```
/*
   Ben Davis
   2/15/24
   EE 371
   Lab 4, Task 2

   This module controls the data_bin file, and the binary search
   of the ram. It has a clock input, a reset input, a start input
   to begin the search, and a finish input so it can adjust the
   datapath to halt the search process. It has three outputs, clear
   for resetting the datapath, comp to execute a cycle of the search
   process, and hold to stall the process (either to synch with the
   ram or after it has completed).

*/

module fsm_bin (
            input logic clk, //clock
            input logic reset, //reset
            input logic start, //start (user input)
            input logic finish, //when the search is finished
            output logic clear, //to clear the datapath
            output logic comp, //execute the search
            output logic hold); //stalls the search


   enum {off, load, run, done} ps, ns;

   //if there is a rest, go to state off
   //otherwise proceed to next state
   always_ff @(negedge clk) begin
      if(reset) begin
         ps <= off;
      end else begin
         ps <= ns;
      end
   end


   always_comb begin
      case(ps)

         off:  if(start) ns <= load; //initial state
               else ns <= off;

         load: ns <= run; //holds, then proceeds to search state

         run:  if(finish) ns <= done; //executes search
               else ns <= load; //proceeds to stop search if data is
                                //found or exhausted
         done: if(start) ns <= done; //holds until reset
               else ns <= off;

      endcase
   end

   //outputs to control datapath based off present state
   assign clear = (ps==off);
   assign comp = (ps==run);
   assign hold = (ps==load);
endmodule
//testbench
module fsm_bin_tb ();

   //recall variables
   logic clk, reset, start, finish, clear, comp, hold;

   fsm_bin dut (.clk, .reset, .start, .finish, .clear, .comp, .hold);

   //clock setup
   parameter clk_pd = 100;
   initial begin
      clk <= 0;
```

```
74          forever #(clk_pd /2) clk <= ~clk;
75       end //of clock setup
76
77       //tests an instance where it takes five cycles to search the ram
78       initial begin
79          reset <= 1; start <= 0; finish <= 0; @(posedge clk);
80          reset <= 0; start <= 1; finish <= 0; @(posedge clk);
81          reset <= 0; start <= 1; finish <= 0; @(posedge clk);
82          reset <= 0; start <= 1; finish <= 0; @(posedge clk);
83          reset <= 0; start <= 1; finish <= 0; @(posedge clk);
84          reset <= 0; start <= 1; finish <= 0; @(posedge clk);
85          reset <= 0; start <= 1; finish <= 1; @(posedge clk);
86          $stop;
87       end
88    endmodule //test end
```