

# 《数字逻辑》

(第4次实验：在Logisim和FPGA开发板上实现同步时序逻辑电路)

厦门大学信息学院软件工程系 曾文华

2024年10月21日

# 目录

- **第一部分：在Logisim上实现同步时序逻辑电路**
  - （一）在Logisim上实现教材第5章的例题
  - （二）在Logisim上实现教材第5章的习题
- **第二部分：在FPGA开发板上实现同步时序逻辑电路**
  - （一）在FPGA开发板上实现教材第5章的例题
  - （二）在FPGA开发板上实现教材第5章的习题

# 第一部分：在Logisim上实现同步时序逻辑电路

请打开设计文件“第5章例题电路的实现.circ”

# (一) 在Logisim上实现教材第5章的例题

## • 1、（验证实验）例题5.1的实现

- 在Logisim上实现例题5.1的“2位二进制数可逆计数器”，该电路的输入为x，状态为 $y_2$ 、 $y_1$ ，CP为时钟信号。
- 验证步骤：
  - 在Logisim中打开“第5章例题电路的实现.circ”中的“例题5.1”电路。
  - 按Ctrl+R，置x=0；然后按时钟CP按钮，观察 $y_2$ 、 $y_1$ 是不是按照00、01、10、11、00……变化（每按1次时钟按钮，变化1次，加1计数器）。
  - 再按Ctrl+R，置x=1；然后按时钟CP按钮，观察 $y_2$ 、 $y_1$ 是不是按照00、11、10、01、00……变化（每按1次时钟按钮，变化1次，减1计数器）。

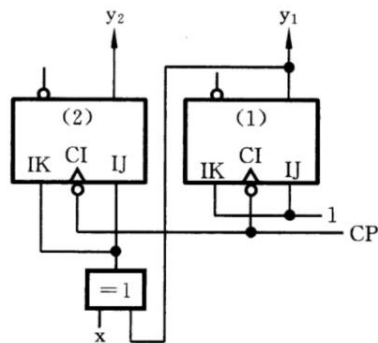
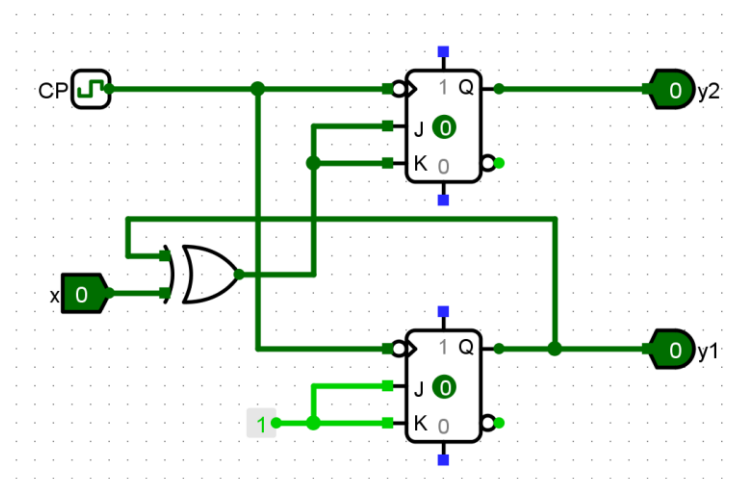


图 5.8 同步时序逻辑电路



## • 2、（验证实验）例题5.2的实现

- 在Logisim上实现例题5.2的“101”序列检测器，该电路的输入为x，状态为 $y_2$ 、 $y_1$ ，输出为Z，CP为时钟信号。

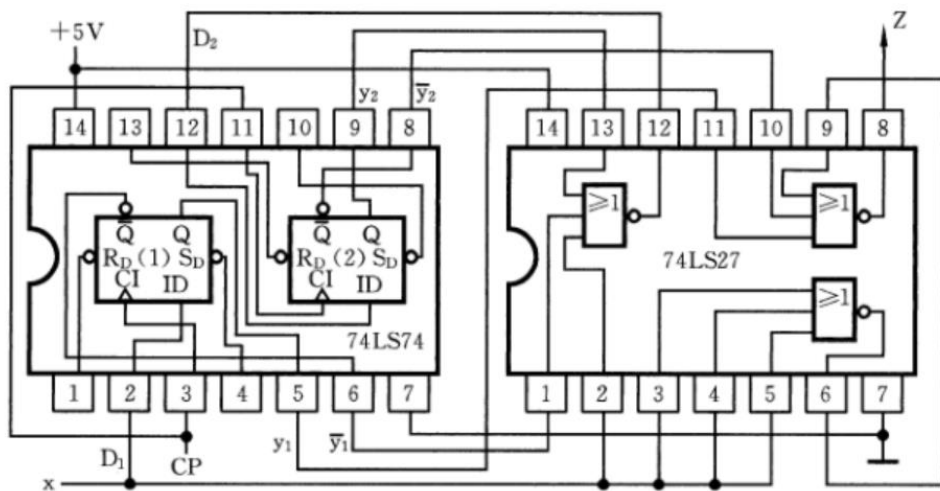
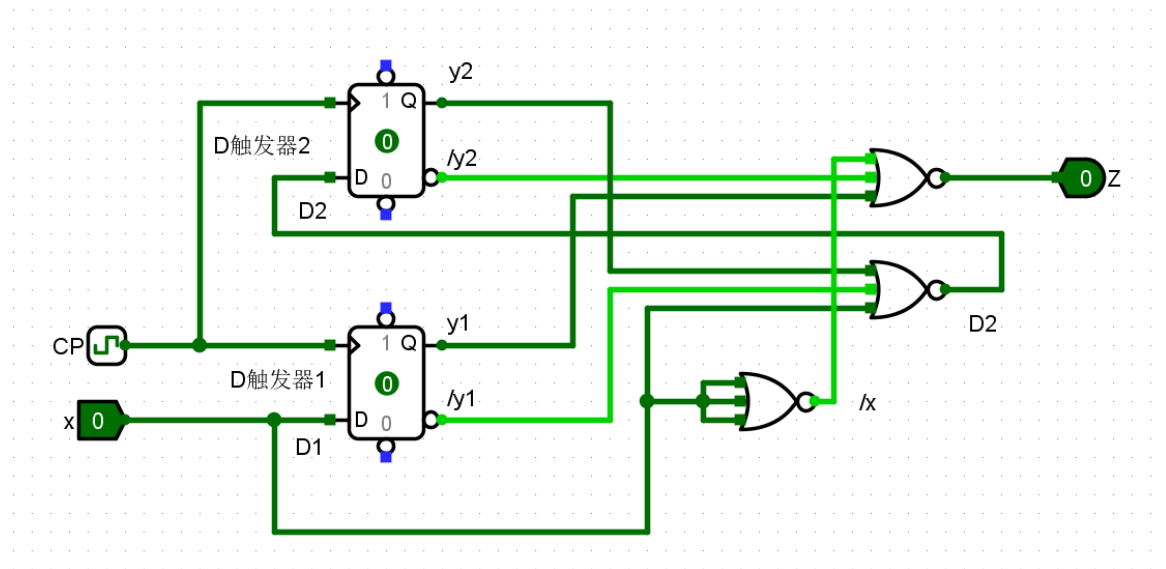
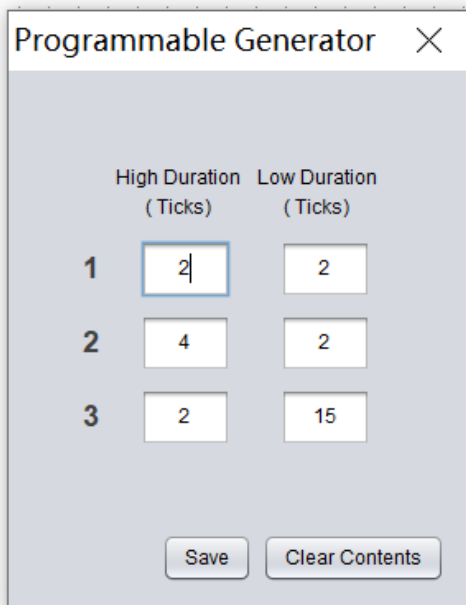


图 5.11 同步时序逻辑芯片连线图



- 验证步骤:

- 在Logisim中打开“第5章例题电路的实现.circ”中的“例题5.2的仿真”电路，观看仿真电路的函数发生器的参数是否正确？
- 然后按Ctrl+R，再按Ctrl+K，观察输入x、状态 ( $y_2$ 、 $y_1$ )、输出Z的波形是否正确？



CP	1	2	3	4	5	6	7	8	9
x	0	<u>1</u>	<u>0</u>	<u>1</u>	<u>1</u>	<u>0</u>	<u>1</u>	0	0
$y_2$	0	0	0	1	0	0	1	0	1
$y_1$	0	0	1	0	1	1	0	1	0
$y_2^{n+1}$	0	0	1	0	0	1	0	1	0
$y_1^{n+1}$	0	1	0	1	1	0	1	0	0
Z	0	0	0	1	0	0	1	0	0

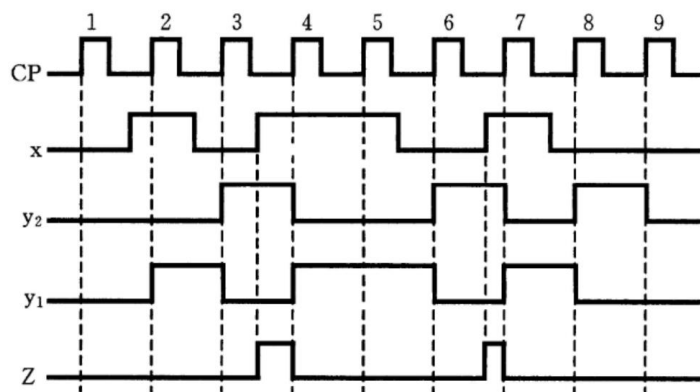
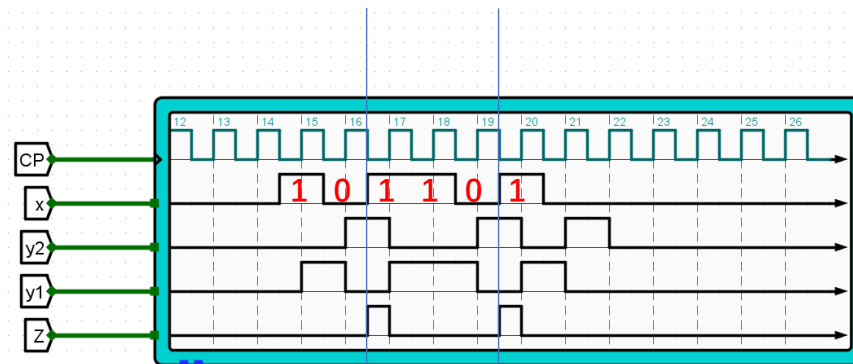
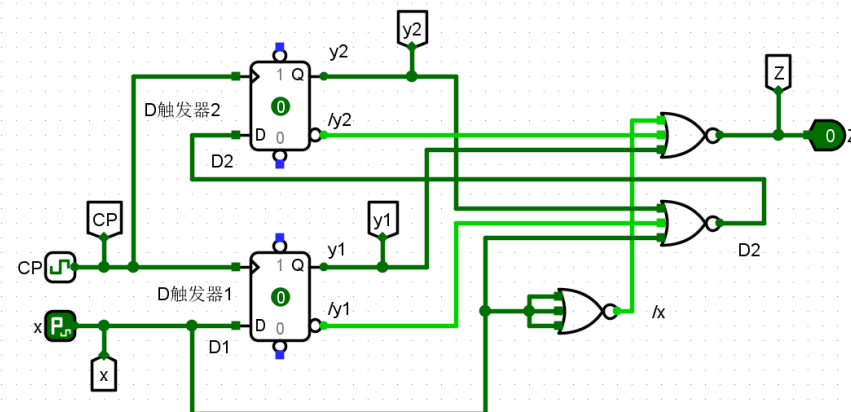


图 5.13 时间图

### “101”序列检测器



### • 3、（验证实验）例题5.3的实现

- 在Logisim上实现例题5.3的**串行加法器**，该电路的输入为 $x_1$ 、 $x_2$ ，状态为 $y$ ，输出为 $Z$ ，CP为时钟信号。

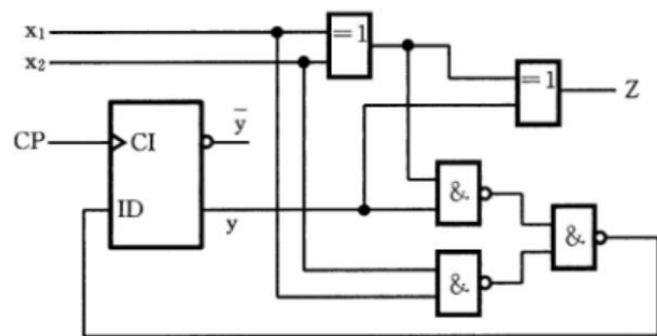
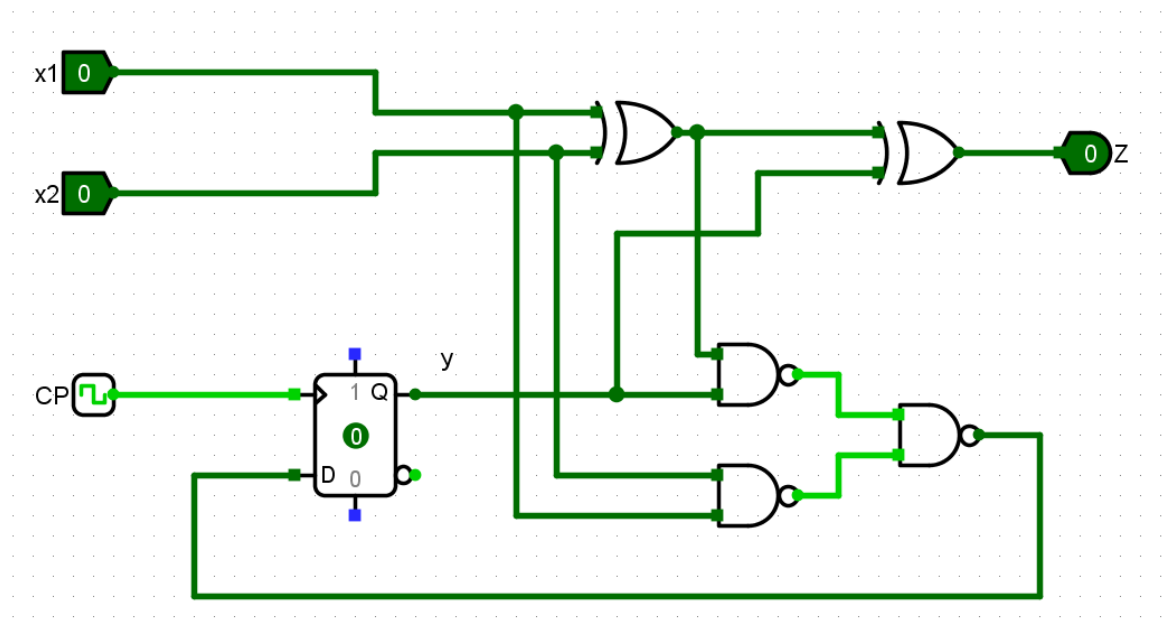
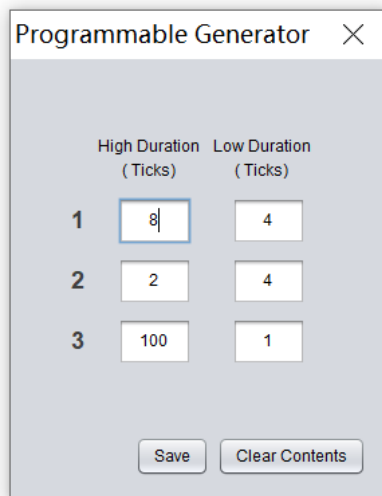


