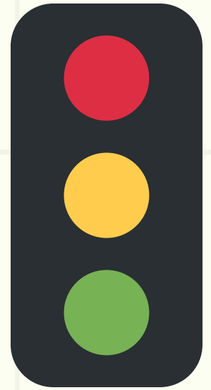
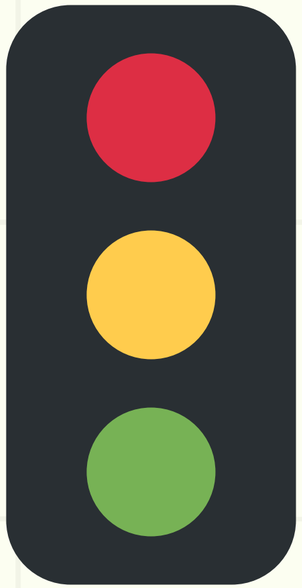
Traffic Light Controller

Mata Kuliah
Perancangan Sistem Digital

Anggota Kelompok PA21:

- Dhafin Hamizan S
- Mirza Adi R
- M Arya Wiandra Utomo
- Wiellona Darlene Oderia S

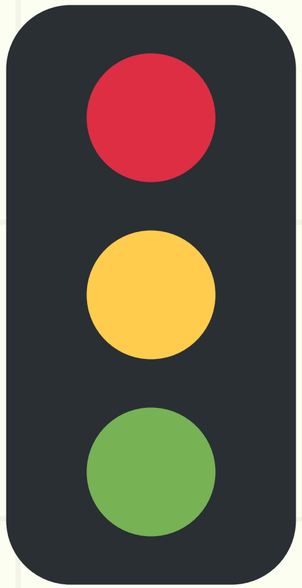




Why?

Dalam era transportasi modern yang ditandai oleh tingginya mobilitas kendaraan dan manusia, lampu merah diperlukan sebagai salah satu alat pengatur lalu lintas untuk memastikan keselamatan, efisiensi, dan ketertiban di jalan raya. Dengan volume kendaraan yang semakin meningkat, potensi kecelakaan dan kemacetan menjadi lebih tinggi jika tidak ada sistem regulasi yang jelas.

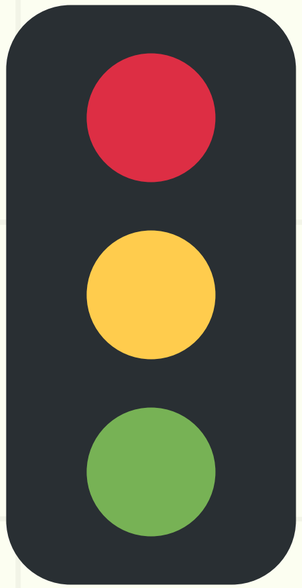
Lampu merah membantu mengatur giliran kendaraan dan pejalan kaki, mengurangi risiko tabrakan di persimpangan, serta memastikan arus lalu lintas berjalan dengan lancar.



Deskripsi Proyek

Sistem lampu lalu lintas yang efisien harus dapat mengintegrasikan semua lampu lalu lintas yang ada. Dalam proyek ini, akan ada 4 lampu lalu lintas (ditandakan dengan 4 mata angin) dan 4 lampu penyebrangan orang.. Proyek ini dirancang untuk membangun sistem lampu lalu lintas terintegrasi berbasis VHDL yang mampu memeriksa masing-masing lampu lalu lintas agar dapat mewujudkan lampu lalu lintas yang ter-regulasi dengan baik untuk sebuah simpangan dengan 4 jalan.

Setiap traffic light dari 4 jalan (North, East, South, West) masing-masing memiliki 3 traffic light,
a. untuk lurus
b. untuk belok kanan
c. untuk penyebrang jalan (ketika mendapat input)



Tujuan

1. Membuat sebuah controller lalu lintas untuk persimpangan yang terdapat 4 jalan, untuk tiga tujuan (lurus, belok, dan penyebrang jalan) sebagai salah Proyek Akhir Praktikum Perancangan Sistem Digital.

