DIGITAL DESIGN

ASSIGNMENT REPORT

ASSIGNMENT ID : 2

Student Name: 罗嘉诚

Student ID: 12112910

Assignment 2

$x \oplus y = xy' + x'y$

或门：

异或门：

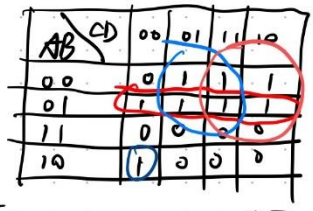
Question 1 10 point

输入 $(ABCD)_2$ $A, B, C, D \in \{0, 1\}$

输出 $(PQRS)_2$ $P, Q, R, S \in \{0, 1\}$

ABCD	PQRS
0000	0000
0001	1111
0010	1110
0011	1101
0100	1100
0101	1011
0110	1010
0111	1001
1000	1060
1001	0111
1010	0110
1011	0101
1100	0100
1101	0011
1110	0010
1111	0001

① P 与 ABCD 卡诺图

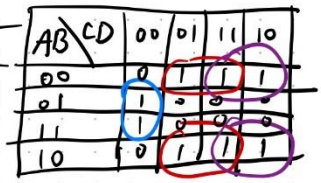


$P = A'B + A'D + A'C + AB'C'D'$

$= A'(B+C+D) + A(B'C'D')$

$= A \oplus (B+C+D)$

② Q 与 ABCD 卡诺图

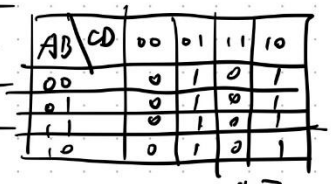


$Q = B'D + B'C + BC'D'$

$= B'(C+D) + B(C'D')$

$= B \oplus (C+D)$

③ R 与 ABCD 卡诺图



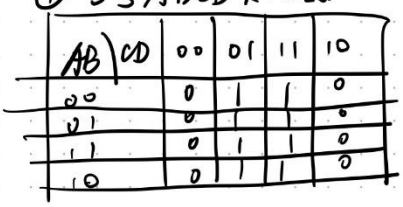
$R = C'D + CD'$

$= C \oplus D$

所以：

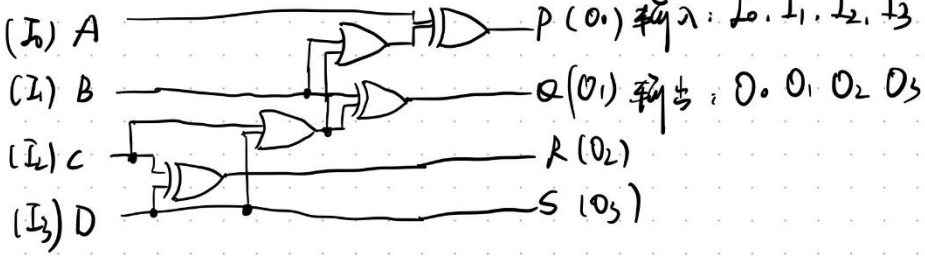
$$\begin{cases} P = A \oplus (B+C+D) \\ Q = B \oplus (C+D) \\ R = C \oplus D \\ S = D \end{cases}$$

④ S 与 ABCD 卡诺图



$S = D$

电路图



Question 2

2 point ✓

$x\ y\ z \rightarrow A\ B\ C$

	x	y	z	A	B	C	
0	0	0	0	0	1	0	2
1	0	0	1	0	1	1	3
2	0	1	0	1	0	0	4
3	0	1	1	0	1	0	2
4	1	0	0	0	1	1	3
5	1	0	1	1	0	0	4
6	1	1	0	1	0	1	5
7	1	1	1	1	1	0	6

所以
$$\begin{cases} A = yz' + xz \\ B = y'z' + x'z + yz \\ C = xz' + x'y'z \end{cases}$$