图 5.14 逻辑电路

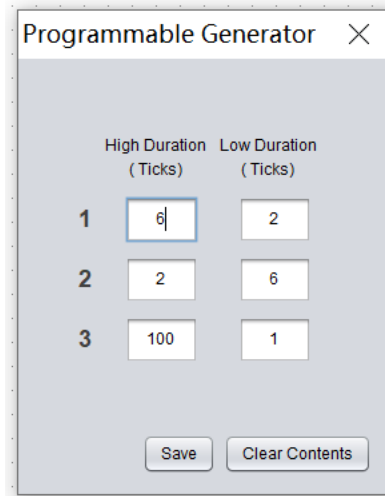


## 验证步骤:

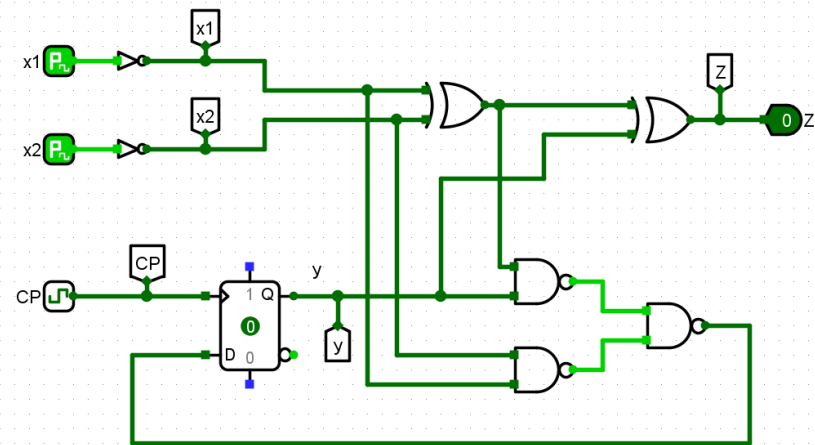
- 在Logisim中打开“第5章例题电路的实现.circ”中的“例题5.3的仿真”电路，观看仿真电路的函数发生器的参数是否正确？
- 然后按Ctrl+R，再按Ctrl+K，观察输入 $x_1$ 、 $x_2$ ，状态 $y$ ，输出 $Z$ 的波形是否正确？



x1函数发生器的参数



x2函数发生器的参数



时钟节拍	1	2	3	4	5	6	7	8
输入 $x_1$	0	0	1	1	0	1	1	0
输入 $x_2$	0	1	0	1	1	1	0	0
状态 $y$	0	0	0	0	1	1	1	1
输出 $Z$	0	1	1	0	0	1	0	1

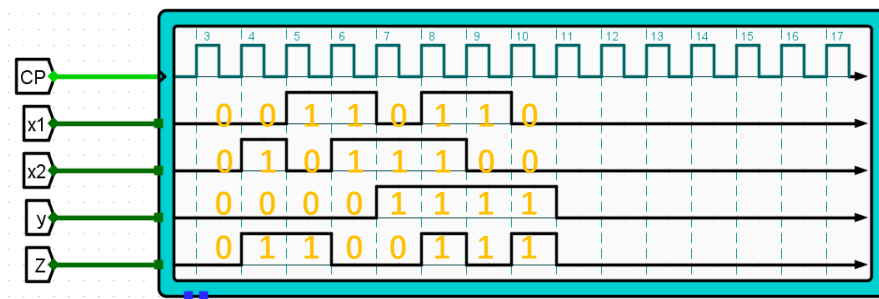
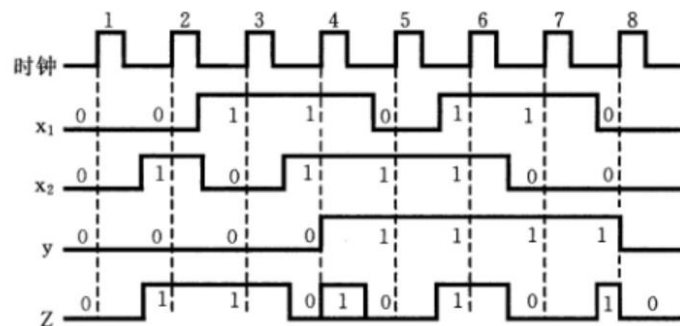


图 5.16 时间图

$x_1=0011\ 0110$ ,  $A=0110\ 1100=6CH$      $x_2=0101\ 1100$ ,  $B=0011\ 1010=3AH$      $S=A+B=6CH+3AH=A6H=1010\ 0110$ ,  $Z=0110\ 0101$



#### • 4、（验证实验）例题5.4的实现

- 在Logisim上实现例题5.4的“3位串行输入移位寄存器”，该电路的输入为 $x$ ，状态为 $y_3$ 、 $y_2$ 、 $y_1$ ，输出为 $Z$ ，CP为时钟信号。

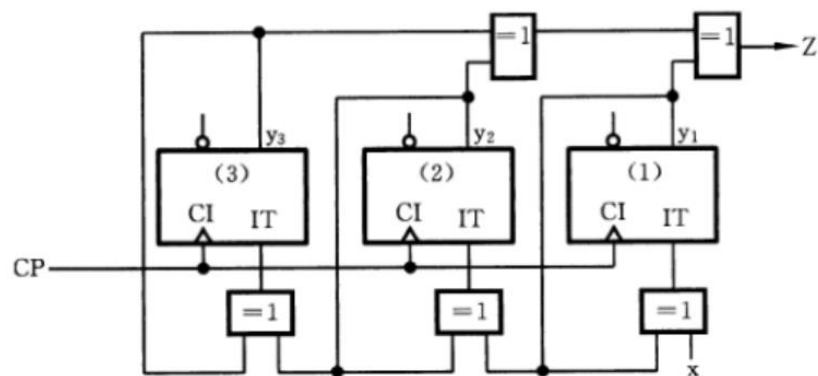
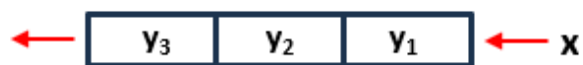
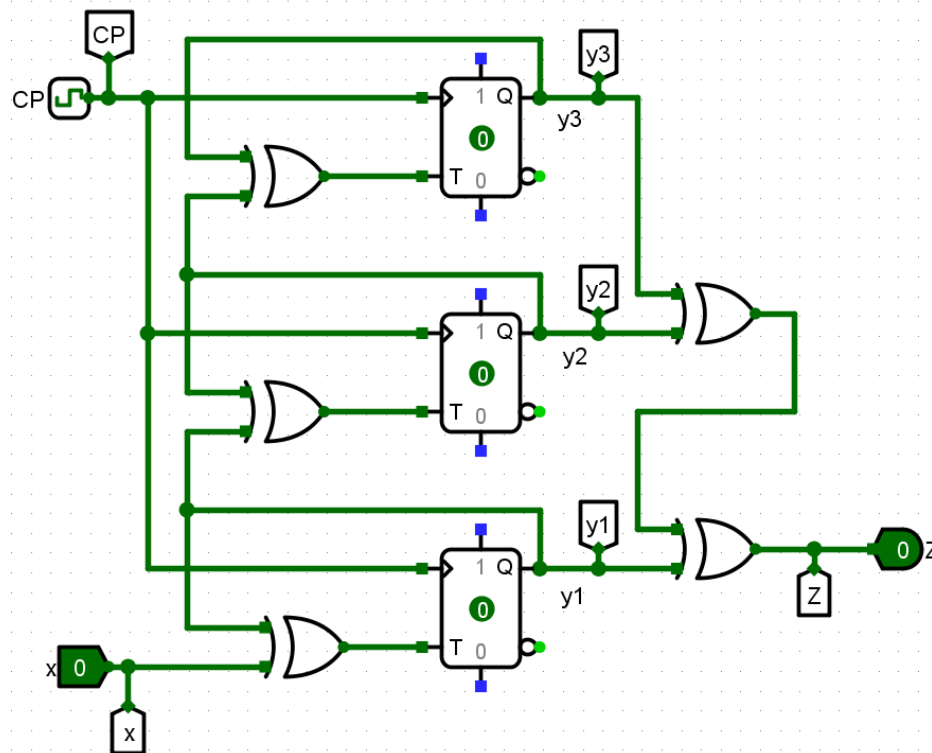
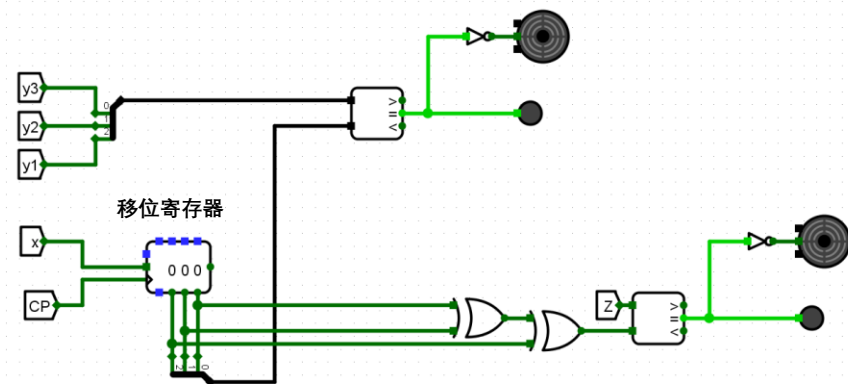
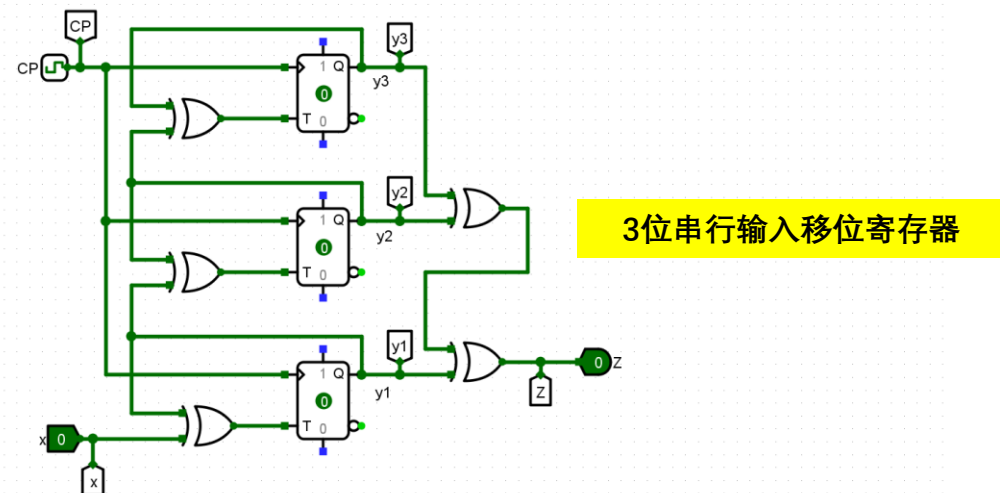
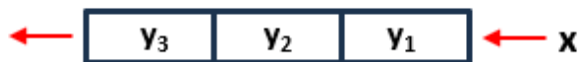


图 5.17 逻辑电路



- 验证步骤:

- 在Logisim中打开“第5章例题电路的实现.circ”中的“例题5.4”电路，按Ctrl+R。
- 置 $x=1$ ，按时钟CP按钮，观察状态  $(y_3、y_2、y_1)$ 、输出Z的变化。
- 再置 $x=0$ ，按时钟CP按钮，观察状态  $(y_3、y_2、y_1)$ 、输出Z的变化。



## • 5、（验证实验）例题5.10的实现

- 在Logisim上实现例题5.10的电路（有2个电路），该电路的输入为 $x$ ，状态为 $y_2$ 、 $y_1$ ，输出为 $Z$ ，CP为时钟信号。

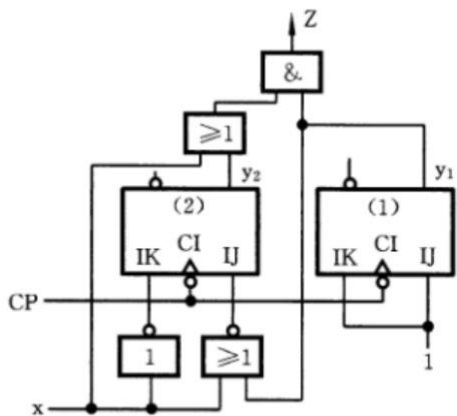


图 5.30 逻辑电路

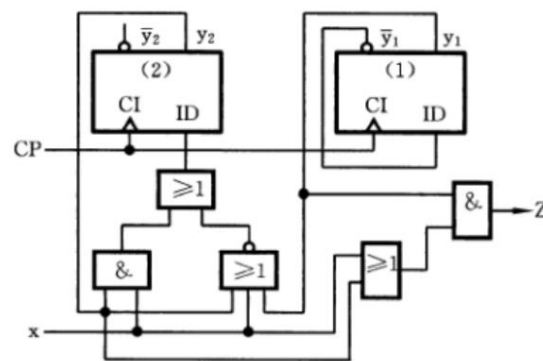
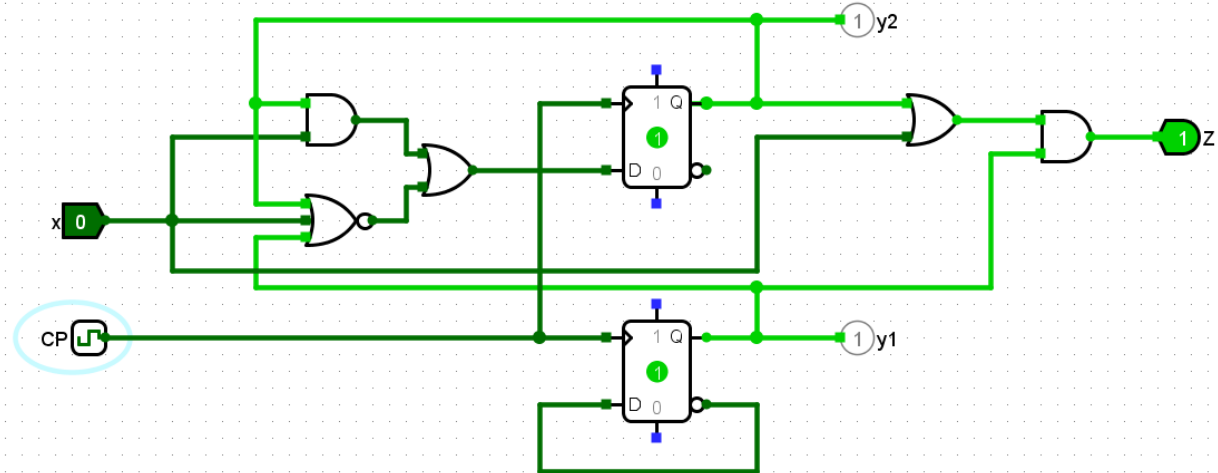
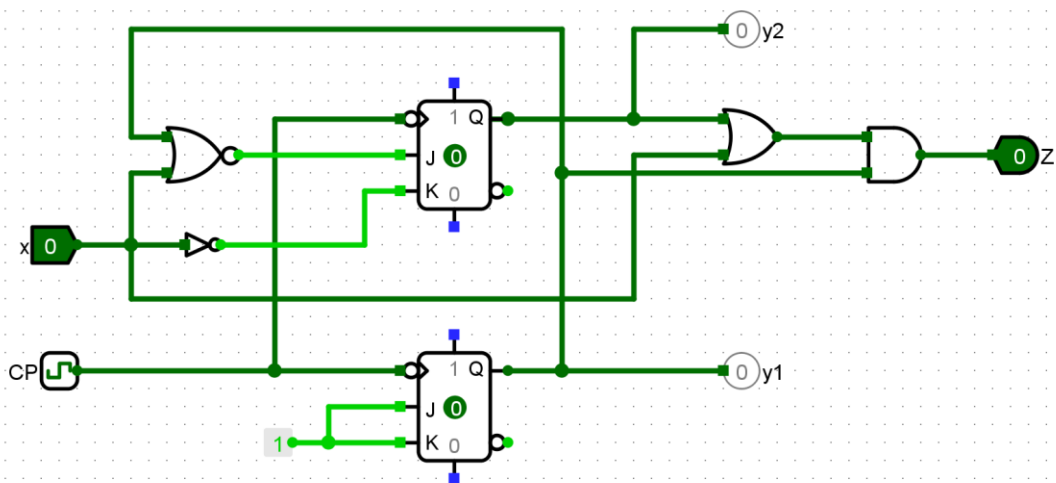


图 5.32 采用 D 触发器的逻辑电路

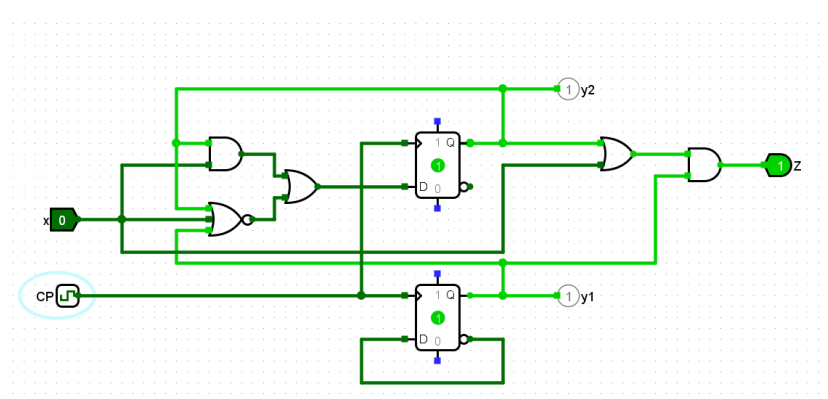
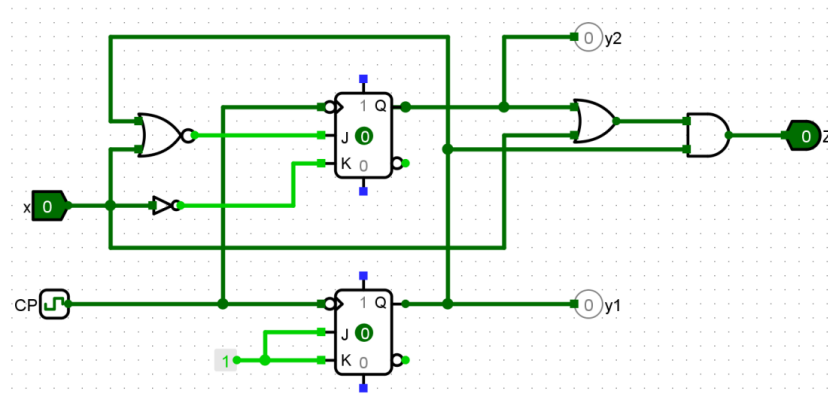


