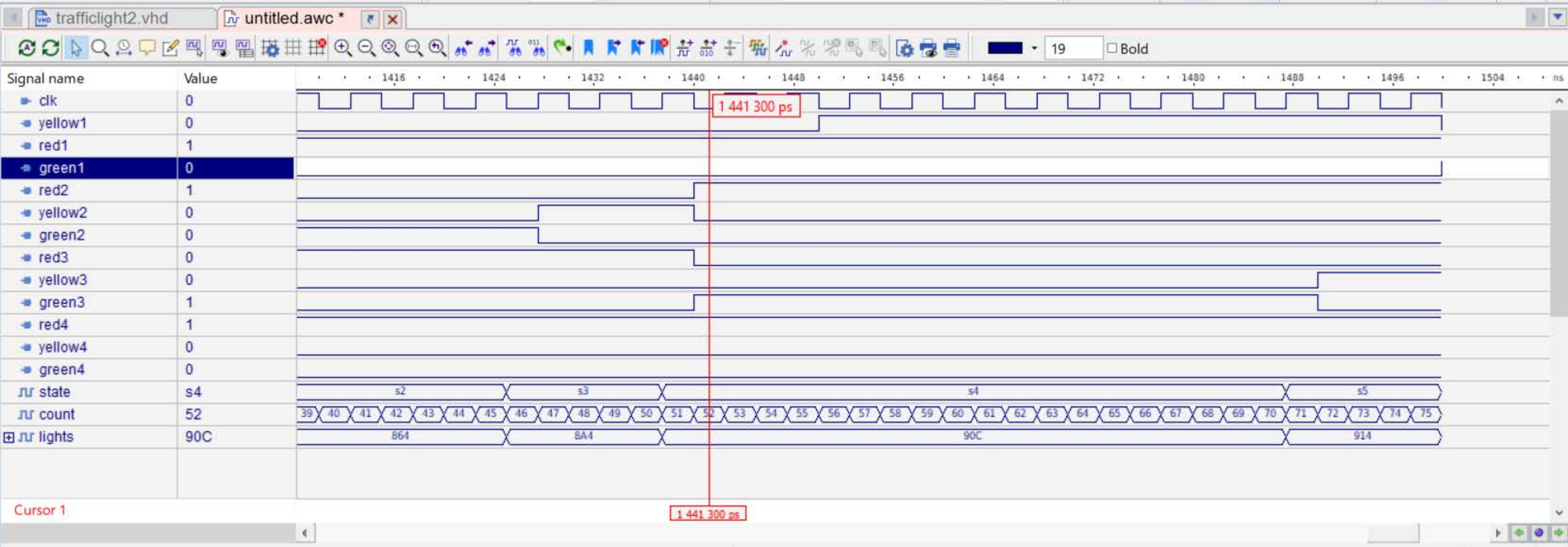
CBS-2
Implement controller for a Traffic light system

```
1   -- TRAFFIC LIGHTS CONTROLLER
2
3   library IEEE;
4   use IEEE.STD_LOGIC_1164.all;
5   use IEEE.STD_LOGIC_unsigned.all;
6
7   entity traffic is
8       port(
9           clk : in STD_LOGIC;
10          -- red1 represents red light of 1st traffic light and similarily
    other lights are represented
11          red1 : out STD_LOGIC;
12          yellow1 : out STD_LOGIC;
13          green1 : out STD_LOGIC;
14          red2 : out STD_LOGIC;
15          yellow2 : out STD_LOGIC;
16          green2 : out STD_LOGIC;
17          red3 : out STD_LOGIC;
18          yellow3 : out STD_LOGIC;
19          green3 : out STD_LOGIC;
20          red4 : out STD_LOGIC;
21          yellow4 : out STD_LOGIC;
22          green4 : out STD_LOGIC
23           );
24  end entity traffic;
25
26  architecture trafficA of traffic is
27  type state_type is (s0, s1, s2, s3, s4, s5,s6,s7);      -- defined state
    for each combination possible
28  signal state : state_type := s0;                        -- initial state is
    s0
29  signal count : integer := 0;                            -- represents time
30          signal lights: std_logic_vector(11 downto 0);   -- a vector that
    represents a state
31  begin
32      STATEpro : process(state)
33          begin
34              case state is
35                  when s0 => lights <= "001100100100";
36                  when s1 => lights <= "010100100100";
37                  when s2 => lights <= "100001100100";
38                  when s3 => lights <= "100010100100";
39                  when s4 => lights <= "100100001100";
40                  when s5 => lights <= "100100010100";
41                  when s6 => lights <= "100100100001";
42                  when s7 => lights <= "100100100010";
43                  when others => lights <= lights;
44              end case;
45          end process;
46
47      LT : process(clk)
48           begin
49
50          case count is
51              when 0 => state <= s0; count <= count + 1;
52              when 20 => state <= s1; count <= count + 1;         -- 1st
    green ends
```

```
53              when 25 => state <= s2; count <= count + 1;          -- 1st
    yellow ends
54              when 45 => state <= s3; count <= count + 1;          -- 2nd
    green ends
55              when 50 => state <= s4; count <= count + 1;          -- 2nd
    yellow ends
56              when 70 => state <= s5; count <= count + 1;          -- 3rd
    green ends
57              when 75 => state <= s6; count <= count + 1;          -- 3rd
    yellow ends
58              when 95 => state <= s7; count <= count + 1;          -- 4th
    green ends
59              when 100 => count <= 0;                              -- 4th
    yellow ends
60              when others => count <= count + 1;
61          end case;
62
63          -- Each bit of state vector represents state of each light i.e. 0
    or 1
64
65          green4 <= lights(0);
66          yellow4 <= lights(1);
67          red4 <= lights(2);
68          green3 <= lights(3);
69          yellow3 <= lights(4);
70          red3 <= lights(5);
71          green2 <= lights(6);
72          yellow2 <= lights(7);
73          red2 <= lights(8);
74          green1 <= lights(9);
75          yellow1 <= lights(10);
76          red1 <= lights(11);
77           end process;
78          -- green4 is represented by last bit i.e. 0th bit ( LSB ) and red1
    is represented by first bit i.e. 11th bit ( MSB )
79
80  end architecture trafficA;
81
```


trafficlight2.vhd
untitled.awc *
19
Bold
Signal name
Value
1416
1424
1432
1440
1448
1456
1464
1472
1480
1488
1496
1504
ns
clk 0
yellow1 0
red1 1
green1 0
red2 1
yellow2 0
green2 0
red3 0
yellow3 0
green3 1
red4 1
yellow4 0
green4 0
state s4
count 52
lights 90C
1 441 300 ps
s2
s3
s4
s5
39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75
864
8A4
90C
914
Cursor 1
1 441 300 ps