A 关于 x, y, z 卡诺图

xy \ z	0	1
00	0	0
01	1	0
11	1	1
10	0	1

$A = yz' + xz$ ✓

B 关于 x, y, z 卡诺图

xy \ z	0	1
00	1	1
01	0	1
11	0	1
10	1	0

$B = y'z' + x'z + yz$ ✓

C 关于 x, y, z 卡诺图

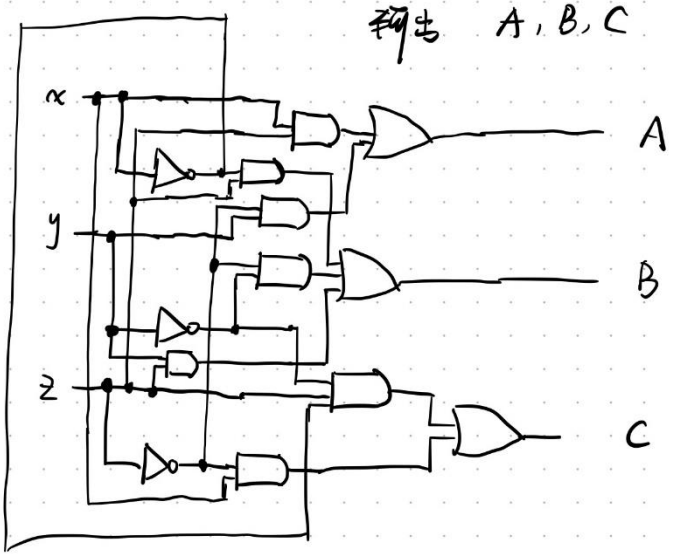
xy \ z	0	1
00	0	1
01	0	0
11	1	0
10	1	0

$C = xz' + x'y'z$ ✓

电路图如下

输入：x, y, z

输出 A, B, C



Question 3 2 point ✓

A	B	C	D	Z
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

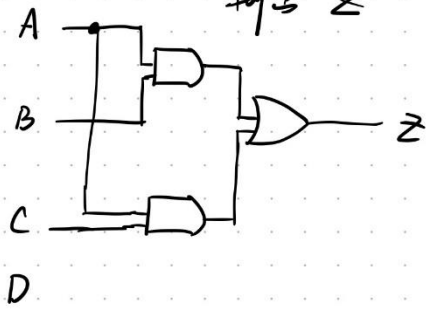
AB \ CD	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	0	0	1	1

$Z = AB + AC$ ✓

电路图：

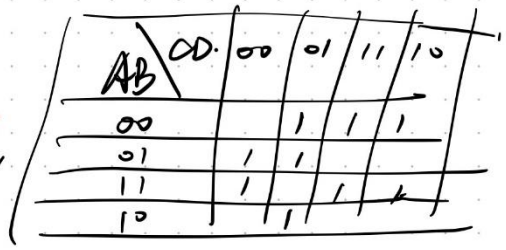
输入 A, B, C, D

输出 Z



Question 4 20 point ✓

8对1多路复用器



$$Y = A'B'C'I_0 + A'B'C\,I_1 + A'BC'\,I_2 + A'BC\,I_3 + AB'C'\,I_4 + AB'C\,I_5 + ABC'\,I_6 + ABC\,I_7$$

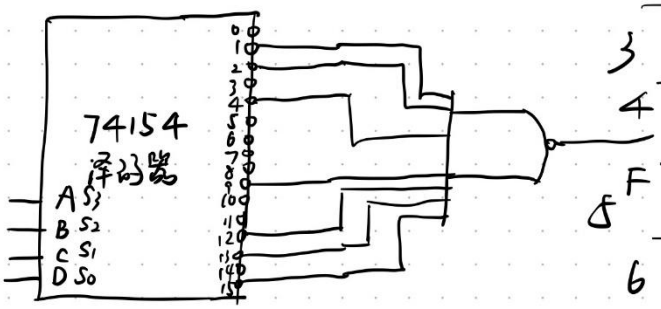
$$= A'B'C'D + A'B'C + A'BC' + AB'C'D + ABC'D' + ABC$$