• **验证步骤**（2个电路的验证步骤一样）：

- 在Logisim中打开“第5章例题电路的实现.circ”中的“**例题5.10 (1)**”（**例题5.10 (2)**）电路，按Ctrl+R。
- 如果 $y_2y_1=00$ ，置 $x=0$ ，按时钟CP按钮，则 $y_2y_1=11$ 、 $Z=0$ 。
- 如果 $y_2y_1=00$ ，置 $x=1$ ，按时钟CP按钮，则 $y_2y_1=01$ 、 $Z=0$ 。
- 如果 $y_2y_1=01$ ，置 $x=0$ ，按时钟CP按钮，则 $y_2y_1=00$ 、 $Z=0$ 。
- 如果 $y_2y_1=01$ ，置 $x=1$ ，按时钟CP按钮，则 $y_2y_1=00$ 、 $Z=1$ 。
- 如果 $y_2y_1=11$ ，置 $x=0$ ，按时钟CP按钮，则 $y_2y_1=00$ 、 $Z=1$ 。
- 如果 $y_2y_1=11$ ，置 $x=1$ ，按时钟CP按钮，则 $y_2y_1=10$ 、 $Z=1$ 。
- 如果 $y_2y_1=10$ ，置 $x=0$ ，按时钟CP按钮，则 $y_2y_1=01$ 、 $Z=0$ 。
- 如果 $y_2y_1=10$ ，置 $x=1$ ，按时钟CP按钮，则 $y_2y_1=11$ 、 $Z=0$ 。

表 5.21 二进制状态表

现 态 $y_2 \ y_1$	次态 $y_2^{n+1}y_1^{n+1}$ /输出 Z	
	$x=0$	$x=1$
0 0	11/0	01/0
0 1	00/0	00/1
1 1	00/1	10/1
1 0	01/0	11/0



## • 6、（验证实验）例题5.14的实现

- 在Logisim上实现例题5.14的“2位二进制减1计数器”，该电路的输入为x，状态为 $y_2$ 、 $y_1$ ，输出为Z，CP为时钟信号。

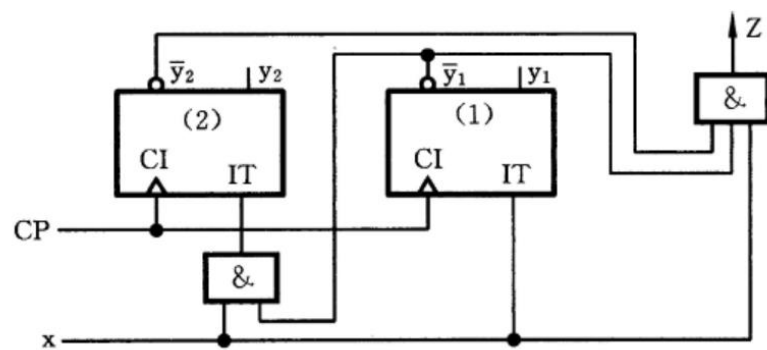
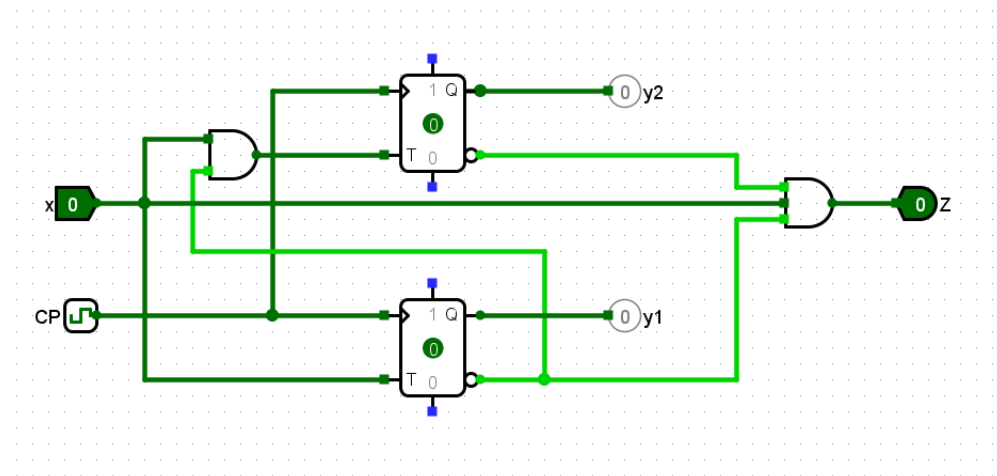


图 5.42 逻辑电路

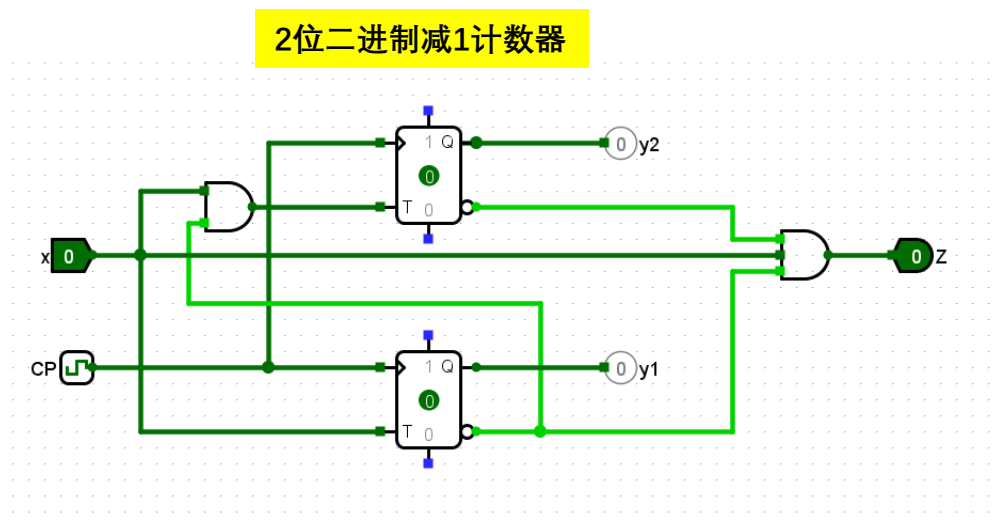


- **验证步骤:**

- 在Logisim中打开“第5章例题电路的实现.circ”中的“**例题5.14**”电路，按Ctrl+R。
- 当x=0时，按时钟CP按钮，y2、y2不变。
- 当x=1时，按时钟CP按钮，y2、y1按照00、11、10、01、00……的规律变化（2位二进制减1计数器）。

表 5.32 状态表

现 态		次态 $y_2^{n+1}y_1^{n+1}$ / 输出 Z	
$y_2$	$y_1$	x=0	x=1
0	0	00/0	11/1
0	1	01/0	00/0
1	1	11/0	10/0
1	0	10/0	01/0



## • 7、（验证实验）例题5.15的实现

- 在Logisim上实现例题5.15的“双向移位寄存器”，该电路的输入为 $x_2$ 、 $x_1$ ，状态为 $y_2$ 、 $y_1$ ，CP为时钟信号。

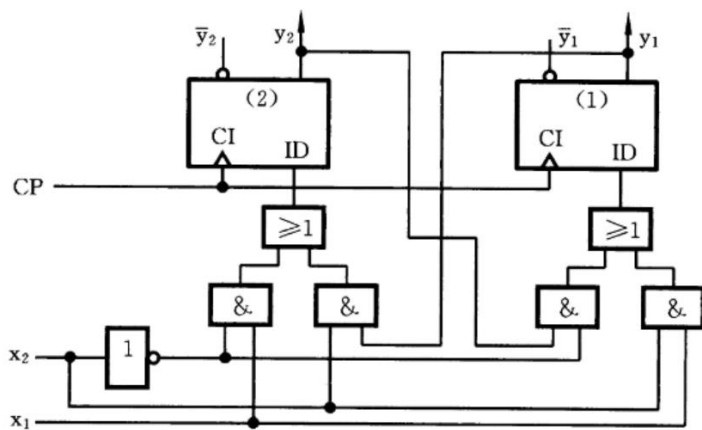
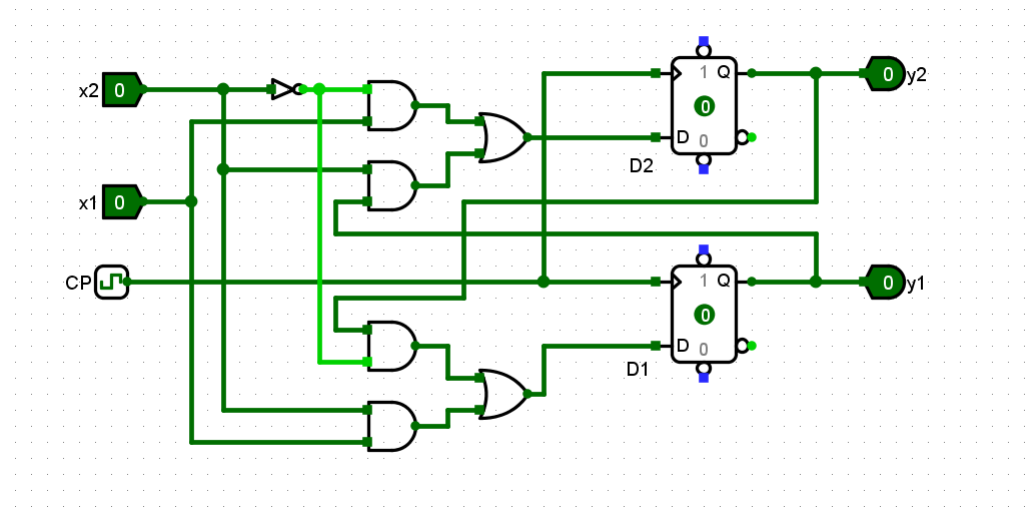
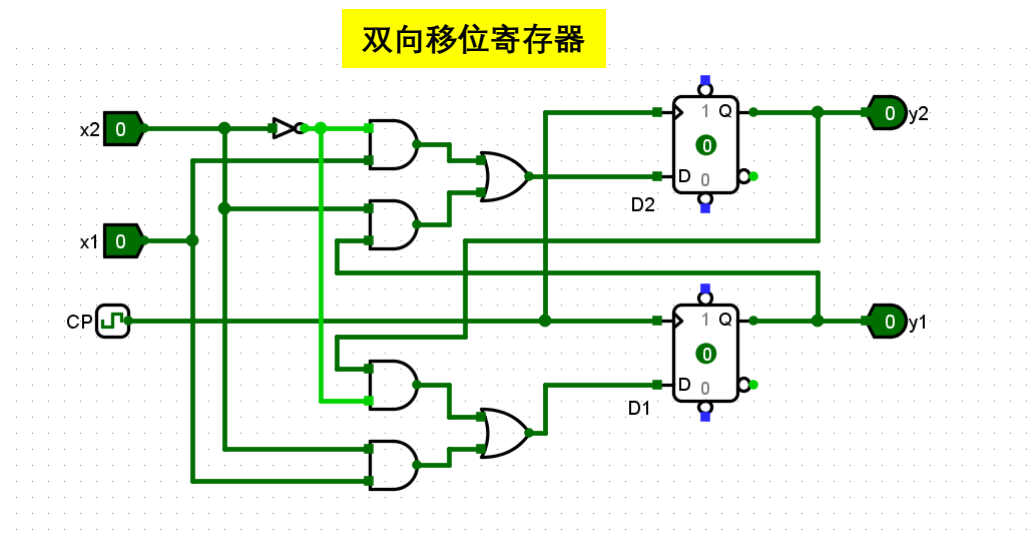


图 5.45 逻辑电路图



- 验证步骤:

- 在Logisim中打开“第5章例题电路的实现.circ”中的“例题5.15”电路，按Ctrl+R。
- 置 $x_2=1$ ,  $x_1=1$ ；按CP，可以看到 $y_1$ 、 $y_2$ 先后置1，表示 $x_1$ 往寄存器低位串行送数，寄存器中的数据“1”从低位移向高位；
- 再置 $x_1=0$ ，按CP，可以看到 $y_1$ 、 $y_2$ 先后置0，表示 $x_1$ 往寄存器低位串行送数，寄存器中的数据“0”从低位移向高位；
- 置 $x_2=0$ ,  $x_1=1$ ，按CP，可以看到 $y_2$ 、 $y_1$ 先后置1，表示 $x_1$ 往寄存器高位串行送数，寄存器中的数据“1”从高位移向低位；
- 再置 $x_1=0$ ，按CP，可以看到 $y_2$ 、 $y_1$ 先后置0，表示 $x_1$ 往寄存器高位串行送数，寄存器中的数据“0”从高位移向低位。





