ECE 551 Homework #1

Michelle Joy Moravan

February 10, 2006

1. *Give the following numbers in Verilog notation:*

(a) *Decimal 28 in binary digits, 10 bits*
10'b11100

(b) *Octal 37 in decimal digits, 6 bits*
6'd31

(c) *Decimal -42 in hex digits, 11 bits*
-11h'2a

2. *Write the following Verilog vector declarations:*

(a) *Input variable A, 3 bits wide, rightmost bit with highest index, leftmost bit at index 0*
input A[0:2];

(b) *Output variable X, 5 bits wide, leftmost bit with highest index, rightmost bit at index 2*
output X[6:2];

3. *Complete the Modelsim tutorial.*

I have completed the Modelsim tutorial. If I had any problems with it, I discussed them with the TA either in person or through email.

The corresponding waveform appears in Figure 1.

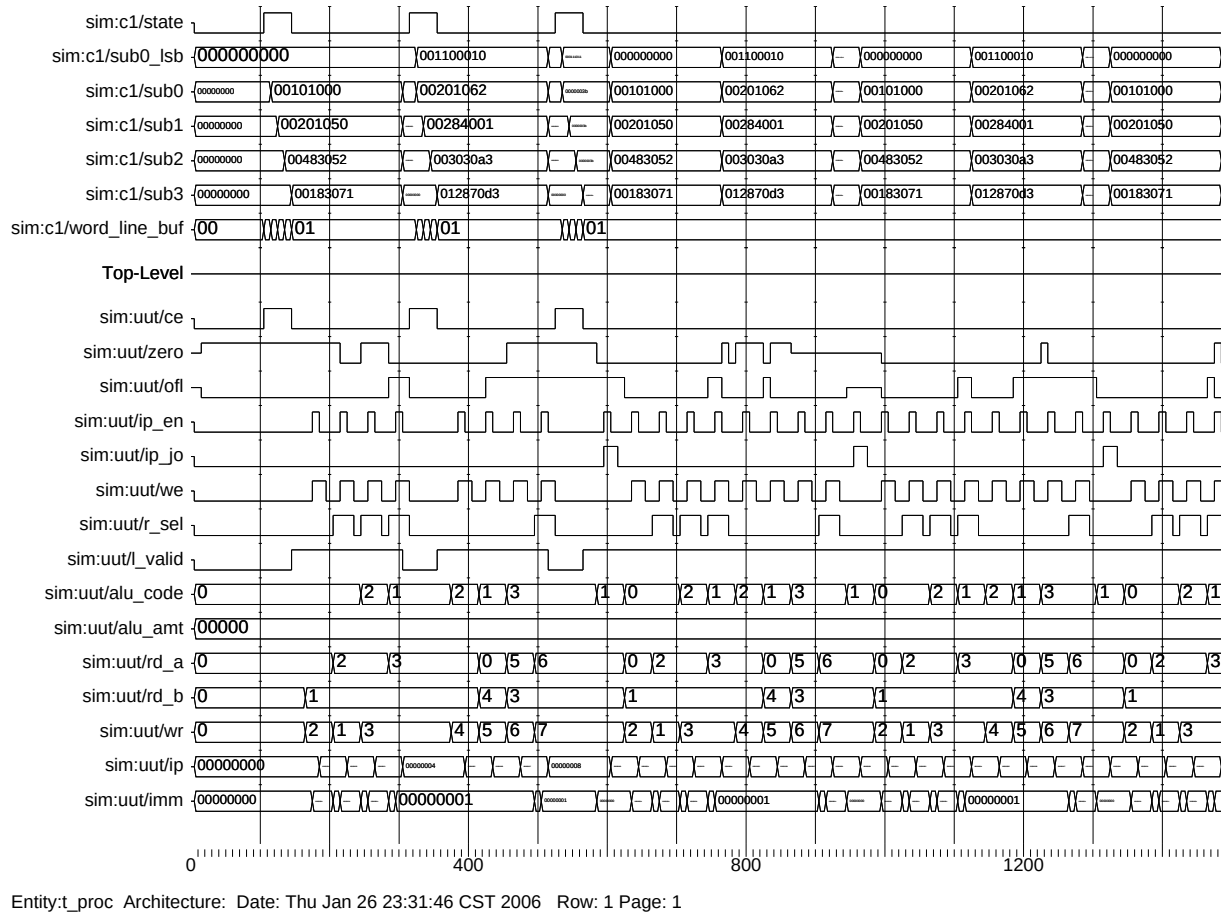

Figure 1: Tutorial Waveform

4. *Create a structural description of the following logic equations...*

```
module problem4(output x, y, input a, b, c, d);
  wire not_b, not_c, not_d;
  wire x1, x2, x3;
  wire y1, y2, y3;

  not(not_b, b);
  not(not_c, c);
  not(not_d, d);

  and(x1, a, not_b);
  and(x2, not_b, not_c);
  and(x3, a, c, not_d);

  and(y1, a, b);
  and(y2, b, c);
  and(y3, a, c);

  or(x, x1, x2, x3);
```

```
  or(y, y1, y2, y3);
endmodule

‘timescale 1ns/1ns
module t_problem4();
    reg[4:0] stim;
    wire xout, yout;

