浙江大学

本科实验报告

课程名称：	计算机逻辑设计基础
姓　　名：	龙永奇
学　　院：	计算机科学与技术学院
系：	本系
专　　业：	计算机科学与技术
学　　号：	3220105907
指导教师：	董亚波

2023 年 1 月 3 日

浙江大学实验报告

课程名称：计算机逻辑设计基础 实验类型：综合

实验项目名称：寄存器和寄存器传输设计

学生姓名：龙永奇 专业：计算机科学与技术 学号：3220105907

同组学生姓名：周楠 指导老师：董亚波

实验地点：东 4-509 实验日期：2023 年 12 月 21 日

1. 实验目的和要求

1. 掌握同步四位二进制计数器 74LS161 的工作原理和设计方法
2. 掌握时钟/定时器的工作原理与设计方法

二、实验内容和原理

内容：

1. 任务 1：采用行为描述设计同步四位二进制计数器 74LS161
2. 任务 2：基于 74LS161 设计时钟应用

原理：

1. 同步四位二进制计数器 74LS161

74LS161 是常用的四位二进制可预置的同步加法计数器,可灵活运用在各种数字电路，实现分频器等很多重要的功能.

其功能描述如下:

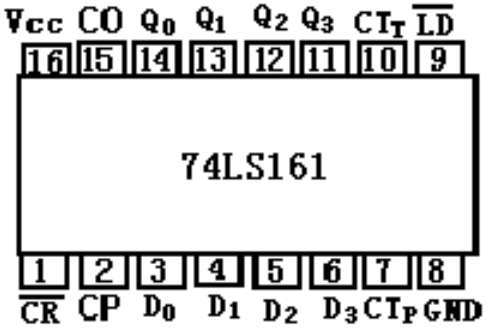

图 2.1 74LS161 引脚分布

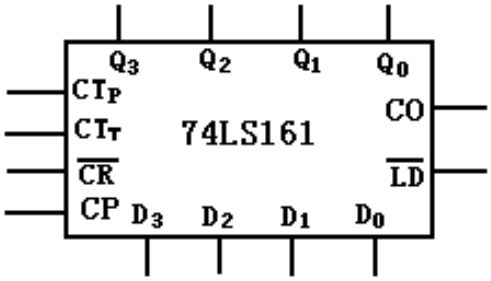

图 2.2 74LS161 引脚功能说明

- 异步清零端 CR：在任何时刻通过 CR 置零使得输出端 Q[3:0] = 0，此时为异步复位
- 同步加载端 LD：CR 不置零，LD = 0 时，在上升沿输出端加载输入端信号
- 使能端 CT_P、CT_T：当两个使能全为 1，CR = LD = 1 时，CP 上升沿时计数器加 1
- 进位输出端 CO：CO = Q0 Q1 Q2 Q3 CT_T

其功能表可表示为：

输　　入						输　出			
$\overline{CR}$	$\overline{LD}$	CT_P	CT_T	CP	$D_3D_2D_1D_0$	Q_3	Q_2	Q_1	Q_0
0	×	×	×	×	××××	0	0	0	0
1	0	×	×	↑	$d_3d_2d_1d_0$	d_3	d_2	d_1	d_0
1	1	0	1	×	××××	保　持			
1	1	×	0	×	××××	保　持			
1	1	1	1	↑	××××	计　数			