## • 8、（验证实验）例题5.16的实现

- 在Logisim上实现例题5.16的“101”序列检测器，该电路的输入为 $x$ ，状态为 $y_2$ 、 $y_1$ ，输出为 $Z$ ，CP为时钟信号。

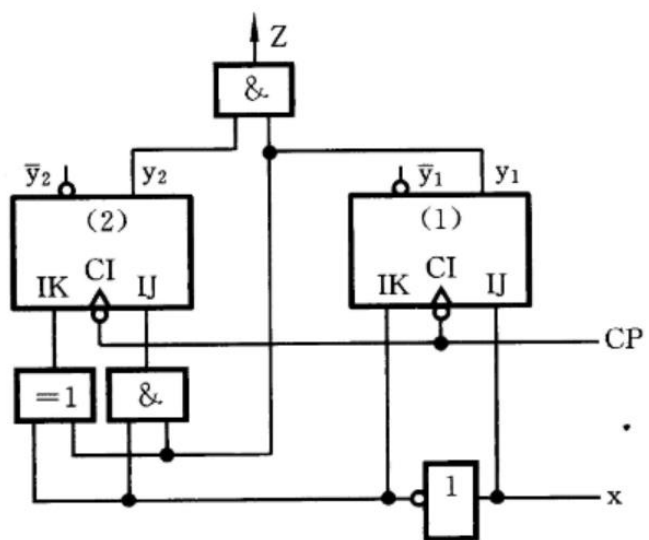
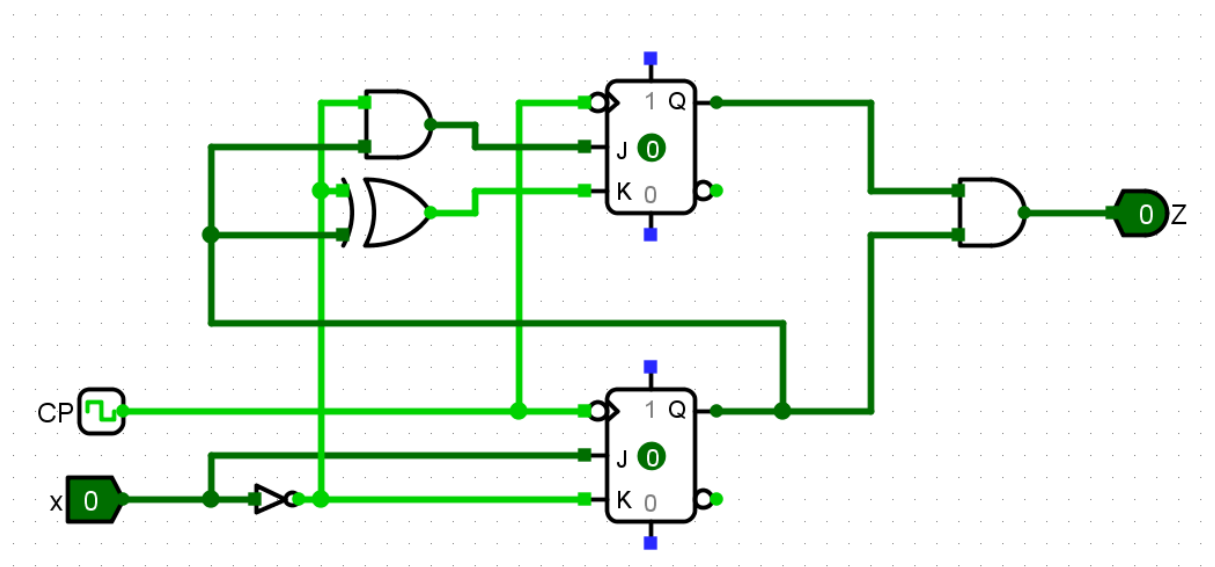
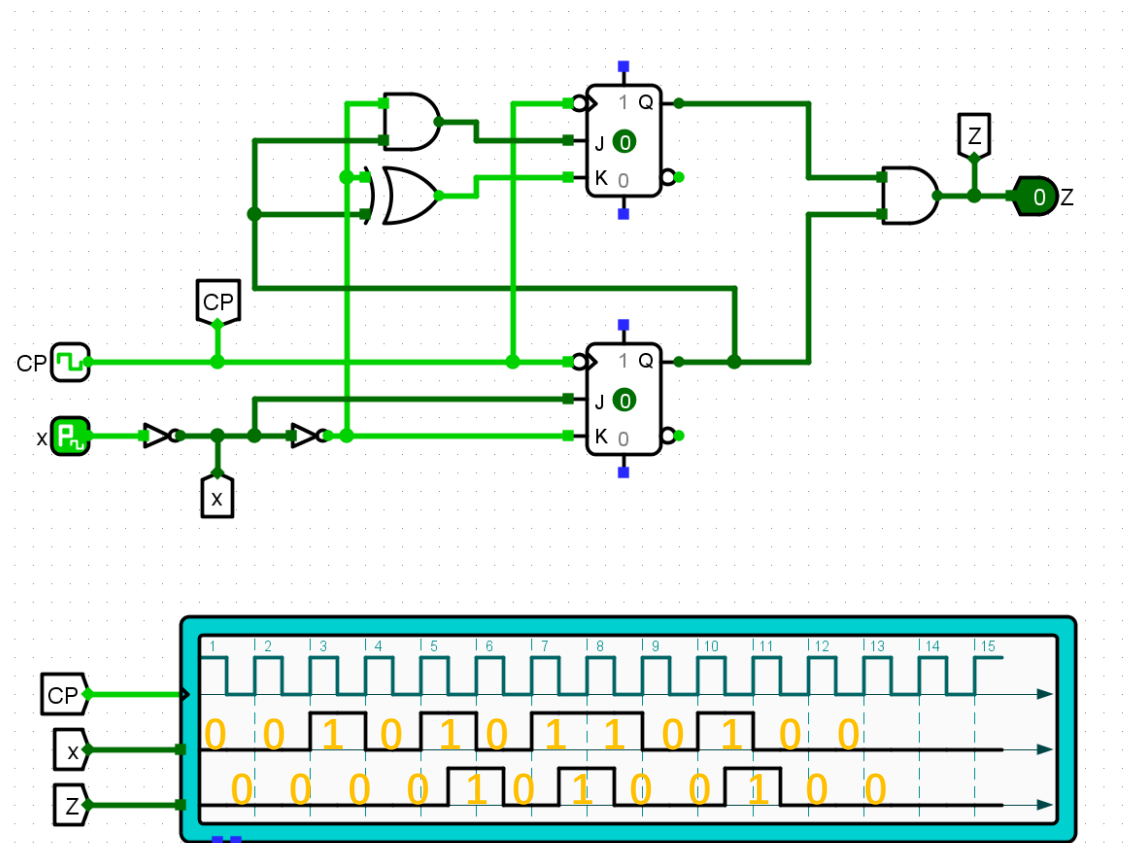
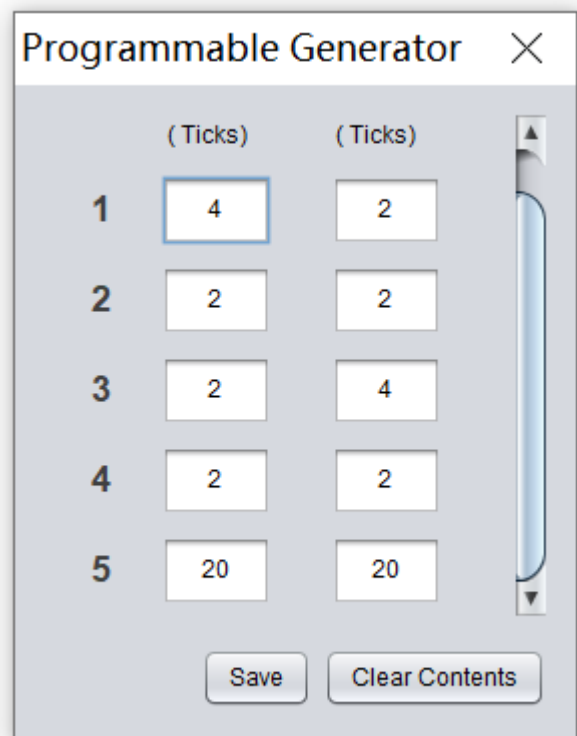


图 5.48 逻辑电路



- 验证步骤:

- 在Logisim中打开“第5章例题电路的实现.circ”中的“例题5.16的仿真”电路，观看仿真电路的函数发生器的参数是否正确？
- 然后按Ctrl+R，再按Ctrl+K，观察输入x、输出Z的波形是否正确？



输入 x	0	0	<u>1</u>	0	<u>1</u>	<u>0</u>	<u>1</u>	<u>1</u>	0	<u>1</u>	0	0
输出 Z	0	0	0	0	1	0	1	0	0	1	0	0
					↑		↑			↑		

## • 9、（验证实验）例题5.17的实现

- 在Logisim上实现例题5.17的“奇偶检测器”，该电路的输入为 $x$ ，状态为 $y_3$ 、 $y_2$ 、 $y_1$ ，输出为 $Z$ ，CP为时钟信号。

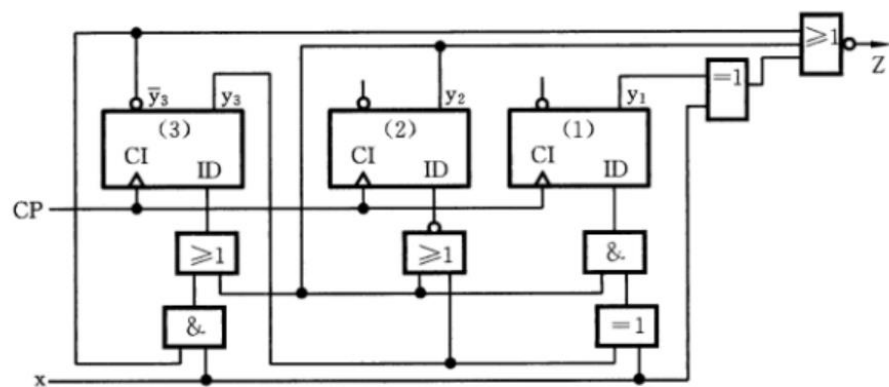
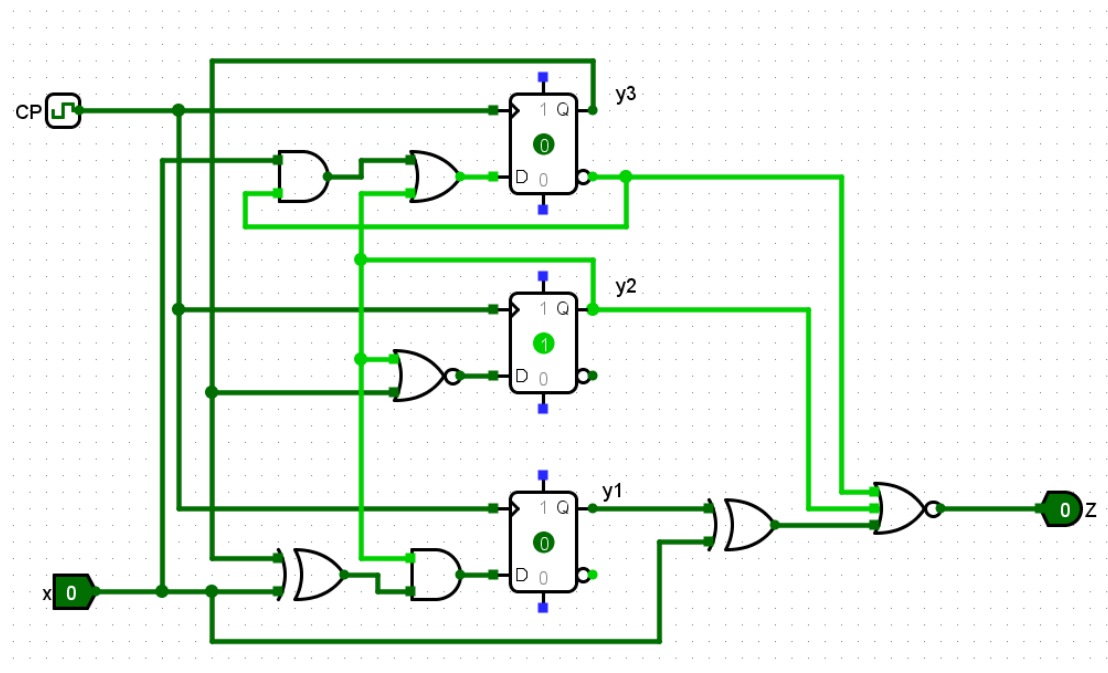
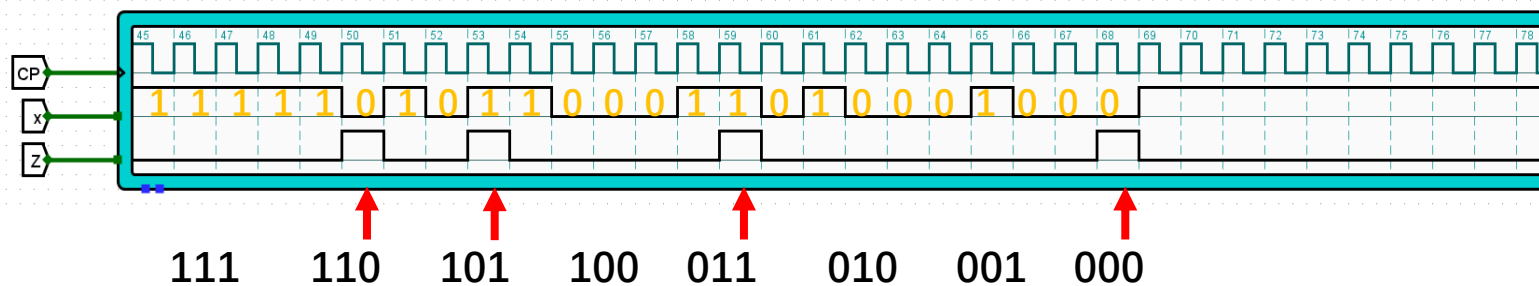
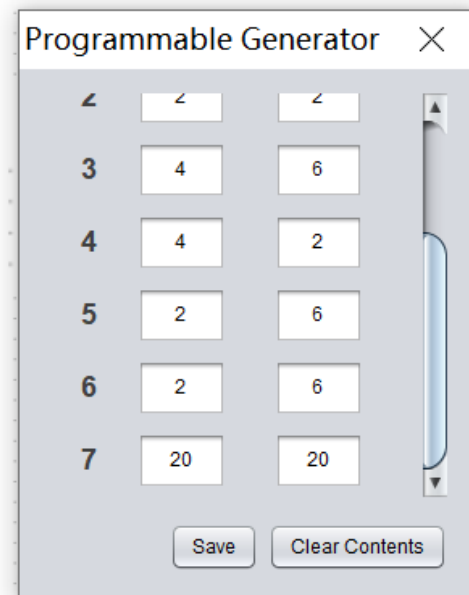
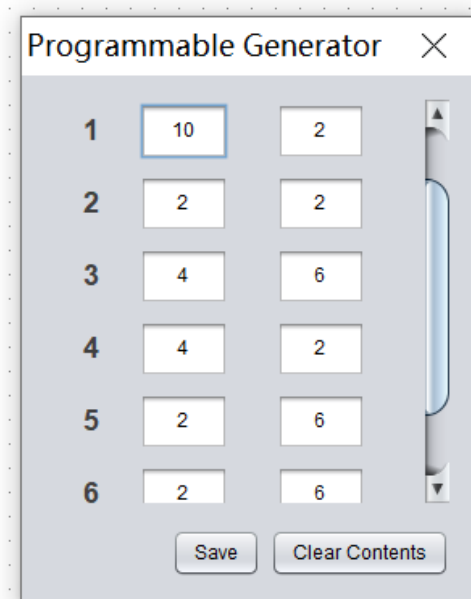
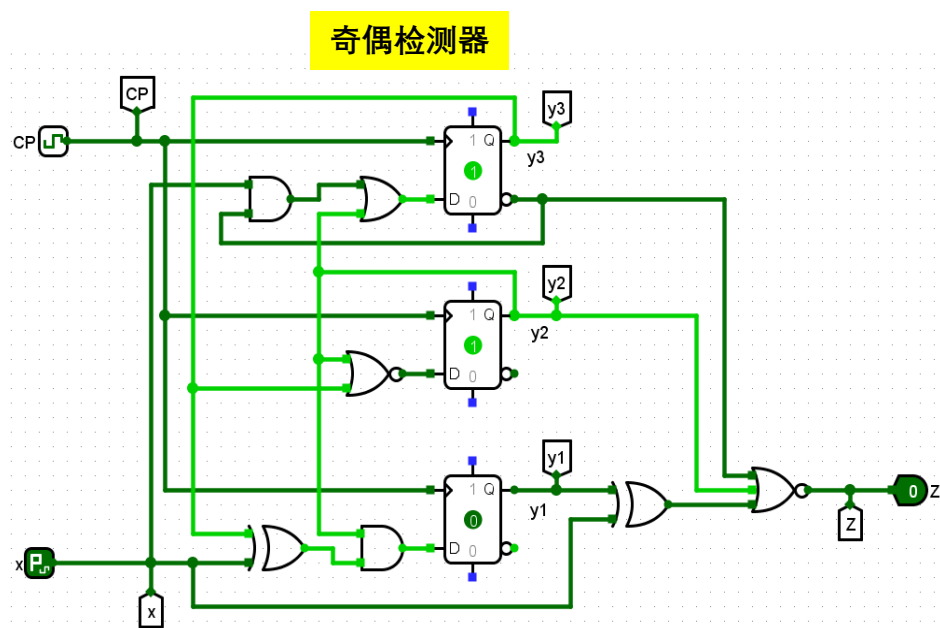


图 5.53 逻辑电路



- 验证步骤:

- 在Logisim中打开“第5章例题电路的实现.circ”中的“例题5.17的仿真”电路，观看仿真电路的函数发生器的参数是否正确?
- 然后按Ctrl+R，再按Ctrl+K，观察输入x、输出Z的波形是否正确? (当检测到3位数中1的个数为偶数个时，输出=1)



## (二) 在Logisim上实现教材第5章的习题

- 1、（设计实验，课后完成）习题5.4的实现
  - 请在Logisim上实现习题5.4的电路（图5.55），并验证该电路的功能。

5.4 分析图 5.55 所示逻辑电路。假定电路初始状态为“00”,说明该电路逻辑功能。

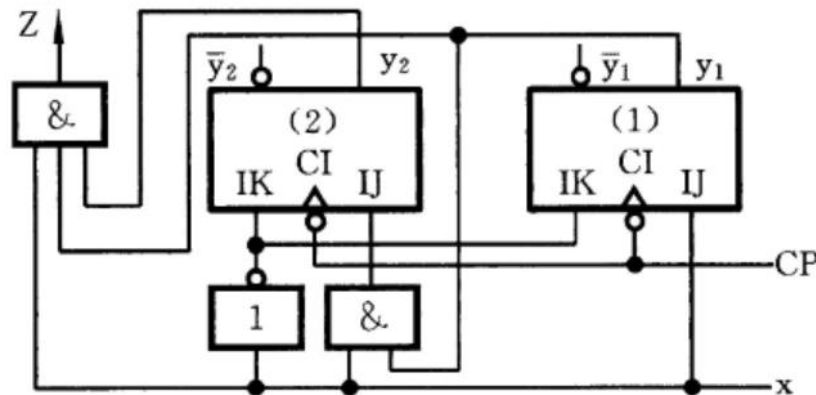


图 5.55 逻辑电路

## • 2、（设计实验，课后完成）习题5.6的实现

- 请在Logisim上实现习题5.6的电路（图5.57），并验证该电路的功能。

5.6 分析图 5.57 所示逻辑电路,说明该电路功能。

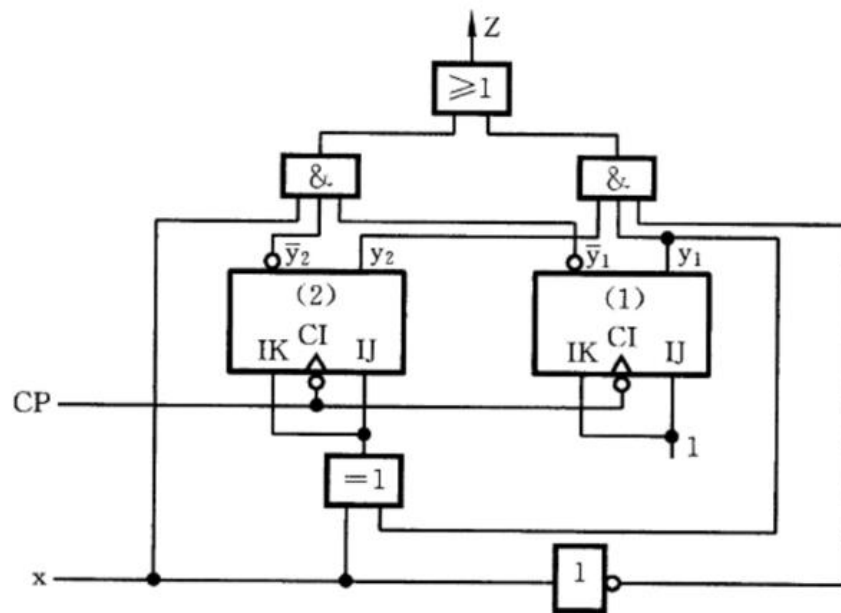


图 5.57 逻辑电路

- 3、（设计实验，课后完成）习题5.8的实现

- 请在Logisim上实现习题5.8的电路（代码检测器），并验证该电路的功能。

5.8 设计一个代码检测器,该电路从输入端  $x$  串行输入余 3 码(先低位后高位),当出现非法数字时,电路输出  $Z$  为 1,否则输出为 0。试作出 Mealy 型状态图。

