Computer Architecture
CPSC 32, Fall Semester 2003
Lab Assignment #4
Due: Week of November 3– November 7, demonstrate in your lab, complete by yourself

1 Assignment

Problem 1 [**15 points**] Write Verilog code that represents a JK flip-flop. Use behavioral code rather than structural code. Recall that a JK flip-flop has the truth table

J	K	$Q(t+1)$
0	0	$Q(t)$
0	1	0
1	0	1
1	1	$\overline{Q}(t)$

Assume that the state changes on a positive edge.

Problem 2 [**15 points**] Write Verilog code that represents a T flip-flop with an asynchronous clear input. Use behavioral code, rather than structural code.

```
module TFF(clk, T, clr, Q);
  input clk, T, clr;
  output Q;
  ...
endmodule
```

A T flip-flop has the following behavior

`clr`	`T`	$Q(t+1)$
0	x	0
1	0	$Q(t)$
1	1	$\overline{Q(t)}$

Problem 3 [**25 points**] Write a three-bit up/down-counter `updown` using the T flip-flops from the previous exercise. It should have a control input `down` such that if `down`=0 then it should behave as an up-counter, and if `down`=1 then it should behave as a down-counter.

```
module updown(clk, clr, Q);
  input clk, clr;
  output [2:0] Q;
  ...
endmodule
```

Write a testbench that lets `updown` count 15 cycles up, and then 5 cycles down, and then finishes the simulation. Use `$monitor` to trace the output of the updown counter. You can use the m555 module discussed in the lecture to create the clock signal.

Problem 4 [**15 points**] A sequential circuit has two inputs w_1 and w_2, and an output z. Its function is to compare the input sequences on the two inputs. If $w_1 = w_2$ during any four consecutive clock cycles, the circuit produces $z = 1$; otherwise $z = 0$. For example

```
w1: 0110111000110
w2: 1110101000111
 z: 0000100001110
```

Derive a suitable circuit.

Problem 5 [**20 points**] Write a finite state machine in Verilog for the previous problem. Is this a Moore machine?

Problem 6 [**10 points**] The following code checks for adjacent ones in an n-bit vector.

```
always @(A)
begin
  f = A[1] & A[0];
  for(k = 2; k<n; k=k+1)
    f = f | (A[k] & A[k-1]);
end
```

With blocking assignments this code produces the desired logic function $f = a_1 a_0 + \cdots a_{n-1} a_{n-2}$. What logic function is produced if we change the code to use non-blocking assignments? Explain.

Demonstrate your solutions in your lab sessions, and turn in written solutions for Problems 4 and 6 at the same time.

Reading Assignment Read chapter 7 of the book by Brown and Vranesic. This chapter is freely available from the McGraw-Hill website (use google: mcgraw-hill brown vranesic).