数字电路作业

Verilog编程-Homework

姜俊彦 2022K8009970011 本科部

> 编程题 1：用Verilog HDL编写一个4路随机仲裁器。

编程题1源码：（四路随机仲裁器）

```
module random_Arb4 #(parameter N=4)
(
    input clk,rstn,
    input [N-1:0] req_i,
    output [N-1:0] ack_i,
    output req_o,
    input ack_o
);
    genvar i;
    wire [N-1:0] prio;
    wire [N-1:0] q;
    assign req_o = |req_i;

    lfsrN #(N)random (.clk(clk),.rstn(rstn),.q(q));
    transfrom transfrom(.q(q),.prio(prio));

    reg [N-1:0] ack_i_reg;

    generate
        for (i = 0; i<N; i=i+1) begin
            if(i==0)begin
              assign ack_i[0]=prio[0]&req_i[0]&ack_o;
            end
            else begin
                assign ack_i[i]=prio[i]&req_i[i]&ack_o;
            end
        end
    endgenerate
endmodule //5B-13_ArbPriority_noroute


module lfsrN #(parameter N=4)(
    input clk,rstn,
    output [N-1:0] q
);
    reg [N-1:0] q;
    always @(posedge clk or negedge rstn) begin
        if(!rstn) begin
            q<=4;
        end
        else begin
        q[N-1:1]<=q[N-2:0];
            if(N==3) q[0]<=q[2]^q[1];
            if(N==4) q[0]<=q[3]^q[2];
            if(N==5) q[0]<=q[4]^q[2];
            if(N==6) q[0]<=q[5]^q[4];
            if(N==7) q[0]<=q[6]^q[5];
            if(N==8) q[0]<=q[7]^q[5]^q[4]^q[3];
        end
    end
endmodule

module transfrom
(
    input [3:0] q,
    output [3:0] prio
);
    reg [3:0]prio_reg;
    always @(q) begin
        if(q>=1&q<=3) begin
            prio_reg=4'b0001;
        end
        else if(q>=5&q<=7) begin
            prio_reg=4'b0010;
        end
        else if(q>=9&q<=11) begin
            prio_reg=4'b0100;
        end
        else if(q>=13&q<=15) begin
            prio_reg=4'b1000;
        end
        else begin
            prio_reg=4'b0000;
        end
    end
    assign prio=prio_reg;
endmodule
```

编程题1源码（四路动态随机优先级仲裁器）

```
module randomprio_Arb4 #(parameter N=4)
(
    input clk,rstn,
    input [N-1:0] req_i,
    output [N-1:0] ack_i,
    output req_o,
    input ack_o
);
    genvar i;
    wire token[2*N-1:0];
    wire [N-1:0] prio;
    wire [N-1:0] q;
    assign req_o = |req_i;

    lfsrN #(N)random (.clk(clk),.rstn(rstn),.q(q));
```

```
    transfrom transfrom(.q(q),.prio(prio));

    reg [N-1:0] ack_i_reg;

    generate
        for (i = 0; i<N; i=i+1) begin
            if(i==0)begin
              assign ack_i[0]=(token[N-1]|prio[0])&req_i[0]&ack_o;
              assign token[0]=(token[N-1]|prio[0])&~req_i[0];
              assign token[N]=token[N-1]&~req_i[0];
            end
            else begin
                assign ack_i[i]=(token[i-1]|prio[i]|token[N+i-
1])&req_i[i]&ack_o;
                assign token[i]=(token[i]|prio[i])&~req_i[i];
                assign token[N+i]=token[N+i-1]&~req_i[i];
            end
        end
    endgenerate
endmodule //5B-13_ArbPriority_noroute


module lfsrN #(parameter N=4)(
    input clk,rstn,
    output [N-1:0] q
);
    reg [N-1:0] q;
    always @(posedge clk or negedge rstn) begin
        if(!rstn) begin
            q<=4;
        end
        else begin
        q[N-1:1]<=q[N-2:0];
            if(N==3) q[0]<=q[2]^q[1];
            if(N==4) q[0]<=q[3]^q[2];
            if(N==5) q[0]<=q[4]^q[2];
            if(N==6) q[0]<=q[5]^q[4];
            if(N==7) q[0]<=q[6]^q[5];
            if(N==8) q[0]<=q[7]^q[5]^q[4]^q[3];
        end
    end
endmodule

module transfrom
(
    input [3:0] q,
    output [3:0] prio
);
    reg [3:0]prio_reg;
    always @(q) begin
        if(q>=1&q<=3) begin
            prio_reg=4'b0001;
        end
        else if(q>=5&q<=7) begin
            prio_reg=4'b0010;
        end
        else if(q>=9&q<=11) begin
            prio_reg=4'b0100;
        end
        else if(q>=13&q<=15) begin
            prio_reg=4'b1000;
        end
        else begin
            prio_reg=4'b0000;
        end
    end
    assign prio=prio_reg;
endmodule
```