图 2.3 功能表

时序图如下：

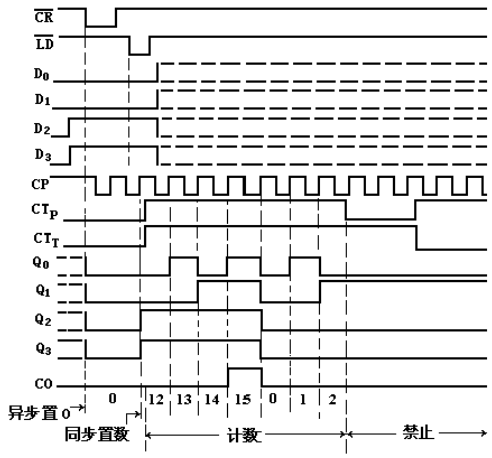

图 2.4 时序图

2. **利用 74LS161 实现十进制计数器**

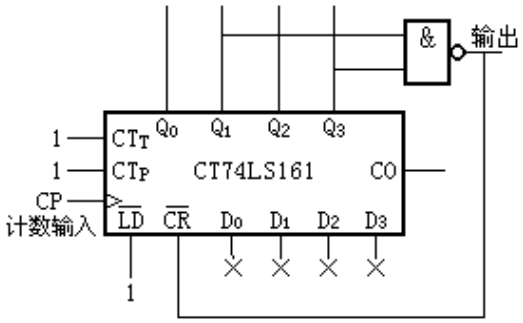

图 2.5 十进制计数器

利用与非门判断终止状态 1010，实现十进制计数 （0000 到 1001）

通过改变与非门的输入信号 ，可以实现其它进制计数。

改变与非门输出信号的功能和输入信号，可以实现同步加载。

3. **16×16 进制计数器**

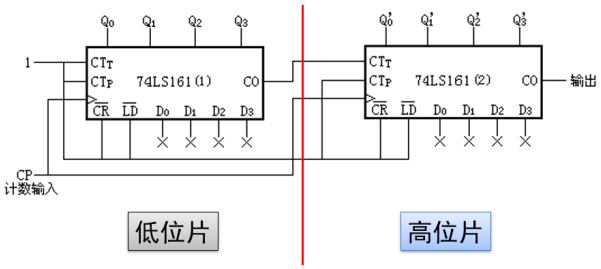

图 2.6 16×16 进制计数器

- 在计到 1111 以前，CO_1＝0，高位片保持原状态不变
- 在计到 1111 时，CO_1＝1，高位片在下一个 CP 加一

其实就是将两个 74LS161 芯片连起来，一个当高位使用，一个当低位使用

4. **实现 50 进制计数器**

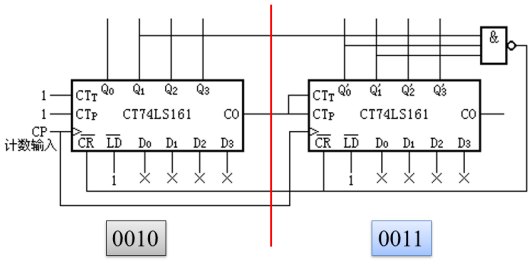

图 2.7 50 进制计数器

十进制数 50 对应的二进制数为 0011 0010，因此需要实现从 0000 0000 到 0011 0001 的 50 进制计数器

三、实验过程和数据记录

1. 采用行为描述设计同步四位二进制计数器 74LS161

(a)新建工程 My74LS161，Top Level Source 为 HDL

(b)新建 Verilog 类型源文件 My74LS161

(c)用 Verilog 代码方式设计，输入代码如下：

(d)建立仿真文件 shift_reg_sim.v，根据仿真要求：

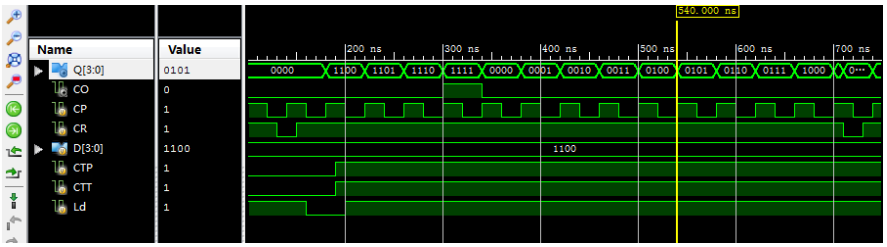

图 3.1 仿真要求

输入以下代码：

```
`timescale 1ns / 1ps
module my74ls161(
        input CR,
        input Ld,
        input CTP, CTT,
        input CP,
        input [3:0] D,
        output reg [3:0] Q,
        output CO
    );

     initial Q = 4'b0;

     always @ (posedge CP or negedge CR)
     begin
        if(CR == 0)
            Q <= 0;
        else if(Ld == 0)
            Q <= D;
        else
        begin
            if(CTT & CTP)
                Q <= Q + 1;
        end
```

```
    end

    assign CO = (&Q) & CTT;


endmodule
```