$$= A'B'C + \underline{A'C'D} + \underline{BC'D'} + ABC + \underline{B'C'D}$$

Question 5 20 point ✓

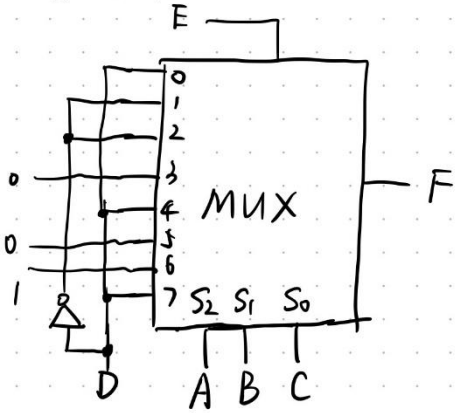
① 使用译码器实现最小项逻辑器

$F(A,B,C,D) = \sum(1,2,4,9,12,13,15)$



	A	B	C	D	F	
0	0	0	0	0	0	D
	0	0	0	1	1	
1	0	0	1	0	1	D'
	0	0	1	1	0	
2	0	1	0	0	1	D'
	0	1	0	1	0	
3	0	1	1	0	0	0
	0	1	1	1	0	
4	1	0	0	0	0	D
	1	0	0	1	1	
F	1	0	1	0	0	0
5	1	0	1	1	0	
6	1	1	0	0	1	1
	1	1	0	1	1	
	1	1	1	0	0	
7	1	1	1	1	1	D

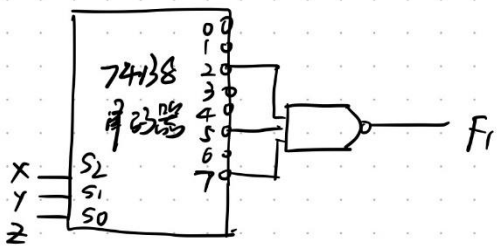
② 使用 MUX 实现最小项逻辑器 ✓



Question 6 10point

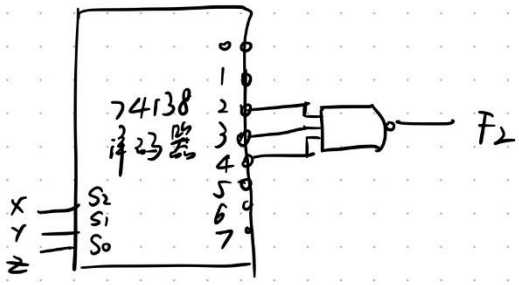
① $F_1 = x'yz' + xz = \sum(2,5,7)$

x	y	z	F_1
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1



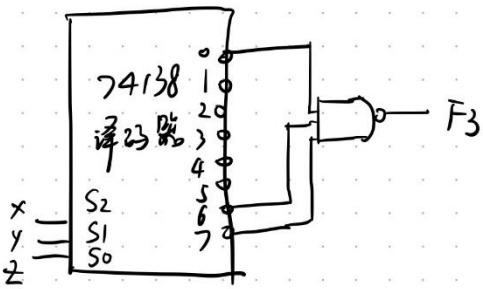
② $F_2 = xy'z' + x'y = \sum(2,3,4)$

x	y	z	F_2
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0



③ $F_3 = x'y'z' + xy = \sum(0,6,7)$

x	y	z	F_3
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1



PART 2: DIGITAL DESIGN LAB (TASK1)

DESIGN

Describe the design of your system by providing the following information:

- *Verilog design (provide the Verilog code)*

Use XOR Gate

```
`timescale 1ns / 1ps
module a2t1_xor(x, y);
input [3: 0] x;
output y;
xor xor1(y, x[0], x[1], x[2], x[3]);
endmodule
```