    problem4 m1(xout, yout, stim[3], stim[2], stim[1], stim[0]);

    initial $monitor("%t: x=%b, y=%b, a=%b, b=%b, c=%b, d=%b",
       $time, xout, yout, stim[3], stim[2], stim[1], stim[0]);

    initial #150 $finish;

    initial begin
       for(stim = 0; stim < 16; stim = stim + 1) begin #5;
       end
    end
endmodule
```

Here is the $monitor output:

```
run 120
#                     0: x=1, y=0, a=0, b=0, c=0, d=0
#                     5: x=1, y=0, a=0, b=0, c=0, d=1
#                    10: x=0, y=0, a=0, b=0, c=1, d=0
#                    15: x=0, y=0, a=0, b=0, c=1, d=1
#                    20: x=0, y=0, a=0, b=1, c=0, d=0
#                    25: x=0, y=0, a=0, b=1, c=0, d=1
#                    30: x=0, y=1, a=0, b=1, c=1, d=0
#                    35: x=0, y=1, a=0, b=1, c=1, d=1
#                    40: x=1, y=0, a=1, b=0, c=0, d=0
#                    45: x=1, y=0, a=1, b=0, c=0, d=1
#                    50: x=1, y=1, a=1, b=0, c=1, d=0
#                    55: x=1, y=1, a=1, b=0, c=1, d=1
#                    60: x=0, y=1, a=1, b=1, c=0, d=0
#                    65: x=0, y=1, a=1, b=1, c=0, d=1
#                    70: x=1, y=1, a=1, b=1, c=1, d=0
#                    75: x=0, y=1, a=1, b=1, c=1, d=1
#                    80: x=1, y=0, a=0, b=0, c=0, d=0
```

The corresponding waveform appears in Figure 2.

