Modul 4
Rangkaian Sekuensial

1. Tujuan

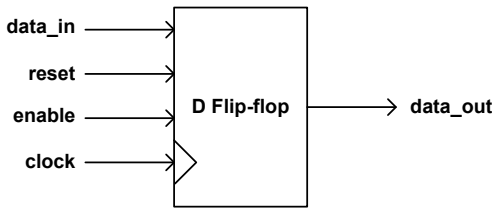
a. Merancang rangkaian sequential menggunakan Verilog.

2. Materi

Rangkaian sekuensial adalah rangkaian yang outputnya tidak hanya tergantung kepada keadaan saat ini, tetapi juga pada keadaan input sebelumnya. Pada bagian ini kita akan membuat rangkaian sekuensial D flip-flop (DFF), counter, dan finite state machine (FSM).

2.1. D Flip-flop

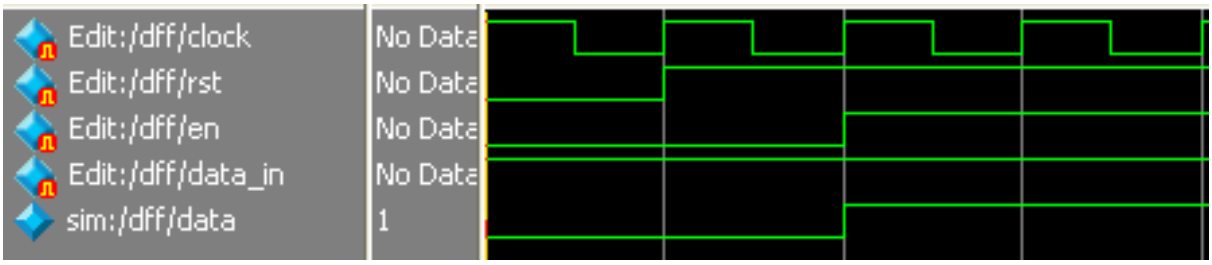
D Flip-flop merupakan elemen memory yang dapat menyimpan data input sebelumnya. D Flip akan menyampling data input apabila nilai clock berubah dari 0 ke 1 (*rising edge*). Dengan demikian, clock digunakan sebagai *sensivity list* dari rangkaian seperti yang terlihat pada gambar 1. Untuk membedakan kapan flip-flop menyampling input data, maka kita perlu menggunakan IF statement seperti pada contoh.



Gambar 1 D Flip Flop

```
module dff(clock,rst,en,data_in,data);

  input clock;
  input rst;
  input en;
  input data_in;

  output data;

  reg data;

  always@(posedge clock)
  begin
       if(!rst)
           data <= 1'b0;
       else if(en)
           data <= data_in;
       else
           data <= data;
  end

endmodule
```

Signal en adalah signal enable yang berfungsi untuk mengaktifkan flip-flop tersebut. Semua transisi hanya boleh terjadi apabila en = ’1’. Statement posedge clock adalah untuk membatasi eksekusi process terjadi pada saat transisi nilai dari clock. Posedge berarti eksekusi akan terjadi pada saat *rising edge*. Sebaliknya, apabila negedge clock, maka process akan di eksekusi pada *falling edge*.
Rangkaian di atas dapat disimulasikan dengan menggunakan rangkaian dff.v. Hasil simulasi dapat ditunjukan dengan timing diagram pada Gambar 2. Pada gambar tersebut terlihat data tersample pada saat rising edge dari clock.



Gambar 2 Timing Diagram DFF

Latihan :

- Simulasikan rangkain DFF
- Ubahlah rangkaian D flip-flop tersebut menjadi 8 bits D flip-flop (8 bits registers) dengan mengubah lebar data menjadi 8 bits dan simulasikan.