Use NAND Gate

```
`timescale 1ns / 1ps
module a2t1_nand(x, y);
input [3: 0] x;
output y;
wire x0_nand_x1;
nand nand1(x0_nand_x1, x[0], x[1]);
wire x0_nand_x1_nand_x0, x0_nand_x1_nand_x1;
nand nand2(x0_nand_x1_nand_x0, x0_nand_x1, x[0]);
nand nand3(x0_nand_x1_nand_x1, x0_nand_x1, x[1]);
wire res1;
nand nand4(res1, x0_nand_x1_nand_x0, x0_nand_x1_nand_x1);
wire res1_nand_x2;
nand nand5(res1_nand_x2, res1, x[2]);
wire res1_nand_x2_nand_res1, res1_nand_x2_nand_x2;
nand nand6(res1_nand_x2_nand_res1, res1_nand_x2, res1);
nand nand7(res1_nand_x2_nand_x2, res1_nand_x2, x[2]);
```

```
wire res2;
nand nand8(res2,  res1_nand_x2_nand_res1, res1_nand_x2_nand_x2);
wire res2_nand_x3;
nand nand9(res2_nand_x3, res2, x[3]);
wire res2_nand_x3_nand_res2, res2_nand_x3_nand_x3;
nand nand10(res2_nand_x3_nand_res2, res2_nand_x3, res2);
nand nand11(res2_nand_x3_nand_x3, res2_nand_x3, x[3]);
nand nand12(y, res2_nand_x3_nand_res2, res2_nand_x3_nand_x3);
endmodule
```

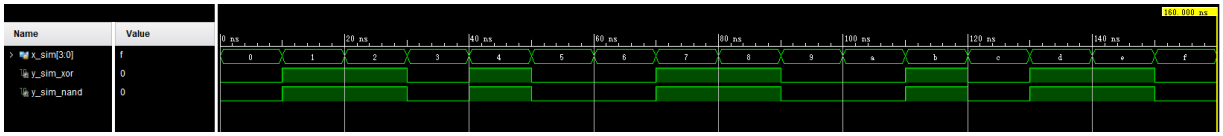
SIMULATION

Describe how you build the test bench and do the simulation.

- *Using Verilog (provide the Verilog code)*

```
`timescale 1ns / 1ps
module a2t1_sim();
reg [3: 0] x_sim;
wire y_sim_xor, y_sim_nand;
a2t1_xor u(x_sim, y_sim_xor);
a2t1_nand v(x_sim, y_sim_nand);
initial begin
   x_sim = 4'b0000;
   while (x_sim < 4'b1111)
   begin
        #10 x_sim = x_sim + 1;
   end
   #10 $finish();
end
endmodule
```

- *Wave form of simulation result (provide screen shots)*



- *The description on whether the simulation result is same as the truth-table, is the function of the design meet the expectation.*

  如上图所示，任务要求我们对于给出的 4-bit 输入 x，输出 1-bit 结果 y，表示 x 的二进制表示中 1 个数的奇偶性（奇数个数为 1，偶数个数为 0）。仿真文件中 x_sim 是一个 4-bit 二进制数表示仿真输入，从 4'h0 依次增加到 4'hF，进行两组仿真，分别得到结果 y_sim_xor 和 y_sim_nand，前者表示进行 XOR 门电路仿真结果，后者表示进行 NAND 门电路仿真结果。可以通过波形图判断，结果均正确，可以说明符合预期结果。

PART 2: DIGITAL DESIGN LAB (TASK2)

DESIGN

Describe the design of your system by providing the following information:

- *Verilog design (provide the Verilog code)*

  Use Sum-of-Minterms

```
`timescale 1ns / 1ps
module a2t2_somin(x, y);
input [3: 0] x;
output y;
wire [3: 0] not_x;
not not1(not_x[0], x[0]);
not not2(not_x[1], x[1]);
not not3(not_x[2], x[2]);
not not4(not_x[3], x[3]);
wire [9: 0] items;
```

```
and and1(items[0], not_x[3], not_x[2], not_x[1], not_x[0]);
and and2(items[1], not_x[3], not_x[2], not_x[1], x[0]);
and and3(items[2], not_x[3], not_x[2], x[1], x[0]);
and and4(items[3], not_x[3], not_x[2], x[1], not_x[0]);
and and5(items[4], not_x[3], x[2], not_x[1], not_x[0]);
and and6(items[5], not_x[3], x[2], not_x[1], x[0]);
and and7(items[6], not_x[3], x[2], x[1], x[0]);
and and8(items[7], not_x[3], x[2], x[1], not_x[0]);
and and9(items[8], x[3], not_x[2], not_x[1], not_x[0]);
and and10(items[9], x[3], not_x[2], not_x[1], x[0]);
or or1(y, items[9], items[8], items[7], items[6], items[5], items[4],
items[3], items[2], items[1], items[0]);
endmodule
```

Use Product-of-Maxterms

```
`timescale 1ns / 1ps
module a2t2_pomax(x, y);
input [3: 0] x;
output y;
wire [3: 0] not_x;
not not1(not_x[0], x[0]);
not not2(not_x[1], x[1]);
not not3(not_x[2], x[2]);
not not4(not_x[3], x[3]);
wire [5: 0] items;
or or1(items[0], not_x[3], not_x[2], x[1], x[0]);
or or2(items[1], not_x[3], not_x[2], x[1], not_x[0]);
or or3(items[2], not_x[3], not_x[2], not_x[1], not_x[0]);
or or4(items[3], not_x[3], not_x[2], not_x[1], x[0]);
or or5(items[4], not_x[3], x[2], not_x[1], not_x[0]);
or or6(items[5], not_x[3], x[2], not_x[1], x[0]);
and and1(y, items[0], items[1], items[2], items[3], items[4], items[5]);
```

```
endmodule
```

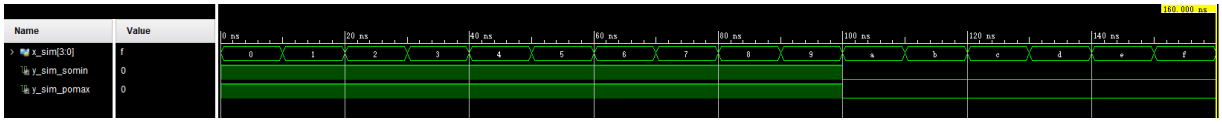
SIMULATION

Describe how you build the test bench and do the simulation.

- *Using Verilog (provide the Verilog code)*

```
`timescale 1ns / 1ps
module a2t2_sim();
reg [3: 0] x_sim;
wire y_sim_somin, y_sim_pomax;
a2t2_somin u(x_sim, y_sim_somin);
a2t2_pomax v(x_sim, y_sim_pomax);
initial begin
   x_sim = 4'b0000;
   while (x_sim < 4'b1111)
   begin
        #10 x_sim = x_sim + 1;
   end
   #10 $finish();
end
endmodule
```

- *Wave form of simulation result (provide screen shots)*



- *The description on whether the simulation result is same as the truth-table, is the function of the design meet the expectation*

如上图所示，任务要求我们对于给出的 4-bit 输入 x，输出 1-bit 结果 y，表示 x 是否为 BCD 码（若 x 在 0-9 之间，则为 1，否则 y 为 0）。仿真文件中 x_sim 是一个 4-bit 二进制数表示仿真输入，从 4'h0 依次增加到 4'hF，进行两组仿真，分别得到

结果 y_sim_somin 和 y_sim_pomax，前者表示使用最小项之和的方式设计电路进行仿真结果，后者表示使用最大项之积的方式设计电路进行仿真结果。可以通过波形图判断，结果均正确，可以说明符合预期结果。

PART 2: DIGITAL DESIGN LAB (TASK3)

DESIGN

Describe the design of your system by providing the following information:

- *Verilog design while using data flow (provide the Verilog code)*

```
`timescale 1ns / 1ps
module a2t3_df(x, y);
input [3: 0] x;
output [3: 0] y;
assign y = (~x) + 1;
endmodule
```

- *Verilog design while using behavioral description styles (provide the Verilog code)*

```
`timescale 1ns / 1ps
module a2t3_bd(x, y);
input [3: 0] x;
output reg [3: 0] y;
always @(x)
begin
    case (x)
        4'b0000: y = 4'b0000;
        4'b0001: y = 4'b1111;
        4'b0010: y = 4'b1110;
        4'b0011: y = 4'b1101;
        4'b0100: y = 4'b1100;
```

```
        4'b0101: y = 4'b1011;
        4'b0110: y = 4'b1010;
        4'b0111: y = 4'b1001;
        4'b1000: y = 4'b1000;
        4'b1001: y = 4'b0111;
        4'b1010: y = 4'b0110;
        4'b1011: y = 4'b0101;
        4'b1100: y = 4'b0100;
        4'b1101: y = 4'b0011;
        4'b1110: y = 4'b0010;
        4'b1111: y = 4'b0001;
    endcase
end
endmodule
```

SIMULATION

Describe how you build the test bench and do the simulation.

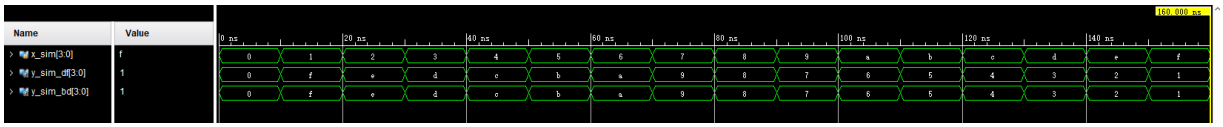
- *Using Verilog (provide the Verilog code)*

```
`timescale 1ns / 1ps
module a2t3_sim();
reg [3: 0] x_sim;
wire [3: 0] y_sim_df, y_sim_bd;
a2t3_df u(x_sim, y_sim_df);
a2t3_bd v(x_sim, y_sim_bd);
initial begin
   x_sim = 4'b0000;
   while (x_sim < 4'b1111)
   begin
        #10 x_sim = x_sim + 1;
   end
   #10 $finish();
```

```
end

endmodule
```

- *Wave form of simulation result (provide screen shots)*



- *The description on whether the simulation result is same as the truth-table, is the function of the design meet the expectation*

如上图所示，任务要求我们对于给出的 4-bit 输入 x，输出 4-bit 结果 y，表示 x 的二进制补码（即每位取反再末尾加 1，取获得的 5-bit 二进制数的后 4 位）。仿真文件中 x_sim 是一个 4-bit 二进制数表示仿真输入，从 4'h0 依次增加到 4'hF，进行两组仿真，分别得到结果 y_sim_df 和 y_sim_bd，前者表示使用数据流设计电路进行仿真结果，后者表示使用行为级描述方式设计电路进行仿真结果。可以通过波形图判断，结果均正确，可以说明符合预期结果。

PART 2: DIGITAL DESIGN LAB (TASK4)

DESIGN

Describe the design of your system by providing the following information:

- *Verilog design while using behavior description style (provide the Verilog code) Design the Decoder (74139)*

```
`timescale 1ns / 1ps
module d74139(ne, x, y);
input ne;
input [1: 0] x;
output reg [3: 0] y;
always @*
begin
    if (~ne)
```

```
        case (x)
            2'b00: y = 4'b1110;
            2'b01: y = 4'b1101;
            2'b10: y = 4'b1011;
            2'b11: y = 4'b0111;
        endcase
    else
        y = 4'hf;
end
endmodule
```

