计数器的设计

计 93　王哲凡　2019011200

一、实验目的

1. 掌握时序逻辑电路的基本分析和设计方法。
2. 理解同步时序电路和异步时序电路的区别。
3. 掌握计数器电路设计原理，用硬件描述语言实现指定功能的计数器设计。
4. 学会利用软件仿真实现对数字电路的逻辑功能进行验证和分析。

二、实验昵绒

1. 使用实验平台上两个数码管显示计数，手动单次时钟进行计数，时钟上升沿计数一次，当计数到 59 时，要求两个数码管都复位到 00 的状态，重新计数。
2. 实验还要求设置一个复位按键，可以随时重新恢复到 00 的状态继续计数。
3. 使用实验平台上的 $1MHz$ 时钟，将计数器改成秒表。
4. 在秒表中使用开关控制秒表启动、暂停。

三、实验代码及简要原理分析

1. D 触发器

```
module DTrigger(
    input wire D,
    input wire clk,
    input wire rst,
    output reg Q,
    output reg nQ
);

always @(posedge clk or posedge rst) begin
    if (rst) begin
        Q <= 1'b0;
        nQ <= 1'b1;
```

```
    end
    else begin
        Q <= D;
        nQ <= !D;
    end
end

endmodule

```

其中 D, Q, nQ 含义即对应一般的 D 触发器，clk 代表时钟信号，单次上升沿进行一次修改，rst = 1 代表异步清零。

2. 10 进制计数器

```
module Count10(
    input wire clk,
    input wire rst,
    output wire [3:0] data
);

wire [3:0] Q;
wire [3:0] nQ;
wire [3:0] D;

genvar i;

generate
    for (i = 0; i < 4; i = i + 1) begin: dt
        DTrigger di(.D(D[i]), .clk(clk), .rst(rst), .Q(Q[i]), .nQ(nQ[i]));
    end
endgenerate

assign D[0] = nQ[0];
assign D[1] = (nQ[0] ^ nQ[1]) & !(Q[0] & Q[3]);
assign D[2] = !((nQ[0] | nQ[1]) ^ Q[2]);
assign D[3] = !(Q[0] & Q[3]) & !(nQ[3] ^ (!(nQ[0] | nQ[1] | nQ[2])));
assign data = Q;

endmodule

```

通过实例化 4 个 D 触发器进行计数表示。

3. 6/3 进制计数器

```
module Count6(
    input wire clk,
    input wire rst,
    input wire v,
    output wire [2:0] data
);

wire [2:0] Q;
wire [2:0] nQ;
wire [2:0] D;

genvar i;

generate
    for (i = 0; i < 3; i = i + 1) begin: dt
        DTrigger di(.D(D[i]), .clk(clk), .rst(rst), .Q(Q[i]), .nQ(nQ[i]));
    end
endgenerate

//assign D[0] = v ^ Q[0];
assign D[0] = (!v & Q[0]) | (v & nQ[1] & nQ[0]);
//assign D[1] = (!v & Q[1]) | (v & (nQ[2] & (Q[1] ^ Q[0])));
assign D[1] = (!v & Q[1]) | (v & nQ[1] & Q[0]);
//assign D[2] = (!v & Q[2]) | (v & ((Q[2] & nQ[0]) | (Q[1] & Q[0])));
assign D[2] = 0;
assign data = Q;

endmodule

```

类比 10 进制计数器，只须修改从现态到次态的转移方程。

其中未注释版本为 3 进制计数器（当场临时要求），注释版本为 6 进制计数器。

为了后面与数码管信号相连，依然采用 4 个 D 触发器。

考虑到，当且仅当 10 进制计数器在 9 变化到 0 时才对 6/3 进制计数器产生贡献，因此特别增加 v 信号表示是否当前是否进行计数。

表达式的计算中只须分为 `v = 0` 和 `v = 1` 两类相或即可。

4. 60/30 进制计数器

```
`include "Dtrigger.v"
module Count60(
    input wire clk,
    input wire rst,
    output wire [3:0] DigitLow,
    output wire [3:0] DigitHigh
);

wire v;
wire D;

Count10 cnt10(.clk(clk), .rst(rst), .data(DigitLow));
Count6 cnt6(.clk(clk), .rst(rst), .v(v), .data(DigitHigh));
DTrigger d(.D(D), .clk(clk), .rst(rst), .Q(v));

assign D = DigitLow[3] & !DigitLow[2] & !DigitLow[1] & !DigitLow[0];

endmodule

```

主要即为对上述两个计数器信号的绑定，特别是需要将 6/3 进制计数器的 v 信号绑定上 10 进制计数器是否为 9。

而如果直接将此信号绑定在 v 上，可能会导致负载问题，于是增加了一个 D 触发器来获取信号传递给 v 以缓冲。

5. 总体计数器

```
module Counter(
    input wire clk,
    input wire rst,
    output wire [3:0] low,
    output wire [3:0] high
);

Count60 cnt(.clk(clk), .rst(rst), .DigitLow(low), .DigitHigh(high));

endmodule

