Upal Mahbub
Assistant Professor, Department of EEE, BUET
E-mail: omeecd@yahoo.com
Phone: +88 01717561826
Website: www.upalmahbub.webnode.com