#### • 4、（设计实验，课后完成）习题5.12的实现

- 请在Logisim上实现习题5.12的电路（有3个电路，分别对应D触发器、T触发器、J-K触发器），并验证该电路的功能。

5.12 分别用 D、T、J-K 触发器作为同步时序逻辑电路的存储元件，实现表 5.45 所示二进制状态表的功能。试写出激励函数和输出函数表达式，比较采用哪种触发器可使电路最简。

表 5.45 状态表

现 态 $y_2 \quad y_1$		次态 $y_2^{n+1} y_1^{n+1}$ / 输出 $Z$	
		$x=0$	$x=1$
0	0	01/0	10/0
0	1	11/0	10/0
1	1	10/1	01/0
1	0	00/1	11/1



- 5、（设计实验，课后完成）习题5.15的实现

- 请在Logisim上实现习题5.15的电路（8421码十进制加1计数器），并验证该电路的功能。

5.15 用 T 触发器作为存储元件,设计一个 8421 码十进制加 1 计数器。

- 6、（选做实验）习题5.5的实现

- 请在Logisim上实现习题5.5的电路（图5.56），并验证该电路的功能。

5.5 分析图 5.56 所示逻辑电路,说明该电路功能。

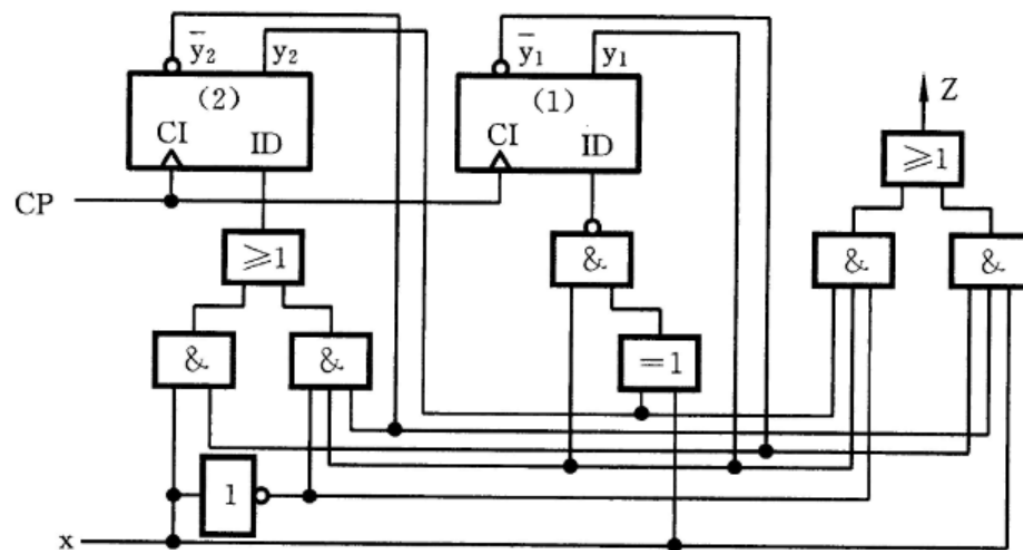


图 5.56 逻辑电路

## • 7、（选做实验）习题5.13的实现

- 请在Logisim上实现习题5.13的电路（有2个电路，分别对应D触发器、J-K触发器），并验证该电路的功能。

5.13 已知某同步时序逻辑电路的激励函数和输出函数表达式为

$$D_2 = \bar{x}y_2 + xy_2\bar{y}_1$$

$$D_1 = \bar{x}y_2 + y_2\bar{y}_1 + x\bar{y}_2y_1$$

$$Z = y_2$$

- 试求出改用 J-K 触发器作为存储元件的最简电路。

## • 8、（选做实验）习题5.14的实现

- 请在Logisim上实现习题5.14的电路（**二进制数比较电路**），并验证该电路的功能。

5.14 设计一个能对两个二进制数  $X=x_1x_2\cdots x_n$  和  $Y=y_1y_2\cdots y_n$  进行比较的同步时序逻辑电路,其中, $X$ 、 $Y$  串行地输入到电路的  $x$ 、 $y$  输入端。比较从  $x_1$ 、 $y_1$  开始,依次进行到  $x_n$ 、 $y_n$ 。电路有两个输出  $Z_x$  和  $Z_y$ ,若比较结果  $X>Y$ ,则  $Z_x$  为 1, $Z_y$  为 0;若  $X<Y$ ,则  $Z_x$  为 0, $Z_y$  为 1;若  $X=Y$ ,则  $Z_x$  和  $Z_y$  都为 1。要求用尽可能少的状态数作出状态图和状态表,并用尽可能少的逻辑门和触发器(采用 J-K 触发器)实现其功能。

## 第二部分： 在FPGA开发板上实现同步时序逻辑电路

请将“第4次实验文档（发给学生）\EGO1\”目录下的所有文件拷贝到实验室电脑（或自己的电脑）的“D:\DigitalLogic\EGO1\”目录下

# (一) 在FPGA开发板上实现教材第5章的例题

## • 1、（验证实验）例题5.1的实现

- 分别用结构化描述方式和行为描述方式实现例题5.1的2位二进制数可逆计数器，该电路的输入为x，状态为 $y_2$ 、 $y_1$ ，CP为时钟信号。
- 输入x和CP为开发板上最左边的拨动开关以及S1按键，状态（ $y_2$ 、 $y_1$ ）为开发板上最左边的2个LED灯。
- 验证步骤：
  - 运行程序后，置x=0；然后按时钟CP按钮，观察 $y_2$ 、 $y_1$ 是不是按照00、01、10、11、00……变化（每按1次时钟按钮，变化1次，加1计数器）。
  - 再置x=1；然后按时钟CP按钮，观察 $y_2$ 、 $y_1$ 是不是按照00、11、10、01、00……变化（每按1次时钟按钮，变化1次，减1计数器）。

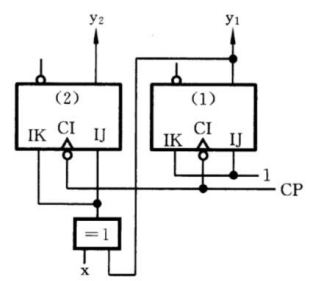
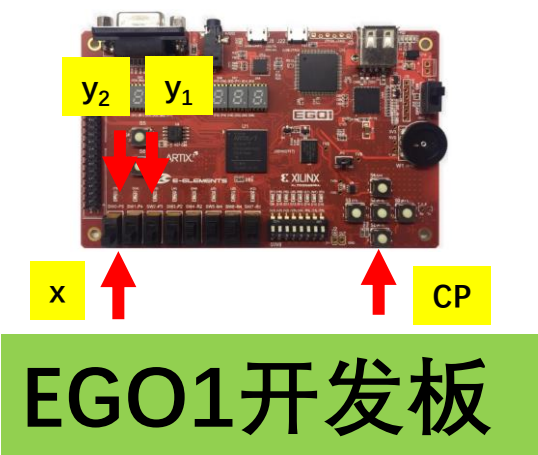


图 5.8 同步时序逻辑电路

表 5.3 次态真值表

输 入 x	现 态		激励函数						次 态	
	$y_2$	$y_1$	$J_2$	$K_2$	$J_1$	$K_1$			$y_2^{n+1}$	$y_1^{n+1}$
0	0	0	0	0	1	1			0	1
0	0	1	1	1	1	1			1	0
0	1	0	0	0	1	1			1	1
0	1	1	1	1	1	1			0	0
1	0	0	1	1	1	1			1	1
1	0	1	0	0	1	1			0	0
1	1	0	1	1	1	1			0	1
1	1	1	0	0	1	1			1	0

//结构化描述方式 (子模块采用行为描述方式) 实现例5.1的2位二进制数可逆计数器, 开发板」

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

```
module xor_gate(
    input a,
    input b,
    output f
);
```

```
    reg y;
    always @(*) //行为描述方式
    begin
        y <= a ^ b;
    end
    assign f = y;
endmodule
```

```
module jk_flip_flop(
    input j, k,
    input cp,
    output q, qn
);
```

```
    reg y;
    always @(negedge cp) //行为描述方式
    begin
        case({j,k})
            1: y <= 0;
            2: y <= 1;
            3: y <= ~y;
        endcase
        assign q = y;
        assign qn = ~y;
    end
endmodule
```

```
module example_5_1(
    input sw_pin[7:0],
    input btn_1,
    output [15:0] led_pin
);
```

//8个拨动开关  
//1个按钮  
//16个led灯

```
    wire y1, y1n, y2, y2n, k2j2;
```

```
    jk_flip_flop U1(j(1),k(1),cp(btn_1),q(y1),qn(y1n));
    xor_gate U2(a(sw_pin[0]),b(y1),f(k2j2));
    jk_flip_flop U3(j(k2j2),k(k2j2),cp(btn_1),q(y2),qn(y2n));
    assign led_pin[0] = y2;
    assign led_pin[1] = y1;
endmodule
```

//结构化描述方式  
//结构化描述方式  
//结构化描述方式

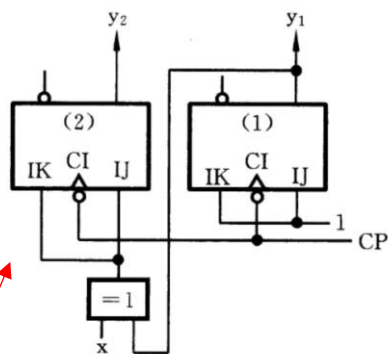


图 5.8 同步时序逻辑电路

## example\_5\_1\_EGO1

```
module jk_flip_flop(
    input j, k,
    input cp,
    output q, qn
);
```

```
    reg y;
    always @(negedge cp) //行为描述方式
    begin
        case({j,k})
            1: y <= 0;
            2: y <= 1;
            3: y <= ~y;
        endcase
        assign q = y;
        assign qn = ~y;
    end
endmodule
```

```
module example_5_1(
    input sw_pin[7:0],
    input btn_1,
    output [15:0] led_pin
);
```

//8个拨动开关  
//1个按钮  
//16个led灯

```
    wire y1, y1n, y2, y2n, k2j2;
    jk_flip_flop U1(j(1),k(1),cp(btn_1),q(y1),qn(y1n));
    xor_gate U2(a(sw_pin[0]),b(y1),f(k2j2));
    jk_flip_flop U3(j(k2j2),k(k2j2),cp(btn_1),q(y2),qn(y2n));
    assign led_pin[0] = y2;
    assign led_pin[1] = y1;
endmodule
```

//结构化描述方式  
//结构化描述方式  
//结构化描述方式

## example\_5\_1\_1\_EGO1

```

always @(negedge btn_1)           //行为描述方式

begin
    case({sw_pin[0], led_pin[0], led_pin[1]})
        0: begin led_pin[0] <= 0; led_pin[1] <= 1; end
        1: begin led_pin[0] <= 1; led_pin[1] <= 0; end
        2: begin led_pin[0] <= 1; led_pin[1] <= 1; end
        3: begin led_pin[0] <= 0; led_pin[1] <= 0; end
        4: begin led_pin[0] <= 1; led_pin[1] <= 1; end
        5: begin led_pin[0] <= 0; led_pin[1] <= 0; end
        6: begin led_pin[0] <= 0; led_pin[1] <= 1; end
        7: begin led_pin[0] <= 1; led_pin[1] <= 0; end
    endcase
end

```

example\_5\_1\_1\_EGO1.v - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

//行为描述方式实现例题5.1的2位二进制数可逆计数器，开发板上的输入（x、

`timescale 1ns / 1ps

```

module example_5_1_1(
    input sw_pin[7:0],           //8个拨动开关
    input btn_1,                 //1个按钮
    output reg [15:0] led_pin    //16个led灯
);

```

always @(negedge btn\_1) //行为描述方式

```

begin
    case({sw_pin[0], led_pin[0], led_pin[1]})
        0: begin led_pin[0] <= 0; led_pin[1] <= 1; end
        1: begin led_pin[0] <= 1; led_pin[1] <= 0; end
        2: begin led_pin[0] <= 1; led_pin[1] <= 1; end
        3: begin led_pin[0] <= 0; led_pin[1] <= 0; end
        4: begin led_pin[0] <= 1; led_pin[1] <= 1; end
        5: begin led_pin[0] <= 0; led_pin[1] <= 0; end
        6: begin led_pin[0] <= 0; led_pin[1] <= 1; end
        7: begin led_pin[0] <= 1; led_pin[1] <= 0; end
    endcase
end

```

endmodule

行为描述方式

表 5.3 次态真值表

输 入	现 态	激励函数	次 态
x	y <sub>2</sub> y <sub>1</sub>	J <sub>2</sub> K <sub>2</sub> J <sub>1</sub> K <sub>1</sub>	y <sub>2</sub> <sup>n+1</sup> y <sub>1</sub> <sup>n+1</sup>
0	0 0	0 0 1 1	0 1
0	0 1	1 1 1 1	1 0
0	1 0	0 0 1 1	1 1
0	1 1	1 1 1 1	0 0
1	0 0	1 1 1 1	1 1
1	0 1	0 0 1 1	0 0
1	1 0	1 1 1 1	0 1
1	1 1	0 0 1 1	1 0



```
module jk_flip_flop(  
    input j, k,  
    input cp,  
    output q, qn  
);  
    reg y;  
    always @(negedge cp)           //行为描述方式  
    begin  
        case({j,k})  
            1: y <= 0;  
            2: y <= 1;  
            3: y <= ~y;  
        endcase  
        assign q = y;  
        assign qn = ~y;  
    endmodule
```

```
module example_5_1_2(  
    input cp, x,  
    input y2, y1,  
    output ny2, ny1  
);  
    wire k2j2;  
    wire ny2n, ny1n;  
    xor_gate U2(.a(x),.b(y1),.f(k2j2));  
    jk_flip_flop U3(.j(k2j2),.k(k2j2),.cp(cp),.q(ny2),.qn(ny2n));  
    jk_flip_flop U1(.j(1),.k(1),.cp(cp),.q(ny1),.qn(ny1n));  
endmodule
```

```
module example_5_1_2_exe(           ←  
    input sw_pin[7:0],              //8个拨动开关  
    input btn_1,                    //1个按钮  
    output [15:0] led_pin           //16个led灯  
);  
    reg y2, y1;  
    example_5_1_2 U(.cp(btn_1), .x(sw_pin[0]), .y2(y2), .y1(y1), .ny2(led_pin[0]), .ny1(led_pin[1]));  
    initial begin                    ↑  
        y2=0;  
        y1=0;  
    end  
    always @(*)                     //行为描述方式  
    begin  
        y2 <= led_pin[0];  
        y1 <= led_pin[1];  
    end  
endmodule
```

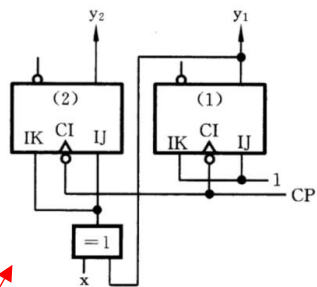


图 5.8 同步时序逻辑电路

## 结构化描述方式

## example\_5\_1\_2\_EGO1

```
module example_5_1_2_exe(  
    input sw_pin[7:0],              //8个拨动开关  
    input btn_1,                    //1个按钮  
    output [15:0] led_pin           //16个led灯  
);  
    reg y2, y1;  
    example_5_1_2 U(.cp(btn_1), .x(sw_pin[0]), .y2(y2), .y1(y1), .ny2(led_pin[0]), .ny1(led_pin[1]));  
    initial begin  
        y2=0;  
        y1=0;  
    end  
    always @(*)                     //行为描述方式  
    begin  
        y2 <= led_pin[0];  
        y1 <= led_pin[1];  
    end  
endmodule
```

