

## Homework #3

Grading will be based on:

- Correctness of the design [40 %]
- Compliance with specifications [10 %]
- Effectiveness of testing [40 %]
- Coding style [10%]

### Problem #1 (20 pts.)

Given the input waveform and the logic circuit of Fig 3.1, use VHDL to model the circuit with the following four different propagation delays.

Out1 <= Input after 8 ns;

Out2 <= Input after 2 ns;

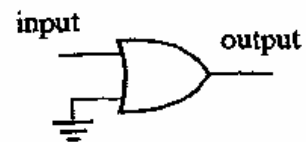
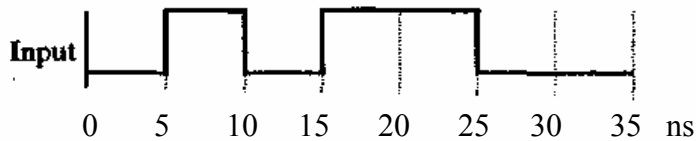
Out3 <= reject 2 ns inertial Input after 8 ns;

Out4 <= reject 10 ns inertial Input after 8 ns;

Simulate the model for 35 ns, and printout the resulting waveforms for Input, Out1, Out2, Out3, and Out4.

(a) Submit your VHDL model.

(b) Submit the waveforms resulting from the simulation of your model.



**FIGURE 3.11**

Problem #2  
(30 pts.)

### Simulation Exercise 3.2: A Single-Bit ALU

Consider a simple one-bit ALU as shown in Figure 3.10 that performs the AND, OR, and ADDITION operations. The result produced at the ALU output depends on the value of signal OPCODE. Write and simulate a model of this ALU, using concurrent signal assignment statements. Test each OPCODE to ensure that the model is accurate by examining the waveforms on the input and output signals. Use a gate delay of 2 ns, a delay of 6 ns through the adder, and a delay of 4 ns through the multiplexor

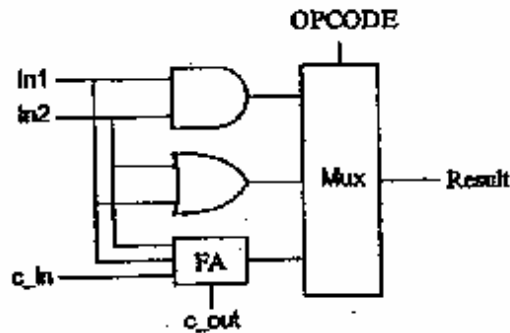


FIGURE 3.10 A single-bit ALU (FA = full adder)

Remember, while the OPCODE field is two bits wide, there are only three valid inputs to the multiplexor.

**Step 1** Compile *alu.vhd*.

**Step 2** Generate a sequence of inputs that you can use to verify that the model is functioning correctly. *(alu\_tb.vhd)*

**Step 3** Open a trace window with the signals you would like to trace. Include internal signals, which are signals that are not entity ports in the model.

**Step 4** Run the simulation for 50 ns.

**Step 5** Check the trace to determine correctness.

**Step 6** Print and record the trace.

- Submit VHDL code for both design and testbench.
- Submit the waveforms used for testing (with comments). Make sure to illustrate that the system works as expected

Problem #3  
(50 pts.)

### Simulation Exercise 4.1: Combinational Shift Logic

This exercise is concerned with the construction of a combinational logic shifter shown in Figure 4.9. The inputs to the shift logic include a 3-bit operand specifying the shift amount, two single-bit signals identifying the direction of the shift operation—left or right—and an 8-bit operand. The output of the shift logic is the shifted 8-bit operand. These shift operations are logical shifts and therefore provide zero fill. For example, a left shift of the number 01101111 by 3 bit positions will produce the output 01111000.

**Step 1.** Create a text file with the entity description and the architecture description of the shift logic. Assume that the delay through the shift logic is fixed at 40 ns, independently of the number of digits that are shifted. While you can implement this behavior in many ways, for this assignment use a single process and the sequential VHDL statements to implement the behavior of the shift logic. You might find it useful to use the concatenation operator & and addressing within arrays to perform the shift operations. For example, we can write the following assignment:

```
dataout <= datain(4 downto 0) & "000";
```

This assignment statement will perform a left shift by three digits with zero fill. Both input and output operands are 8-bit numbers. For VHDL'93, you may use the VHDL built-in shift operators. Use the case statement to structure your process.

- Step 2.** Use the types `std_logic` and `std_logic_vector` for the input and output signals. Declare and reference the library `IEEE` and the package `std_logic_1164`.
- Step 3.** Create a sequence of test vectors. Each of the test vectors will specify the values of (1) the `shiftright` and `shiftright` single-bit control signals, (2) an 8-bit input operand, and (3) a 3-bit number that specifies the number of digits the input operand is to be shifted. Your test cases should be sufficient to ensure that the model is operating correctly.
- Step 4.** Load the simulation model into the simulator. Set the simulator step time to be equal to the value of the propagation delay through the shift logic.

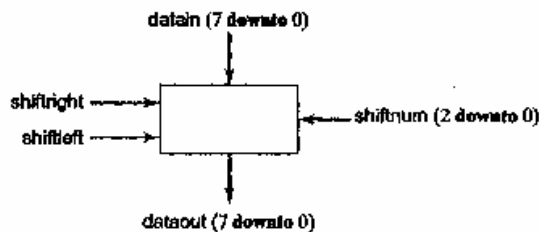


FIGURE 4.9 Interface description for a combinational logic shifter

- Step 5.** Using the facilities available within the simulator, generate the input stimulus and open a trace window to view both the input stimulus and the output operand value.
- Step 6.** Exercise the simulator by running the simulation long enough to cover your test cases. Verify correct operation from the trace.

- (a) Submit VHDL code for both design and testbench.
- (b) Submit the waveforms used for testing (with comments). Make sure to illustrate that the system works as expected