(e)得到波形图如下：

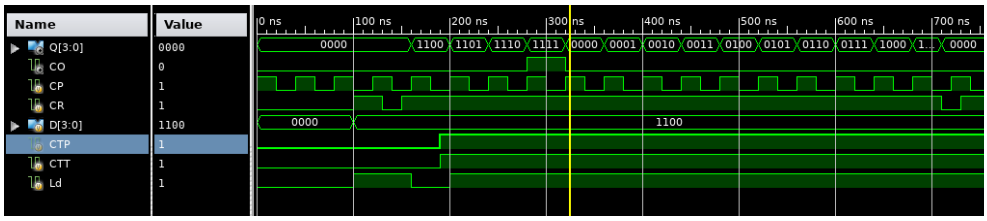

图 3.2 仿真波形图

仿真波形符合预期

2. 基于 74LS161 设计时钟应用

(a)新建工程 MyClock，Top Level Source 为 HDL

(b)使用行为描述设计，要求如下：

- 调用 My74LS161
- 调用分频模块，用 100ms 作为秒的驱动时钟
- 调用实验 13 的 8 位数码管显示模块显示时分秒
- 用 SW[15]初始化时钟为 23:58:30

(c)在工程中导入 Lab13 完成的 SEGP2S 模块

```
`timescale 1ns / 1ps
module SEGP2S(
        input wire clk,
        input [31:0] reg_num,
        input [7:0] LE,
        input start_sig,
        output SEG_CLK,
        output SEG_CLR,
        output SEG_EN,
        output SEG_DT,
        output wire [63:0] num
    );
        wire finish, start, SL;
        wire [63:0] disp_num;
        wire [64:0] Segment;
```

```
        assign SEG_CLK = clk | finish;
        assign SEG_CLR = 1'b1;
        assign SEG_EN = 1'b1;
        assign SEG_DT = Segment[64];

        assign finish = Segment[0] & Segment[1] & Segment[2] &
Segment[3] & Segment[4] & Segment[5] &Segment[6] & Segment[7]
                            & Segment[8] & Segment[9] &
Segment[10] & Segment[11] & Segment[12] & Segment[13] &
Segment[14] & Segment[15]
                            &Segment[16] & Segment[17]
&Segment[18] & Segment[19] & Segment[20] & Segment[21] &
Segment[22] & Segment[23]
                            & Segment[24] &Segment[25] &
Segment[26] &Segment[27] & Segment[28] & Segment[29] &
Segment[30] & Segment[31]
                            & Segment[32] & Segment[33]
&Segment[34] & Segment[35] &Segment[36] & Segment[37] &
Segment[38] & Segment[39]
                            & Segment[40] & Segment[41] &
Segment[42] &Segment[43] & Segment[44] &Segment[45] &
Segment[46] & Segment[47]
                            & Segment[48] & Segment[49] &
Segment[50] & Segment[51] &Segment[52] & Segment[53]
&Segment[54] & Segment[55]
                            & Segment[56] & Segment[57] &
Segment[58] & Segment[59] & Segment[60] &Segment[61] &
Segment[62] &Segment[63];

        MyMC14495 m0(.point(1'b0),.LE(LE[0]),.D0(reg_num[0]
                    ),.D1(reg_num[1]),.D2(reg_num[2]),.D3(reg_
num[3]
                    ),.a(disp_num[0]),.b(disp_num[1]),.c(disp_
num[2]
                    ),.d(disp_num[3]),.e(disp_num[4]),.f(disp_
num[5]
                    ),.g(disp_num[6]),.p(disp_num[7]));

        MyMC14495 m1(.point(1'b0),.LE(LE[1]),.D0(reg_num[4]
                    ),.D1(reg_num[5]),.D2(reg_num[6]),.D3(reg_
num[7]
                    ),.a(disp_num[8]),.b(disp_num[9]),.c(disp_
num[10
```

```
                    ]),.d(disp_num[11]),.e(disp_num[12]),.f(di
sp_num
                    [13]),.g(disp_num[14]),.p(disp_num[15]));

        MyMC14495 m2(.point(1'b0),.LE(LE[2]),.D0(reg_num[8]
                    ),.D1(reg_num[9]),.D2(reg_num[10]),.D3(reg
_num[
                    11]),.a(disp_num[16]),.b(disp_num[17]),.c(
                    disp_num[18]),.d(disp_num[19]),.e(disp_num
[20])
                    ,.f(disp_num[21]),.g(disp_num[22]),.p(disp
