3-4 用 Verilog 设计一个 3-8 译码器。要求分别用 case 语句和 if_else 语句。比较这两种方式

3-5 图 3-16 所示的是双 2 选 1 多路选择器构成的电路 MUXK。对于其中 MUX21A，当 s=0 和 s=1 时，分别有 y=a 和 y=b。试在一个模块结构中用两个过程来表达此电路。

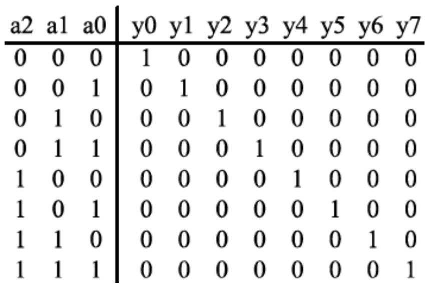
3-6 给出一个四选一多路选择器的 Verilog 描述。选通控制端有四个输入：S0,S1,S2,S3。当且仅当 S0=0 时 Y=A;S1=0 时，Y=B;S2=0 时，Y=C;S3=0 时，Y=D。

解：

（1）解题思路

3-4 解题思路：

3-8 译码器真值表



a2	a1	a0	y0	y1	y2	y3	y4	y5	y6	y7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

一个宽度为 3 的输入，一个宽度为 8 的输出。

先设计用 case 语句的译码器，再设计用 if_else 语句的译码器。

decoder38a 采用 case 语句

decoder38b 采用 if_else 语句

3-5 解题思路：

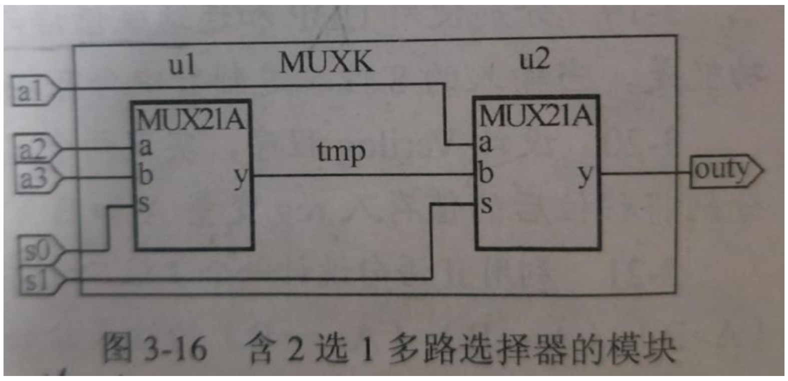


图 3-16 含 2 选 1 多路选择器的模块

就直接设计两个过程，第二个过程的其中一个输入 b 来自第一个过程的输出 y。主要用了 wire 和= ？ : 语句。

3-6 解题思路：

采用 if_else 语句，方便，清晰的表示输入控制和输出的关系。

因为当且仅当 S0=0 时 Y=A;S1=0 时，Y=B;S2=0 时，Y=C;S3=0 时，Y=D。所以我先判断 S3,再判断 S2，以此类推，相当于有控制的优先级。

注意，这里我设置如果输入均为 1，那么 Y 为 0.

（2）核心模块代码

3-4

采用 case 的 decoder38a:

```
module decoder38a(data_in,result);
input [2:0]data_in;
output reg[7:0]result;
always@(*)
 begin
 case(data_in)
  3'b000: result = 8'b00000001;
  3'b001: result = 8'b00000010;
  3'b010: result = 8'b00000100;
  3'b011: result = 8'b00001000;
  3'b100: result = 8'b00010000;
  3'b101: result = 8'b00100000;
  3'b110: result = 8'b01000000;
  3'b111: result = 8'b10000000;
 endcase
 end
endmodule
```

采用 if_else 的 decoder38b:

```
module decoder38b(data_in,result);
input [2:0]data_in;
output reg[7:0]result;
reg[3:0]SEL;
always@(*)
 begin
 SEL={data_in[2],data_in[1],data_in[0]};
 if(SEL==0)
    result = 8'b00000001;
 else if(SEL==1)
    result = 8'b00000010;
 else if(SEL==2)
    result = 8'b00000100;
     else if(SEL==3)
    result = 8'b00001000;
     else if(SEL==4)
    result = 8'b00010000;
     else if(SEL==5)
    result = 8'b00100000;
     else if(SEL==6)
    result = 8'b01000000;
     else if(SEL==7)
    result = 8'b10000000;
 end
endmodule
```

