Link to Github

Lab 08

1. Preparation Tasks

Completed state table

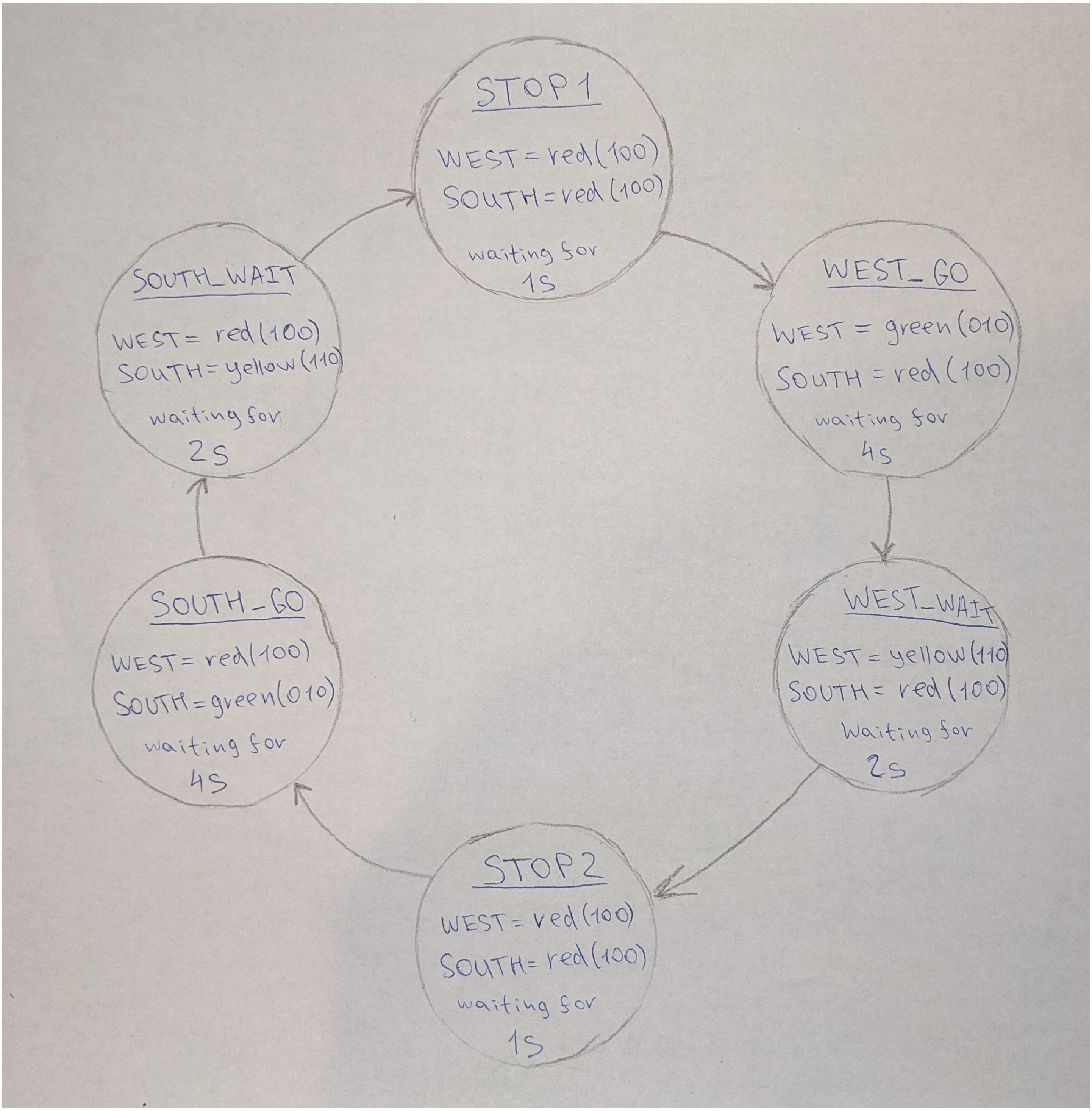
Input P	0	0	1	1	0	1	0	1	1	1	1	0	0	1	1	1
Clock	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑	↑
State	A	A	B	C	C	D	A	B	C	D	B	B	B	C	D	B
Output R	`0`	`0`	`0`	`0`	`0`	`1`	`0`	`0`	`0`	`1`	`0`	`0`	`0`	`0`	`1`	`0`