_num[
                    23]));

        MyMC14495 m3(.point(1'b0),.LE(LE[3]),.D0(reg_num[12
                    ]),.D1(reg_num[13]),.D2(reg_num[14]),.D3(r
eg_num
                    [15]),.a(disp_num[24]),.b(disp_num[25]),.c
(
                    disp_num[26]),.d(disp_num[27]),.e(disp_num
[28])
                    ,.f(disp_num[29]),.g(disp_num[30]),.p(disp
_num[
                    31]));

        MyMC14495 m4(.point(1'b0),.LE(LE[4]),.D0(reg_num[16
                    ]),.D1(reg_num[17]),.D2(reg_num[18]),.D3(r
eg_num
                    [19]),.a(disp_num[32]),.b(disp_num[33]),.c
(
                    disp_num[34]),.d(disp_num[35]),.e(disp_num
[36])
                    ,.f(disp_num[37]),.g(disp_num[38]),.p(disp
_num[
                    39]));

        MyMC14495 m5(.point(1'b0),.LE(LE[5]),.D0(reg_num[20
                    ]),.D1(reg_num[21]),.D2(reg_num[22]),.D3(r
eg_num
                    [23]),.a(disp_num[40]),.b(disp_num[41]),.c
(
                    disp_num[42]),.d(disp_num[43]),.e(disp_num
[44])
```

```
                    ,.f(disp_num[45]),.g(disp_num[46]),.p(disp
_num[
                    47]));

        MyMC14495 m6(.point(1'b0),.LE(LE[6]),.D0(reg_num[24
                    ]),.D1(reg_num[25]),.D2(reg_num[26]),.D3(r
eg_num
                    [27]),.a(disp_num[48]),.b(disp_num[49]),.c
(
                    disp_num[50]),.d(disp_num[51]),.e(disp_num
[52])
                    ,.f(disp_num[53]),.g(disp_num[54]),.p(disp
_num[
                    55]));

        MyMC14495 m7(.point(1'b0),.LE(LE[7]),.D0(reg_num[28
                    ]),.D1(reg_num[29]),.D2(reg_num[30]),.D3(r
eg_num
                    [31]),.a(disp_num[56]),.b(disp_num[57]),.c
(
                    disp_num[58]),.d(disp_num[59]),.e(disp_num
[60])
                    ,.f(disp_num[61]),.g(disp_num[62]),.p(disp
_num[
                    63]));

        SLReg9b
m8(.clk(clk), .S_L(SL) ,.s_in(1'b1),.p_in({disp_num[7:0] ,
1'b0}), .Q(Segment[8:0]));
        SLReg8b
m9(.clk(clk), .S_L(SL) ,.s_in(Segment[8]),.p_in({disp_num[15:8
]}), .Q(Segment[16:9]));
        SLReg8b
m10(.clk(clk), .S_L(SL) ,.s_in(Segment[16]),.p_in({disp_num[23
:16]}), .Q(Segment[24:17]));
        SLReg8b
m11(.clk(clk), .S_L(SL) ,.s_in(Segment[24]),.p_in({disp_num[31
:24]}), .Q(Segment[32:25]));
        SLReg8b
m12(.clk(clk), .S_L(SL) ,.s_in(Segment[32]),.p_in({disp_num[39
:32]}), .Q(Segment[40:33]));
        SLReg8b
m13(.clk(clk), .S_L(SL) ,.s_in(Segment[40]),.p_in({disp_num[47
:40]}), .Q(Segment[48:41]));
```

```
        SLReg8b
m14(.clk(clk), .S_L(SL) ,.s_in(Segment[48]),.p_in({disp_num[55
:48]}), .Q(Segment[56:49]));
        SLReg8b
m15(.clk(clk), .S_L(SL) ,.s_in(Segment[56]),.p_in({disp_num[63
:56]}), .Q(Segment[64:57]));

        SR_LATCH m16(.S(start & finish), .R(~finish),.Q(SL));
        Load_Gen
m17(.clk(clk), .btn_in(start_sig), .Load_out(start));

endmodule
```

