~\MobaXterm\home\QuartusProjects\Homework\hw8\hw8_serial_parser\serial.vhd

```
1  -- Name: Ian Flury
2  -- Assignment: HW8, P1 programmable square wave generator.
3  -- Class: CPEN430 Digital System Design
4  -- Date: 10/18/2023
5
6  library ieee;
7  use ieee.std_logic_1164.all;
8  use ieee.std_logic_unsigned.all;
9  use ieee.numeric_std.all;
10 use work.all;
11
12
13 entity serial is
14     port (
15         clock : in std_logic;
16         RST : in std_logic;
17         serial : in std_logic;
18         wave_out : out std_logic
19     );
20 end serial;
21
22 architecture rtl of serial is
23
24 type state_t is (zero, one, two, three);
25 signal state, next_state : state_t;
26
27
28 begin
29
30     the_machine: process(state, serial, RST)
31     begin
32         if RST = '1' then
33             wave_out <= '0';
34                 next_state <= zero;
35         else
36             case state is
37                 when zero =>
38                     if serial = '1' then
39                         next_state <= one;
40                     else
41                                 next_state <= zero;
42                       end if;
43                       wave_out <= '0';
44                 when one =>
45                     if serial = '1' then
46                         next_state <= two;
47                     else
48                         next_state <= zero;
49                     end if;
50                       wave_out <= '0';
51                 when two =>
52                     if serial = '1' then
```

```
53                          next_state <= three;
54                      else
55                          next_state <= zero;
56                      end if;
57                            wave_out <= '0';
58                  when three =>
59                      if serial = '1' then
60                          wave_out <= '1';
61                          next_state <= three;
62                      else
63                                  wave_out <= '0';
64                                  next_state <= zero;
65                            end if;
66                  when others =>
67              end case;
68          end if;
69      end process the_machine;
70
71      the_registers: process(clock, RST)
72      begin
73          if RST = '1' then
74              state <= zero;
75          elsif clock'EVENT and clock = '1' then
76              state <= next_state;
77          end if;
78      end process the_registers;
79
80  end rtl;
```