3-5 doublemux21a

```
module doublemux21a(a1,a2,a3,s0,s1,Y);
    input a1,a2,a3,s0,s1;
    output Y;
    wire tp=s0 ? a3 : a2;
    wire Y = s1 ? tp : a1;
endmodule
```

3-6 mux41

```
module mux41 (Y,A,B,C,D,S3,S2,S1,S0);
    output reg Y;
    input A,B,C,D,S3,S2,S1,S0;  //有优先级顺序的
    always@(*)
    begin
    if(S3==0)
    Y=D;
    else if(S2==0)
    Y=C;
    else if(S1==0)
    Y=B;
    else if(S0==0)
    Y=A;
    else
    Y=1'b0;
    end
endmodule
```

（3）测试模块代码

3-4
采用 case 的 decoder38a_test_tb.v：

```
`timescale 1ns/1ps
module decoder38a_test_tb;

    reg [2:0] data_in;
    wire [7:0] result;

    decoder38a DUT (
        .data_in(data_in),
        .result(result)
    );

    initial begin

        data_in = 0;

        #20;

        data_in = 3'b000;
          #20;

          data_in = 3'b001;
          #20;

          data_in = 3'b010;
          #20;

          data_in = 3'b011;
          #20;

          data_in = 3'b100;
          #20;

          data_in = 3'b101;
          #20;

          data_in = 3'b110;
          #20;

          data_in = 3'b111;
    end
endmodule
```

采用 if_else 的 decoder38b_test_tb.v：

```
`timescale 1ns/1ps
module decoder38b_test_tb;

    reg [2:0] data_in;
    wire [7:0] result;

    decoder38b DUT (
        .data_in(data_in),
        .result(result)
    );

    initial begin

        data_in = 0;

        #20;

        data_in = 3'b000;
          #20;

          data_in = 3'b001;
          #20;

          data_in = 3'b010;
          #20;

          data_in = 3'b011;
          #20;

          data_in = 3'b100;
          #20;

          data_in = 3'b101;
          #20;

          data_in = 3'b110;
          #20;
          data_in = 3'b111;
    end
endmodule
```

3-5 doublemux21_test_tb.v

```
`timescale 1ns/1ps
module doublemux21a_test_tb;
    reg a1,a2,a3,s1,s0;
    wire y;
    doublemux21a DUT  //Device Under Test
    (
        .a1(a1),
        .a2(a2),
        .a3(a3),
        .s1(s1),
        .s0(s0),
        .Y(y)
    );

    initial begin
        a1 = 1'b1;
        a2 = 1'b1;
        a3 = 1'b0;
        s0 = 1'b0;
        s1 = 1'b0;
        #450 $finish;
    end

    always #20 a1 = ~a1;
    always #40 a2 = ~a2;
    always #60 a3 = ~a3;
    always #30 s0 = ~s0;
    always #20 s1 = ~s1;

    always @(y);
//      $finish;
//  end
endmodule
```

3-6 mux41_test.tb.v

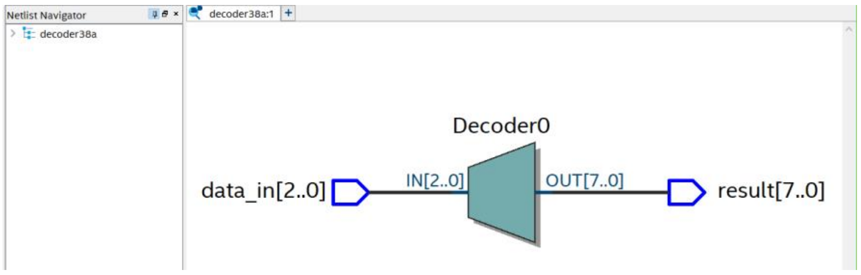
```
`timescale 1ns/1ps
```

```
module mux41_test_tb;
    reg a,b,c,d,s3,s2,s1,s0;
    wire y;
    mux41 DUT  //Device Under Test
    (
        .A(a),
        .B(b),
        .C(c),
        .D(d),
        .S3(s3),
        .S2(s2),
        .S1(s1),
        .S0(s0),
        .Y(y)
    );

    initial begin
        a = 1'b1;
        b = 1'b1;
        c = 1'b0;
        d = 1'b0;
        s0 = 1'b1;
        s1 = 1'b0;
        s2 = 1'b0;
        s3 = 1'b0;
        #450 $finish;
    end

    always #20 a = ~a;
    always #40 b = ~b;
    always #60 c = ~c;
    always #80 d = ~d;
    always #30 s0 = ~s0;
    always #40 s1 = ~s1;
    always #50 s2 = ~s2;
    always #60 s3 = ~s3;

    always @(y);
//      $finish;
//  end
endmodule
```