2. Menerapkan modul-modul yang telah dipelajari pada mata kuliah Perancangan Sistem Digital

Algoritma

Kode di smaping merupakan proses utama untuk transisi state dan perubahan arah, yaitu fitur utama dari lampu merah perempatan yang telah dibuat. State-state yang ada antara lain Idle, Red, Yellow, dan Green, atau warna-warna yang ada pada sebuah lampu merah, dan perubahan arah yaitu antara 4 lampu merah pada perempatan.

```
begin
    -- Proses utama untuk transisi state dan perubahan arah
    process(clk, reset)
    begin
        if reset = '1' then
            current_state <= IDLE_STATE;
            previous_state <= IDLE_STATE;
            counter <= 0;
            light_index <= "00";
        elsif rising_edge(clk) then
            if toggle = '1' then
                if counter = 1 then
                    counter <= 0;
                    previous_state <= current_state; -- menyimpan state sebelumnya
                    current_state <= next_state;

                    -- Pergantian arah setelah siklus penuh
                    if current_state = GREEN_STATE and next_state = RED_STATE then
                        light_index <= std_logic_vector(unsigned(light_index) + 1);
                    end if;
                else
                    counter <= counter + 1;
                end if;
            else
                current_state <= IDLE_STATE;
                previous_state <= IDLE_STATE;
                counter <= 0;
            end if;
        end if;
    end process;

    -- Proses untuk transisi state
    process(previous_state, current_state, Pedestrian_button_north, Pedestrian_button_south, Pedestrian_button_west, Pedestrian_button_east, Timer)
    begin
        case current_state is
            when IDLE_STATE =>
                if toggle = '1' then
                    next_state <= RED_STATE;
                else
                    next_state <= IDLE_STATE;
                end if;
```

Algoritma

Kode di smaping merupakan kondisi dimana seorang pedestrian memberikan input melalui tombol penyebrangan, dan membuat lampu untuk menyebrang menjadi state hijau, dan lampu bagi kendaraan menjadi state merah agar pedestrian dapat menyebrang.

```
            -- Jika tombol pedestrian ditekan, masuk ke state pedestrian
            if Pedestrian_button_north = '1' then
                next_state <= PEDESTRIAN_STATE;
            elsif Pedestrian_button_south = '1' then
                next_state <= PEDESTRIAN_STATE;
            elsif Pedestrian_button_west = '1' then
                next_state <= PEDESTRIAN_STATE;
            elsif Pedestrian_button_east = '1' then
                next_state <= PEDESTRIAN_STATE;
            else
                next_state <= YELLOW_STATE;
            end if;

        when YELLOW_STATE =>
            if previous_state = RED_STATE then
                next_state <= GREEN_STATE;
            else
                next_state <= RED_STATE;
            end if;

        when GREEN_STATE =>
            next_state <= YELLOW_STATE;

        when PEDESTRIAN_STATE =>
            -- Timer untuk pedestrian hanya diatur di sini
            pedestrian_timer <= to_integer(unsigned(Timer)); -- Mengambil nilai dari input Timer
            if pedestrian_timer = 0 then
                next_state <= RED_STATE; -- Kembali ke RED state setelah pedestrian selesai
            end if;

        when others =>
            next_state <= IDLE_STATE;
    end case;
end process;
```

Signal yang Digunakan

Signal-signal yang terdapat pada kode di samping digunakan untuk menentukan durasi dari setiap state lampu, dan juga untuk timer pedestrian agar setiap lampu, baik untuk kendaraan ataupun pedestrian terbagi dengan baik