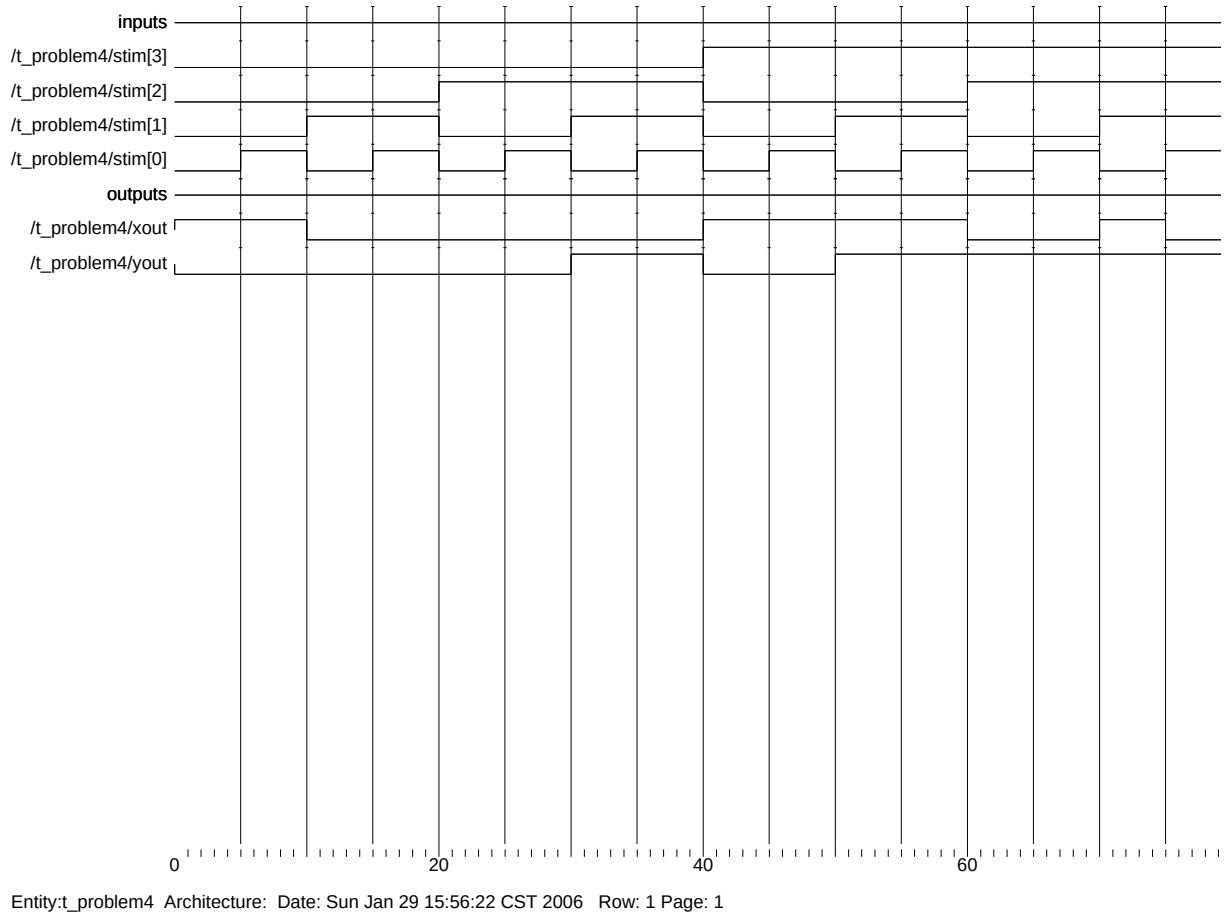

Figure 2: Problem 4 Waveform

5. *Structurally implement a gray-code counter...*

```
module comb_gray_code(output [2:0] out,
                      input [2:0] in);
    wire [2:0] not_in;
    wire x2a, x2b_x1b, x1a, x0a, x0b;

    not(not_in[2], in[2]);
    not(not_in[1], in[1]);
    not(not_in[0], in[0]);

    and(x2a, in[0], in[2]);
    and(x2b_x1b, not_in[0], in[1]);
    and(x1a, in[0], not_in[2]);
    and(x0a, not_in[1], not_in[2]);
    and(x0b, in[1], in[2]);

    or(out[2], x2a, x2b_x1b);
    or(out[1], x1a, x2b_x1b);
    or(out[0], x0a, x0b);
```

```
endmodule

module problem5(output[2:0] out, input clk, rst);
    wire[2:0] next_state;

    comb_gray_code cgc(next_state, out);
    dff dff2(out[2], next_state[2], clk, rst);
    dff dff1(out[1], next_state[1], clk, rst);
    dff dff0(out[0], next_state[0], clk, rst);
endmodule

module t_problem5();
    wire [2:0] out;
    reg clk, rst;

    problem5 gray_counter(out, clk, rst);

    initial $monitor("%t out=%b, rst=%b", $time, out, rst);
    initial #500 $finish;

    initial begin
        clk = 0; forever #5 clk = ~clk;
    end

    initial begin
        rst = 1; #10 rst = 0;
    end
endmodule
```

Here is the $monitor output:

```
run 90
#                    0 out=xxx, rst=1
#                    5 out=000, rst=1
#                   10 out=000, rst=0
#                   15 out=001, rst=0
#                   25 out=011, rst=0
#                   35 out=010, rst=0
#                   45 out=110, rst=0
#                   55 out=111, rst=0
#                   65 out=101, rst=0
#                   75 out=100, rst=0
#                   85 out=000, rst=0
```

The corresponding waveform appears in Figure 3.

