实验一：计数器

一、 实验目的

1. 掌握简单时序逻辑电路的设计方法
2. 了解任意进制计数器的设计方法

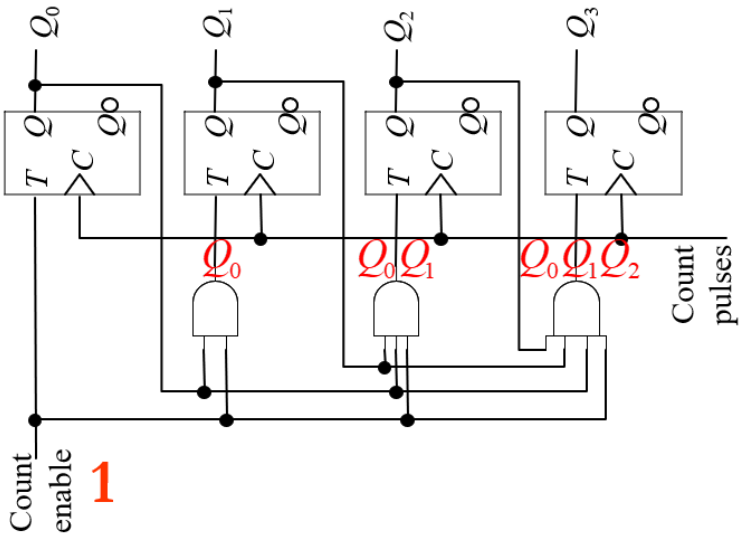
二、 设计方案

1. 计数器的原理

计数器是一种常用的时序电路，按照规定的方式改变内部各触发器的状态，以记录输入的时钟脉冲的个数。按照规定的计数顺序的不同，计数器又可分为加法计数器、减法计数器、可逆计数器和不同进制的计算器；按照工作方式的不同，又可以分为异步计数器和同步计数器。

2. 同步计数器

同步计数器中，各个触发器使用同一个计数控制时钟，每一位在时钟上升沿到来时是否反转取决于比其低的位是否都是“1”。其中，触发器的翻转是在时钟上升沿同步进行的，其翻转稳定时间仅仅取决于单级触发器的翻转时间，而与计数器的位数无关。

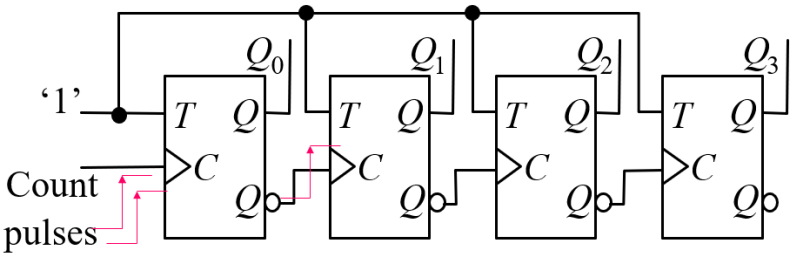


3. 异步计数器

异步计数器中，各个触发器使用的是不同的计数控制时钟，计数控制信号是在各级间逐级传递的，种种计数器从时钟脉冲上升沿到达最后一个触发器翻转到规定的状态，需要较长的延时，计数器位数越多，翻转到稳定的时间就越长。