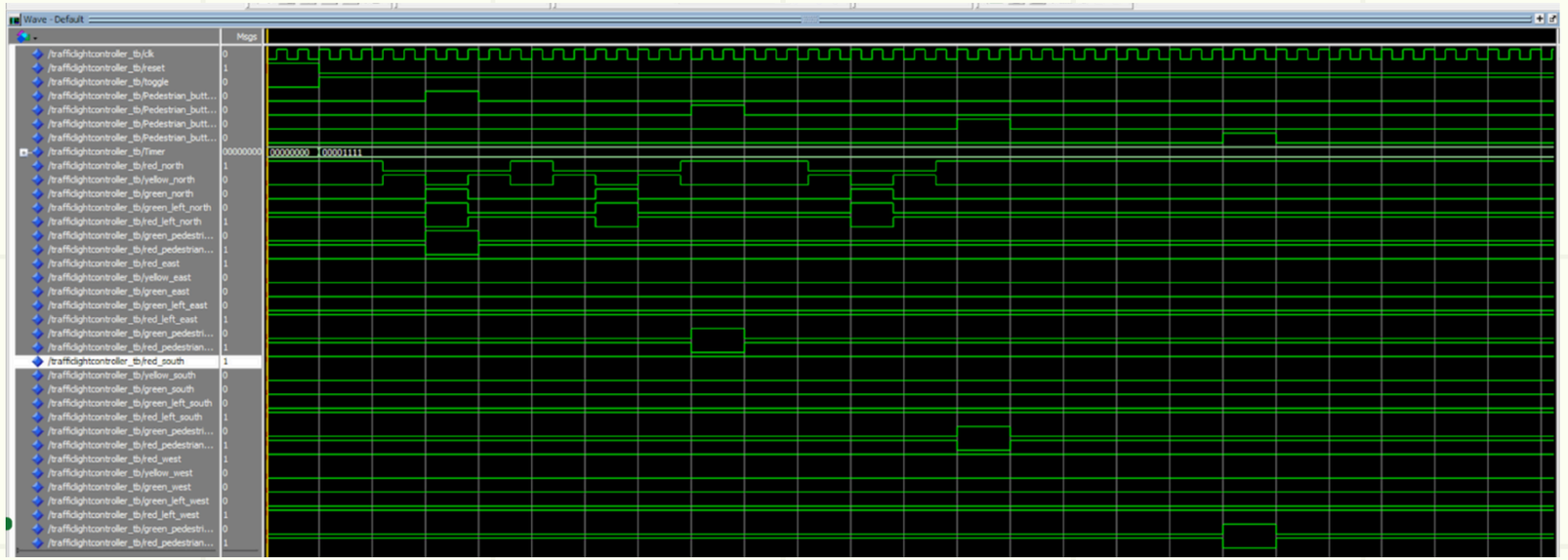
```
type State_Type is (IDLE_STATE, RED_STATE, YELLOW_STATE, GREEN_STATE, PEDESTRIAN_STATE);
signal previous_state, current_state, next_state: State_Type;

signal light_index : STD_LOGIC_VECTOR(1 downto 0) := "00";
signal counter : INTEGER range 0 to 1 := 0;

-- Timer for pedestrian green light
signal pedestrian_timer : INTEGER range 0 to 255 := 0; -- Max duration for pedestrian light (in clock cycles)
```

Cuplikan Kode Testbench

Pada cuplikan kode testbench berikut, "Clock Generation" bertugas untuk menghasilkan sinyal clock yang berganti antara '0' dan '1' setiap 10 ns, yang berlangsung terus-menerus. Sedangkan bagian "Stimulus Process" memberikan serangkaian perintah untuk menguji sistem. Proses ini dimulai dengan mereset sistem, lalu mengaktifkan beberapa tombol pejalan kaki secara bergantian untuk mensimulasikan interaksi pengguna. Setiap tombol ditekan selama beberapa waktu, dan setelah beberapa waktu, sistem dihentikan dengan mematikan sinyal toggle. Proses ini bertujuan untuk menguji bagaimana sistem bereaksi terhadap input yang diberikan.