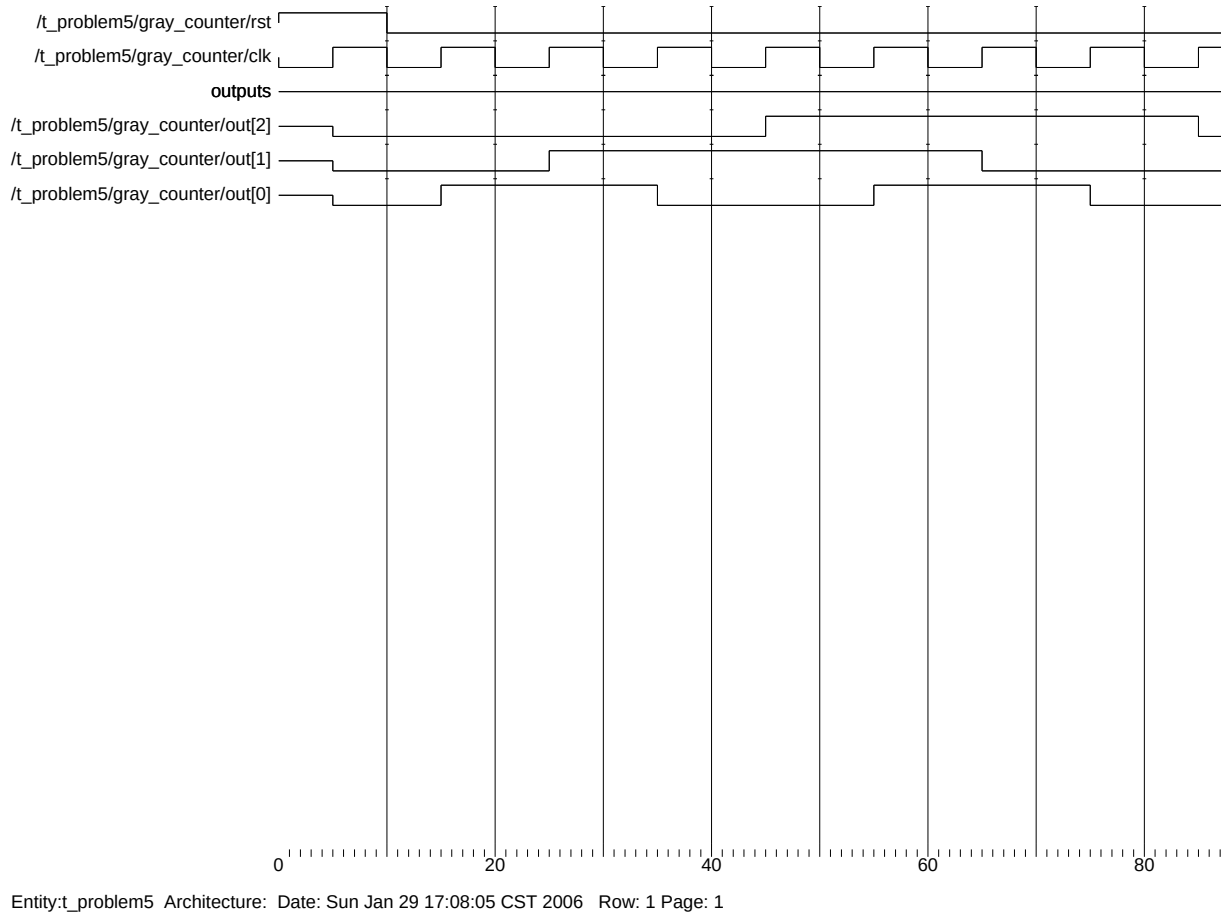

Figure 3: Problem 5 Waveform

6. *Structurally design a finite state machine...*

(a) *Draw a FSM diagram for the circuit.*

The FSM diagram appears in Figure 4. The circles represent the states, and the bits in each circle represent the current state. The arcs show transitions; the corresponding bits show the inputs that lead to each transition. The bits are ordered: dec, inc, rst. All possible combinations that can lead to a transition are listed. If more than one is possible, the answers are comma separated. Finally, an x in an input indicates a don't care.

