CPE462 - VHDL: Simulation and Synthesis - Fall'11

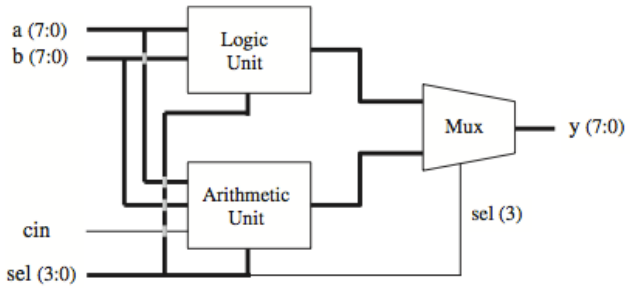
Prof. Nuno Alves (nalves@wne.edu), College of Engineering

Homework Assignment #8

Due Date: Friday, October 14th 2011

Homework policy: If I see something unusual about your work, you can be sure I will ask you about it. If you are unable to explain it to me why you chose some implementation method I will have to consider your entire solution wrong. Now, what is my definition of "unusual work"? Well, things like suspicious parts of code that may have been copied from a website or chunks of your implementation that are very close to one of your colleagues. I really encourage communication amongst yourselves, and I want you guys to have a good time learning while doing these exercises... but please don't blindly copy code. If you really must, at least acknowledge your sources.

1. An ALU (Arithmetic Logic Unit) is shown in below. As the name says, it is a circuit capable of executing both kinds of operations, arithmetic as well as logical. Its operation is described in the associated truth table. The output (arithmetic or logical) is selected by the MSB of sel, while the specific operation is selected by sel's other three bits.



sel	Operation	Function	Unit
0000	y <= a	Transfer a	Arithmetic
0001	y <= a+1	Increment a	
0010	y <= a-1	Decrement a	
0011	y <= b	Transfer b	
0100	y <= b+1	Increment b	
0101	y <= b-1	Decrement b	
0110	y <= a+b	Add a and b	
0111	y <= a+b+cin	Add a and b with carry	
1000	y <= NOT a	Complement a	Logic
1001	y <= NOT b	Complement b	
1010	y <= a AND b	AND	
1011	y <= a OR b	OR	
1100	y <= a NAND b	NAND	
1101	y <= a NOR b	NOR	
1110	y <= a XOR b	XOR	
1111	y <= a XNOR b	XNOR	

Using the code below as a template fill in the blanks to provide the ALU with the desired functionality. You don't have to load the code in active HDL, but it may help to validate your work.

```
1  ------------------------------------------
2  LIBRARY ieee;
3
4
5  ------------------------------------------
6  ENTITY ALU IS
7     PORT (a, b: IN STD_LOGIC_VECTOR (7 DOWNTO 0);
8                sel: IN STD_LOGIC_VECTOR (3 DOWNTO 0);
9                cin: IN STD_LOGIC;
10               y: OUT STD_LOGIC_VECTOR (7 DOWNTO 0));
11 END ALU;
12 ------------------------------------------
13 ARCHITECTURE dataflow OF ALU IS
14    SIGNAL arith, logic: STD_LOGIC_VECTOR (7 DOWNTO 0);
15 BEGIN
16    ----- Arithmetic unit: ------
17    WITH sel(2 DOWNTO 0) SELECT
18       arith <=  a WHEN "000",
19                 a+1 WHEN "001",
20
21
22
23
24
25
26    ----- Logic unit: -----------
27    WITH sel(2 DOWNTO 0) SELECT
28       logic <=
29
30
31
32
33
34
35
36    -------- Mux: ---------------
37    WITH sel(3) SELECT
38       y <=
39
40 END dataflow;
41 ------------------------------------------
```

2. In we class saw the design of a multiplexer using WHEN statements. Those circuits were for a pre-defined number of inputs (4 inputs) and a pre-defined number of bits per input (1 bit). A truly generic mux is depicted in the figure below. In it, n represents the number of bits of the selection input (sel), while m indicates the number of bits per input. The circuit has 2n inputs (notice that there is no relationship between m and n).

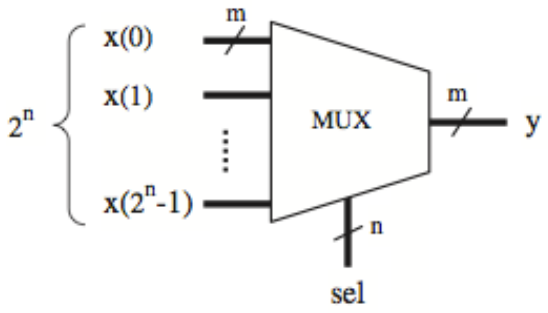


Diagram for a generic multiplexer

a) Using a GENERIC statement to specify n, and assuming m=1, design this circuit and turn in a copy of the code.

Hint #1: To covert a std_logic_vector data-type into an integer you can use the function conv_integer(xxx) where 'xxx' is the signal with the std_logic_vector data-type. Remember that the library use *ieee.std_logic_unsigned.all;* must be declared.

Hint #2: Remember that in WHEN statements, you can use other comparison attributes, which means you are not constrained to the "=" operator.

b) Design a test-bench that will test this circuit. Turn in copy of your test-bench and the outcome of the test-bench simulation when n=3.