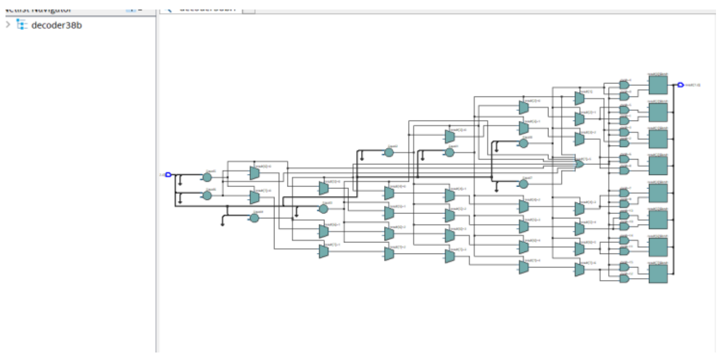
（4）RTL View 的网表图

3-4

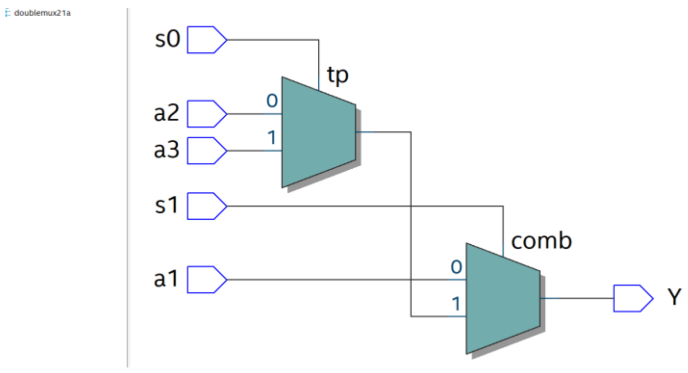
采用 case 的 decoder38a



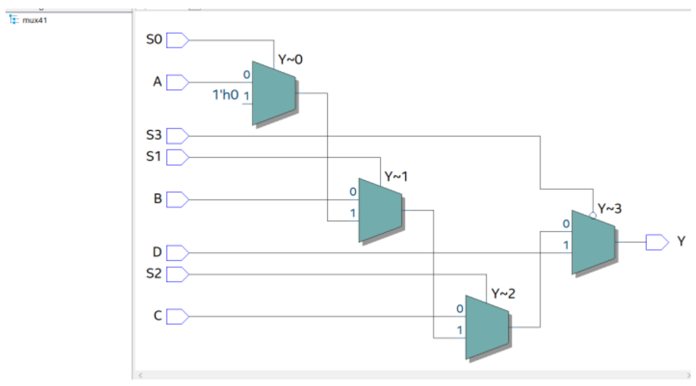
采用 if_else 的 decoder38b



3-5 doublemux21a



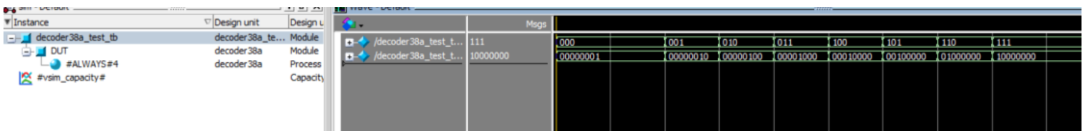
3-6 mux41



（5）ModelSim 仿真的结果图

3-4

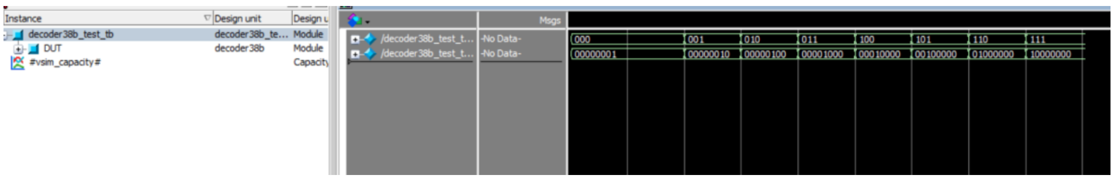
采用 case 的 decoder38a



Instance	Design unit	Design u
decoder38a_test_tb	decoder38a_te...	Module
DUT	decoder38a	Module
#ALWAYS#4	decoder38a	Process
#vsim_capacity#		Capacity

```
/decoder38a_test_t...  111       000       001       010       011       100       101       110       111
/decoder38a_test_t...  10000000  00000001  00000010  00000100  00001000  00010000  00100000  01000000  10000000
```