(b) *Write the structural Verilog to implement the circuit.*

```
module counter(output [2:0] next_state, input [2:0] cur_state, input inc, dec);
    wire [2:0] cur_state_inv;
    wire inc_inv, dec_inv;

    wire ns2_max_a, ns2_max_b, ns2_max_c, ns2_max_d, ns2_max_e, ns2_max_f;
    wire ns1_max_a, ns1_max_b, ns1_max_c, ns1_max_d, ns1_max_e, ns1_max_f;
    wire ns0_max_a, ns0_max_b, ns0_max_c, ns0_max_d;

    not(cur_state_inv[2], cur_state[2]);
    not(cur_state_inv[1], cur_state[1]);
    not(cur_state_inv[0], cur_state[0]);
```

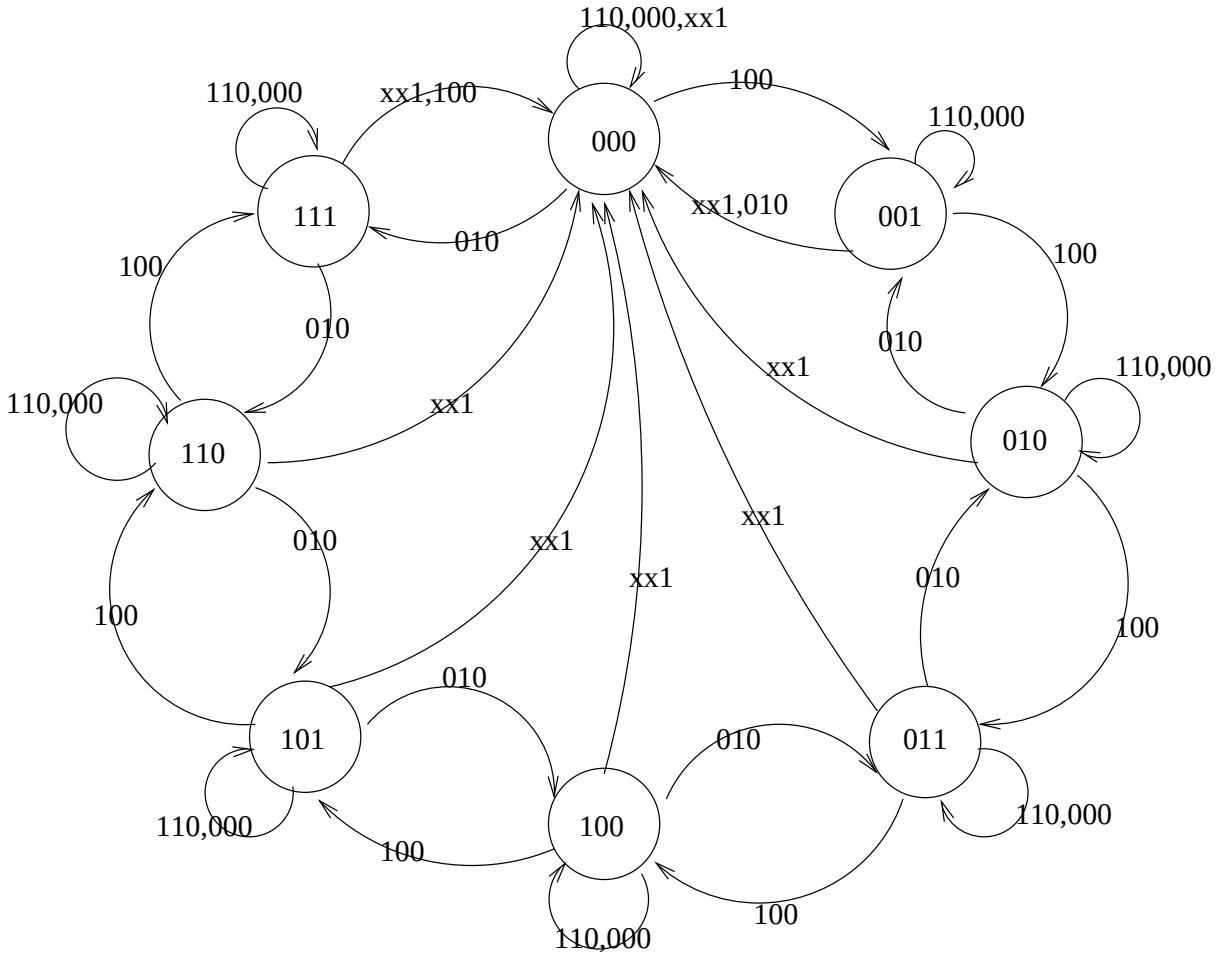

Figure 4: Problem 6a: Finite State Machine

```
not(inc_inv, inc);
not(dec_inv, dec);

and(ns2_max_a, inc_inv, dec, cur_state_inv[2], cur_state_inv[1],
    cur_state_inv[0]);
and(ns2_max_b, inc, dec_inv, cur_state_inv[2], cur_state[1], cur_state[0]);
and(ns2_max_c, inc_inv, dec_inv, cur_state[2]);
and(ns2_max_d, cur_state[2], cur_state[1], cur_state_inv[0]);
and(ns2_max_e, dec, cur_state[2], cur_state[0]);
and(ns2_max_f, inc, cur_state[2], cur_state_inv[1]);

and(ns1_max_a, inc_inv, dec_inv, cur_state[1]);
and(ns1_max_b, inc, dec, cur_state[1]);
and(ns1_max_c, inc, cur_state[1], cur_state_inv[0]);
and(ns1_max_d, inc_inv, dec, cur_state_inv[1], cur_state_inv[0]);
and(ns1_max_e, inc_inv, dec, cur_state[1], cur_state[0]);
and(ns1_max_f, inc, dec_inv, cur_state_inv[1], cur_state[0]);

and(ns0_max_a, inc_inv, dec_inv, cur_state[0]);
```

```
    and(ns0_max_b, inc, dec, cur_state[0]);
    and(ns0_max_c, inc_inv, dec, cur_state_inv[0]);
    and(ns0_max_d, inc, dec_inv, cur_state_inv[0]);

    or(next_state[2], ns2_max_a, ns2_max_b, ns2_max_c, ns2_max_d, ns2_max_e,
       ns2_max_f);
    or(next_state[1], ns1_max_a, ns1_max_b, ns1_max_c, ns1_max_d, ns1_max_e,
       ns1_max_f);
    or(next_state[0], ns0_max_a, ns0_max_b, ns0_max_c, ns0_max_d);
endmodule

module problem6(output [2:0] state, input clk, rst, inc, dec);
    wire [2:0] next_state;

    counter cnt(next_state, state, inc, dec);
    dff dff2(state[2], next_state[2], clk, rst);
    dff dff1(state[1], next_state[1], clk, rst);
    dff dff0(state[0], next_state[0], clk, rst);
endmodule