每来一个计数脉冲，最低位 QA 的状态变化一次，其后每位则在低一位触发器

的状态由 1 变为 0 时发生状态变化，时钟输入是上一级的输出，这样一来就构成了异步计数器。



三、 关键代码

1. 同步计数器

a. 计数器代码

```
1   module add_dec(hit,cout,reset,ud);
2     input hit;
3     input reset;
4     input ud;
5     output [6:0] cout;
6
7     wire [3:0] cnt;
8     BCD7 U0(.din(cnt),.dout(cout));
9     count U1(.reset(reset),.hit(hit),.cnt(cnt),.ud(ud));
10  endmodule
```

```
12  module BCD7(din,dout);
13    input  [3:0] din;
14    output [6:0] dout;
15
16    assign dout=(din==4'h0)?7'b1000000:
17                (din==4'h1)?7'b1111001:
18                (din==4'h2)?7'b0100100:
19                (din==4'h3)?7'b0110000:
20                (din==4'h4)?7'b0011001:
21                (din==4'h5)?7'b0010010:
22                (din==4'h6)?7'b0000010:
23                (din==4'h7)?7'b1111000:
24                (din==4'h8)?7'b0000000:
25                (din==4'h9)?7'b0010000:
26                (din==4'hA)?7'b0001000:
27                (din==4'hB)?7'b0000011:
28                (din==4'hC)?7'b1000110:
29                (din==4'hD)?7'b0100001:
30                (din==4'hE)?7'b0000110:
31                (din==4'hF)?7'b0001110:7'b0;
32  endmodule
```

```
34  module count(hit,reset,cnt,ud);
35    input hit;
36    input reset;
37    input ud;
38    output [3:0] cnt;
39    reg [3:0] cnt;
40    always@(negedge hit or negedge reset)
41      begin
42        if(~reset) cnt<=4'b0;
43        else
44          begin
45             if(ud)
46               begin
47                     if(~hit)
48                      begin
49                       if(cnt==4'b1111) cnt=4'b0;
50                       else
51                         begin
52                          if(cnt[1]&cnt[2]&cnt[0]) cnt[3]=~cnt[3];
53                          if(cnt[0]&cnt[1]) cnt[2]=~cnt[2];
54                          if(cnt[0]) cnt[1]=~cnt[1];
55                          cnt[0]=~cnt[0];
56                         end
57                      end
58                end
59            else
60              begin
61               if(~hit)
62                begin
63                  if(cnt==4'b0) cnt=4'b1111;
64                else
65                  begin
66                    if(!(cnt[1]||cnt[2]||cnt[0])) cnt[3]=~cnt[3];
67                    if(!(cnt[0]||cnt[1])) cnt[2]=~cnt[2];
68                    if(!(cnt[0])) cnt[1]=~cnt[1];
69                    cnt[0]=~cnt[0];
70                  end
71                end
72              end
73            end
74        end
```

b. 测试代码

```
`timescale 1ns/1ps;
module add_dec_tb;
reg hit,reset,ud;
wire [6:0] cout1;
add_dec uut(.ud(ud),.hit(hit),.reset(reset),.cout(cout1));
initial
  begin
    ud=1;
    reset=0;
    hit=1;
    #10
    reset=1;
    #100
    hit=0;
    #10

    #40
    hit=1;
    #100
    hit=0;
    #50
    hit=1;
    #100
    hit=0;
    #50
    hit=1;
    #100
    hit=0;
    #50
    hit=1;
    #100
    hit=0;
    #50
    hit=1;
    #100
    hit=0;
    #50
    hit=1;
    #100
    hit=0;
    #50
    hit=1;
    #100
    hit=0;
    $stop;
  end
endmodule
```