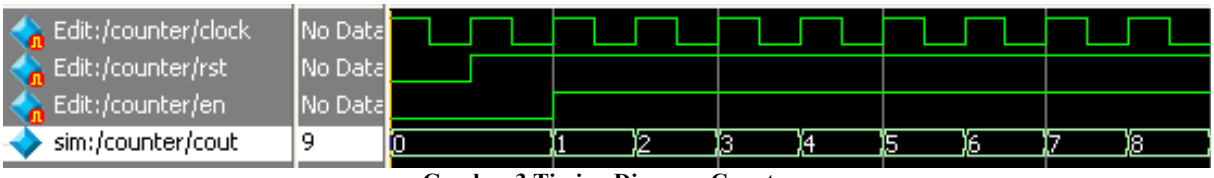
2.2. Counter

Berikut adalah deskripsi Verilog untuk rangkaian counter 4 bit.

```
module counter(clock,rst,en,cout);

   input clock;
   input rst;
   input en;

   output [3:0] cout;

   reg [3:0] cout;

   always@(posedge clock)
   begin
        if(!rst)
            cout <= 4'd0;
        else if(en)
            cout <= cout + 4'd1;
        else
            cout <= cout;
   end

endmodule
```

Seperti terlihat diatas, nilai register count dapat dinaikan dengan memberi harga ‘1’ pada signal en = 1. Nilai register count menjadi “0000” apabila di reset atau counter mencapai nilai maksimum “1111”.

Rangkaian diatas dapat disimulasikan dan hasilnya sebagai berikut



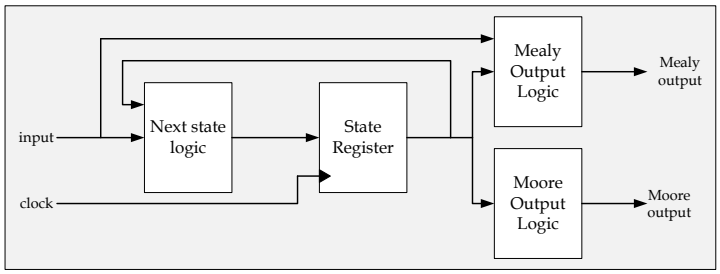
Gambar 3 Timing Diagram Counter

Latihan :

- Simulasikan rangkain Counter 4 bit
- Ubahlah rangkaian counter tersebut, sehingga ia menghitung mundur dari 14 ke 0.
- Ubahlah rangkaian counter tersebut menjadi counter 8 bit.

2.3. State Machine

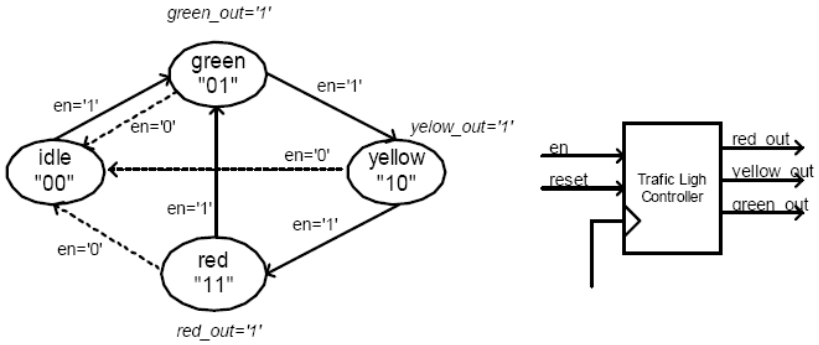
Desain FSM sangat penting dalam implementasi ke dalam FPGA. Ada banyak cara bagi kita untuk membuat FSM, tetapi kompiler FPGA memiliki keterbatasan dalam mensintesis sebuah FSM. Cara yang paling aman dalam mendesain FSM adalah dengan memisahkan bagian kombinasional dan sekuensial. Bagian kombinasional berfungsi untuk menghasilkan sinyal state selanjutnya (next state) dan nilai output. Sedangkan bagian sekuensial untuk menentukan sinyal state saat ini (perpindahan state).



Gambar 4 Desain State Machine

Terdapat dua jenis state machine, yaitu mealy machine dan moore machine. Pada mealy machine output state machine tergantung pada state saat ini dan nilai inputnya. Sedangkan pada moore machine output state machine hanya tergantung pada state saat ini.

