ECE C03
LAB#4
Design of Latches, Flip-Flops, and a Finite State Machine

Spring 1998
Assigned: Tuesday, May 12, 1998
Due: Tuesday, May 26, 1998
at the BEGINNING of class

1 Introduction

In this lab you will compare the behavior of the following: SR latch with control input, master-slave SR F/F, D F/F, 4 bit register, and 8 register register file. The first two circuits share the same function table (Table 1) but differ significantly in the conditions under which their outputs will change. The third circuit (D F/F) has a slightly different function table (Table 2). A register is a collection of D F/F's that can store complete words of data, and a register file is a collection of registers.

TABLE 1

S	R	Q $\overline{Q}$
0	0	no change
0	1	Q=0, $\overline{Q}$=1
1	0	Q=1, $\overline{Q}$=0
1	1	Undefined

SR Function Table

TABLE 2

D	Q $\overline{Q}$
0	Q=0, $\overline{Q}$=1
1	Q=1, $\overline{Q}$=0

D Function Table

In this lab you will compare two different ways to implement a single FSM. You will be given a FSM and two different state assignments. You will need to determine the logic equations for both state assignments necessary to implement the FSM, pick the most efficient one, and im plement it using Design Archictect.

For this lab, create a new subdirectory, *lab4,* and place all your circuits, symbols, and simulation files in this directory.

2 Circuit Diagrams

2.1 SR Latch with Control Input

Use Design Architect to enter the clocked SR latch of Fig. 1 below. Call the circuit inputs S, R, and C and the outputs Q and Qbar. Leave the rise and fall times of each gate at zero and label your circuits as in previous labs.

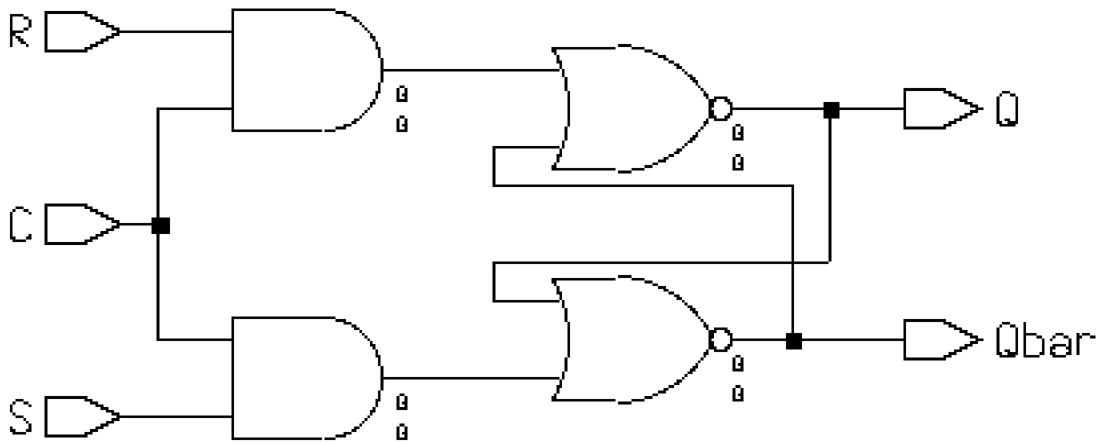


Figure 1 SR Latch with Control Input

Use the automatic symbol generator to create a symbol for your SR latch similar to Fig 2. This will be used in the master-slave F/F of Section 2.2. Remember to save the symbol, and then check the symbol and the schematic . If you get warnings, save the symbol again and check the symbol again. The Mentor system is very picky, and sometimes doesn't instantiate the cell correctly if it has not been checked successfully.