编程题1测试模块源码：

```
`timescale 1ns/1ps

module  test;
    reg clk,rstn;
    reg [3:0] req_i;
    reg ack_o;
    wire [3:0] ack_i;
    wire req_o;

    random_Arb4 #(4)
dut(.clk(clk),.rstn(rstn),.req_i(req_i),.ack_i(ack_i),.req_o(req_o),.ack_o(ack
_o));
    //randomprio_Arb4 #(4)
dut(.clk(clk),.rstn(rstn),.req_i(req_i),.ack_i(ack_i),.req_o(req_o),.ack_o(ack
_o));

    initial begin
        clk=0;
        forever begin
            #5 clk=~clk;
        end
    end
    initial begin
        req_i<=0;
        ack_o=1;
        #2000
        ack_o=0;
        #100
        ack_o=1;
    end
    initial begin
        rstn<=1;
        #3 rstn<=0;
        #13 rstn<=1;
    end
```

```
    always @(posedge clk) begin
        if(~req_i[0]) begin
            req_i[0]<=$random;
        end
        if(~req_i[1]) begin
            req_i[1]<=$random;
        end
        if(~req_i[2]) begin
            req_i[2]<=$random;
        end
        if(~req_i[3]) begin
            req_i[3]<=$random;
        end
    end

    always @(negedge clk) begin
        if(ack_i[0])begin
            req_i[0]<=0;
        end
        if(ack_i[1])begin
            req_i[1]<=0;
        end
        if(ack_i[2])begin
            req_i[2]<=0;
        end
        if(ack_i[3])begin
            req_i[3]<=0;
        end

    end


    integer n0=0,n1=0,n2=0,n3=0;
    always @(negedge clk) begin
        if(ack_i[0]) begin
            n0=n0+1;
        end
        if(ack_i[1]) begin
            n1=n1+1;
        end
        if(ack_i[2]) begin
            n2=n2+1;
        end
        if(ack_i[3]) begin
            n3=n3+1;
        end
    end



    initial begin
        $dumpfile("random_Arb4_test.vcd");
        $dumpvars(0);
        #400000
        $display($stime, ": port 0 is answered %d times.\n",n0);
        $display($stime, ": port 1 is answered %d times.\n",n1);
        $display($stime, ": port 2 is answered %d times.\n",n2);
        $display($stime, ": port 3 is answered %d times.\n",n3);
        $finish();
    end

endmodule
```

编程题1仿真结果：

```
20149@Jiuhao-S-System-Center MINGW64 ~/Verilog-demo
$ vvp random_Arb4
VCD info: dumpfile random_Arb4_test.vcd opened for output.
   4000000: port 0 is answered       63194 times.

   4000000: port 1 is answered       69768 times.

   4000000: port 2 is answered       71475 times.

   4000000: port 3 is answered       65773 times.


20149@Jiuhao-S-System-Center MINGW64 ~/Verilog-demo
$ iverilog -o randomprio_Arb4 -s test src/randomprio_Arb4.v test/randomprio_Arb4
_test.v

20149@Jiuhao-S-System-Center MINGW64 ~/Verilog-demo
$ vvp randomprio_Arb4
VCD info: dumpfile randomprio_Arb4_test.vcd opened for output.
    400000: port 0 is answered       10965 times.

    400000: port 1 is answered        9652 times.

    400000: port 2 is answered       10041 times.

    400000: port 3 is answered        9920 times.
```