Sebagai contoh state machine sederhana adalah state machine traffic light seperti terlihat pada Gambar 2.



Gambar 5 State Transition Traffic Light

Seperti terlihat pada state diagram, pada kondisi awal sistem di reset dan berada di state idle. Apabila en = ’1’, maka state akan berubah idle→green, green→yellow, yellow→red, red→green. Apabila en = ’0’ pada setiap kondisi maka state akan pindah ke idle. Dari state machine tersebut kita dapat mendesain deskripsi Verilog sebagai berikut:

- Menentukan jumlah state. Dari diagram kita dapat melihat bahwa sistem terdiri dari 4 state. Dengan demikian kita dapat merepresentasikan state tersebut dengan register 2 bit, dalam hal ini state. Dalam code Verilog :

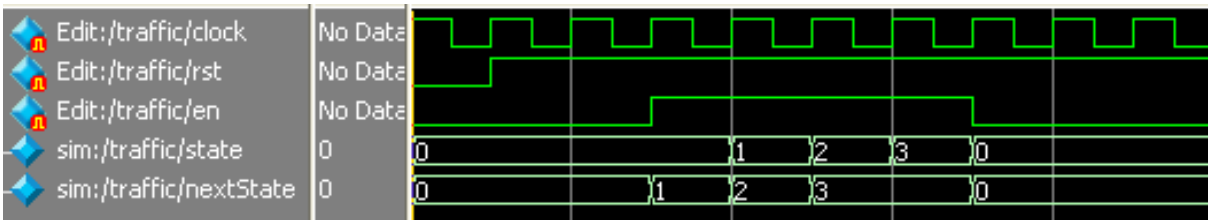
  reg [1:0] state;

- Menentukan Sensivity list. Dalam hal ini, perubahan state adalah terjadi pada saat rising edge daripada clock, sehingga sensivity list dapat ditentukan adalah clock.
- Syarat perubahan state. Dari state diagram dapat dilihat bahwa state berubah atas fungsi signal en.

Dari tahapan tersebut kita dapat membuat deskripsi Verilog state machine tersebut sebagai berikut:

```
module traffic(clock,rst,en,state);

   input clock;
   input rst;
   input en;
```

```
    output state;

    parameter IDLE   = 2'd0;
    parameter GREEN  = 2'd1;
    parameter YELLOW = 2'd2;
    parameter RED    = 2'd3;

    reg  [1:0] state;
    wire [1:0] nextState;

    //state transition
    always@(posedge clock)
    begin
       if(!rst)
           state <= IDLE;
       else if(!en)
           state <= IDLE;
       else
           state <= nextState;
    end

    //next state logic
    assign nextState = (state == IDLE)&en   ? GREEN:
                       (state == GREEN)&en  ? YELLOW:
                       (state == YELLOW)&en ? RED:
                                              state;

endmodule
```



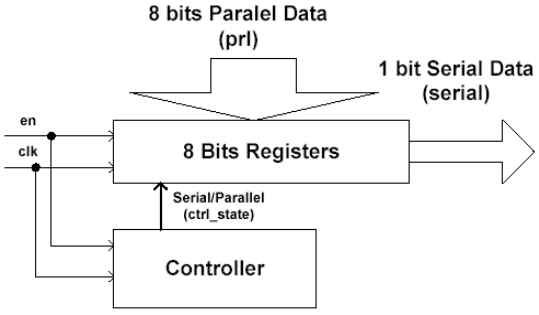
Gambar 6 Timing Diagram State Machine Traffic Light

Latihan :

- Simulasikan rangkain state machine traffic light.
- Ubahlah deskripsi Verilog traffic light controller di atas, sehingga di setiap state, sistem berhenti beberapa clock, (red=10 clock, yellow=5 clock, green=10 clock). Pertama-tama buat dahulu diagram state transition dari traffic light controller tersebut.
  (HINT: Gunakan counter).