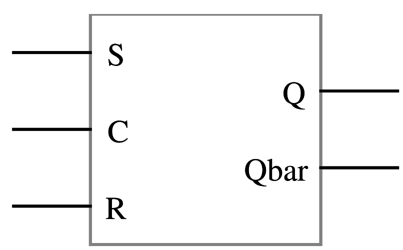


Figure 2 SR Latch Symbol

2.2 Master-Slave SR Flip-Flop

Use Design Architect to enter the master-slave SR F/F of Fig. 3 below. The inverter is placed from *inv* in *gen_lib* and the master and slave latches by activating the symbol you created in Section

2.1. Include *portout* connectors on the Y and Ybar nets so that these signals can be displayed when you simulate.

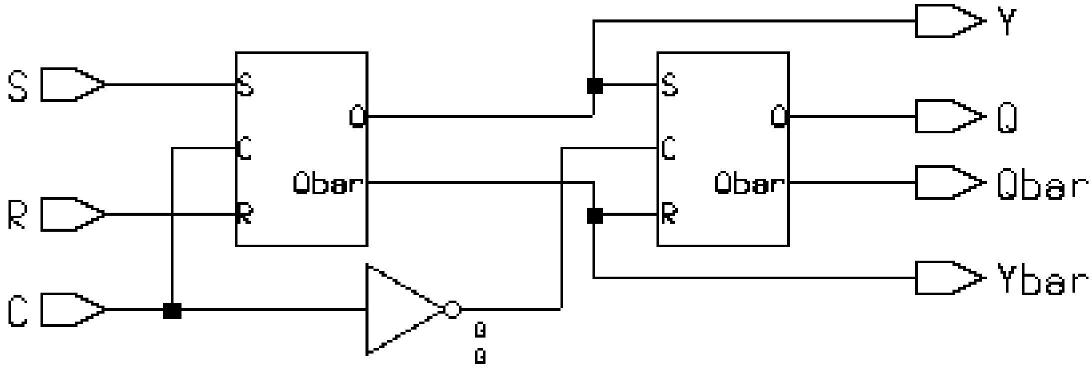


Figure 3 Master-Slave SR F/F

2.3 D Flip-Flop

Use Design Architect to enter the D F/F circuit of Fig. 4 below.

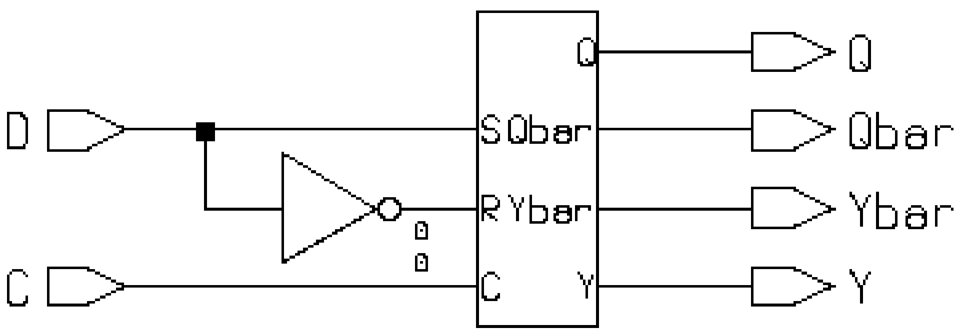


Figure 4 D Flip-Flop

2.4 Register

The register is made up of four D Flip-Flops from the *gen_lib* (not the D FF you designed earlier). These each have a preset and a clear signal. The preset signal is pulled high to disable it, and the clear is taken to form an active low reset.