Completed table with color settings

RGB LED	Artix-7 pin names	Red	Yellow	Green
LD16	N15, M16, R12	`1,0,0`	`1,1,0`	`0,1,0`
LD17	N16, R11, G14	`1,0,0`	`1,1,0`	`0,1,0`

2. Traffic light controller

State diagram



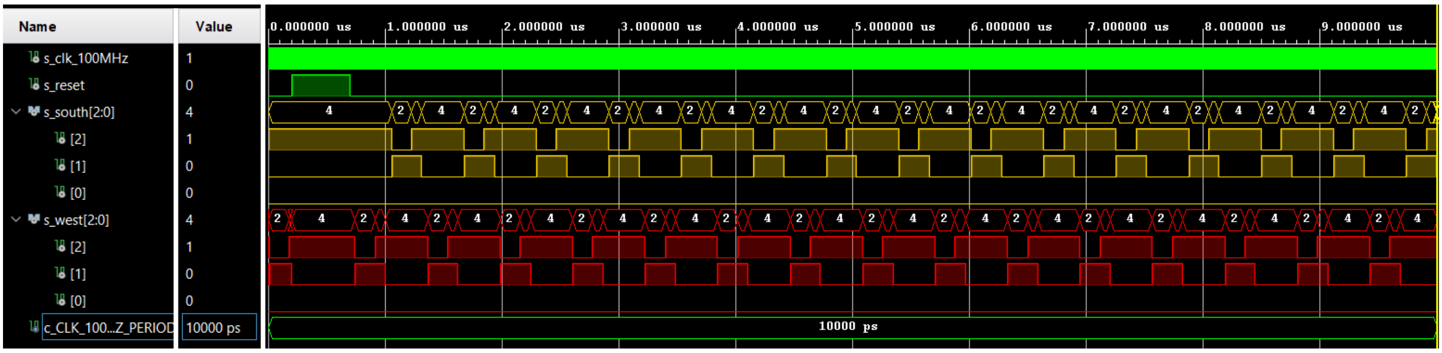
Listing of VHDL code of sequential process `p_traffic_fsm`

```
p_traffic_fsm : process(clk)
begin
    if rising_edge(clk) then
        if (reset = '1') then
            s_state <= STOP1;
            s_cnt   <= c_ZERO;
        elsif (s_en = '1') then
            case s_state is
                when STOP1 =>
                    if (s_cnt < c_DELAY_1SEC) then
                        s_cnt <= s_cnt + 1;
                    else
                        s_state <= WEST_GO;
                        s_cnt   <= c_ZERO;
                    end if;

                when WEST_GO =>
                    if (s_cnt < c_DELAY_4SEC) then
                        s_cnt <= s_cnt + 1;
                    else
                        s_state <= WEST_WAIT;
                        s_cnt   <= c_ZERO;
                    end if;

                when WEST_WAIT =>
                    if (s_cnt < c_DELAY_2SEC) then
                        s_cnt <= s_cnt + 1;
                    else
                        s_state <= STOP2;
                        s_cnt   <= c_ZERO;
                    end if;

                when STOP2 =>
                    if (s_cnt < c_DELAY_1SEC) then
                        s_cnt <= s_cnt + 1;
                    else
                        s_state <= SOUTH_GO;
                        s_cnt   <= c_ZERO;
                    end if;

                when SOUTH_GO =>
                    if (s_cnt < c_DELAY_4SEC) then
                        s_cnt <= s_cnt + 1;
                    else
                        s_state <= SOUTH_WAIT;
                        s_cnt   <= c_ZERO;
                    end if;

                when SOUTH_WAIT =>
                    if (s_cnt < c_DELAY_2SEC) then
                        s_cnt <= s_cnt + 1;
                    else
                        s_state <= STOP1;
                        s_cnt   <= c_ZERO;
                    end if;

                when others =>
                    s_state <= STOP1;

            end case;
        end if;
    end if;
end process p_traffic_fsm;
```