```

即为对 60/30 进制计数器再进行包装（实际可直接利用 60/30 进制计数器）。

6. 秒表

```
module StopWatch(
    input wire sys_clk,
    input wire rst,
    input wire pause,
    output wire [3:0] low,
    output wire [3:0] high
);

reg clk;
reg [31:0] cnt;
wire lessHalf;
wire lessFull;

initial begin
    clk = 0;
    cnt = 0;
end

Counter counter(.clk(clk), .rst(rst), .low(low), .high(high));

always @(posedge sys_clk or posedge rst) begin
    if (rst) begin
        cnt <= 32'd0;
        clk <= 1'b0;
    end
    else if (sys_clk) begin
        if (!pause) begin
            if (lessFull) begin
                cnt <= cnt + 1'b1;
                if (lessHalf) begin
                    clk <= 1'b0;
                end
                else begin
                    clk <= 1'b1;
                end
            end
            else begin
                cnt <= 0;
                clk <= 1'b0;
            end
       end
    end
end

assign lessHalf = cnt < 32'd500000;
assign lessFull = cnt < 32'd1000000;
```

```
47
48 endmodule
49
```

由于板上时钟为 $1MHz$，因此若需要 $1Hz$ 的时钟，则须进行分频操作。

通过 cnt 记录当前 $1MHz$ 时钟次数，当 `cnt < 32'd500000` 时，时钟处于低电平，反之处于高电平，当 cnt 达到 1000000 时，将其清零即为完成一个周期。

pause 用于实现暂停功能，即当 pause 处于高电平时，不进行 cnt 的计数操作。

最终将 cnt 分频得到的新时钟 clk 传入 Counter 即可实现秒表。

7. 整合双功能（手动/自动）计数器

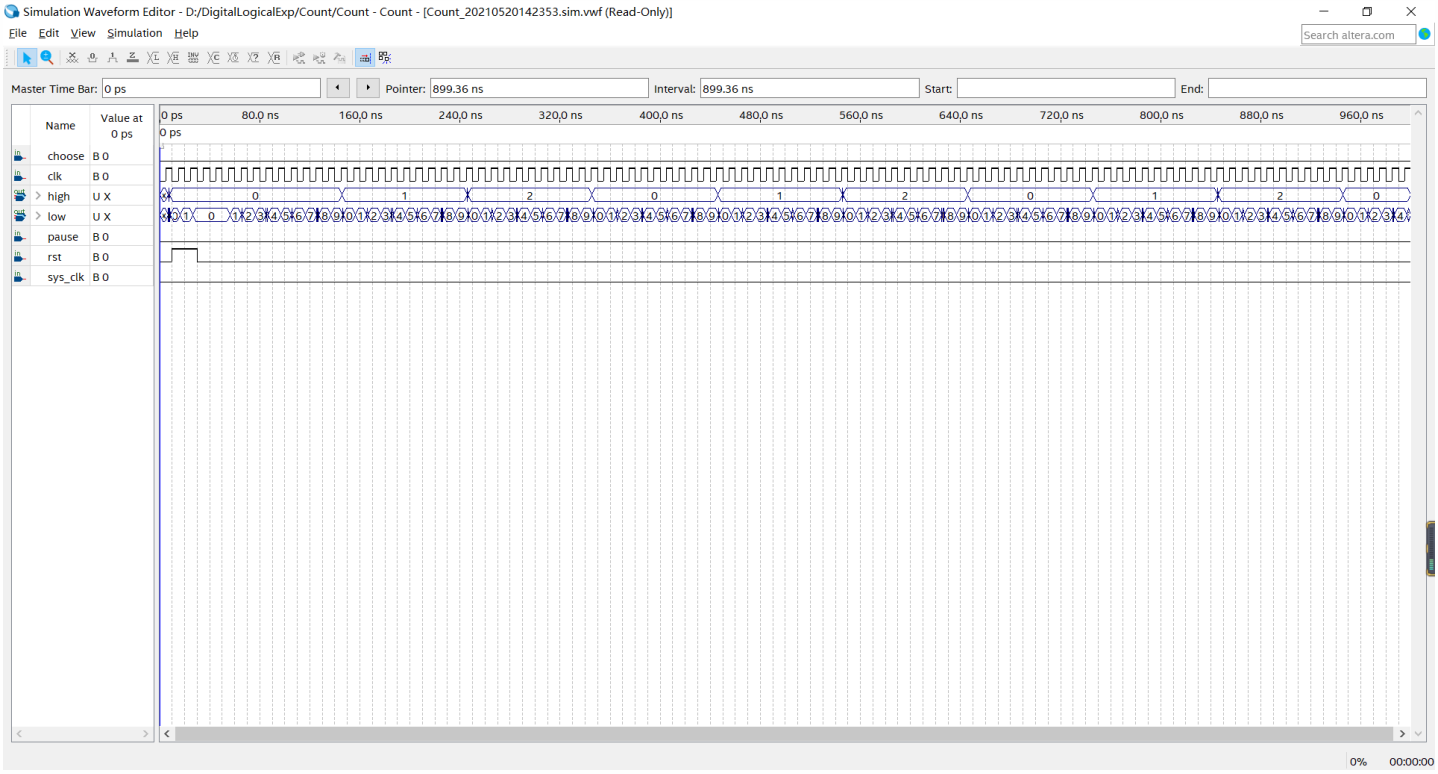
```
1  module Count(
2      input wire sys_clk,
3      input wire clk,
4      input wire rst,
5      input wire pause,
6      input wire choose,
7      output wire [3:0] low,
8      output wire [3:0] high
9  );
10
11 wire [3:0] low1;
12 wire [3:0] low2;
13 wire [3:0] high1;
14 wire [3:0] high2;
15
16 Counter counter(.clk(clk), .rst(rst), .low(low1), .high(high1));
17 StopWatch watch(.sys_clk(sys_clk), .rst(rst), .pause(pause), .low(low2),
   .high(high2));
18
19 assign low = choose ? low2 : low1;
20 assign high = choose ? high2 : high1;
21
22 endmodule
23
```