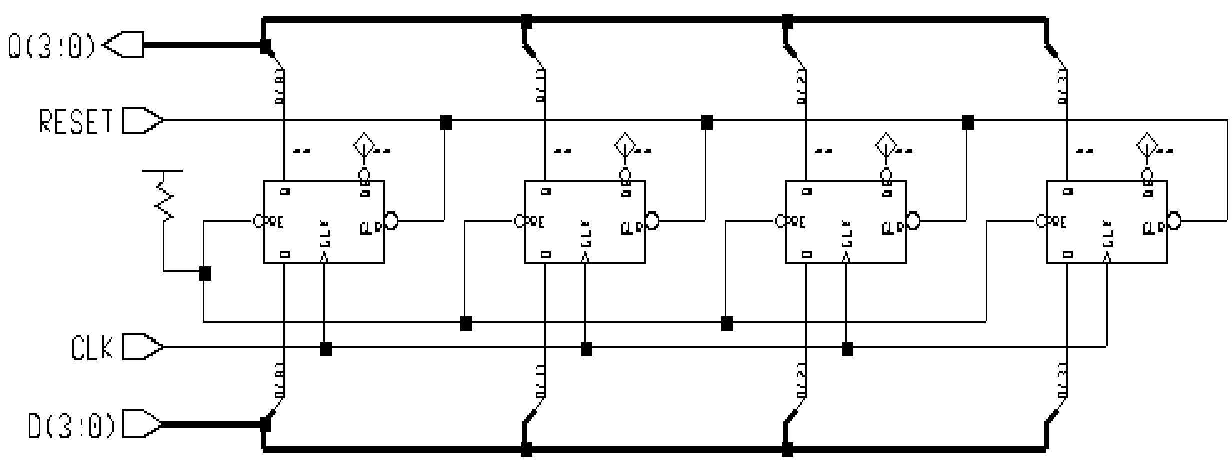


Figure 5 Register

3 Testing and Simulation for Registers

From an *xterm* window, type *ecec03* to enter your class work space and type *cd lab4* to change your current working directory to the *lab4* directory that you created. Use *vi* (or another editor) to enter the following into a simulation command file called *sr_test*.

```
$$force_clock_menu_item("C",10,0,"1",5,"0",void,@absolute,@clear);
$$force_multiple_values_menu_item("S",void,@absolute,
  [0,7,8,22,23,59,61],["0","1","0","1","0","1","0"]);
$$force_multiple_values_menu_item("R",void,@absolute,
  [0,2,3,44,46,63],["0","1","0","1","0","1"]);
```

The first line of this file generates a periodic signal C with period=10.

The *force_multiple_values* commands work as follows: The first parameter is the signal which is being controlled. The second and third parameters can be disregarded, just use *void* and *@absolute*. The last two parameters are a list of times and corresponding signal values to be forced at those times. For example, the force command for the signal *"S"* means that the signal S will have the following properties: S will be 0 at time 0, 1 at time 7, 0 at time 8, 1 at time 22, 0 at time 23, 1 at time 59, and 0 after time 61. The list of times and corresponding values must be in separate lists, bounded by square brackets with commas between elements.

Use Quicksim and the above simulation file to simulate each of your circuits: SR latch with control input, master-slave SR F/F, and positive-edge-triggered SR F/F. To do this, with the arrow in the Quicksim window, type on the keyboard *dofile lab4/sr_test*. A small window should popup when you type the text. Upon hitting return, Quicksim will execute the commands in the file. Add a trace window, showing the traces for C, S, R, Q, and Qbar for each test. In addition, add traces for Y and Ybar when simulating the master-slave F/F and the D F/F. For this lab Initialize each simulation by doing *Run>Initialize:* and entering "Xr" when prompted for a state value, and then run each simulation for time <u>80</u>. You will need to simulate each circuit in turn, leaving Quicksim between tests. Add force commands to *sr_test* to send CLR low at t=0 and high at t=1.

Simulate a single register by showing the traces for D, Q, Reset, and Clock. Create a new simulation file to test that the reset (active low) and the register itself works correctly.

Generate plots for all your tests. Label your simulation plots (by hand) with the name of the circuit. You should have a plot of your register, sr latch, ms f/f, and d-f/f giving a total of 4 trace files.

