```
1   // Manages timing for updating GoL state (delay between GoL iterations)
2
3   module controlUnit (clk, reset, start, enable_update);
4      input logic clk, reset;
5      input logic start;
6      output logic enable_update;
7
8      logic [31:0] counter;
9      parameter UPDATE_INTERVAL = 250000;
10
11     always_ff @(posedge clk or posedge reset) begin
12        if (reset) begin
13           counter <= 0;
14           enable_update <= 0;
15        end else if (start) begin
16           if (counter == UPDATE_INTERVAL) begin
17              counter <= 0;
18              enable_update <= 1;
19           end else begin
20              counter <= counter + 1;
21              enable_update <= 0;
22           end
23        end else begin
24           enable_update <= 0;
25        end
26     end
27  endmodule
28
29
30
31  module controlUnit_testbench();
32      logic clk, reset, start;
33      logic enable_update;
34
35      controlUnit dut (clk, reset, start, enable_update);
36
37      // Set up a simulated clock.
38      parameter CLOCK_PERIOD = 100;
39      initial begin
40          clk <= 0;
41          forever #(CLOCK_PERIOD/2) clk <= ~clk; // Forever toggle the clock
42      end
43
44      // Test the design.
45      initial begin
46          // Reset the design
47          reset <= 1; start <= 0; @(posedge clk); // reset every time we start
48          @(posedge clk);
49          reset <= 0; @(posedge clk); @(posedge clk);
50
51          // Test case 1: No start signal
52          start <= 0; @(posedge clk); @(posedge clk);
53
54          // Test case 2: Start signal, enable_update should be asserted after UPDATE_INTERVAL
55          start <= 1;
56          repeat (250001) @(posedge clk);
57
58          // Test case 3: Keep the start signal, observe enable_update behavior
59          repeat (500000) @(posedge clk);
60
61          // Test case 4: Remove start signal, enable_update should not assert
62          start <= 0; @(posedge clk); @(posedge clk);
63
64          // Test case 5: Reapply reset and start again
65          reset <= 1; @(posedge clk); reset <= 0; @(posedge clk);
66          start <= 1;
67          repeat (250001) @(posedge clk);
68
69          $stop;
70      end
71  endmodule
72
```