Listing of VHDL code of combinatorial process p_output_fsm

```
p_output_fsm : process(s_state)
begin
    case s_state is
        when STOP1 =>
            south_o <= "100";   -- Red
            west_o  <= "100";   -- Red

        when WEST_GO =>
            south_o <= "100";   -- Red
            west_o  <= "010";   -- Green
        when WEST_WAIT =>
            south_o <= "100";    -- Red
            west_o  <= "110";    -- Yellow

        when STOP2 =>
            south_o <= "100";    -- Red
            west_o  <= "100";    -- Red

        when SOUTH_GO =>
            south_o <= "010";    -- Green
            west_o  <= "100";    -- Red

        when SOUTH_WAIT =>
            south_o <= "110";    -- Yellow
            west_o  <= "100";    -- Red

        when others =>
            south_o <= "100";    -- Red
            west_o  <= "100";    -- Red
    end case;
end process p_output_fsm;
```

Screenshot(s) of the simulation, from which it is clear that controller works correctly



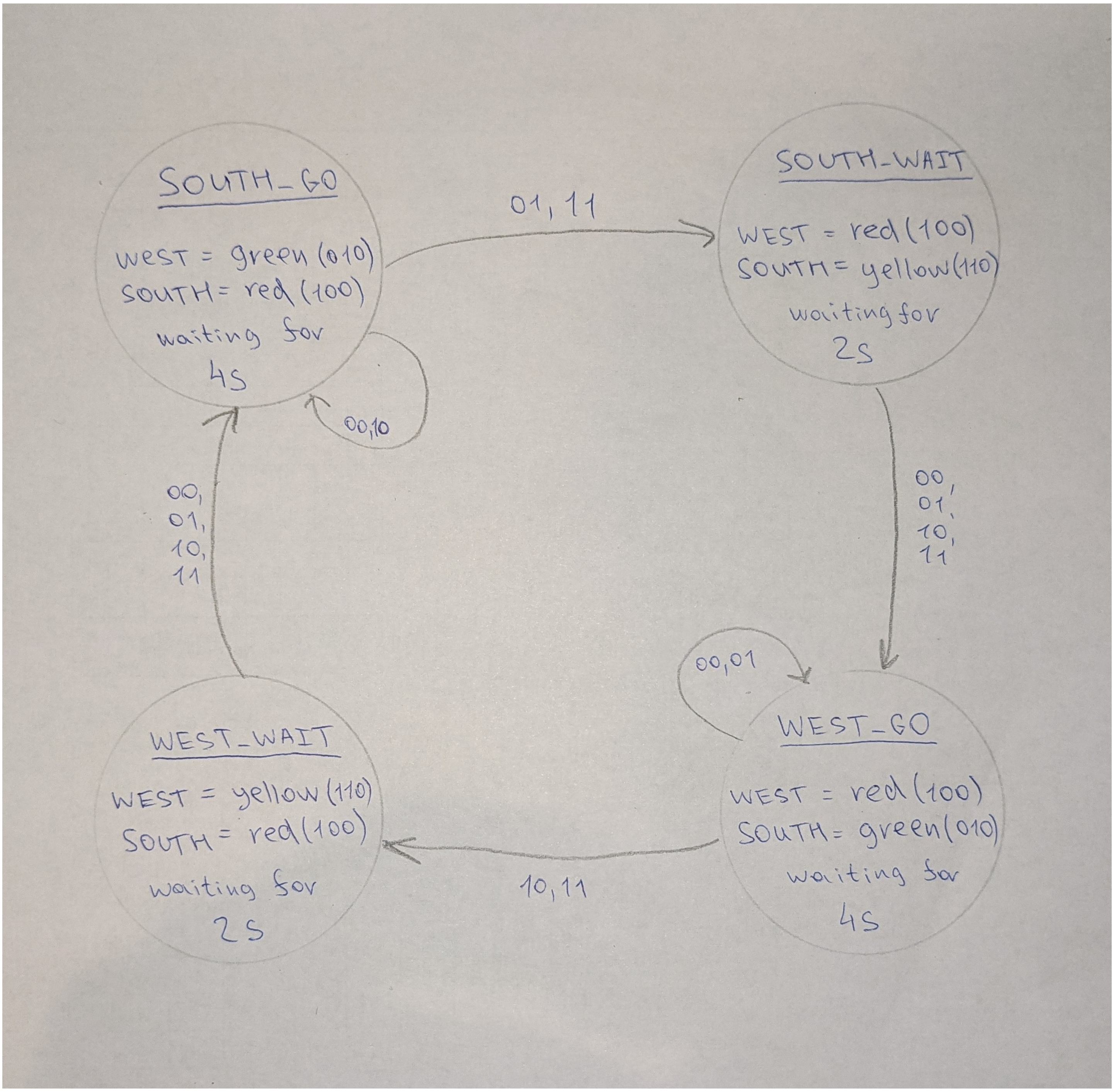
Name	Value
s_clk_100MHz	1
s_reset	0
s_south[2:0]	4
[2]	1
[1]	0
[0]	0
s_west[2:0]	4
[2]	1
[1]	0
[0]	0
c_CLK_100...Z_PERIOD	10000 ps