为了方便检查，将自动计数（秒表）与手动计数合并到一个电路上，通过 choose 表示选择手动还是自动。

其中 sys_clk 为 $1MHz$ 时钟，clk 为手动单次时钟。

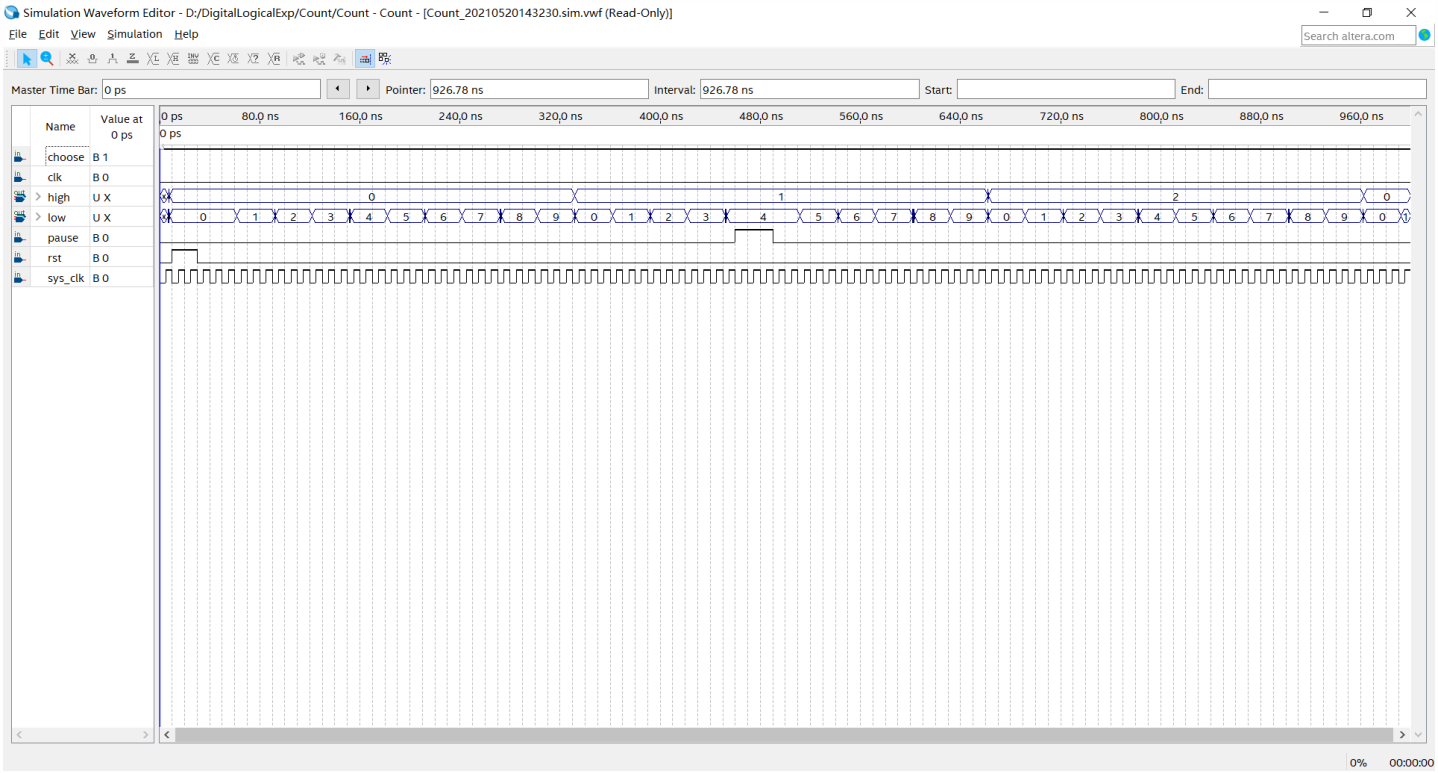
四、仿真分析

1. 计数器仿真



2. 秒表仿真

为方便分析，将分频器改为了 2 分频。



五、实验收获

通过实验，我了解到了结构化的设计方法，对于时序逻辑电路的设计有了更深的理解，对于触发器的原理和设计也有了更好的解释。

此外，我也体验到了一些电路设计中的竞争与冒险问题，对于设计上要做出的取舍有了一定实际体验。