```
1   // divided_clocks[0] = 25MHz, [1] = 12.5Mhz, ... [23] = 3Hz, [24] = 1.5Hz, [25] = 0.75Hz, ..
2   module clock_divider (clock, reset, divided_clocks);
3      input logic reset, clock;
4      output logic [31:0] divided_clocks = 0;
5
6      always_ff @(posedge clock) begin
7         divided_clocks <= divided_clocks + 1;
8      end
9
10  endmodule
11
12
13  module clock_divider_testbench();
14     logic clk;
15     logic reset;
16     logic [31:0] divided_clocks;
17
18     clock_divider dut (.clock(clk), .reset, .divided_clocks);
19
20     // Set up a simulated clock.
21     parameter CLOCK_PERIOD=100;
22     initial begin
23        clk <= 0;
24        forever #(CLOCK_PERIOD/2) clk <= ~clk; // Forever toggle the clock
25     end
26
27     // Test the design.
28     initial begin
29          reset <= 1;                                   @(posedge clk);   // reset in beginning
30                                                        @(posedge clk);
31          reset <= 0;                                   @(posedge clk);
32                                                        @(posedge clk);
33                                                        @(posedge clk);
34                                                        @(posedge clk);
35                                                        @(posedge clk);   // observe the
    pseudo-random Q vals
36                                                        @(posedge clk);
37                                                        @(posedge clk);
38                                                        @(posedge clk);
39                                                        @(posedge clk);
40                                                        @(posedge clk);
41                                                        @(posedge clk);
42                                                        @(posedge clk);
43                                                        @(posedge clk);
44                                                        @(posedge clk);
45                                                        @(posedge clk);
46                                                        @(posedge clk);
47                                                        @(posedge clk);
48                                                        @(posedge clk);
49                                                        @(posedge clk);
50                                                        @(posedge clk);
51                                                        @(posedge clk);
52                                                        @(posedge clk);
53          $stop;
54      end
55  endmodule
```