Design the 16-to-1 MUX

```
module  mux16to1(x, sel, y);
input [15: 0] x;
input [3: 0] sel;
output reg y;
always @*
begin
    case (sel)
        4'h0: y = x[0];
        4'h1: y = x[1];
        4'h2: y = x[2];
        4'h3: y = x[3];
        4'h4: y = x[4];
        4'h5: y = x[5];
        4'h6: y = x[6];
        4'h7: y = x[7];
        4'h8: y = x[8];
        4'h9: y = x[9];
        4'ha: y = x[10];
        4'hb: y = x[11];
        4'hc: y = x[12];
```

```
        4'hd: y = x[13];
        4'he: y = x[14];
        4'hf: y = x[15];
    endcase
end
endmodule
```

- *Verilog design while using structured design (provide the Verilog code)*

Use the decoder (74139) and external gates

```
`timescale 1ns / 1ps
module a2t4_dc(x, y);
input [4: 0] x;
output y;
wire [3: 0] t0, t1, t2, t3;
wire [3: 0] xor_0, xor_1, xor_2, xor_3;
wire [3: 0] res;
d74139 d74139_1(x[4], 2'b00, t0);
xor xor1(xor_0[3], t0[3], x[3]);
xor xor2(xor_0[2], t0[2], x[2]);
xor xor3(xor_0[1], t0[1], x[1]);
xor xor4(xor_0[0], t0[0], x[0]);
and and1(res[0], xor_0[3], xor_0[2], xor_0[1], xor_0[0]);
d74139 d74139_2(x[4], 2'b01, t1);
xor xor5(xor_1[3], t1[3], x[3]);
xor xor6(xor_1[2], t1[2], x[2]);
xor xor7(xor_1[1], t1[1], x[1]);
xor xor8(xor_1[0], t1[0], x[0]);
and and2(res[1], xor_1[3], xor_1[2], xor_1[1], xor_1[0]);
d74139 d74139_3(x[4], 2'b10, t2);
xor xor9(xor_2[3], t2[3], x[3]);
xor xor10(xor_2[2], t2[2], x[2]);
xor xor11(xor_2[1], t2[1], x[1]);
```

```
xor xor12(xor_2[0], t2[0], x[0]);
and and3(res[2], xor_2[3], xor_2[2], xor_2[1], xor_2[0]);
d74139 d74139_4(x[4], 2'b11, t3);
xor xor13(xor_3[3], t3[3], x[3]);
xor xor14(xor_3[2], t3[2], x[2]);
xor xor15(xor_3[1], t3[1], x[1]);
xor xor16(xor_3[0], t3[0], x[0]);
and and4(res[3], xor_3[3], xor_3[2], xor_3[1], xor_3[0]);
or or1(y, res[0], res[1], res[2], res[3]);
endmodule
```

Use the 16-to-1 MUX and external gates

```
`timescale 1ns / 1ps
module a2t4_mux(x, y);
input [4: 0] x;
output y;
wire not_x4;
not not1(not_x4, x[4]);
wire res1;
nor nor1(res1, not_x4, x[3], x[2], x[1], x[0]);
wire item;
mux16to1 mux16to1_1(16'b0000000100010110, x[3: 0], item);
wire res2;
and and1(res2, item, not_x4);
or or1(y, res1, res2);
endmodule
```

SIMULATION

Describe how you build the test bench and do the simulation.