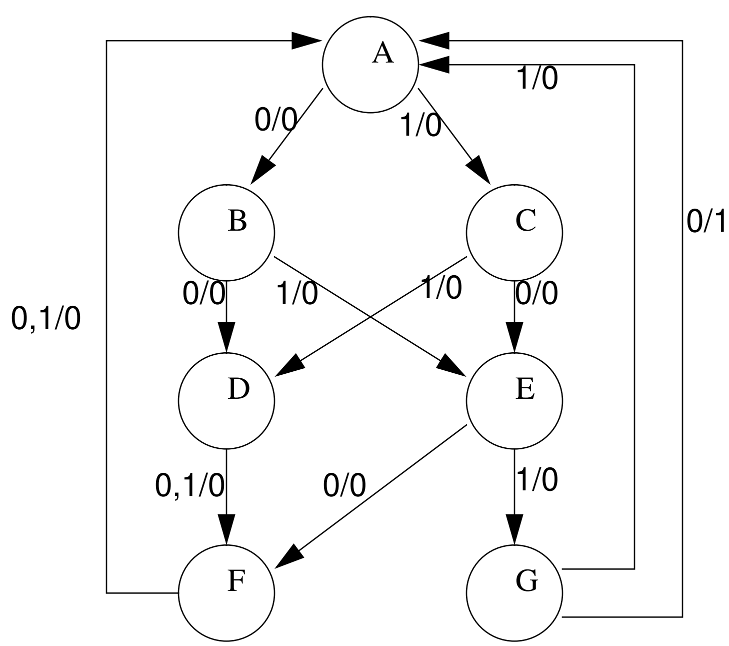
4 Finite State Machine

Figure 1 is the drawing of a simple finite state machine with 7 states. This FSM acts as a 0110 or 1010 pattern detector, thus the output is a 1 whenever one of the two patterns are detected and a zero otherwise. As this state machine has 7 states, you will need 3 bits to encode the values of all the states, one of the state encodings will be left unused and this can be used to optimize the number of gates by including this as a don't care. The two state encodings are listed in Table 1, you will need to determine the equations for them both, and implement the most efficient one.

You should implement this as a Moore machine, with three D F/F's to hold the present state. Design your FSM using the D F/F's in the Mentor Graphics *gen_lib* library, so that you can reset the entire FSM to the initial state, using the method used in the last lab to reset the registers.

State	Assignment #1	Assignment #2
A	000	000
B	001	010
C	010	011
D	011	110
E	100	111
F	101	100
G	111	101

Table 1



5 Testing and Simulation

From an *xterm* window, type *ecec03* to enter your class work space and type *cd lab5* to change your current working directory to the *lab5* directory that you created. Use *vi* (or another editor) to enter the following into a simulation command file called *fsm_test*.

```
$$force_multiple_values_menu_item("/reset",void,@absolute,[0,1],["1","0"]);
$$force_clock_menu_item("/Clk",10,0,"1",5,"0",void,@absolute,@clear);
$$force_multiple_values_menu_item("/x",void,@absolute,
[  0,
   1, 11, 21, 31,
  41, 51, 61, 71,
  81, 91,101,111,
 121,131,141,151,
 161,171,181,191,
 201,211,221,231,
 241,251,261,271],
["0",
 "1","0","1","0",
 "1","1","1","0",
 "0","1","1","1",
 "0","1","1","0",
 "0","1","0","1",
 "1","1","0","0",
 "0","0","1","1"]);
```

Use Quicksim and the above simulation file to simulate your circuit. To do this, with the arrow in the Quicksim window, type on the keyboard dofile lab5/fsm_test. A small window should popup when you type the text. Upon hitting return, Quicksim will execute the commands in the file. Add a trace window, showing the traces for each of the flip-flop outputs, the input, the clock and the output. For this lab Initialize each simulation by doing Run>Initialize: and entering ”Xr” when prompted for a state value, and then run each simulation for time 300.

After simulation, generate a plot of your test, and label it with the name of your circuit

TURN IN the following items for a grade (**stapled** in this order):

- √ A sheet plot of each circuit (5 total).
- √ A simulation trace of each circuit (5 total) clearly marked as to which circuit the trace corresponds to.

Be sure to include your name and section number on each page.