(d)新建 Verilog 类型源文件 Top，代码如下：

```
`timescale 1ns / 1ps module Top(
                    input clk1,
                    input wire SW,
                    output SEG_EN,
                    output SEG_CLR,
                    output SEG_CLK,
                    output SEG_DT

                );
       wire clk;
       wire [7:0] hour;
       wire [7:0] min;
       wire [7:0] sec;

       wire [7:0] hour_in;
       wire [7:0] min_in;
       wire [7:0] sec_in;

       wire CTT;
       wire CTP;

       wire [23:0] num;
       wire [63:0] disp_num;

       assign CTT = 1;
       assign CTP = 1;
       assign num [7:0] = sec;
       assign num [15:8] = min;
       assign num [23:16] = hour;

       assign hour_in [7:0] = 8'b00100011;
```

```
        assign min_in [7:0] = 8'b01011000;
        assign sec_in [7:0] = 8'b00110000;

clk_100ms d0(.clk(clk1), .clk_100ms(clk));

My74LS161 m1(.CR(~(sec[3] &
          sec[1])), .Ld(SW), .CTP(CTP), .CTT(CTT), .CP(clk), .D
          (sec_in [3:0]), .Q(sec [3:0]));
My74LS161 m2(.CR(~(sec[6] & sec[5])), .Ld(SW), .CTP(sec[3] &
          sec[0]), .CTT(sec[3] & sec[0]), .CP(clk), .D(sec_in
          [7:4]), .Q(sec [7:4]));

My74LS161 m3(.CR(~(min[3] & min[1])), .Ld(SW), .CTP(sec[6] &
          sec[4] & sec[3] & sec[0]),
              .CTT(sec[6] & sec[4] & sec[3] &
          sec[0]), .CP(clk), .D(min_in [3:0]), .Q(min [3:0]));
My74LS161 m4(.CR(~(min[6] & min[5])), .Ld(SW), .CTP(min[3] &
          min[0] & sec[6] & sec[4] & sec[3] & sec[0]),
              .CTT(min[3] & min[0] & sec[6] & sec[4] & sec[3] &
          sec[0]), .CP(clk), .D(min_in [7:4]), .Q(min [7:4]));

My74LS161 m5(.CR(~(hour[2] & hour[5] | hour[3] &
          hour[1])), .Ld(SW),
              .CTP(min[6] & min[4] & min[3] & min[0] & sec[6] &
          sec[4] & sec[3] & sec[0]), .CTT(min[6] & min[4] &
          min[3] & min[0] & sec[6] & sec[4] & sec[3] & sec[0]),
              .CP(clk), .D(hour_in [3:0]), .Q(hour [3:0]));

My74LS161 m6(.CR(~(hour[2] & hour[5])), .Ld(SW),
              .CTP(hour[3] & hour[0] & min[6] & min[4] & min[3]
          & min[0] & sec[6] & sec[4] & sec[3] & sec[0]),
              .CTT(hour[3] & hour[0] & min[6] & min[4] & min[3]
          & min[0] & sec[6] & sec[4] & sec[3] & sec[0]),
              .CP(clk), .D(hour_in [7:4]), .Q(hour [7:4]));

SEGP2S m7(.clk(clk1), .reg_num({8'b00000000,
          num}), .LE(8'b11000000), .start_sig(clk), .SEG_EN(SEG
          _EN), .SEG_CLR(SEG_CLR), .SEG_CLK(SEG_CLK), .SEG_DT(S
          EG_DT), .num(disp_num));

endmodule
```

(e)建立仿真文件 MyClock_sim.v，根据仿真要求：

- 分频模块改为 128 个 clk 周期驱动秒计数

- 将时钟初始值改为 23:59:58
- 控制 clk 周期宽度和仿真时长，仿真到时钟输出 00:00:02
- 在 Top 模块中将 6 个 74LS161 模块的输出合并为 num[23:0]输出，在仿真时用 16 进制显示
- 同时输出 SEGCLK 和 SEGDT，观察是否有正确输出

