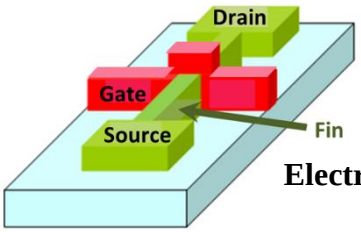

UNIVERSITY OF TEHRAN
Electrical and Computer Engineering Department
ECE (8101) 432

Object Oriented Modeling of Electronic Circuits – Spring 93-94
Midterm Exam

Computer Account#_____

First Name:_____

Last Name:_____

Student Number:_____

Signature:_____

Grade:

Problem 1. _____/25

Problem 2. _____/25

Problem 3. _____/25

Problem 4. _____/25

Total: _____/100

DO NOT USE LAPTOPS
EXTRA SHEETS WILL NOT BE ACCEPTED
YOU MUST SHOW COMPLETE WORK ON ALL PROBLEMS
YOU HAVE EXACTLY 150 MINUTES FOR WORKING ON THIS TEST
THIS IS AN OPEN BOOK OPEN NOTE EXAM, NO SHARING ALLOWED

Background) A single bit comparator circuit (*bit_comparator*) has two data inputs, three control inputs, and three compare outputs. The logical symbol for this circuit is shown in Figure 1. The three control inputs provide a mechanism for generation of multi-bit comparators by cascading several *bit_comparator*s.

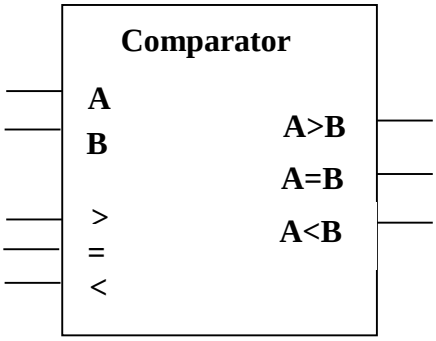

FIGURE 1 Logical symbol of a single bit comparator

The *A*>*B* output is 1 if the *A* input is greater than the *B* input (*AB* is 10) or if *A* is equal to *B* and the > input is 1. The *A*=*B* output is 1 if *A* is equal to *B* and the = input is 1. The *A*<*B* output is the opposite of the *A*>*B* output. This line becomes 1 if *A* input is less than *B* output (*AB* is 01) or if *A* is equal to *B* and the < input is 1. The gate-level circuit diagram of the *bit_comparator* is shown in Figure 2.

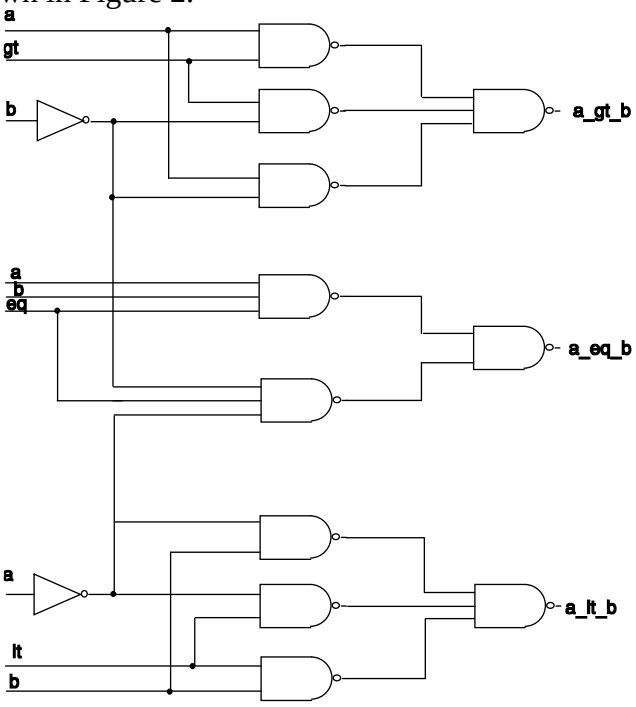

FIGURE 2 Logical diagram of bit_comparator

A 4-bit comparator with two 4-bit data inputs, three control inputs, and three compare outputs is shown in Figure 3. The *A*>*B* output is 1 when data on the *A* input, treated as a 4-bit positive number, is greater than the 4-bit positive number on *B*, or when data on *A* and *B* are equal and the > input is 1. This arrangement can easily be built by cascading four *bit_comparator*s as shown in Figure 4.

