IC设计-HW_2

2021211039 刘沁雨

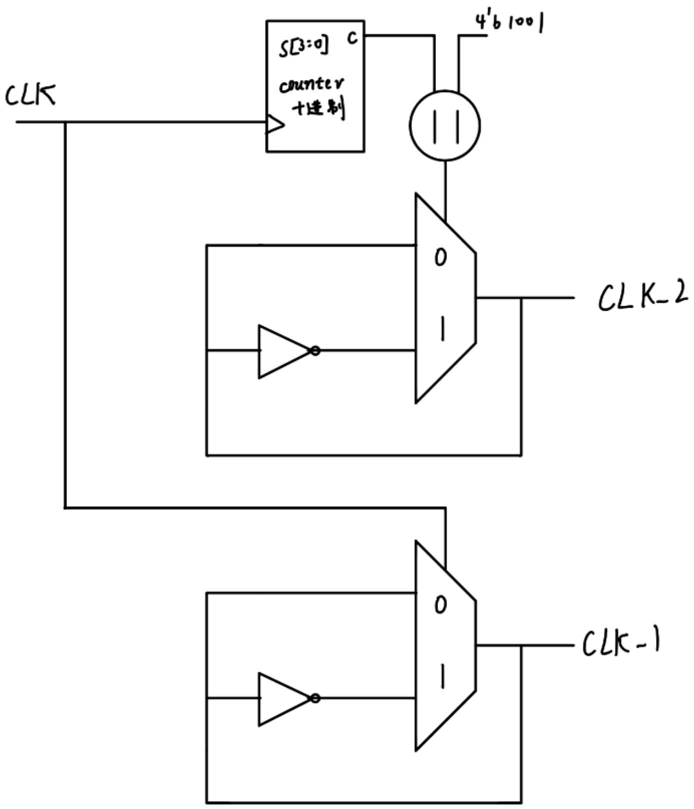
hw_2_1

源代码

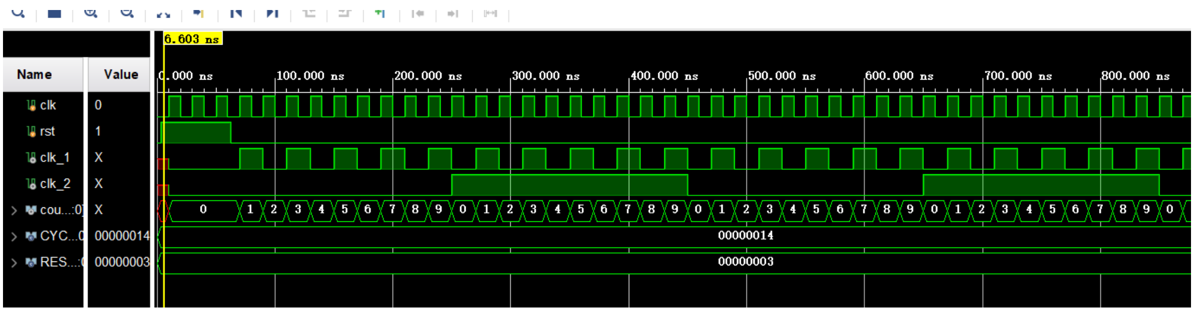
```verilog
module hw_2_1 (
    input CLK,RESET,
    output CLK_1,CLK_2
);
    reg clk_1,clk_2;
    reg [3:0]counter;

assign CLK_1 = clk_1;
assign CLK_2 = clk_2;

    //5M
    always@(posedge CLK)
        begin
            if(RESET)
            clk_1 <= 0;
            else
            clk_1 <= ~clk_1;
        end

    //500k
    always @(posedge CLK) begin
        if(RESET | counter == 9)
        counter <= 0;
        else
        counter <= counter +1;
    end

    always @(posedge CLK) begin
        if(RESET)
        clk_2 <= 0;
        else if(counter == 9)
        clk_2 <= ~clk_2;
        else
        clk_2 <= clk_2;
    end
endmodule
```