将 top 代码修改为：

```
`timescale 1ns / 1ps
module Top(
        input clk1,
        input CTT,
        input CTP,
        input [7:0] hour_in,
        input [7:0] min_in,
        input [7:0] sec_in,
        input hour_ld,
        input min_ld,
        input sec_ld,
        output clk,
        output [23:0] num,
        output SEG_CLK,
        output SEG_DT,
        output [63:0] disp_num
    );
        wire [7:0] hour;
        wire [7:0] min;
        wire [7:0] sec;

        assign num[7:0] = sec;
        assign num[15:8] = min;
        assign num[23:16] = hour;

        clk_128 d0(.clk(clk1), .clk_128(clk));

        My74LS161
m1(.CR(~(sec[3]&sec[1])),.Ld(sec_ld),.CTP(CTP),.CTT(CTT),.CP(
clk),.D(sec_in[3:0]),.Q(sec[3:0]));//9
        My74LS161
m2(.CR(~(sec[6]&sec[5])),.Ld(sec_ld),.CTP(sec[3]&sec[0]),.CTT
(sec[3]&sec[0]),.CP(clk),.D(sec_in[7:4]),.Q(sec[7:4]));//9
```

```
        My74LS161
m3(.CR(~(min[3]&min[1])),.Ld(min_ld),.CTP(sec[6]&sec[4]&sec[3
]&sec[0]),
        .CTT(sec[6]&sec[4]&sec[3]&sec[0]),.CP(clk), .D(min_in
[3:0]), .Q(min[3:0]));
        My74LS161
m4(.CR(~(min[6]&min[5])),.Ld(min_ld),.CTP(min[3]&min[0]&sec[6
]&sec[4]&sec[3]&sec[0]),
        .CTT(min[3]&min[0]&sec[6]&sec[4]&sec[3]&sec[0]),.CP(c
lk),.D(min_in[7:4]),.Q(min[7:4]));

        My74LS161 m5(.CR(~(hour[2]&hour[5] |
hour[3]&hour[1])),.Ld(hour_ld),
        .CTP(min[6]&min[4]&min[3]&min[0]&sec[6]&sec[4]&sec[3]
&sec[0]),.CTT(min[6]&min[4]&min[3]&min[0]&sec[6]&sec[4]&sec[3
]&sec[0]),
        .CP(clk),.D(hour_in[3:0]),.Q(hour[3:0]));

        My74LS161 m6(.CR(~(hour[2]&hour[5])),.Ld(hour_ld),
        .CTP(hour[3]&hour[0]&min[6]&min[4]&min[3]&min[0]&sec[
6]&sec[4]&sec[3]&sec[0]),
        .CTT(hour[3]&hour[0]&min[6]&min[4]&min[3]&min[0]&sec[
6]&sec[4]&sec[3]&sec[0]),
        .CP(clk),.D(hour_in[7:4]),.Q(hour[7:4]));



        SEGP2S
m7(.clk(clk1),.reg_num(num),.LE(8'b11000000),.start_sig(clk),
 .SEG_CLK(SEG_CLK),.SEG_DT(SEG_DT),.num(disp_num));
endmodule
```

其中 clk_128 模块为：

```
`timescale 1ns / 1ps
module clk_128(clk, clk_128);
    input wire clk;
    output reg clk_128;
    reg [31:0] cnt;
    initial clk_128 = 0;
    always@(posedge clk) begin
        if(cnt < 64)begin
            cnt <= cnt + 1;
        end else begin
            cnt <= 0;
            clk_128 = ~clk_128;
        end
```

```
    end

endmodule
```

输入以下仿真代码：

```
`timescale 1ns / 1ps
module Top_sim;
    // Inputs
    reg clk1;
    reg CTT;
    reg CTP;
    reg [7:0] hour_in;
    reg [7:0] min_in;
    reg [7:0] sec_in;
    reg hour_ld;
    reg min_ld;
    reg sec_ld;

    // Outputs
    wire clk;
    wire [23:0] num;
    wire SEG_CLK;
    wire SEG_DT;
    wire [63:0] disp_num;

    // Instantiate the Unit Under Test (UUT)
    Top uut (
        .clk1(clk1),
        .CTT(CTT),
        .CTP(CTP),
        .hour_in(hour_in),
        .min_in(min_in),
        .sec_in(sec_in),
        .hour_ld(hour_ld),
        .min_ld(min_ld),
        .sec_ld(sec_ld),
        .clk(clk),
        .num(num),
        .SEG_CLK(SEG_CLK),
        .SEG_DT(SEG_DT),
        .disp_num(disp_num)
    );

    initial begin
        // Initialize Inputs
```

```
        clk1 = 0;
        CTT = 0;
        CTP = 0;
        hour_in = 0;
        min_in = 0;
        sec_in = 0;
        hour_ld = 0;
        min_ld = 0;
        sec_ld = 0;
        #100;

        CTT=0;
        CTP=0;
        sec_ld = 1;
        min_ld = 1;
        hour_ld = 1;

        hour_in[7:4]=2;
        hour_in[3:0]=3;
        min_in[7:4]=5;
        min_in[3:0]=9;
        sec_in[7:4]=5;
        sec_in[3:0]=8;
        CTT=0;
        CTP=0;
        #500;

        sec_ld = 0;
        min_ld = 0;
        hour_ld = 0;
        #3000;

        sec_ld = 1;
        min_ld = 1;
        hour_ld = 1;
        CTT=1;
        CTP=1;
        end

        always begin
        clk1=0;#10;
        clk1=1;#10;
        end
endmodule
```

