Name: KEY **Id#**

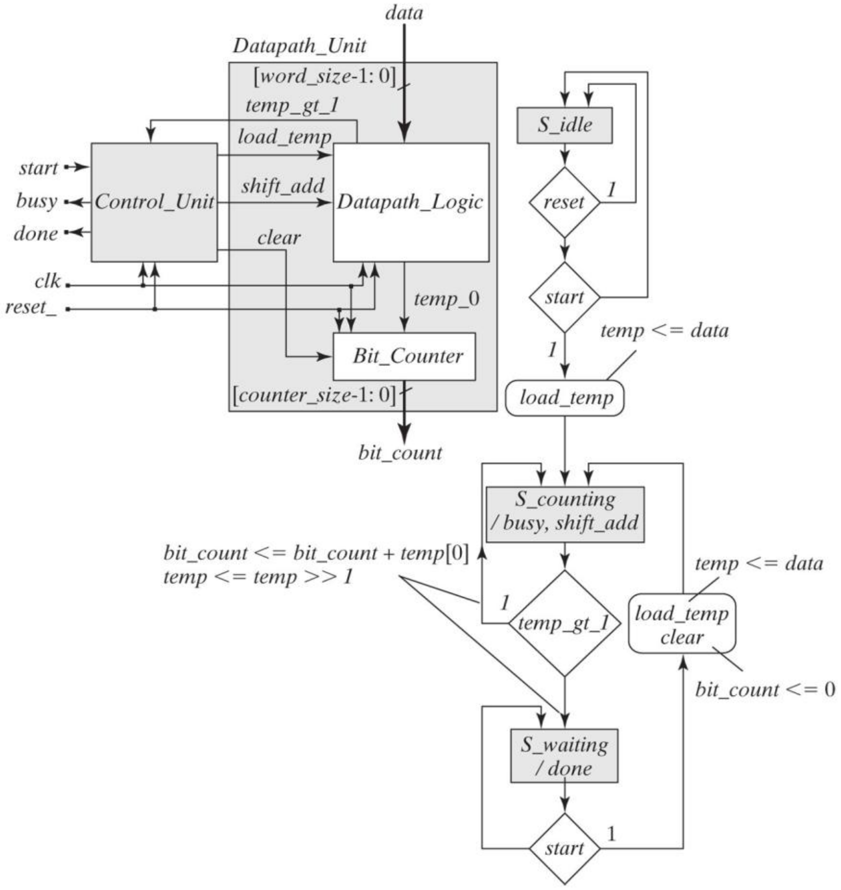
COE 405, Term 122

Design & Modeling of Digital Systems

Quiz# 3

Date: Monday, March 15, 2013

Q.1. The ASMD chart given below describes a state machine that counts 1's in a word and terminates activity as soon as possible. The machine remains in its reset state, *S_idle*, until an external agent asserts *start*. This action asserts the output, *load_temp*, which will cause *data* to be loaded into register *temp* when the state makes a transition to *S_counting* at the next active edge of *clk*. The machine remains in *S_counting* while *temp* contains a 1. Two actions occur concurrently at each subsequent clock: (1) *temp* is shifted towards its LSB and (2) *temp[0]* is added to *bit_count*. When *temp* finally has a 1 in only the LSB, the machine's state moves to *S_waiting*, where *done* is asserted. The state remains in *S_waiting* until *start* is reasserted. Assume that when the synchrnous *reset* input is asserted the machine is reset to the state *S_idle* and *bit_count* and *temp* are initialized to 0.



(i) Write a Verilog model to model the data-path.

```
module OnesCount_DPU #(parameter word_size=4, counter_size=3)(
output [counter_size-1: 0] bit_count,
output temp_gt_1,
input [word_size-1:0] data,
input load_temp, shift_add, clear, clock, reset
);

reg [word_size-1:0] temp;
reg [counter_size-1:0] counter;

assign bit_count = counter;

assign temp_gt_1 = | temp[word_size-1:1];

always @ (posedge clock)
  if (load_temp) temp <= data;
  else if (shift_add) temp <= temp >> 1;

always @ (posedge clock)
  if (reset || clear) counter <= 0;
  else if (shift_add) counter <= counter + temp[0];

endmodule
```

(ii) Write a Verilog model to model the control unit based on the ASMD chart i.e. not based on equations.

```
module OnesCount_CU  (output reg load_temp,
shift_add, clear, done, busy,
input start, temp_gt_1, clock, reset);

parameter S_idle = 2'b00, S_counting = 2'b01, S_waiting = 2'b10;

reg [1:0] state, next_state;

  always @(posedge clock)
   if (reset) state <= S_idle;
   else state <= next_state;

  always @(state, start, temp_gt_1) begin
   load_temp=0; shift_add=0; clear=0; done=0; busy=0;
   case (state)
      S_idle:
                  if (start) begin
                    load_temp = 1;
```

```
          next_state = S_counting;
           end
           else next_state = S_idle;
   S_counting: begin
               shift_add = 1; busy = 1;
               if (temp_gt_1) next_state = S_counting;
               else   next_state = S_waiting;
           end
   S_waiting:  begin
              done = 1;
              if (start) begin
                      load_temp = 1;  clear = 1;
                      next_state = S_counting;
              end
              else next_state = S_waiting;
             end
   default: begin
            next_state = 2'bxx;
            load_temp='bx; shift_add='bx; clear='bx; done='bx; busy='bx;
          end
  endcase
 end
endmodule
```

(iii) Write a Verilog model to model the whole design.

```
module OnesCount #(parameter word_size=4, counter_size=3)(
output [counter_size-1: 0] bit_count,
output done, busy,
input  [word_size-1:0] data,
input  start, clock, reset
);

OnesCount_CU  M1 (load_temp,shift_add, clear, done, busy, start, temp_gt_1, clock, reset);
OnesCount_DPU M2 (bit_count, temp_gt_1, data, load_temp, shift_add, clear, clock, reset);

endmodule
```