3. Smart controller

State table

Current state	Controllers lights (SW)	Controllers values (SW)	No cars	West	South	Both
SOUTH_GO	GREEN, RED	010, 100	SOUTH_GO	SOUTH_WAIT	SOUTH_GO	SOUTH_WAIT
SOUTH_WAIT	YELLOW, RED	110, 100	SOUTH_WAIT	SOUTH_WAIT	SOUTH_WAIT	SOUTH_WAIT
WEST_GO	RED, GREEN	100, 010	WEST_GO	WEST_GO	WEST_WAIT	WEST_WAIT
WEST_WAIT	RED, YELLOW	100, 110	WEST_WAIT	WEST_WAIT	WEST_WAIT	WEST_WAIT

State diagram



Listing of VHDL code of sequential process `p_smart_traffic_fsm`

```
p_smart_traffic_fsm : process(clk)
begin
    if rising_edge(clk) then
        if (reset = '1') then
            s_state <= WEST_GO ;
            s_cnt   <= c_ZERO;

        elsif (s_en = '1') then
            case s_state is
                when SOUTH_GO =>
                    -- sensors check
                    if (s_cnt < c_DELAY_2SEC and (sens_i = "00" or sens_i = "10")) then
                        s_cnt <= s_cnt + 1;
                    else
                        s_state <= SOUTH_WAIT;
                        s_cnt <= c_ZERO;
                    end if;

                when SOUTH_WAIT =>
                    if (s_cnt < c_DELAY_1SEC) then
                        s_cnt <= s_cnt + 1;
                    else
                        s_state <= WEST_GO;
                        s_cnt <= c_ZERO;
                    end if;

                when WEST_GO =>
                    -- sensors check
                    if (s_cnt < c_DELAY_2SEC and (sens_i = "00" or sens_i = "01")) then
                            s_cnt <= s_cnt + 1;
                        end if;
                    else
                        s_state <= WEST_WAIT;
                        s_cnt <= c_ZERO;
                    end if;

                when WEST_WAIT =>
                    if (s_cnt < c_DELAY_1SEC) then
                        s_cnt <= s_cnt + 1;
                    else
                        s_state <= SOUTH_GO;
                        s_cnt <= c_ZERO;
                    end if;

                when others =>
                    s_state <= SOUTH_GO;
            end case;
        end if;
    end if;
end process p_smart_traffic_fsm;
```