module t_problem6();
    wire [2:0] state;
    reg clk;
    reg [2:0] commands;

    problem6 p6counter(state, clk, commands[2], commands[1], commands[0]);

    initial $monitor("%t rst=%d, inc=%d, dec=%d, st2=%d, st1=%d, st0=%d",
       $time, commands[2], commands[1], commands[0], state[2], state[1], state[0]);

    initial #500 $finish;

    initial begin
        clk = 0; forever #5 clk = ~clk;
    end

    initial begin
        commands = 3'b100;
        #10 commands = 3'b010;
        #100 commands = 3'b011;
        #10 commands = 3'b001;
        #50 commands = 3'b100;
    end

endmodule
```

(c) *Output the results of simulation...*

```
run 10
#                    0 rst=1, inc=0, dec=0, st2=x, st1=x, st0=x
#                    5 rst=1, inc=0, dec=0, st2=0, st1=0, st0=0
run 100
#                   10 rst=0, inc=1, dec=0, st2=0, st1=0, st0=0
```

```
#                  15 rst=0, inc=1, dec=0, st2=0, st1=0, st0=1
#                  25 rst=0, inc=1, dec=0, st2=0, st1=1, st0=0
#                  35 rst=0, inc=1, dec=0, st2=0, st1=1, st0=1
#                  45 rst=0, inc=1, dec=0, st2=1, st1=0, st0=0
#                  55 rst=0, inc=1, dec=0, st2=1, st1=0, st0=1
#                  65 rst=0, inc=1, dec=0, st2=1, st1=1, st0=0
#                  75 rst=0, inc=1, dec=0, st2=1, st1=1, st0=1
#                  85 rst=0, inc=1, dec=0, st2=0, st1=0, st0=0
#                  95 rst=0, inc=1, dec=0, st2=0, st1=0, st0=1
#                 105 rst=0, inc=1, dec=0, st2=0, st1=1, st0=0
run 10
#                 110 rst=0, inc=1, dec=1, st2=0, st1=1, st0=0
run 50
#                 120 rst=0, inc=0, dec=1, st2=0, st1=1, st0=0
#                 125 rst=0, inc=0, dec=1, st2=0, st1=0, st0=1
#                 135 rst=0, inc=0, dec=1, st2=0, st1=0, st0=0
#                 145 rst=0, inc=0, dec=1, st2=1, st1=1, st0=1
#                 155 rst=0, inc=0, dec=1, st2=1, st1=1, st0=0
#                 165 rst=0, inc=0, dec=1, st2=1, st1=0, st0=1
run 10
#                 170 rst=1, inc=0, dec=0, st2=1, st1=0, st0=1
#                 175 rst=1, inc=0, dec=0, st2=0, st1=0, st0=0
```

The corresponding waveform appears in Figure 5.