# example\_5\_1\_3\_EGO1

```
module example_5_1_3_exe(
    input sw_pin[7:0],           //8个拨动开关
    input btn_1,                 //1个按钮
    output [15:0] led_pin        //16个led灯
);

    reg y2, y1;

    example_5_1_3 U(cp(btn_1), .x(sw_pin[0]), .y2(y2), .y1(y1), .ny2(led_pin[0]), .ny1(led_pin[1]));

    initial begin
        y2=0;
        y1=0;
    end

    always @(*)                //行为描述方式
    begin
        y2 <= led_pin[0];
        y1 <= led_pin[1];
    end

endmodule
```

example\_5\_1\_3\_EGO1.v - 记事本  
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)  
//行为描述方式实现例题5.1的2位二进制数可逆计数器，开发板上的输入（x、CP）为开发板上最左边的拨动开关以及S1按键，输出

```
`timescale 1ns / 1ps

module example_5_1_3(
    input cp, x,
    input y2, y1,
    output reg ny2, ny1
);

    always @(negedge cp)        //行为描述方式
    begin
        case({x, y2, y1})
            0: begin ny2 <= 0; ny1 <= 1; end
            1: begin ny2 <= 1; ny1 <= 0; end
            2: begin ny2 <= 1; ny1 <= 1; end
            3: begin ny2 <= 0; ny1 <= 0; end
            4: begin ny2 <= 1; ny1 <= 1; end
            5: begin ny2 <= 0; ny1 <= 0; end
            6: begin ny2 <= 0; ny1 <= 1; end
            7: begin ny2 <= 1; ny1 <= 0; end
        endcase
    end

endmodule
```

## 行为描述方式

// x y2 y1 = 000	y2n+1 y1n+1 = 01
// x y2 y1 = 001	y2n+1 y1n+1 = 10
// x y2 y1 = 010	y2n+1 y1n+1 = 11
// x y2 y1 = 011	y2n+1 y1n+1 = 00
// x y2 y1 = 100	y2n+1 y1n+1 = 11
// x y2 y1 = 101	y2n+1 y1n+1 = 00
// x y2 y1 = 110	y2n+1 y1n+1 = 01
// x y2 y1 = 111	y2n+1 y1n+1 = 10

表 5.3 次态真值表

输 入	现 态		激励函数				次 态	
x	y2	y1	J2	K2	J1	K1	y2 <sup>n+1</sup>	y1 <sup>n+1</sup>
0	0	0	0	0	1	1	0	1
0	0	1	1	1	1	1	1	0
0	1	0	0	0	1	1	1	1
0	1	1	1	1	1	1	0	0
1	0	0	1	1	1	1	1	1
1	0	1	0	0	1	1	0	0
1	1	0	1	1	1	1	0	1
1	1	1	0	0	1	1	1	0

```
module example_5_1_3_exe(
    input sw_pin[7:0],
    input btn_1,
    output [15:0] led_pin
);

    reg y2, y1;

    example_5_1_3 U(cp(btn_1), .x(sw_pin[0]), .y2(y2), .y1(y1), .ny2(led_pin[0]), .ny1(led_pin[1]));

    initial begin
        y2=0;
        y1=0;
    end

    always @(*)                //行为描述方式
    begin
        y2 <= led_pin[0];
        y1 <= led_pin[1];
    end

endmodule
```

## • 2、（验证实验）例题5.2的实现

- 分别用结构化描述方式和行为描述方式实现例题5.2的“101”序列检测器，该电路的输入为x，状态为 $y_2$ 、 $y_1$ ，输出为Z，CP为时钟信号。
- 请编写仿真程序（不需要开发板），验证在下面的输入x序列下， $y_2^{n+1}$ 、 $y_1^{n+1}$ 和Z的输出是否正确？

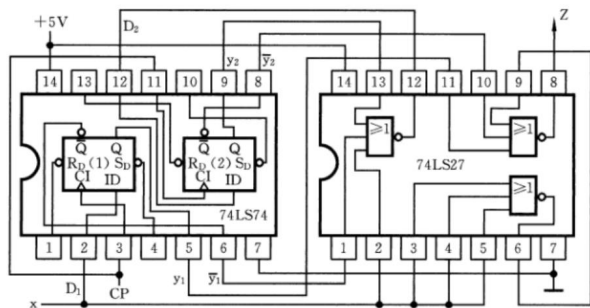
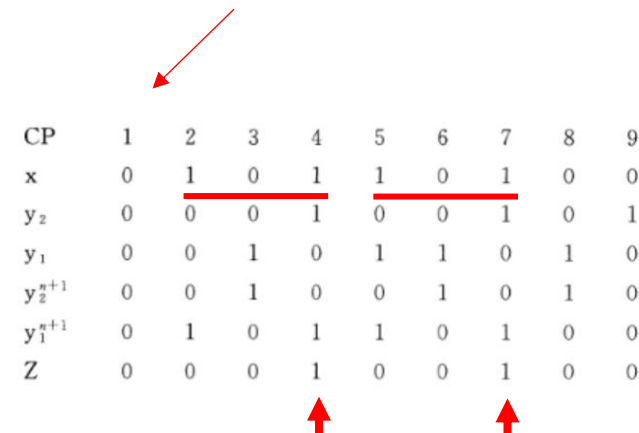


图 5.11 同步时序逻辑芯片连线图

表 5.6 状态表

现态 $y_2 \ y_1$	次态/输出 ( $y_2^{n+1} y_1^{n+1} / Z$ )	
	x=0	x=1
0 0	00/0	01/0
0 1	10/0	01/0
1 1	00/0	01/0
1 0	00/0	01/1



CP	1	2	3	4	5	6	7	8	9
x	0	1	0	1	1	0	1	0	0
$y_2$	0	0	0	1	0	0	1	0	1
$y_1$	0	0	1	0	1	1	0	1	0
$y_2^{n+1}$	0	0	1	0	0	1	0	1	0
$y_1^{n+1}$	0	1	0	1	1	0	1	0	0
Z	0	0	0	1	0	0	1	0	0

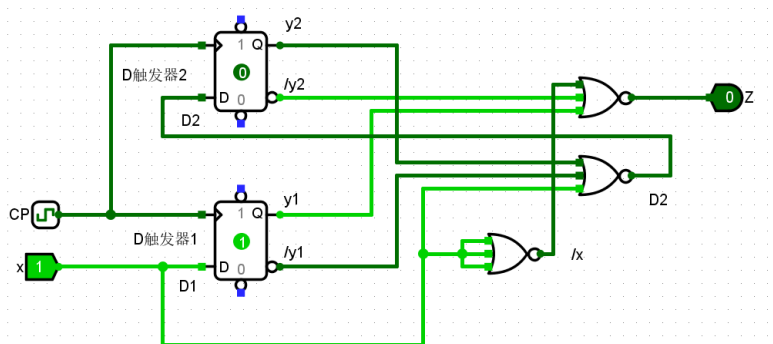


图 5.13 时间图

```
input a,
input b,
input c,
output f
);
reg y;
always @(*)           //行为描述方式
begin
    y <= ~(a | b | c);
end
assign f = y;
endmodule
```

## example\_5\_2\_EGO1

```
module d_flip_flop(
input d,
input cp,
input rd,
output q, qn
);
reg y;
always @(posedge cp or negedge rd)
begin
    if(rd==0)
        y <= 0;
    else
        case(d)
            0: y <= 0;
            1: y <= 1;
        endcase
    end
    assign q = y;
    assign qn = ~y;
endmodule
```

//行为描述方式

// rd=0 y=0

// rd=1

// d = 0 y=0

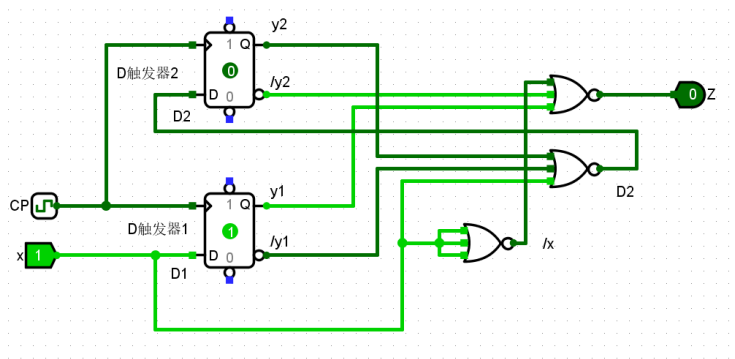
// d = 1 y=1

```
module example_5_2(
input cp,
input x,
input rd,
input y2, y2n, y1, y1n,
output ny2, ny2n, ny1, ny1n, z
);
wire xn, d2;
nor_gate U1(.a(x),.b(y1n),.c(y2),.f(d2));
d_flip_flop U2(.d(d2),.cp(cp),.rd(rd),.q(ny2),.qn(ny2n));
d_flip_flop U3(.d(x),.cp(cp),.rd(rd),.q(ny1),.qn(ny1n));
nor_gate U4(.a(x),.b(x),.c(x),.f(xn));
nor_gate U5(.a(xn),.b(ny2n),.c(ny1),.f(z));
endmodule
```

//结构化描述方式

## 结构化描述方式

//结构化描述方式



`timescale 1ns / 1ps

module example\_5\_2\_1\_sim();

```
reg cp, x;
wire ny2, ny1, z;
reg y2, y1;
```

example\_5\_2\_1 U(.cp(cp), .x(x), .y2(y2), .y1(y1), .ny2(ny2), .ny1(ny1), .z(z));

initial begin

```
#0
y2=0;
y1=0;
cp=0;
x=0;
```

```
#20
x=1;
```

```
#20
x=0;
```

```
#20
x=1;
```

```
#20
x=1;
```

```
#20
x=0;
```

```
#20
x=1;
```

```
#20
x=0;
```

```
#20
x=0;
```

end

always #10 cp <= ~cp;

always #20 begin y2 <= ny2; y1 <= ny1; end

endmodule

## 仿真程序

CP	1	2	3	4	5	6	7	8	9
x	0	1	0	1	1	0	1	0	0
y <sub>2</sub>	0	0	0	1	0	0	1	0	1
y <sub>1</sub>	0	0	1	0	1	1	0	1	0
y <sub>2</sub> <sup>n+1</sup>	0	0	1	0	0	1	0	1	0
y <sub>1</sub> <sup>n+1</sup>	0	1	0	1	1	0	1	0	0
Z	0	0	0	1	0	0	1	0	0

//行为描述方式实现例题5.2的“101”序列检测器,

## example\_5\_2\_1\_EGO1

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

```
module example_5_2_1(  
    input cp,  
    input x,  
    input rd,  
    input y2, y1,  
    output reg ny2, ny1, z  
);
```



```
always @(posedge cp or negedge rd)
```

```
begin  
    if(rd==0)  
        begin ny2 <= 0; ny1 <= 0; z=0; end  
    else  
        case({y2, y1, x})  
            0: begin ny2 <= 0; ny1 <= 0; z=0; end  
            1: begin ny2 <= 0; ny1 <= 1; z=0; end  
            2: begin ny2 <= 1; ny1 <= 0; z=0; end  
            3: begin ny2 <= 0; ny1 <= 1; z=0; end  
            4: begin ny2 <= 0; ny1 <= 0; z=0; end  
            5: begin ny2 <= 0; ny1 <= 1; z=1; end  
            6: begin ny2 <= 0; ny1 <= 0; z=0; end  
            7: begin ny2 <= 0; ny1 <= 1; z=0; end  
        endcase  
    end  
end
```

```
endmodule
```

表 5.6 状态表

现态 $y_2 \quad y_1$		次态/输出 ( $y_2^{n+1} y_1^{n+1} / Z$ )	
		x=0	x=1
0	0	00/0	01/0
0	1	10/0	01/0
1	1	00/0	01/0
1	0	00/0	01/1

// rd=0

// rd=1

## 行为描述方式

## 仿真程序

```
reg cp, x;  
wire ny2, ny1, z;
```

```
reg rd;  
reg y2, y1;
```

```
example_5_2_1 U(.cp(cp), .x(x), .rd(rd), .y2(y2), .y1(y1), .ny2(ny2), .ny1(ny1), .z(z));
```

```
initial begin
```

```
    #0  
    cp=0;  
    rd=0;  
    x=0;
```

```
    #5  
    rd=1;
```

```
    #20  
    x=1;
```

```
    #20  
    x=0;
```

```
    #20  
    x=1;
```

```
    #20  
    x=1;
```

```
    #20  
    x=0;
```

```
    #20  
    x=1;
```

```
    #20  
    x=0;
```

```
    #20  
    x=0;
```

```
end
```

```
always #10 cp <= ~cp;
```

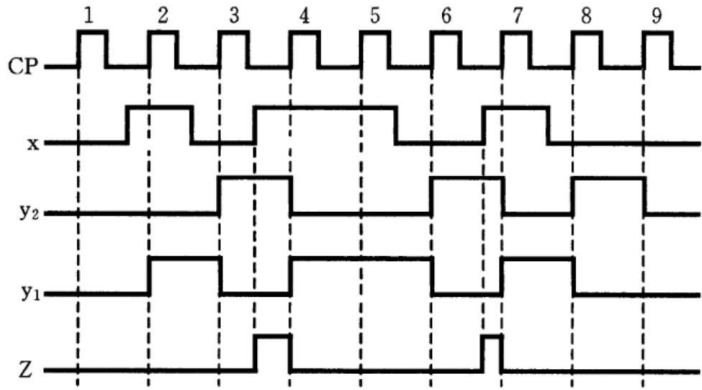
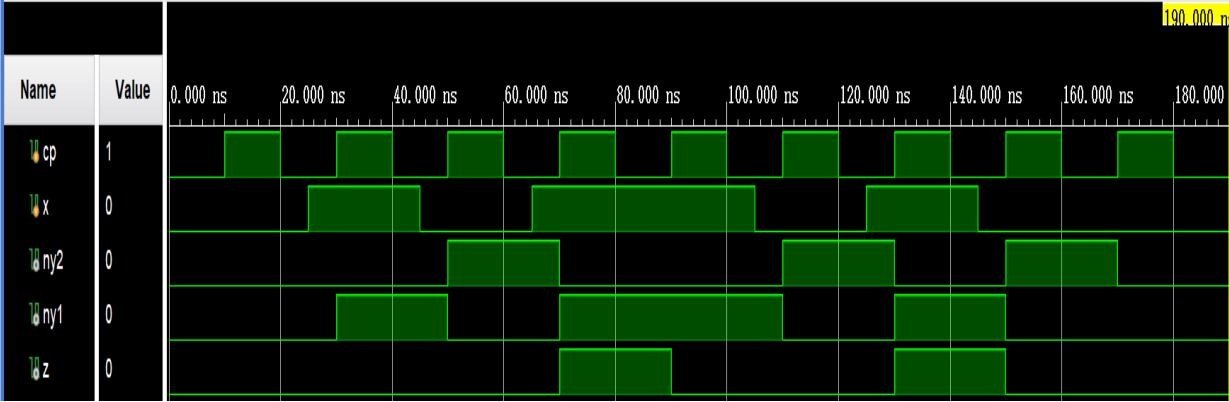
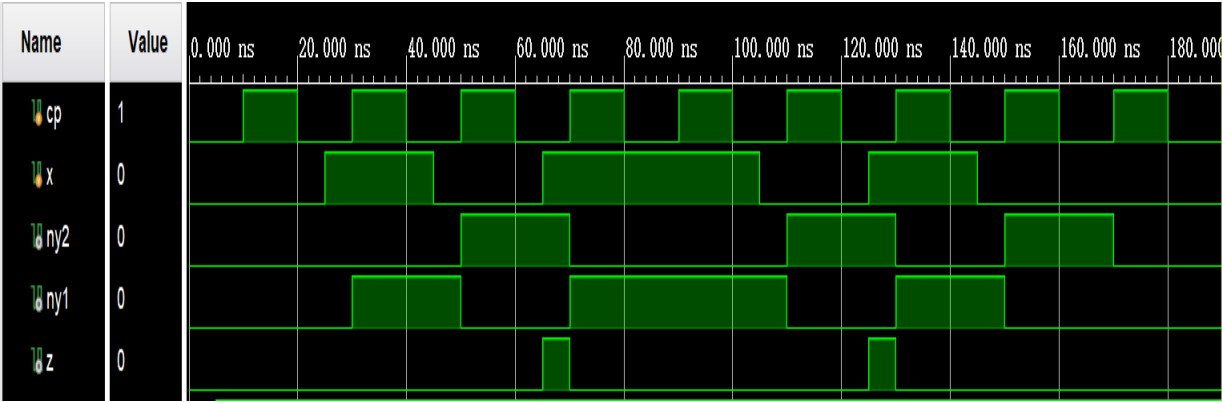
```
always #20 begin y2 <= ny2; y1 <= ny1; end
```

```
endmodule
```

CP	1	2	3	4	5	6	7	8	9
x	0	1	0	1	1	0	1	0	0
y <sub>2</sub>	0	0	0	1	0	0	1	0	1
y <sub>1</sub>	0	0	1	0	1	1	0	1	0
y <sub>2</sub> <sup>n+1</sup>	0	0	1	0	0	1	0	1	0
y <sub>1</sub> <sup>n+1</sup>	0	1	0	1	1	0	1	0	0
Z	0	0	0	1	0	0	1	0	0

结构化描述方式

行为描述方式



CP	1	2	3	4	5	6	7	8	9
x	0	1	0	1	1	0	1	0	0
y <sub>2</sub>	0	0	0	1	0	0	1	0	1
y <sub>1</sub>	0	0	1	0	1	1	0	1	0
y <sub>2</sub> <sup>n+1</sup>	0	0	1	0	0	1	0	1	0
y <sub>1</sub> <sup>n+1</sup>	0	1	0	1	1	0	1	0	0
Z	0	0	0	1	0	0	1	0	0