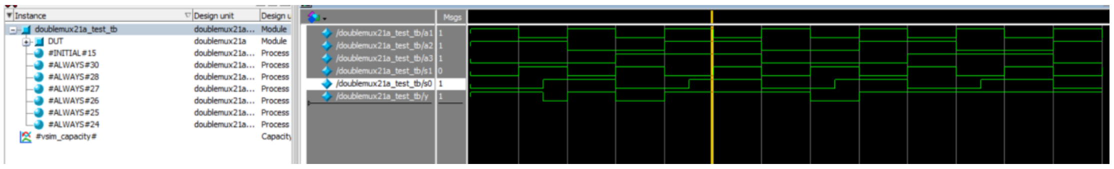
采用 if_else 的 decoder38b



Instance	Design unit	Design u
decoder38b_test_tb	decoder38b_te...	Module
DUT	decoder38b	Module
#vsim_capacity#		Capacity

```
/decoder38b_test_t...  -No Data-  000       001       010       011       100       101       110       111
/decoder38b_test_t...  -No Data-  00000001  00000010  00000100  00001000  00010000  00100000  01000000  10000000
```

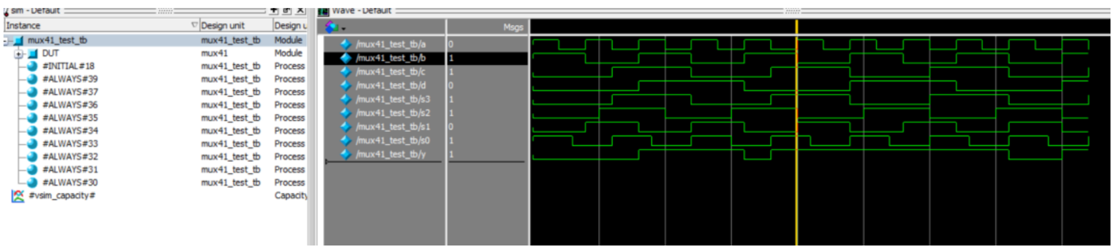
3-5 doublemux21a



Instance	Design unit	Design u
doublemux21a_test_tb	doublemux21a...	Module
DUT	doublemux21a	Module
#INITIAL#15	doublemux21a...	Process
#ALWAYS#30	doublemux21a...	Process
#ALWAYS#28	doublemux21a...	Process
#ALWAYS#27	doublemux21a...	Process
#ALWAYS#26	doublemux21a...	Process
#ALWAYS#25	doublemux21a...	Process
#ALWAYS#24	doublemux21a...	Process
#vsim_capacity#		Capacity

```
/doublemux21a_test_tb/a1  1
/doublemux21a_test_tb/a2  1
/doublemux21a_test_tb/a3  1
/doublemux21a_test_tb/s1  0
/doublemux21a_test_tb/s0  1
/doublemux21a_test_tb/y   1
```

3-6 mux41



Instance	Design unit	Design u
mux41_test_tb	mux41_test_tb	Module
DUT	mux41	Module
#INITIAL#18	mux41_test_tb	Process
#ALWAYS#39	mux41_test_tb	Process
#ALWAYS#37	mux41_test_tb	Process
#ALWAYS#36	mux41_test_tb	Process
#ALWAYS#35	mux41_test_tb	Process
#ALWAYS#34	mux41_test_tb	Process
#ALWAYS#33	mux41_test_tb	Process
#ALWAYS#32	mux41_test_tb	Process
#ALWAYS#31	mux41_test_tb	Process
#ALWAYS#30	mux41_test_tb	Process
#vsim_capacity#		Capacity

```
/mux41_test_tb/a   0
/mux41_test_tb/b   1
/mux41_test_tb/c   1
/mux41_test_tb/d   0
/mux41_test_tb/s3  1
/mux41_test_tb/s2  1
/mux41_test_tb/s1  0
/mux41_test_tb/s0  1
/mux41_test_tb/y   1
```