

浙江大学



计算机逻辑设计基础

Lab 14

Author: 苏煜程

Student ID: 3220105481

Date: 2024 年 1 月 4 日

浙江大学实验报告

课程名称：计算机逻辑设计基础

实验名称：计数器、定时器设计与应用

学生姓名：苏煜程 专业：人工智能（图灵班） 学号：3220105481

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

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

2 实验任务

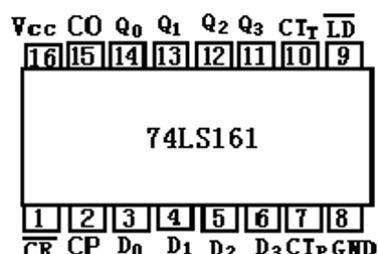
1. 采用行为描述设计同步四位二进制计数器 74LS161
2. 基于 74LS161 设计时钟应用

3 实验原理

3.1 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

3.1.1 功能描述

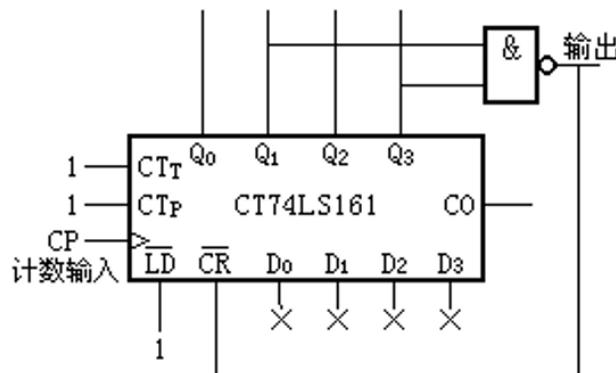


- 异步清零端 \overline{CR}
- 同步加载端 \overline{LD}
- 使能端 CT_P, CT_T , 当都为 1 时, 计数器工作
- 进位输出端 CO

输入					输出			
\overline{CR}	\overline{LD}	CT_p	CT_T	CP	$D_3 D_2 D_1 D_0$	$Q_3 Q_2 Q_1 Q_0$		
0	\times	\times	\times	\times		$\times \times \times \times$	0 0 0 0	
1	0	\times	\times	\uparrow		$d_3 d_2 d_1 d_0$	$d_3 d_2 d_1 d_0$	
1	1	0	1	\times		$\times \times \times \times$	保 持	
1	1	\times	0	\times		$\times \times \times \times$	保 持	
1	1	1	1	\uparrow		$\times \times \times \times$	计 数	

图 1: 功能表

3.2 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）
- 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

3.3 数字时钟应用设计

- 设计一个数字钟，使用 74LS161 模块，设计 60 进制和 24 进制计数器，实现 24 小时内时间的实时显示。
- 数字钟的初值通过计数器同步置位的方式实现，默认加载 23:58:30。选择大实验板上的 6 个数码管显示，前两位显示小时的十位和个位，中间两位显示分钟的十位和个位，最后两位显示秒的十位和个位。

4 实验内容与步骤

4.1 任务 1：采用行为描述设计同步四位二进制计数器 74LS161

- 新建工程，工程名称用 My74LS161，Top Level Source Type 用 HDL

2. 用行为描述设计 (CR 是异步清零, 低电平有效; LD 是同步置位, 低电平有效)

```
1 module My74LS161(
2     input wire CP,
3     input wire CR,
4     input wire CTP,
5     input wire CTT,
6     input wire LD,
7     input wire [3:0] D,
8     output reg [3:0] Q,
9     output wire Co
10 );
11
12     always @ (posedge CP or negedge CR) begin
13         if (~CR) Q <= 4'b0;
14         else if (~LD) Q <= D;
15         else if (CTP & CTT) Q <= Q + 1'b1;
16     end
17
18     assign Co = (Q == 4'b1111) ? 1'b1 : 1'b0;
19
20 endmodule
```

3. 仿真

仿真激励代码:

```
1 initial begin
2     // Initialize Inputs
3     CR = 0;
4     CTP = 0;
5     CTT = 0;
6     LD = 0;
7     D = 0;
8
9     // Wait 100 ns for global reset to finish
10    #100;
11
12    // Add stimulus here
13    CR = 1;
14    LD = 1;
15    D = 4'b1100;
16    CTT = 0;
17    CTP = 0;
18    #30;
19    CR = 0; #20;
```

```

20      CR = 1; #20;
21      LD = 0; #20;
22      CTT = 1;
23      CTP = 1; #20;
24      LD = 1; #20;
25
26      #400;
27
28      CTT = 0;
29      CTP = 0;
30      CR = 0; #20;
31      CR = 1; #20;
32  end
33
34  always begin
35      CP = 0; #10;
36      CP = 1; #10;
37  end

```

仿真结果：



可见，当 CTP, CTT 都为 1 且有 LD 信号时，开始计数。经过 CP 信号的一个周期，寄存器的值 Q 加一，且在 Q 为 15 时产生进位信号。当 CTP, CTT 不同时为 1 时，停止计数。符合预期。

4.2 任务 2：基于 74LS161 设计时钟应用

1. 新建工程，工程名称用 MyClock，Top Level Source Type 用 HDL

2. 用行为描述设计。

- 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

```

1 module MyClock(
2     input wire clk,
3     input wire LD,
4     output wire SEG_CLK,
5     output wire SEG_CLR,

```

```

6   output wire SEG_EN,
7   output wire SEG_DT
8 );
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52