2.4. 8 Bit Paralel to Serial Converter

Pada bagian ini akan dijelaskan tahapan perancangan 8 bits parallel to serial converter. Sistem ini berfungsi untuk mengubah data 8 bits yang datang setiap 8 clock. Data serial akan di keluarkan melalaui port serial setiap clock. Block diagram dari sistem ini dapat dilihat pada gambar berikut:



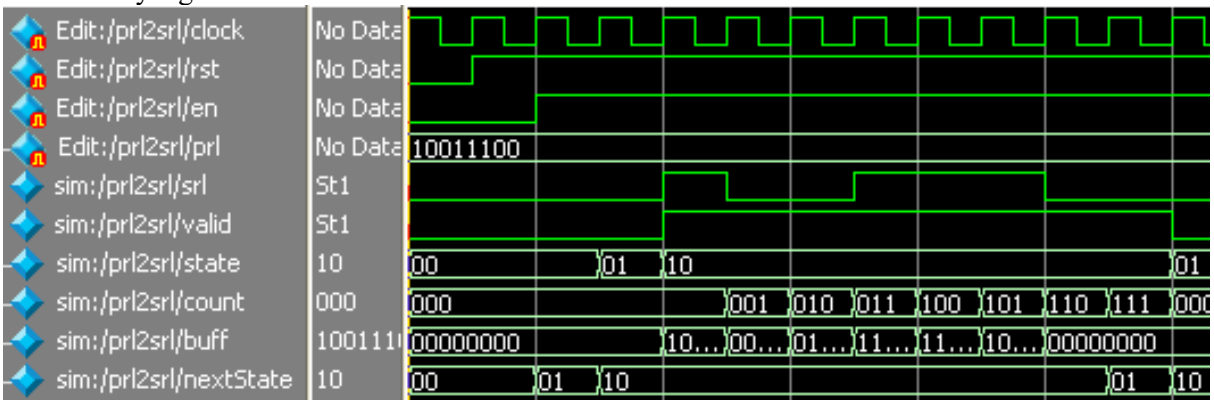
Gambar 7 Arsitektur Paralel to Serial Converter

Sistem ini dapat dibagi atas register dan controller. Register berfungsi menerima parallel 8 bits data pada saat mode register parallel, dan melakukan shift secara serial ketika register pada mode serial. Selain register, sistem memiliki controller, yang berfungsi menentukan kapan register menerima data parallel dan kapan men-shift data secara serial.

Berikut adalah deskripsi Verilog untuk rangkaian 8 bits parallel to serial converter.

```
module prl2srl(clock,rst,en,prl,srl,valid);

   input         clock;
   input         rst;
   input         en;
   input   [7:0] prl;

   output  srl;
   output  valid;

   parameter IDLE    = 2'b00;
   parameter PARALEL = 2'b01;
   parameter SERIAL  = 2'b10;

   reg  [1:0] state;
   reg  [2:0] count;
   reg  [7:0] buff;
   wire [1:0] nextState;

   //state machine
   always@(posedge clock)
   begin
        if(!rst)
            state <= IDLE;
        else
            state <= nextState;
   end

   assign nextState = (state == IDLE) & en                      ? PARALEL:
                      (state == PARALEL) & en                   ? SERIAL:
                      (state == SERIAL) & en & (count == 3'd7)  ? PARALEL:
                                                                  state;

   //counter
   always@(posedge clock)
   begin
        if(!rst)
            count <= 3'd0;
        else if(state == SERIAL)
            count <= count + 3'd1;
        else
            count <= count;
   end
   always@(posedge clock)
   begin
        if(!rst)
            buff <= 8'd0;
        else if(state == PARALEL)
            buff <= prl;
        else if(state == SERIAL)
            buff <= {buff[6:0],1'b0};
        else
            buff <= buff;
   end
   assign srl   = buff[7];
   assign valid = (state == SERIAL);
endmodule
```

Waveform yang dihasilkan adalah



Gambar 8 Timing Diagram Paralel to Serial Converter