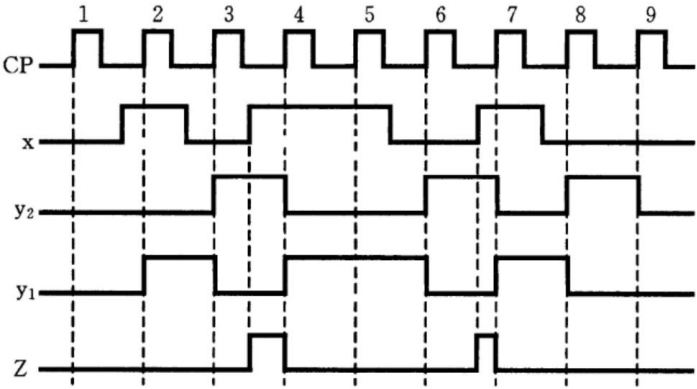


图 5.13 时间图

图 5.13 时间图

### • 3、（设计实验，课后完成）例题5.3的实现

- 请分别用结构化描述方式和行为描述方式实现例题5.3的串行加法器，该电路的输入为 $x_1$ 、 $x_2$ ，状态为 $y$ ，输出为 $Z$ ，CP为时钟信号。
- 请编写仿真程序（不需要开发板），验证在下面的输入（ $x_1$ 、 $x_2$ ）序列下， $y$ 和 $Z$ 的输出是否正确？
- EGO1开发板（结构化描述方式）：工程命名为example\_5\_3\_EGO1，设计文件命名为example\_5\_3\_EGO1.v，仿真文件命名为example\_5\_3\_sim\_EGO1.v，约束文件为EGO1.xdc。
- EGO1开发板（行为描述方式）：工程命名为example\_5\_3\_1\_EGO1，设计文件命名为example\_5\_3\_1\_EGO1.v，仿真文件命名为example\_5\_3\_1\_sim\_EGO1.v，约束文件为EGO1.xdc。

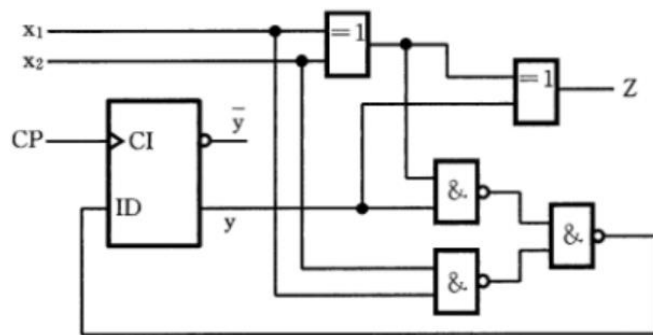


图 5.14 逻辑电路

表 5.7 状态表

现态 $y$	次态/输出 ( $y^{n+1}/Z$ )			
	$x_1 x_2 = 00$	$x_1 x_2 = 01$	$x_1 x_2 = 11$	$x_1 x_2 = 10$
0	0/0	0/1	1/0	0/1
1	0/1	1/0	1/1	1/0

时钟节拍	1	2	3	4	5	6	7	8
输入 $x_1$	0	0	1	1	0	1	1	0
输入 $x_2$	0	1	0	1	1	1	0	0
状态 $y$	0	0	0	0	1	1	1	1
输出 $Z$	0	1	1	0	0	1	0	1

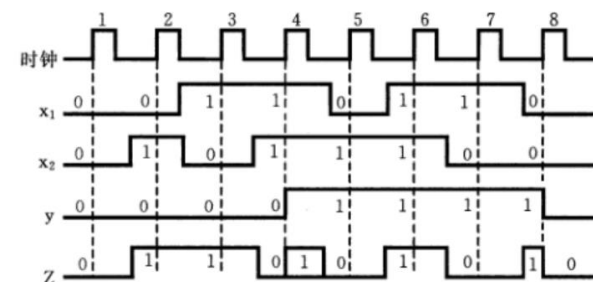
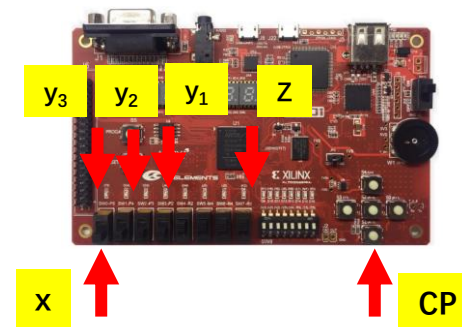


图 5.16 时间图



## • 4、（设计实验，课后完成）例题5.4的实现

- 请分别用**结构化描述方式**和**行为描述方式**实现例题5.4的**3位串行输入移位寄存器**，该电路的输入为 $x$ ，状态为  $(y_3、y_2、y_1)$ ，输出为 $Z$ ，CP为时钟信号。
- 输入 $x$ 和CP**为开发板上最左边的拨动开关以及S1按键，**状态  $(y_3、y_2、y_1)$  和输出 $Z$** 为开发板上最左边的3个LED灯以及最右边的LED灯。
- 验证步骤**（在开发板上）：
  - 运行程序后，置 $x=1$ ，按CP按钮，观察状态  $(y_3、y_2、y_1)$ 、输出 $Z$ 的变化是否正确？
  - 再置 $x=0$ ，按CP按钮，观察状态  $(y_3、y_2、y_1)$ 、输出 $Z$ 的变化是否正确？
- EGO1开发板（结构化描述方式）：工程命名为**example\_5\_4\_EGO1**，设计文件命名为example\_5\_4\_EGO1.v，约束文件为EGO1.xdc。
- EGO1开发板（行为描述方式）：工程命名为**example\_5\_4\_1\_EGO1**，设计文件命名为example\_5\_4\_1\_EGO1.v，约束文件为EGO1.xdc。



EGO1开发板

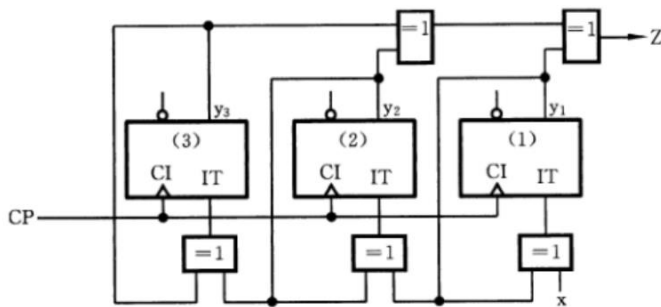


图 5.17 逻辑电路

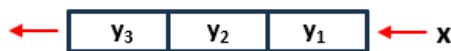


表 5.8 状态表

现 态 $y_3 y_2 y_1$	次态 $y_3^{n+1} y_2^{n+1} y_1^{n+1}$		输出 $Z$
	$x=0$	$x=1$	
0 0 0	0 0 0	0 0 1	0
0 0 1	0 1 0	0 1 1	1
0 1 0	1 0 0	1 0 1	1
0 1 1	1 1 0	1 1 1	0
1 0 0	0 0 0	0 0 1	1
1 0 1	0 1 0	0 1 1	0
1 1 0	1 0 0	1 0 1	0
1 1 1	1 1 0	1 1 1	1



## • 5、（选做实验）例题5.10的实现

- 请分别用**结构化描述方式**和**行为描述方式**实现例题5.10的电路（有2个电路），该电路的输入为 $x$ ，状态为 $(y_2, y_1)$ ，输出为 $Z$ ，CP为时钟信号。
- （结构化描述方式）**输入 $x$ 和CP**为开发板上最左边的拨动开关以及S1按键，用J-K触发器实现的电路的**状态 $(y_2, y_1)$ 和输出 $Z$** 为开发板上最左边的3个LED灯，用D触发器实现的电路的**状态 $(y_2, y_1)$ 和输出 $Z$** 为开发板上最右边的3个LED灯。
- （行为描述方式）**输入 $x$ 和CP**为开发板上最左边的拨动开关以及S1按键，**状态 $(y_2, y_1)$ 和输出 $Z$** 为开发板上最左边的3个LED灯。
- **验证步骤**（在开发板上）：
  - 运行程序，记录现在的状态（现态， $y_2, y_1$ ）；置 $x=0$ ，观察输出 $Z$ 的值，是否与状态表中的内容一致？按CP按钮，观看变化后的状态（次态， $y_2, y_1$ ），是否与状态表中的内容一致？
  - 反复进行上述操作，直到状态表中的**8种情况都得到验证**。

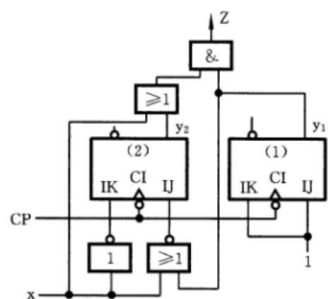


图 5.30 逻辑电路

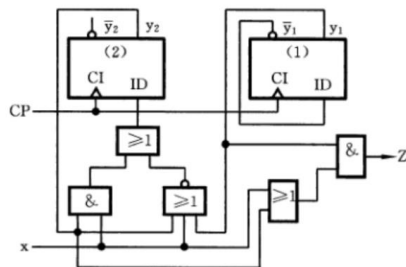
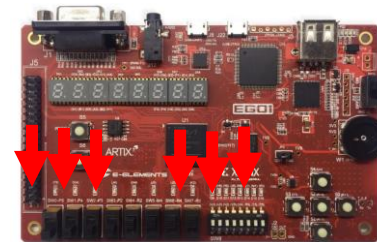


图 5.32 采用 D 触发器的逻辑电路

表 5.21 二进制状态表

现 态 $y_2$ $y_1$	次态 $y_2^{n+1} y_1^{n+1}$ / 输出 $Z$	
	$x=0$	$x=1$
0 0	11/0	01/0
0 1	00/0	00/1
1 1	00/1	10/1
1 0	01/0	11/0



EGO1开发板

## • 6、（选做实验） 例题5.14的实现

- 请分别用**结构化描述方式**和**行为描述方式**实现例题5.14的**2位二进制减1计数器**，该电路的输入为 $x$ ，状态为 $(y_2, y_1)$ ，输出为 $Z$ ，CP为时钟信号。
- **输入 $x$ 和CP**为开发板上最左边的拨动开关以及S1按键，**状态 $(y_2, y_1)$ 和输出 $Z$** 分别开发板上最左边的2个LED灯以及最右边的LED灯。
- **验证步骤（在开发板上）：**
  - 运行程序，记录现在的状态（现态， $y_2, y_1$ ）；置 $x=0$ ，观察输出 $Z$ 的值，是否与状态表中的内容一致？按CP按钮，观看变化后的状态（次态， $y_2, y_1$ ），是否与状态表中的内容一致？
  - 反复进行上述操作，直到状态表中的**8种情况都得到验证**。

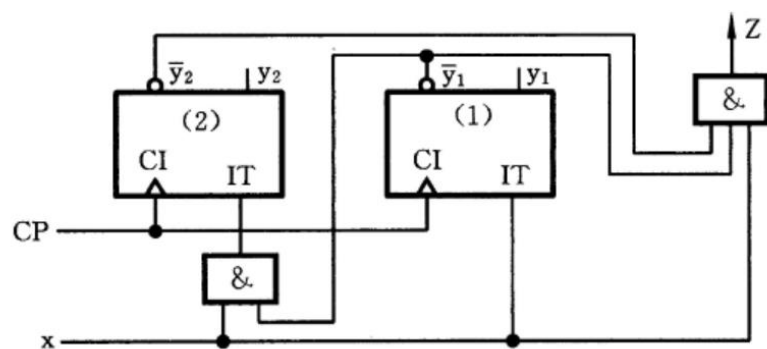
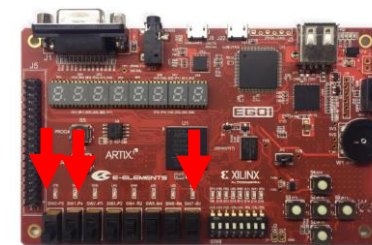


图 5.42 逻辑电路

表 5.32 状态表

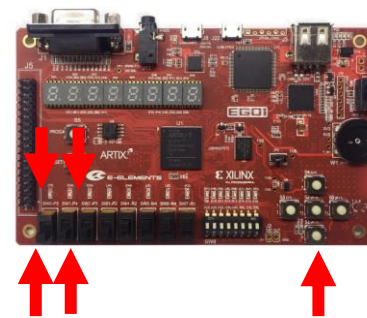
现 态 $y_2 \quad y_1$		次态 $y_2^{n+1} y_1^{n+1}$ / 输出 $Z$	
		$x=0$	$x=1$
0	0	00/0	11/1
0	1	01/0	00/0
1	1	11/0	10/0
1	0	10/0	01/0



EGO1开发板

## • 7、（选做实验） 例题5.15的实现

- 请分别用**结构化描述方式**和**行为描述方式**实现例题5.15的**双向移位寄存器**，该电路的输入为  $(x_2、x_1)$ ，状态为  $(y_2、y_1)$ ，CP为时钟信号。
- **输入  $(x_2、x_1)$  和CP**为开发板上最左边的2个拨动开关以及S1按键，**状态  $(y_2、y_1)$** 为开发板上最左边的2个LED灯。
- **验证步骤（在开发板上）：**
  - 置 $x_2=1, x_1=1$ ；按CP，可以看到 $y_1、y_2$ 先后置1，表示 $x_1$ 往寄存器低位串行送数，寄存器中的数据“1”**从低位移向高位**；
  - 再置 $x_1=0$ ，按CP，可以看到 $y_1、y_2$ 先后置0，表示 $x_1$ 往寄存器低位串行送数，寄存器中的数据“0”从低位移向高位；
  - 置 $x_2=0, x_1=1$ ，按CP，可以看到 $y_2、y_1$ 先后置1，表示 $x_1$ 往寄存器高位串行送数，寄存器中的数据“1”**从高位移向低位**；
  - 再置 $x_1=0$ ，按CP，可以看到 $y_2、y_1$ 先后置0，表示 $x_1$ 往寄存器高位串行送数，寄存器中的数据“0”从高位移向低位。



EGO1开发板

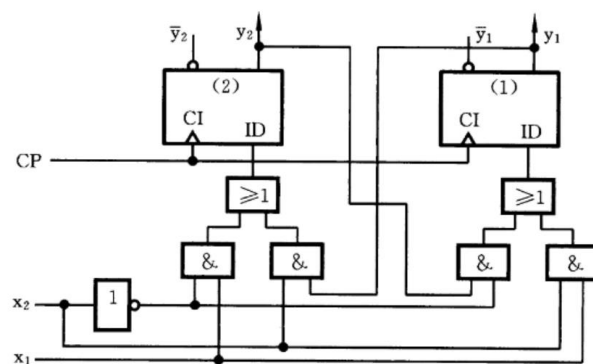


图 5.45 逻辑电路图

表 5.33 二进制状态表

现 态	次态 $y_2^{n+1}y_1^{n+1}$			
$y_2 \ y_1$	$x_2x_1=00$	$x_2x_1=01$	$x_2x_1=11$	$x_2x_1=10$
0 0	0 0	1 0	0 1	0 0
0 1	0 0	1 0	1 1	1 0
1 1	0 1	1 1	1 1	1 0
1 0	0 1	1 1	0 1	0 0

## • 8、（选做实验）例题5.16的实现

- 请分别用**结构化描述方式**和**行为描述方式**实现例题5.16的“101”序列检测器，该电路的输入为 $x$ ，状态为 $(y_2, y_1)$ ，输出为 $Z$ ，CP为时钟信号。
- 请编写**仿真程序**（不需要开发板），验证在下面的输入 $x$ 序列下， $Z$ 的输出是否正确？

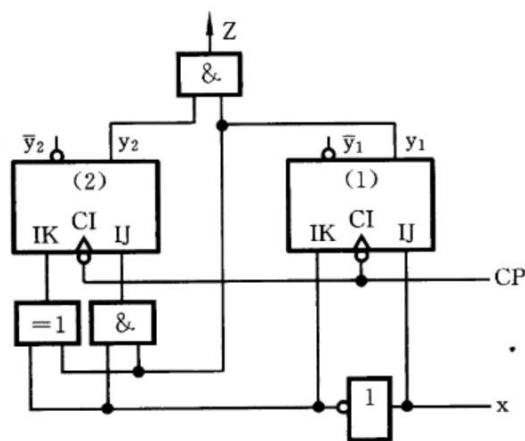


图 5.48 逻辑电路

表 5.35 二进制状态表

现态		次态 $y_2^{n+1} y_1^{n+1}$		输出 $Z$
		$x=0$	$x=1$	
$y_2$	$y_1$			
0	0	0 0	0 1	0
0	1	1 0	0 1	0
1	0	0 0	1 1	0
1	1	1 0	0 1	1

输入  $x$     0   0   1   0   1   0   1   1   0   1   0   0  
 输出  $Z$     0   0   0   0   1   0   1   0   0   1   0   0