```

```

    wire [7:0] num_hour, num_min, num_sec;
    wire [31:0] num;
    wire [1:0] Co;
    wire CP;
    wire clk_100ms;

    assign num = {num_hour, 4'b0, num_min, 4'b0, num_sec};
    assign CP = clk_100ms;

    clk_100ms clk0(.clk(clk),
                  .clk_100ms(clk_100ms));

    MyCounter60 cnt0(.CP(CP),
                     .CR(1'b1),
                     .CTP(1'b1),
                     .CTT(1'b1),
                     .LD(LD),
                     .D(8'b0011_0000),
                     .Q(num_sec),
                     .Co(Co[0]));

    MyCounter60 cnt1(.CP(CP),
                     .CR(1'b1),
                     .CTP(Co[0]),
                     .CTT(Co[0]),
                     .LD(LD),
                     .D(8'b0101_1000),
                     .Q(num_min),
                     .Co(Co[1]));

    MyCounter24 cnt2(.CP(CP),
                     .CR(1'b1),
                     .CTP(Co[0]),
                     .CTT(Co[1]),
                     .LD(LD),
                     .D(8'b0010_0011),
                     .Q(num_hour),
                     .Co());

    SEGP2S seg0(.clk(clk),
                .load(CP),
                .num(num),
                .SEG_CLK(SEG_CLK),

```

```

53      .SEG_CLR(SEG_CLR),
54      .SEG_EN(SEG_EN),
55      .SEG_DT(SEG_DT));
56
endmodule

```

3. 仿真

进行以下修改：

- 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 *SEG_CLK* 和 *SEG_DT*，观察是否有正确输出

```

1  module MyClock(
2    input wire clk,
3    input wire LD,
4    output wire SEG_CLK,
5    output wire SEG_CLR,
6    output wire SEG_EN,
7    output wire SEG_DT
8    output wire [31:0] num
9  );
10
11  wire [7:0] num_hour, num_min, num_sec;
12  // wire [31:0] num;
13  wire [1:0] Co;
14  wire CP;
15  // wire clk_100ms;
16
17  assign num = {num_hour, 4'b0, num_min, 4'b0, num_sec};
18  assign CP = clk_128;
19
20  /////////////////////////////////
21
22  wire clk_128;
23  wire [31:0] clk_div;
24
25  clkdiv clkdiv(.clk(clk),
26                .rst(1'b0),
27                .clkdiv(clk_div));
28

```

```

29 assign clk_128 = clk_div[6];
30
31 ///////////////////////////////////////////////////////////////////
32
33 // clk_100ms clk0(.clk(clk),
34 //                 .clk_100ms(clk_100ms));
35
36 MyCounter60 cnt0(.CP(CP),
37                     .CR(1'b1),
38                     .CTP(1'b1),
39                     .CTT(1'b1),
40                     .LD(LD),
41                     .D(8'b0011_0000),
42                     .Q(num_sec),
43                     .Co(Co[0]));
44
45 MyCounter60 cnt1(.CP(CP),
46                     .CR(1'b1),
47                     .CTP(Co[0]),
48                     .CTT(Co[0]),
49                     .LD(LD),
50                     .D(8'b0101_1000),
51                     .Q(num_min),
52                     .Co(Co[1]));
53
54 MyCounter24 cnt2(.CP(CP),
55                     .CR(1'b1),
56                     .CTP(Co[0]),
57                     .CTT(Co[1]),
58                     .LD(LD),
59                     .D(8'b0010_0011),
60                     .Q(num_hour),
61                     .Co());
62
63 SEGP2S seg0(.clk(clk),
64             .load(CP),
65             .num(num),
66             .SEG_CLK(SEG_CLK),
67             .SEG_CLR(SEG_CLR),
68             .SEG_EN(SEG_EN),
69             .SEG_DT(SEG_DT));
70
71 endmodule

```

仿真激励代码:

```

1 initial begin

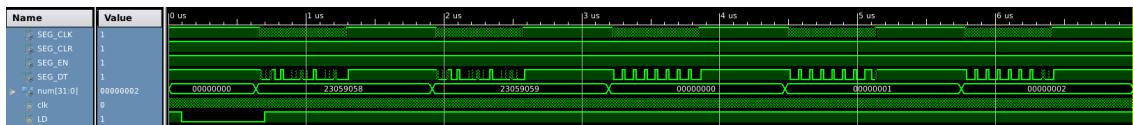
```

```

2 // Initialize Inputs
3 LD = 1;
4
5 // Wait 100 ns for global reset to finish
6 #100;
7
8 // Add stimulus here
9 LD = 0; #600;
10 LD = 1;
11
12 end
13
14 always begin
15   clk = 0; #5;
16   clk = 1; #5;
17 end

```

仿真结果：



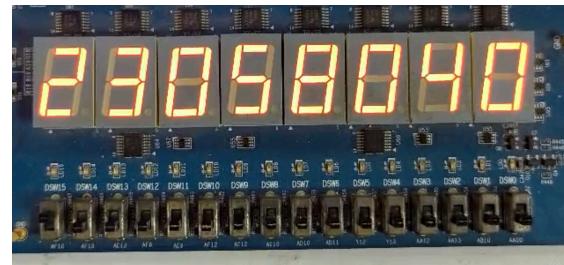
可见，时钟在 23:59:58 时开始计数，直到时钟变为 00:00:02，且 *SEG_CLK* 和 *SEG_DT* 有正确输出。符合预期。

4. 下载验证

将仿真时进行的修改还原，下载到实验板上，验证功能。



(a) 时钟初始值为 23:58:30



(b) 打开 SW[15] 开始计时



(c) 进位之前



(d) 进位

可见上板结果符合预期。

5 实验结果分析

实验结果在前一节已经分析，符合实验要求。

6 讨论与心得

本次实验工作量适中，没有遇到仿真结果和上板结果不一致的情况，总体还算顺利。