原理图

十进制计数器部分没有展开画，用RTL实现是用4个D触发器串联并通过外接门电路来实现计数功能，并用几个逻辑门把等于9时的进位信号引出，展开略为复杂。



仿真结果

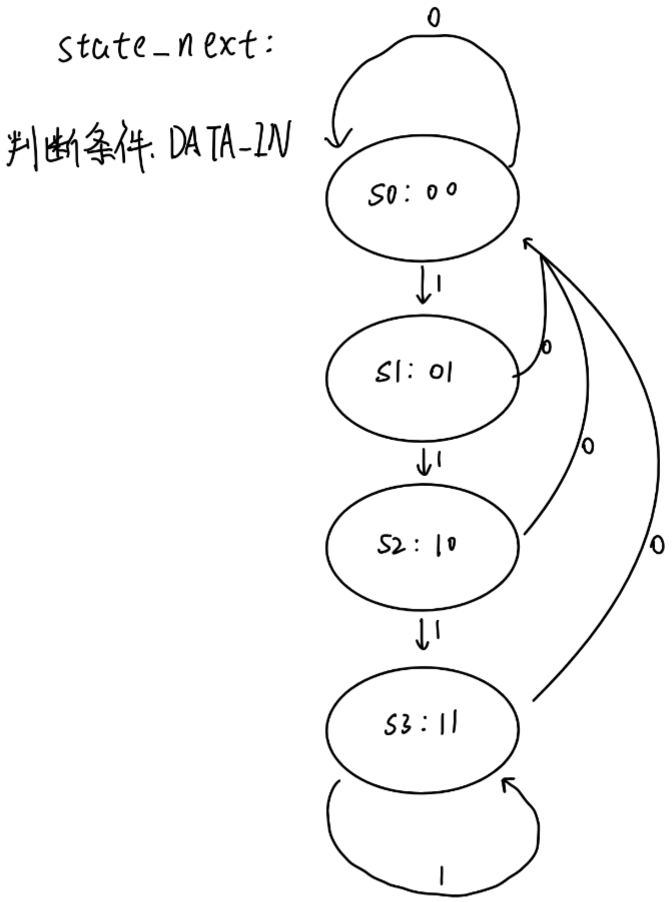


hw_2_2

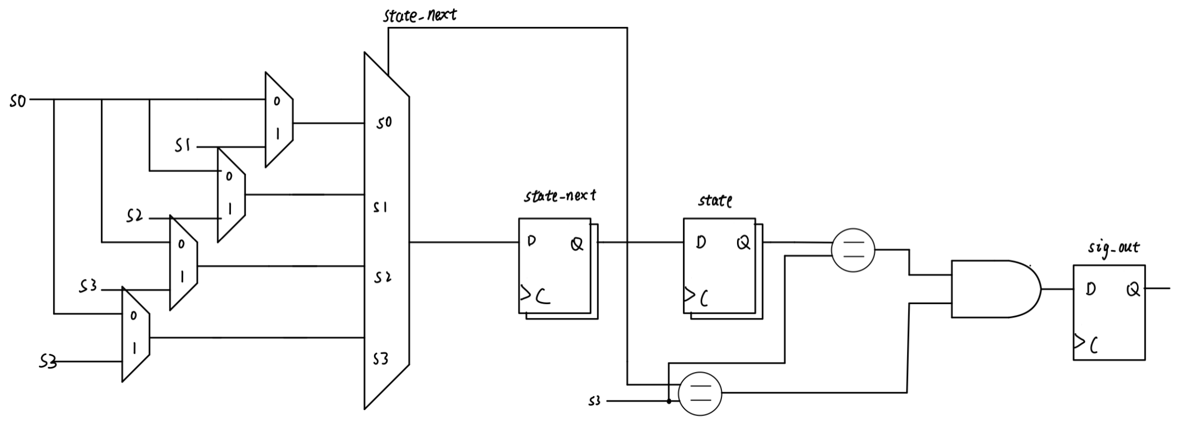
代码

```
`timescale 1ns/1ns
//series data detector
module hw_2_2 (
    input DATA_IN,CLK,RST,
    output SIG_OUT
);
    reg [1:0] state, state_next;
    reg sig_out;

    parameter s0 = 2'b00, s1 = 2'b01, s2 = 2'b10, s3 = 2'b11;
    assign SIG_OUT = sig_out;

    always @(posedge CLK) begin
        if(RST)
        sig_out <= 0;
        else if(state == s3 && state_next == s3)
        sig_out <= 1;
        else
        sig_out <= 0;
    end

    always@(posedge CLK) begin
        if(RST)
        state <= 0;
        else
        state <= state_next;
    end

    always @(*) begin
        if(RST)
        state_next = 0;
        else begin
            case (state)
            s0: if(!DATA_IN) state_next = s0;  else  state_next = s1;
            s1: if(!DATA_IN) state_next = s0;  else  state_next = s2;
            s2: if(!DATA_IN) state_next = s0;  else  state_next = s3;
            s3: if(!DATA_IN) state_next = s0;  else  state_next = s3;
            default:         state_next = state_next;
        endcase
        end
    end
endmodule
```