## • 9、（选做实验）例题5.17的实现

- 请分别用**结构化描述方式**和**行为描述方式**实现例题5.17的**奇偶检测器**，该电路的输入为x，状态为（ $y_3$ 、 $y_2$ 、 $y_1$ ），输出为Z，CP为时钟信号。
- 请编写**仿真程序**（不需要开发板），验证在下面的输入x序列下，Z的输出是否正确？

输入x = 111    110    101    100    011    010    001    000  
 输出Z = 0      1      1      0      1      0      0      1

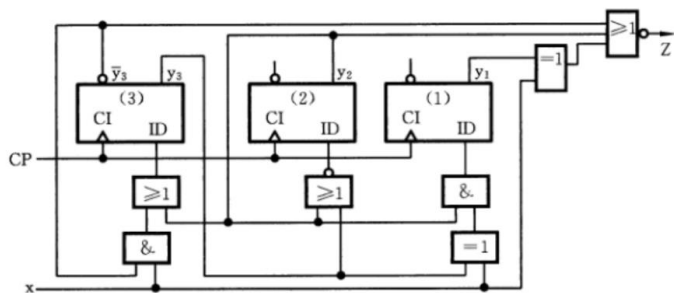


图 5.53 逻辑电路

表 5.39 二进制状态表

$y_3$	$y_2$	$y_1$	$y_3^{n+1}y_2^{n+1}y_1^{n+1}/Z$	
			x=0	x=1
0	0	0	010/0	110/0
0	1	0	100/0	101/0
1	1	0	101/0	100/0
1	0	0	000/1	000/0
1	0	1	000/0	000/1

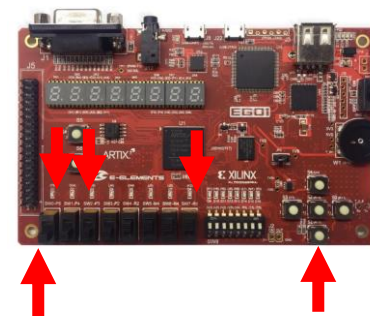
表 5.40 无效状态检查表

输入 x	现 态 $y_3 y_2 y_1$			激励函数 $D_3 D_2 D_1$			次 态 $y_3^{n+1} y_2^{n+1} y_1^{n+1}$			输出 Z
	$y_3$	$y_2$	$y_1$	$D_3$	$D_2$	$D_1$	$y_3^{n+1}$	$y_2^{n+1}$	$y_1^{n+1}$	
0	0	0	1	0	1	0	0	1	0	0
0	0	1	1	1	0	0	1	0	0	0
0	1	1	1	1	0	1	1	0	1	0
1	0	0	1	1	1	0	1	1	0	0
1	0	1	1	1	0	1	1	0	1	0
1	1	1	1	1	0	0	1	0	0	0

## (二) 在FPGA开发板上实现教材第5章的习题

### • 1、（选做实验）习题5.4的实现

- 请分别用结构化描述方式和行为描述方式实现习题5.4的电路，该电路的输入为 $x$ ，状态为 $(y_2, y_1)$ ，输出为 $Z$ ，CP为时钟信号。
- 输入 $x$ 和CP为开发板上最左边的拨动开关以及S1按键，状态 $(y_2, y_1)$ 和输出 $Z$ 为开发板上最左边的2个LED灯以及最右边的LED灯。
- 请在开发板上（或编写仿真程序）验证该电路的功能。



EGO1开发板

5.4 分析图 5.55 所示逻辑电路。假定电路初始状态为“00”，说明该电路逻辑功能。

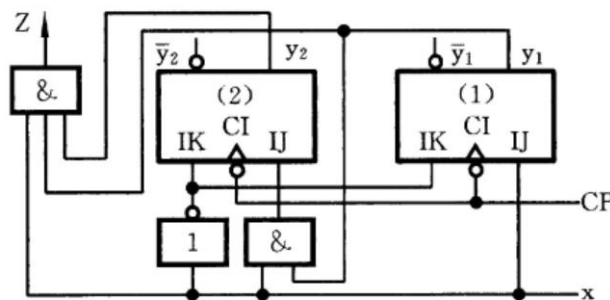


图 5.55 逻辑电路



## • 2、（选做实验）习题5.5的实现

- 请分别用结构化描述方式和行为描述方式实现习题5.5的电路，该电路的输入为 $x$ ，状态为 $(y_2, y_1)$ ，输出为 $Z$ ， $CP$ 为时钟信号。
- 输入 $x$ 和 $CP$ 为开发板上最左边的拨动开关以及S1按键，状态 $(y_2, y_1)$ 和输出 $Z$ 为开发板上最左边的2个LED灯以及最右边的LED灯。
- 请在开发板上（或编写仿真程序）验证该电路的功能。

5.5 分析图 5.56 所示逻辑电路，说明该电路功能。

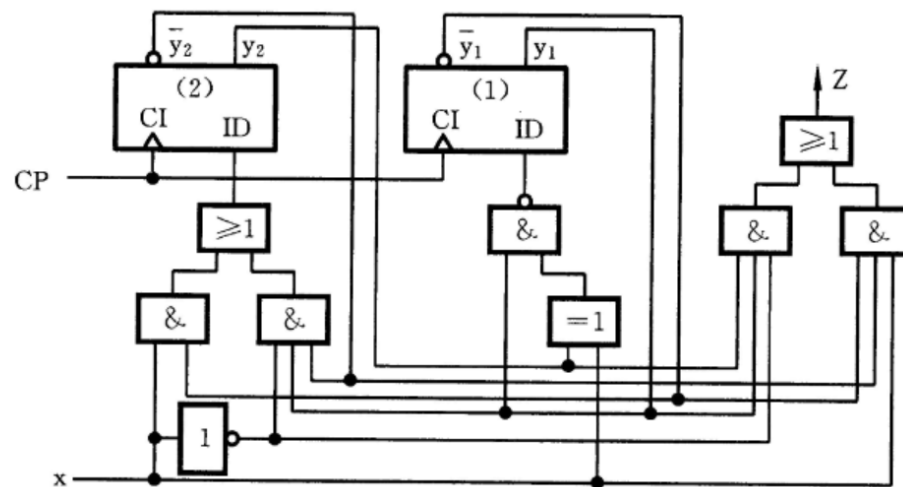
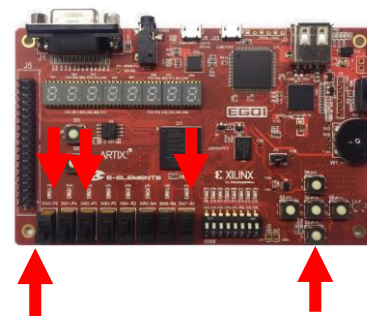


图 5.56 逻辑电路



EGO1开发板

### • 3、（选做实验）习题5.6的实现

- 请分别用结构化描述方式和行为描述方式实现习题5.6的电路，该电路的输入为 $x$ ，状态为 $(y_2, y_1)$ ，输出为 $Z$ ， $CP$ 为时钟信号。
- 输入 $x$ 和 $CP$ 为开发板上最左边的拨动开关以及S1按键，状态 $(y_2, y_1)$ 和输出 $Z$ 为开发板上最左边的2个LED灯以及最右边的LED灯。
- 请在开发板上（或编写仿真程序）验证该电路的功能。

5.6 分析图 5.57 所示逻辑电路，说明该电路功能。

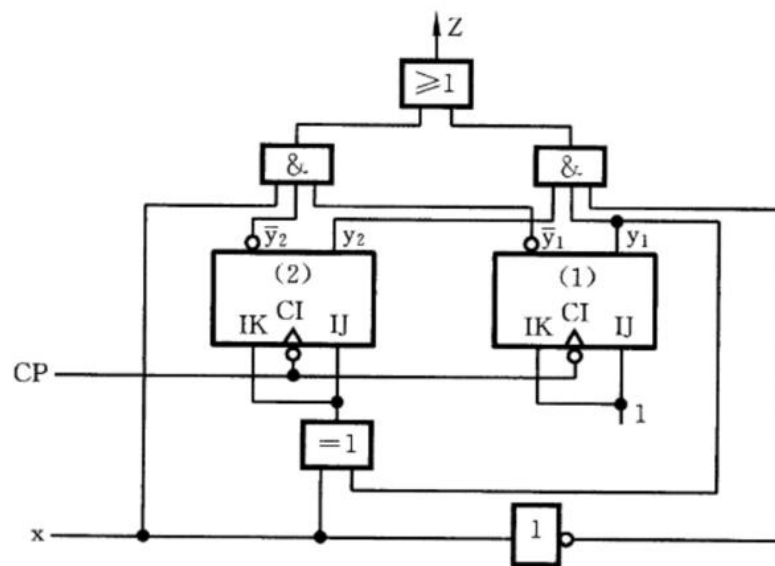
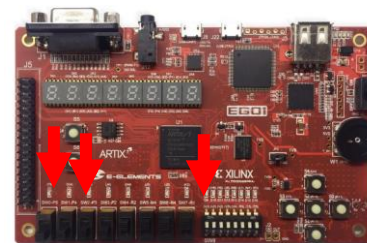


图 5.57 逻辑电路

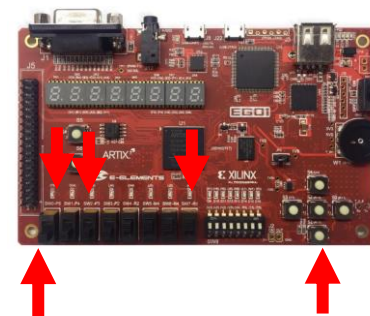


EGO1开发板



## • 4、（选做实验）习题5.8的实现

- 请分别用**结构化描述方式**和**行为描述方式**实现习题5.8的电路（**代码检测器**），该电路的输入为 $x$ ，状态为 $(y_n、\cdots、y_1)$ ，输出为 $Z$ ，CP为时钟信号。
- **输入 $x$ 和CP**为开发板上最左边的拨动开关以及S1按键，**状态 $(y_n、\cdots、y_1)$ 和输出 $Z$** 为开发板上最左边的 $n$ 个LED灯以及最右边的LED灯。
- 请在**开发板**上（或编写**仿真程序**）验证该电路的功能。



EGO1开发板

5.8 设计一个代码检测器,该电路从输入端  $x$  串行输入余 3 码(先低位后高位),当出现非法数字时,电路输出  $Z$  为 1,否则输出为 0。试作出 Mealy 型状态图。

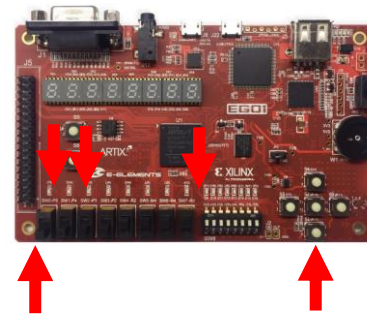
## • 5、（选做实验）习题5.12的实现

- 请分别用结构化描述方式和行为描述方式实现习题5.12的电路（有3个电路，分别对应D、T、J-K触发器），该电路的输入为x，状态为（ $y_2$ 、 $y_1$ ），输出为Z，CP为时钟信号。
- 输入x和CP为开发板上最左边的拨动开关以及S1按键，状态（ $y_2$ 、 $y_1$ ）和输出Z为开发板上最左边的2个LED灯以及最右边的LED灯。
- 请在开发板上（或编写仿真程序）验证该电路的功能。

5.12 分别用 D、T、J-K 触发器作为同步时序逻辑电路的存储元件，实现表 5.45 所示二进制状态表的功能。试写出激励函数和输出函数表达式，比较采用哪种触发器可使电路最简。

表 5.45 状态表

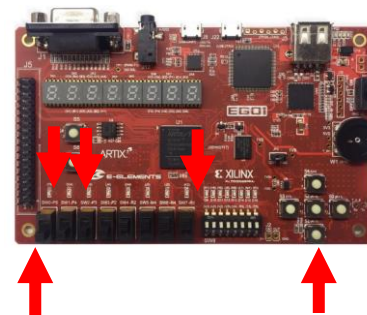
现 态 $y_2 \quad y_1$		次态 $y_2^{n+1} y_1^{n+1}$ / 输出 Z	
		x=0	x=1
0	0	01/0	10/0
0	1	11/0	10/0
1	1	10/1	01/0
1	0	00/1	11/1



EGO1开发板

## • 6、（选做实验）习题5.13的实现

- 请分别用结构化描述方式和行为描述方式实现习题5.13的电路，该电路的输入为x，状态为 $(y_2、y_1)$ ，输出为Z，CP为时钟信号。
- 输入x和CP为开发板上最左边的拨动开关以及S1按键，状态 $(y_2、y_1)$ 和输出Z为开发板上最左边的2个LED灯以及最右边的LED灯。
- 请在开发板上（或编写仿真程序）验证该电路的功能。



EGO1开发板

5.13 已知某同步时序逻辑电路的激励函数和输出函数表达式为

$$D_2 = \bar{x}y_2 + xy_2\bar{y}_1$$

$$D_1 = \bar{x}y_2 + y_2\bar{y}_1 + x\bar{y}_2y_1$$

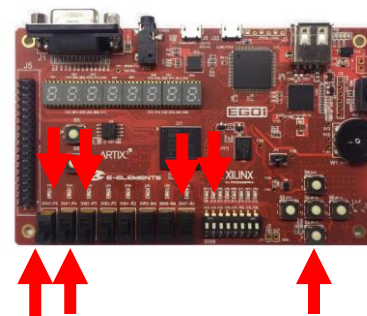
$$Z = y_2$$

• 试求出改用 J-K 触发器作为存储元件的最简电路。

## • 7、（选做实验）习题5.14的实现

- 请分别用结构化描述方式和行为描述方式实现习题5.14的电路（两个二进制数比较电路），该电路的输入为  $(x_2, x_1)$ ，分别对应题目中的X和Y），状态为  $(y_n, \dots, y_1)$ ，输出为  $(Z_x, Z_y)$ ，CP为时钟信号。
- 输入  $(x_2, x_1)$  和CP为开发板上最左边的2个拨动开关以及S1按键，状态  $(y_n, \dots, y_1)$  和输出  $(Z_x, Z_y)$  为开发板上最左边的n个LED灯以及最右边的2个LED灯。
- 请在开发板上（或编写仿真程序）验证该电路的功能。

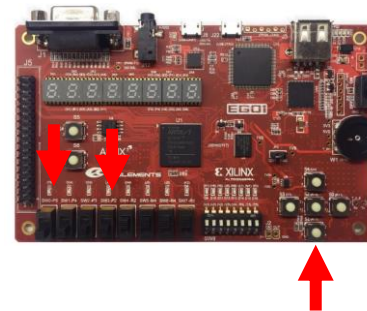
5.14 设计一个能对两个二进制数  $X = x_1 x_2 \dots x_n$  和  $Y = y_1 y_2 \dots y_n$  进行比较的同步时序逻辑电路，其中，X、Y 串行地输入到电路的  $x, y$  输入端。比较从  $x_1, y_1$  开始，依次进行到  $x_n, y_n$ 。电路有两个输出  $Z_x$  和  $Z_y$ ，若比较结果  $X > Y$ ，则  $Z_x$  为 1,  $Z_y$  为 0；若  $X < Y$ ，则  $Z_x$  为 0,  $Z_y$  为 1；若  $X = Y$ ，则  $Z_x$  和  $Z_y$  都为 1。要求用尽可能少的状态数作出状态图和状态表，并用尽可能少的逻辑门和触发器（采用 J-K 触发器）实现其功能。



EGO1开发板

## • 8、（选做实验）习题5.15的实现

- 请分别用结构化描述方式和行为描述方式实现习题5.15的电路（8421码十进制加1计数器），该电路的状态为  $(y_n, \dots, y_1)$ ，输出有状态相同，CP为时钟信号。
- CP为开发板上最左边的拨动开关以及S1按键，状态  $(y_n, \dots, y_1)$  为开发板上最左边的n个LED灯（输出与状态相同）。
- 请在开发板上（或编写仿真程序）验证该电路的功能。



EGO1开发板

5.15 用 T 触发器作为存储元件,设计一个 8421 码十进制加 1 计数器。

# 实验要求

- 1、在Logisim上完成**验证实验**，将实验过程中的主要结果通过截屏（或拍照）的方式，拷贝到实验报告中，并给予适当的文字说明。
- 2、在FPGA开发板上完成**验证实验**，将实验过程中的主要结果通过截屏（或拍照）的方式，拷贝到实验报告中，并给予适当的文字说明。
- 3、在Logisim上完成**设计实验**，设计文件命名为：**第5章习题电路的实现.circ**。
- 4、在FPGA开发板上完成**设计实验**，工程文件、设计文件、约束文件请严格按照规定的要求命名。
- 5、实验报告命名为：**学号+姓名+第4次实验报告.docx**。
- 6、将设计文件、实验报告打包压缩成1个压缩文件，命名为：**学号+姓名+第4次实验.zip**（或**.rar**），并上传到FTP上，上传截止日期：**2024年11月3日晚上24点**。
- 7、鼓励有兴趣的同学完成**选做实验**。

**Thanks**