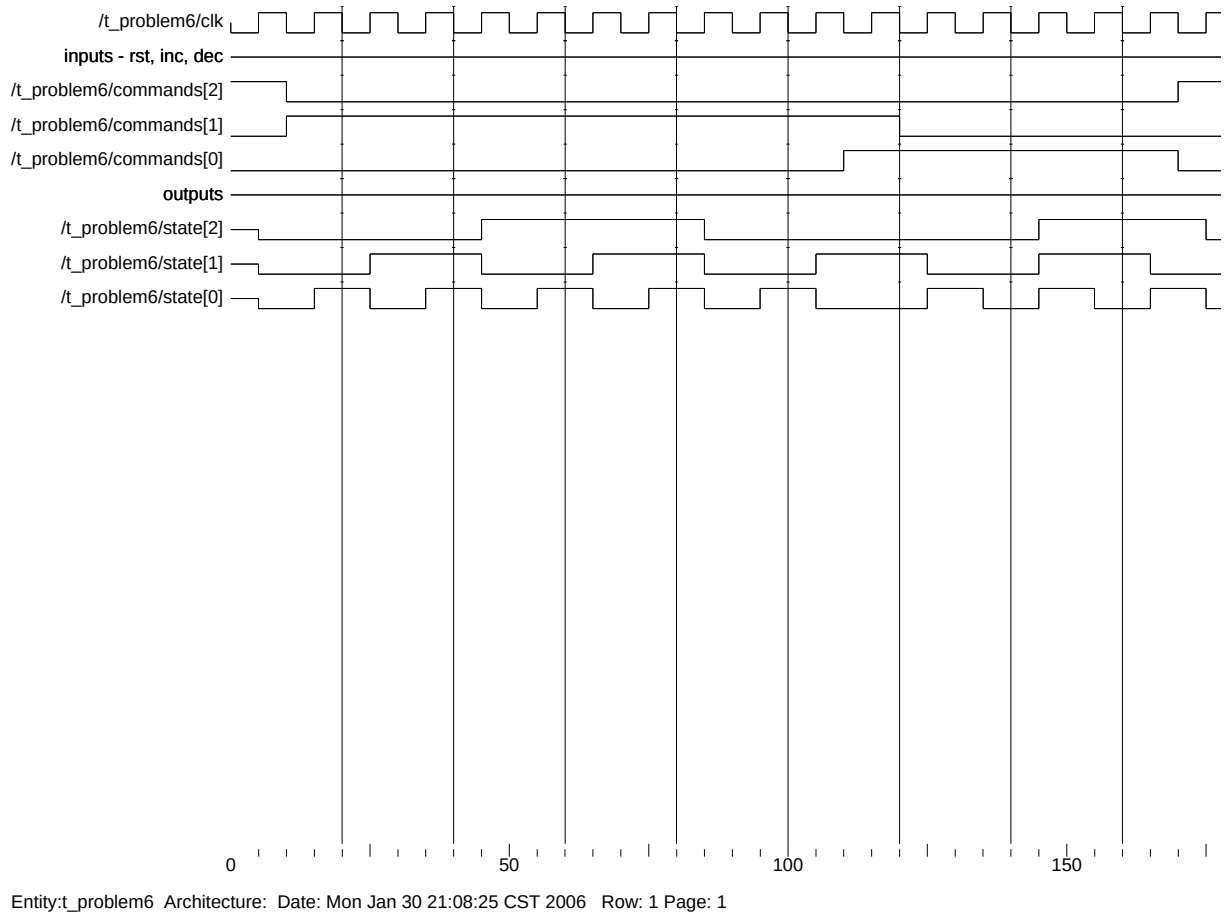

Figure 5: Problem 6 Waveform