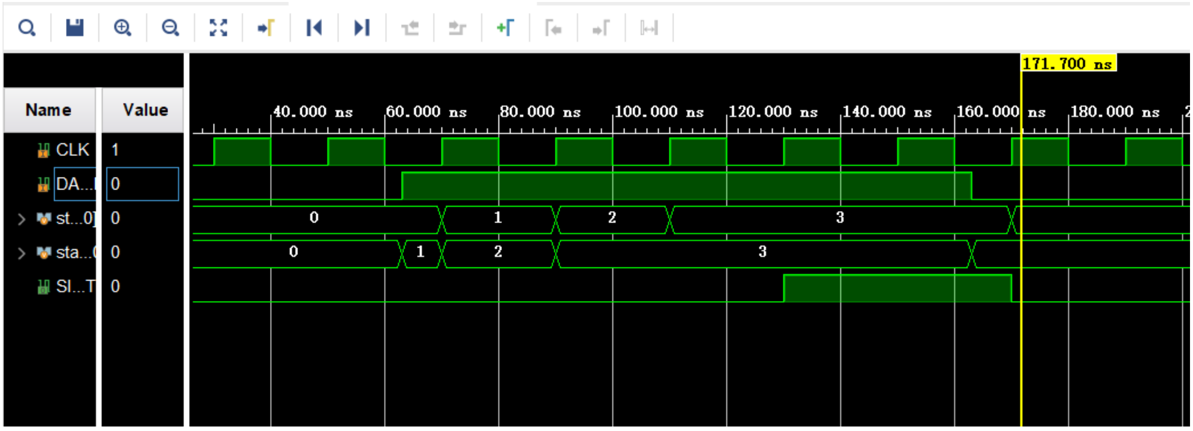
状态转换图



RTL原理图



仿真结果



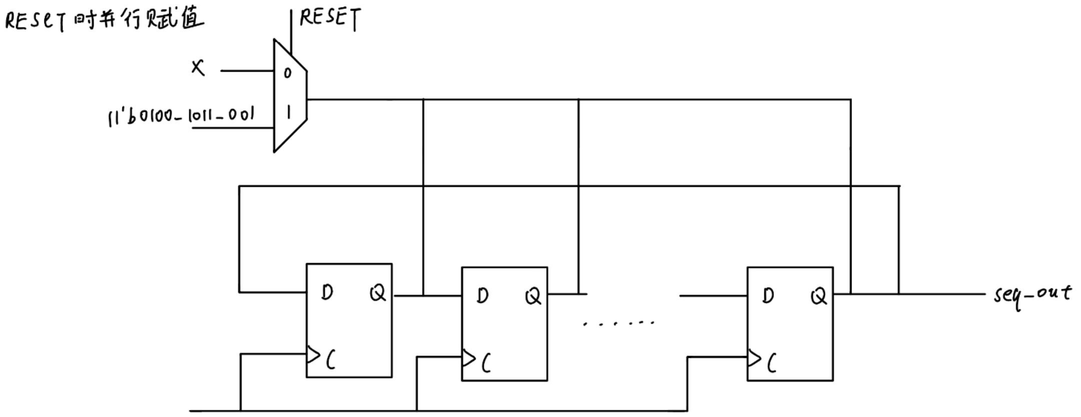
hw_2_3

利用循环移位寄存器实现

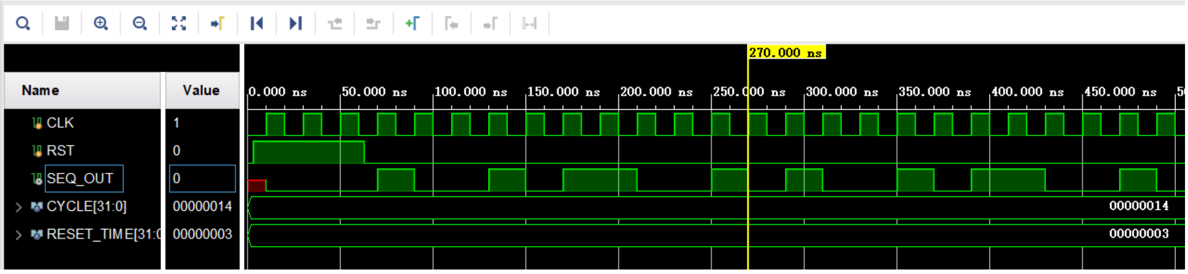
代码

```
//data array generator 0100_1011_001
module hw_2_3(
    input CLK,RST,
    output SEQ_OUT
    );
    reg  [10:0] sequence ;
    wire seq_out;

    assign SEQ_OUT = seq_out;
always@(posedge CLK)
begin
    if(RST)
        sequence <= 11'b0100_1011_001;
    else
        sequence <= {sequence[9:0],sequence[10]};
end
    assign seq_out = sequence[10];
endmodule

```

原理图



仿真结果



Name | Value
CLK 1
RST 0
SEQ_OUT 0
CYCLE[31:0] 00000014
RESET_TIME[31:0] 00000003