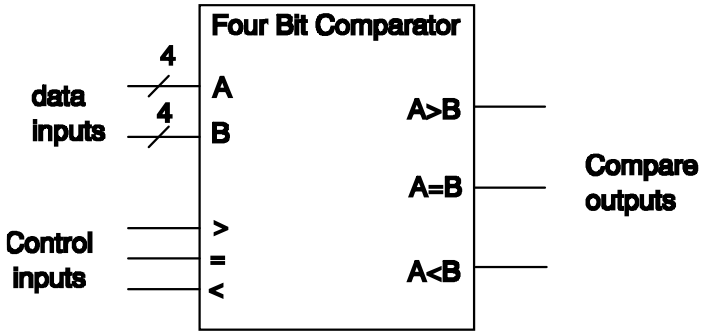

FIGURE 3 Logical symbol of a 4-bit comparator

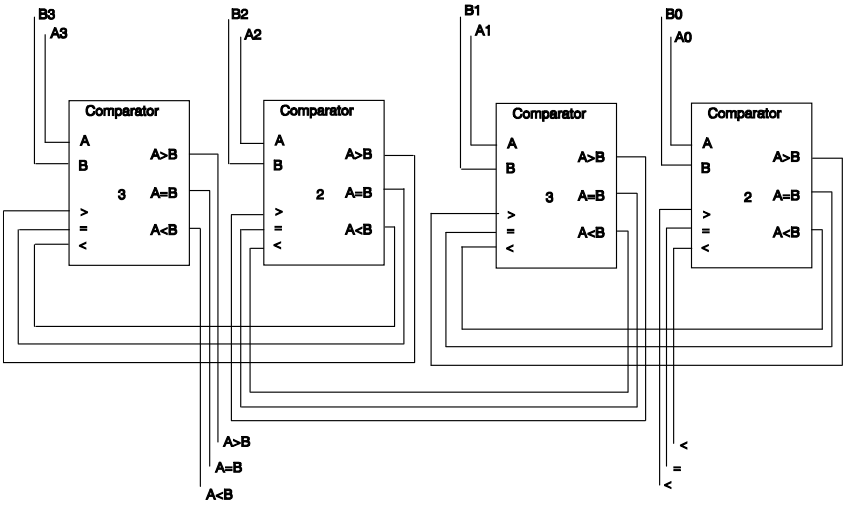

FIGURE 4 A 4-bit comparator using four single bit comparators

C/C++ Gate Level Classes

1. In this problem, you are to:

A) Write a C++ **primitive** class for a 1-bit comparator (shown above). This class has an *evl()* function that sets its outputs according to the comparator functionality .

B) Write a 4-bit comparator using the 1-bit comparator that you write in previous part.

C) Write a testbench and verify your comparator.

C/C++ RTL Components

2. In this problem you are to design an 8-bit multiplier. This multiplication is performed by adding the multiplicand *A* to itself *B* times, where *B* denotes the multiplier. A multiplier circuit has two 8-bit inputs *A* and *B* and its 16-bit output, *W*. After a complete positive pulse appears on *go*, the circuit reads the *A* and *B* operands from the corresponding busses, after which the multiplication process begins. After the completion of the multiplication process, the multiplier places *W* on the corresponding busses and issues a one-clock duration pulse of *ready*. Following this, the circuit returns to its starting state waiting for another pulse on *go*. In this assignment you should:

A) Show the complete datapath of this multiplier.
B) Show the complete state diagram for the controller.
C) Write datapath of this circuit using RTL C++.
D) Write the controller description using RTL C++.
E) Complete the circuit by putting together the datapath components and controller classes in the proper order for correct simulation.

SystemC Linguistic

3. The following code is a complete SystemC example.

```
#include "systemc.h"
class comp1 : public sc_module {
    public:
        sc_out<bool> sigout;
        SC_CTOR(comp1) { }
};
class comp2 : public sc_module {
    public:
        sc_in<bool> sigin;
        SC_CTOR(comp2) { } };
// main function for preparing the system to
verify,
// and run the simulation
int sc_main(int argc, char *argv[]) {
    comp1 compa("compa");
    comp2 compb("compb");
    comp2 compc("compc");
    sc_signal<bool> x;
    compa.sigout(x);
    compb.sigin(x);
    compc.sigin(x);
    sc_start();
    return 0; }
```

A) Draw a diagram of the hardware described in this example.
B) Explain what the line "SC_CTOR(comp1) { } ;" does.
C) If compiled and executed the simulator will run on this example. What line of code invokes the simulator?
D) Write a method for comp1 so that it toggles its output every 17 ns with the initial value of 0.
E) Write a method for comp2 so that it prints "Activated" every time there is an event on its input.

SystemC RTL Design

4. Show a SystemC model for an 8-bit serial adder with two serial data inputs *ain* and *bin*, and a control input, *start*. The serial adder block diagram is shown in Figure 5. The circuit has an eight bit result output (*result*) and a *ready* signal. After a complete pulse on *start*, operand data bits start showing up on *ain* and *bin* with every clock with least significant bits coming in first. In eight clock pulses as input data come into the circuit, they are added and the result becomes ready on *result*. At this time the *ready* signal becomes 1 and it remains 1 until a 1 is detected on the *start* input. While the circuit is performing its data collection and addition, pulses on *start* are ignored.

A) Show the complete datapath of this adder.
B) Show the complete state diagram for the controller.
C) Write datapath of this circuit using RTL SystemC.
D) Write the controller description using RTL SystemC.
E) Complete the circuit by putting together the datapath components and controller.

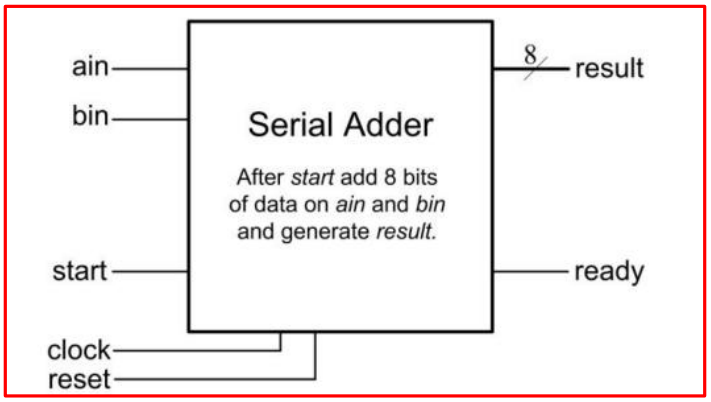

Figure 5 Serial Adder Block Diagram