2. 异步计数器

a. 计数器代码

```
1   module add_dec(hit,cout,reset);
2     input hit;
3     input reset;
4     output [6:0] cout;
5
6     wire [3:0] cnt;
7
8     BCD7 U0
9     (
10      .din  (cnt),
11      .dout (cout)
12    );
13
14    count U1
15    (
16      .reset  (reset),
17      .hit    (hit),
18      .cnt    (cnt)
19    );
20
21
22  endmodule
```

```
25  module BCD7(din,dout);
26    input  [3:0] din;
27    output [6:0] dout;
28
29    assign dout=(din==4'h0)?7'b1000000:
30                (din==4'h1)?7'b1111001:
31                (din==4'h2)?7'b0100100:
32                (din==4'h3)?7'b0110000:
33                (din==4'h4)?7'b0011001:
34                (din==4'h5)?7'b0010010:
35                (din==4'h6)?7'b0000010:
36                (din==4'h7)?7'b1111000:
37                (din==4'h8)?7'b0000000:
38                (din==4'h9)?7'b0010000:
39                (din==4'hA)?7'b0001000:
40                (din==4'hB)?7'b0000011:
41                (din==4'hC)?7'b1000110:
42                (din==4'hD)?7'b0100001:
43                (din==4'hE)?7'b0000110:
44                (din==4'hF)?7'b0001110:7'b0;
45  endmodule
```

```
47  module count(hit,reset,cnt);
48    input hit;
49    input reset;
50    output [3:0] cnt;
51
52    reg [3:0] cnt;
53
54    always@(negedge hit or negedge reset)
55      begin
56        if(~reset) cnt[0]=0;
57        else
58          cnt[0]=~cnt[0];
59      end
60
61    always@(negedge cnt[0] or negedge reset)
62      begin
63        if(~reset) cnt[1]=0;
64        else
65          cnt[1]=~cnt[1];
66      end
67
68     always@(negedge cnt[1] or negedge reset)
69      begin
70        if(~reset) cnt[2]=0;
71        else
72          cnt[2]=~cnt[2];
73      end
74
75      always@(negedge cnt[2] or negedge reset)
76      begin
77        if(~reset) cnt[3]=0;
78        else
79          cnt[3]=~cnt[3];
80      end
81
82  endmodule
```

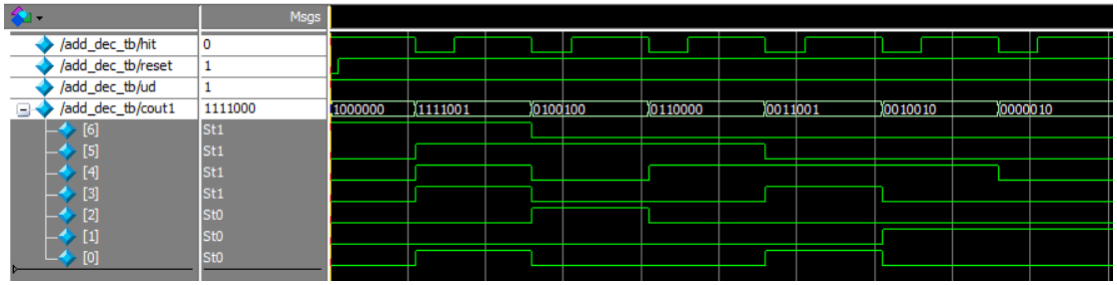
b. 测试代码

```
1  `timescale 1ns/1ps;
2  module add_dec_tb;
3  reg hit,reset;
4  wire [6:0] cout1;
5  add_dec uut(.hit(hit),.reset(reset),.cout(cout1));
6  initial
7    begin
8      reset=0;
9      hit=1;
10     #10
11     reset=1;
12     #100
13     hit=0;
14     #50
15     hit=1;
16     #100
17     hit=0;
18     #50
19     hit=1;
20     #100
21     hit=0;
22     #50
23     hit=1;
24     #100
25     hit=0;
26     #50
27     hit=1;
28     #100
29     hit=0;
30     #50
31     hit=1;
32     #100
33     hit=0;
34     #50
35     hit=1;
36     #100
37     hit=0;
38     #50
39     hit=1;
40     #100
41     hit=0;
42     $stop;
43   end
44 endmodule
```