得到仿真波形为：

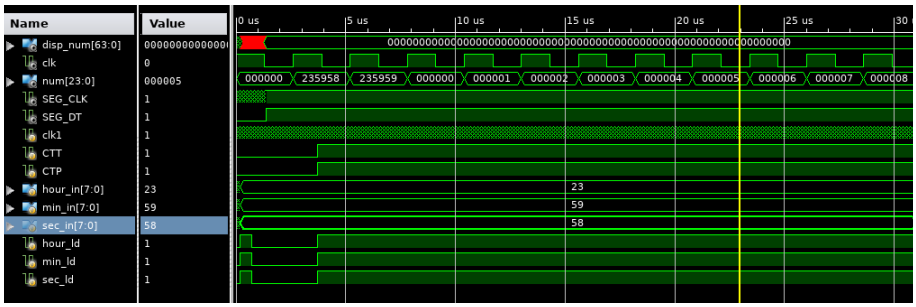

图 3.4 仿真波形

仿真开始时 Ld = 0，将 Ld 置为 1 后，在上升沿时读入 23 59 58，之后开始移位，可以看到 23 59 59 后 reset 变为 00 00 00

(f)建立 K7.ucf 文件，输入代码如下：

```
NET "clk1" LOC = AC18 | IOSTANDARD = LVCMOS18;
NET "SW" LOC = AF10 | IOSTANDARD = LVCMOS15;

NET "SEG_CLK"       LOC = M24   | IOSTANDARD = LVCMOS33;
NET "SEG_CLR"       LOC = M20   | IOSTANDARD = LVCMOS33;
NET "SEG_DT"        LOC = L24    | IOSTANDARD = LVCMOS33;
NET "SEG_EN"        LOC = R18    | IOSTANDARD = LVCMOS33;
```

(g)生成 bit 文件，上板验证

图 3.5 初始界面

图 3.6 开始

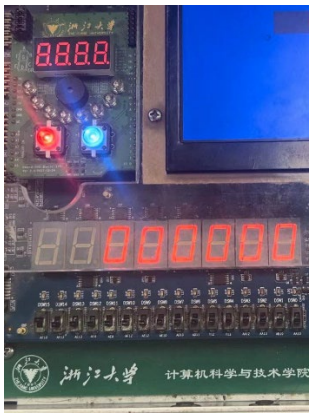

图 3.5 进位

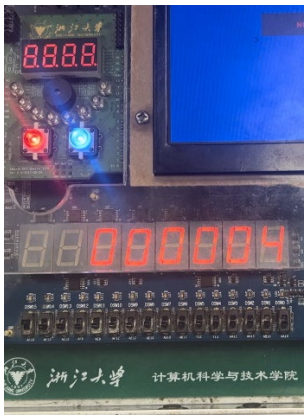

图 3.6 进位后

四、实验结果分析

本次实验通过仿真并使用 SWORD 板进行验证，实验结果符合预期。

在任务一的模块创建过程中，我和同组的同学学习到了对于 Q[3:0]这样的多位信号可以通过 &Q 对其每一位进行与操作，相比于每一位写出来可以使得代码更简洁。

五、讨论与心得

本次实验实现的 74LS161 结构简单明晰，并且应用范围非常广、功能强大，可以非常方便的调用此芯片设计任意计数器（详情见第二部分实验介绍）。通过设计自己的 Clock，我们也在自己的 project 上运用了此模块😊

感谢老师和同学在本学期的帮助,期末冲冲冲!