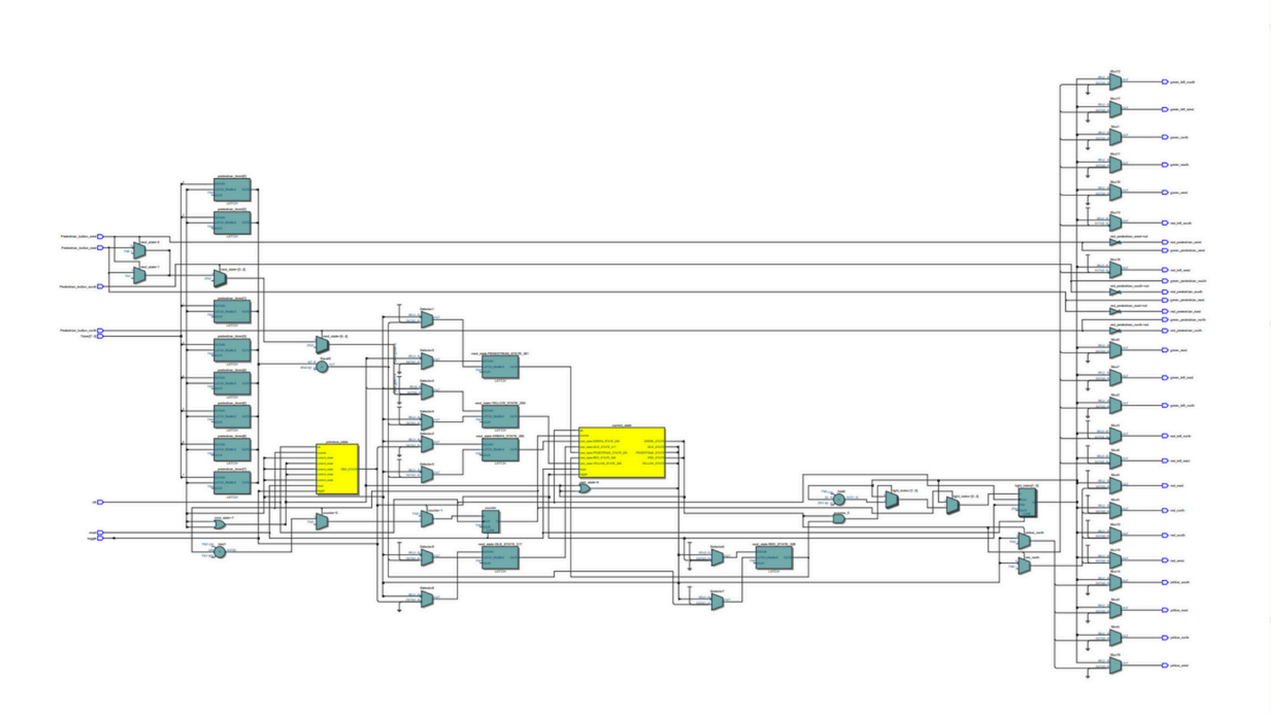
```
-- Clock Generation
clk_process: process
begin
    while True loop
        clk <= '0';
        wait for 10 ns;
        clk <= '1';
        wait for 10 ns;
    end loop;
end process;

-- Stimulus Process
stimulus: process
begin
    -- Reset the system
    reset <= '1';
    wait for 50 ns;
    reset <= '0';

    -- Enable toggle to start system
    toggle <= '1';
    Timer <= "00001111"; -- Timer set to 15 clock cycles

    -- Simulate pedestrian button press
    wait for 100 ns;
    Pedestrian_button_north <= '1';
    wait for 50 ns;
    Pedestrian_button_north <= '0';

    -- Simulate pedestrian button press for east
    wait for 200 ns;
    Pedestrian_button_east <= '1';
    wait for 50 ns;
    Pedestrian_button_east <= '0';

    -- Simulate pedestrian button press for south
    wait for 200 ns;
    Pedestrian_button_south <= '1';
    wait for 50 ns;
    Pedestrian_button_south <= '0';

    -- Simulate pedestrian button press for west
    wait for 200 ns;
    Pedestrian_button_west <= '1';
    wait for 50 ns;
    Pedestrian_button_west <= '0';

    -- Stop the toggle
    wait for 500 ns;
    toggle <= '0';
```

Testing



Simulasi wave pada ModelSIM digunakan untuk menguji fungsionalitas program testbench dan memverifikasi bahwa implementasi perangkat keras sesuai dengan spesifikasi desain. Pada jendela wave, ditampilkan bentuk gelombang digital dari berbagai sinyal yang terlibat, disinkronkan dengan clock, sehingga perubahan nilai sinyal dapat diamati dengan jelas seiring waktu simulasi

Result



Setelah dilakukan testing dengan wave test dan sintesis, hasil yang didapat sudah sesuai dengan hipotesis awal, dengan segala kondisi yang sudah ditentukan. Selanjutnya, dilakukan sintesis pada software Quartus dan dapat dilihat bahwa tidak terdapat error yang menunjukkan bahwa design yang telah dibuat sudah sesuai, dan juga dapat diimplementasikan pada FPGA.

Analisis

Proyek Traffic Light Controller ini dibuat dan diuji untuk membantu sebuah perempatan yang dinamai dengan mata angin. Traffic Light Controller yang telah dirancang telah melewati berbagai jenis ujian untuk memastikan fungsionalitas dari Traffic Light Controller tersebut.

Dari hasil simulasi wave pada modelsim menunjukkan sistem Traffic Light Controller mampu menjalankan alur lampu lalu lintas yang efisien. Setiap lampu lalu lintas berjalan sesuai dengan alur yang sudah dibuat yaitu yang pertama akan hijau adalah north kemudian dilanjutkan dengan east, south, dan west. Setiap pedestrian menekan tombol juga akan menjalankan lampu pedestrian.