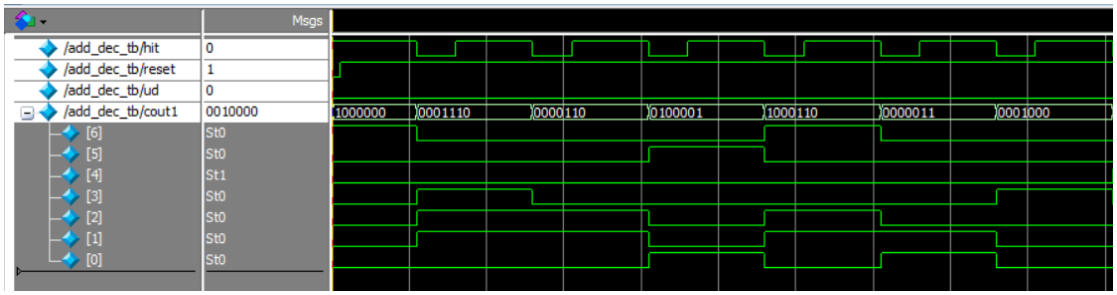
四、 仿真结果及分析

1. 同步计数器

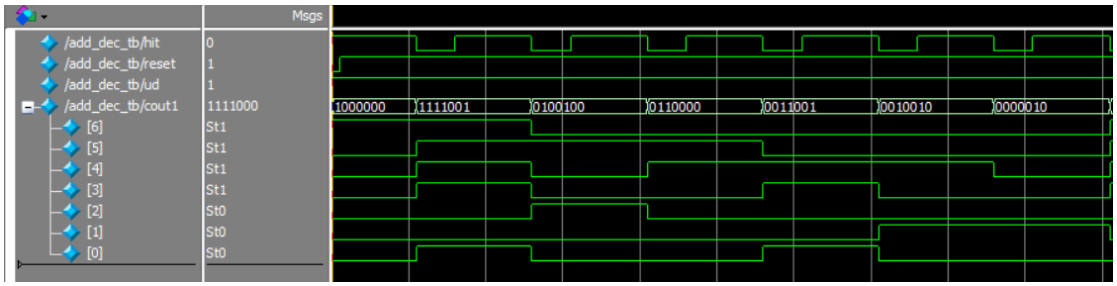
Ud=1（顺序）：



Ud=0（逆序）：



2. 异步计数器



五、 综合情况

1. 同步时序电路

Total logic elements	12
Total combinational functions	12
Dedicated logic registers	4
Total registers	4
Total pins	10
Total virtual pins	0
Total memory bits	0
Embedded Multiplier 9-bit elements	0
Total PLLs	0

Clocks

	Clock Name	Type	Period	Frequency	Rise	Fall
1	clk_50M	Base	20.000	50.0 MHz	0.000	10.000

Slow Model Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	653.59 MHz	450.05 MHz	clk_50M	limit due to minimum period restriction (max I/O toggle rate)

2. 异步时序电路

Total logic elements	11
Total combinational functions	11
Dedicated logic registers	4
Total registers	4
Total pins	9
Total virtual pins	0
Total memory bits	0
Embedded Multiplier 9-bit elements	0
Total PLLs	0

Clocks

	Clock Name	Type	Period	Frequency	Rise	Fall
1	count:U1|cnt[0]	Base	1.000	1000.0 MHz	0.000	0.500
2	count:U1|cnt[1]	Base	1.000	1000.0 MHz	0.000	0.500
3	count:U1|cnt[2]	Base	1.000	1000.0 MHz	0.000	0.500
4	hit	Base	1.000	1000.0 MHz	0.000	0.500

Slow Model Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	1610.31 MHz	500.0 MHz	count:U1|cnt[2]	limit due to low minimum pulse width violation (tcl)

六、 实验总结

我首先做的是同步计数器，仿真正确之后，在 Quartus 上编译，一遍就通过，在设置引脚的时候，把两个引脚设置反了，导致在板子上操作的时候有问题，总是反的。并且除了显示输出的那一个有数字，其余的都是输出都是亮着的，后来发现自己没有设置缺省的值，因此是亮着的。之后把缺省设置好，管脚设置好，可以再板子上很好的实现计数器的功能。

之后做的是异步计数器，因为刚开始没有充分理解异步计数器，结果写成了同步计数器的另一种表达形式，后来经过检查发现了这一问题并改正过来了。

这个实验让我对计数器有了更多的认识，同时对同步和异步差异的认识也更加深刻。