- *Using Verilog (provide the Verilog code)*

  Test bench for decoder and 16-to-1 MUX

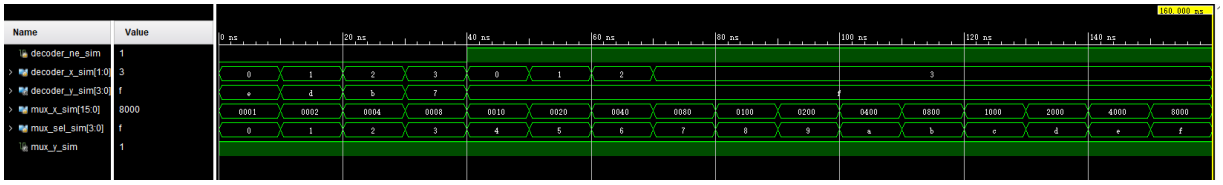
```
`timescale 1ns / 1ps
module decoder_mux_sim();
reg decoder_ne_sim;
reg [1: 0] decoder_x_sim;
wire [3: 0] decoder_y_sim;
d74139 u(decoder_ne_sim, decoder_x_sim, decoder_y_sim);
reg [15: 0] mux_x_sim;
reg [3: 0] mux_sel_sim;
wire mux_y_sim;
mux16to1 v(mux_x_sim, mux_sel_sim, mux_y_sim);
initial begin
    decoder_ne_sim = 1'b0;
    decoder_x_sim = 2'b0;
    while ({decoder_ne_sim, decoder_x_sim} < 3'b111)
    begin
        # 10 {decoder_ne_sim, decoder_x_sim} = {decoder_ne_sim,
decoder_x_sim} + 1;
    end
end
initial begin
    mux_sel_sim = 4'b0;
    mux_x_sim = 16'b0;
    mux_x_sim[mux_sel_sim] = 1'b1;
    while (mux_sel_sim < 4'b1111)
    begin
        #10 mux_sel_sim = mux_sel_sim + 1; mux_x_sim = 16'b0;
mux_x_sim[mux_sel_sim] = 1'b1;
    end
    #10 $finish();
end
endmodule
```

Test bench for the circuit

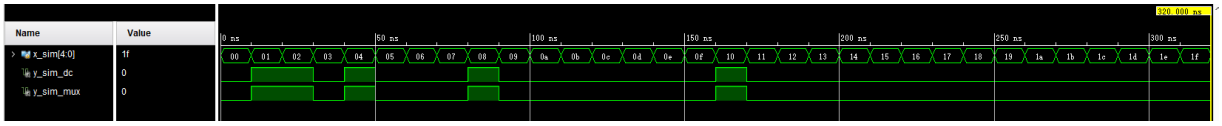
```
module a2t4_sim();
reg [4: 0] x_sim;
wire y_sim_dc, y_sim_mux;
a2t4_dc u(x_sim, y_sim_dc);
a2t4_mux v(x_sim, y_sim_mux);
initial begin
    x_sim = 5'b0;
    while (x_sim < 5'b11111)
    begin
        # 10 x_sim = x_sim + 1;
    end
    # 10 $finish();
end
endmodule
```

- *Wave form of simulation result (provide screen shots)*

Result of test bench for decoder and 16-to-1 MUX



Result of test bench for the circuit



- *The description on whether the simulation result is same as the truth-table, is the function of the design meet the expectation*

Analysis for result of test bench for decoder and 16-to-1 MUX

如上图所示，我们遍历 2-4 译码器的所有状态作为仿真输入，即 decoder_ne_sim 表示仿真使能端、decoder_x_sim 表示输入端，获得仿真输出 decoder_y_sim，我们采用作业补充说明中指出的办法不遍历 16-to-1 数据选择器所有输入状态，而是选择作业补充说明中指出的 16 个特殊输入作为测试，mux_x_sim 表示信号输入端，mul_sel_sim 表示信号选择端，上述两个作为仿真输入，共 16 个仿真输入，mul_y_sim 表示仿真输出端，可以通过波形图判断，结果均正确，可以说明符合预期结果。

Analysis for result of test bench for the circuit

如上图所示，任务要求我们对于给出的 5-bit 输入 x，输出 1-bit 结果 y，表示 x 是否是独热码（二进制表示中有且仅有一个 1，则为 1 否则为 0）。仿真文件中 x_sim 是一个 5-bit 二进制数表示仿真输入，从 5'b00000 依次增加到 5'b11111，进行两组仿真，分别得到结果 y_sim_dc 和 y_sim_mux，前者表示使用译码器设计电路进行仿真结果，后者表示使用数据选择器设计电路进行仿真结果。可以通过波形图判断，结果均正确，可以说明符合预期结果。