> 编程题 2：用Verilog HDL编写一个输出为寄存器直接输出的FIFO缓 冲，缓冲深度为8，数据宽度为4位。

编程题2源代码

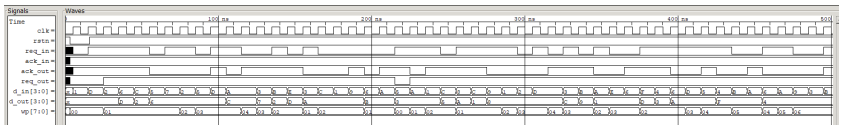
```
module fifo_register_out #(parameter dw=4,L=8)
(
    input clk,rstn,
    input [dw-1:0] d_in,
    input req_in,
    output ack_in,
    output [dw-1:0] d_out,
    output req_out,
    input ack_out
);
    reg [dw-1:0] data [L-1:0];
    reg [L-1:0] wp;
    reg [dw-1:0]out;


    assign req_out=|wp;
    assign ack_in=(wp==L)?0:1;
    assign d_out=out;
```

```
    always @(posedge clk or negedge rstn) begin
        if(!rstn) begin
            wp<=0;
        end
        else begin
            if(req_in&ack_in&(~(req_out&ack_out))) begin
                wp<=(wp==L)?L:wp+1;
                data[wp]<=d_in;
            end
            else if(req_out&ack_out&(~(req_in&ack_in))) begin
                wp<=(wp==0)?0:wp-1;
                data[3]<=0;
                data[2]<=data[3];
                data[1]<=data[2];
                data[0]<=data[1];
                out<=data[0];
            end
            else if(req_in&ack_in&req_out&ack_out)begin
                data[2]<=data[3];
                data[1]<=data[2];
                data[0]<=data[1];
                out<=data[0];
                data[wp-1]<=d_in;
            end
            else begin
                ;
            end
        end
    end
endmodule


//深度为L的FIFO缓冲
```

编程题2测试代码

```
`timescale 1ns/1ps
module test;
    parameter dw=4,L=8;
    reg clk,rstn;
    reg [dw-1:0] d_in;
    reg req_in;
    wire ack_in;
    wire [dw-1:0] d_out;
    wire req_out;
    reg ack_out;

    reg [dw-1:0]localfifo[L-1:0];
    integer wp=0;
    integer data_num=0;
    integer full_num=0;
    integer empty_num=0;

    fifo_register_out #(dw,L)
dut(clk,rstn,d_in,req_in,ack_in,d_out,req_out,ack_out);

    initial begin
        clk=0;
        forever begin
            #5 clk=~clk;
        end
    end

    initial begin
        rstn=1;
        #3 rstn=0;
        #13 rstn=1;
    end

    always @(posedge clk) begin
        req_in<=$random;
        d_in<=$random;
        ack_out<=$random;
    end


    initial begin
        $dumpfile("fifo_register_out.vcd");
        $dumpvars(0);
        #10000 $finish();
    end
endmodule
```

编程题2仿真波形



> 编程题3：用Verilog HDL编写一个交通灯，按照红灯10个时钟周期， 黄灯1个时钟周期，绿灯9个时钟周期，黄灯2个时钟周期 的顺序循环。

编程题3源码

```
module traffic_light
(
    input clk,rstn,
    output [1:0]signal  //2'b00: Error, 2'b01:red, 2'b10:green, 2'b11:yellow
);
    reg [4:0] cnt;
    reg [1:0] signal_reg;

    always @(posedge clk or negedge rstn) begin
        if(!rstn) begin
            cnt<=22;
```

```
        end
        else if(cnt <21) begin
            cnt<=cnt+5'b00001;
        end
        else begin
            cnt<=5'b00000;
        end
    end

    always @(cnt) begin
        if(cnt<10) begin
            signal_reg=2'b01;
        end
        else if(cnt<11) begin
            signal_reg=2'b11;
        end
        else if(cnt<20) begin
            signal_reg=2'b10;
        end
        else if(cnt<22) begin
            signal_reg=2'b11;
        end
        else begin
            signal_reg=2'b00;   //Error Signal
        end
    end

    assign signal=signal_reg;
endmodule
```

编程题3测试代码

```
`timescale 1ns/1ps

module test;
    reg clk;
    reg rstn;
    wire [1:0]signal;

    traffic_light dut(.clk(clk),.rstn(rstn),.signal(signal));

    initial begin
        clk=0;
        forever begin
            #5 clk=~clk;
        end
    end

    initial begin
        rstn=1;
        #3
        rstn=0;
        #13 rstn=1;
    end

    initial begin
        $dumpfile("traffic_light_test.vcd");
        $dumpvars(0);
        #300 $finish();
    end
endmodule
```

编程题3仿真波形