北 京 邮 电 大 学

实 验 报 告

课程名称 数字逻辑与数字电路实验

实验名称 在系统编程基础篇&提高篇

计算机学院 20232113XX 班 姓名 Yokumi

教师 李晶 成绩

2024 年 11 月 26 日

实验五 在系统编程基础篇

一、 实验目的

①熟悉数码管的两种驱动方法及其应用；

②掌握基本 verilog 语言的语法；

③学习使用 Quartus II 软件进行设计与仿真；

④学习在系统可编程器件的下载。

二、 实验器件和设备

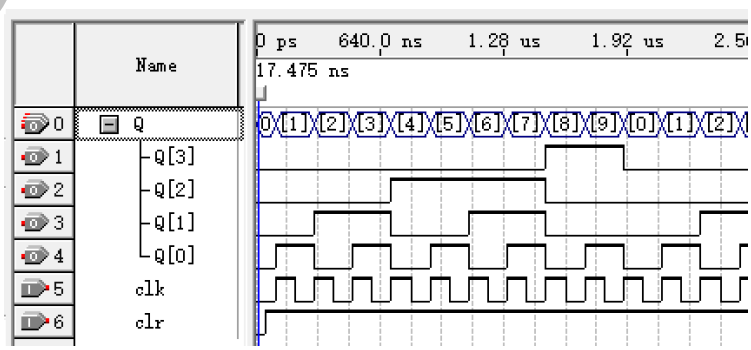
- TEC8 数字电路实验系统 1 台
- TBS1102B-EDU 双踪示波器 1 台
- QUARTUS II 软件

三、 实验过程及结果

问题描述：用 verilog 语言设计并测试一个 8421 码十进制计数器及七段数码管显示系统。

子任务 2.1：用 QUARTUS II 软件设计异步复位 8421 码十进制计数器，对该计数器进行功能仿真；

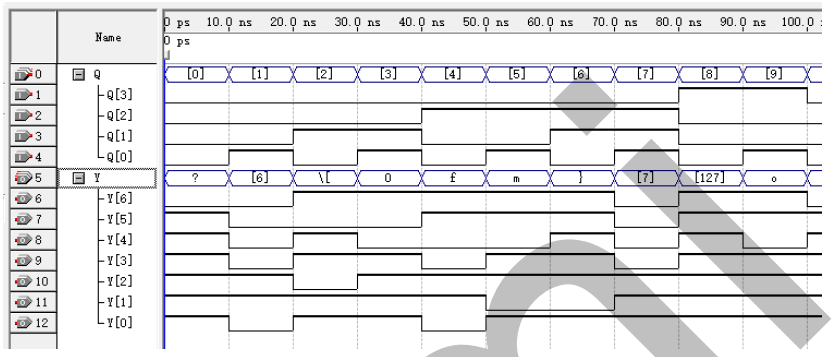
```
module counter_8421(clk, clr, Q);
    input wire clk;
    input wire clr;
    output reg [3:0] Q;

    always @(posedge clk ,negedge clr)
        begin
        if (!clr)
            begin
                Q <= 4'b0;
            end
        else
            begin
                if (Q == 4'b1001)
                    begin
                        Q <= 4'b0;
                    end
                else
                    begin
                        Q <= Q + 1;
                    end
            end
        end
endmodule
```



代码和仿真截图如上，实现思路较为基础，不赘述。设计的 8421 码十进制计数器为<u>上升沿有效</u>、<u>异步复位（复位端为低有效）</u>。

子任务 2.2：用 QUARTUS II 软件设计七段数码管显示，对该数码管显示进行功能仿真；

```
module display7(Q, Y);
    input wire [3:0] Q;

    output reg [6:0] Y;

    always @(*)
        begin
            case(Q)
                4'b0000: Y = 7'b0111111;
                4'b0001: Y = 7'b0000110;
                4'b0010: Y = 7'b1011011;
                4'b0011: Y = 7'b1001111;
                4'b0100: Y = 7'b1100110;
                4'b0101: Y = 7'b1101101;
                4'b0110: Y = 7'b1111101;
                4'b0111: Y = 7'b0000111;
                4'b1000: Y = 7'b1111111;
                4'b1001: Y = 7'b1101111;
                default: Y = 7'b0000000;
            endcase
        end
endmodule
```



代码和仿真截图如上，编写用到 case 语句，根据七段显示数码管的真值表进行译码，仿真经检查无误（出现乱码是由于视图问题以及七段显示数码管显示并不是十进制）。

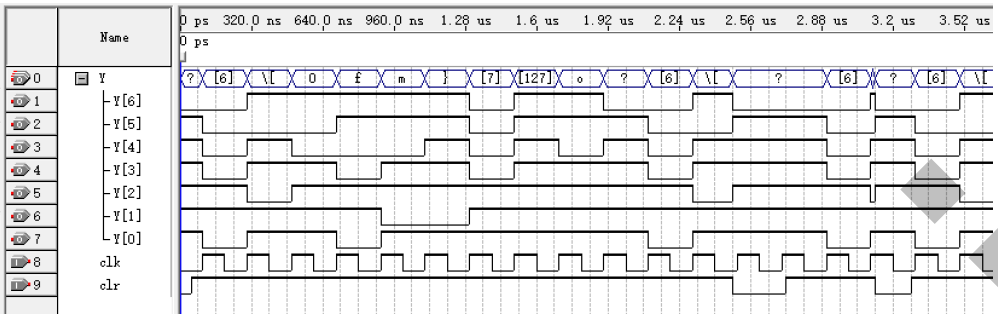
子任务 2.3：用 QUARTUSII 软件实现十进制计数器并采用七段数码管显示，对该系统进行功能仿真；

一、顶层文件代码：

```
1. module lab5(clk, clr, Y);
2.     input wire clk;
3.     input wire clr;
4.     output wire [6:0] Y;
5.
6.     wire [3:0] Q;
7.
8.     counter_8421 u1 (
9.         .clk(clk),
10.        .clr(clr),
11.        .Q(Q)
12.    );
13.
14.    display7 u2 (
15.        .Q(Q),
16.        .Y(Y)
17.    );
18.
19. endmodule
```

分别将基础模块 counter_8421 和 display7 实例化为 u1 和 u2，综合得到的文件，**输入有时钟信号 clk、清零信号 clr 和 7 位输出 Y**。

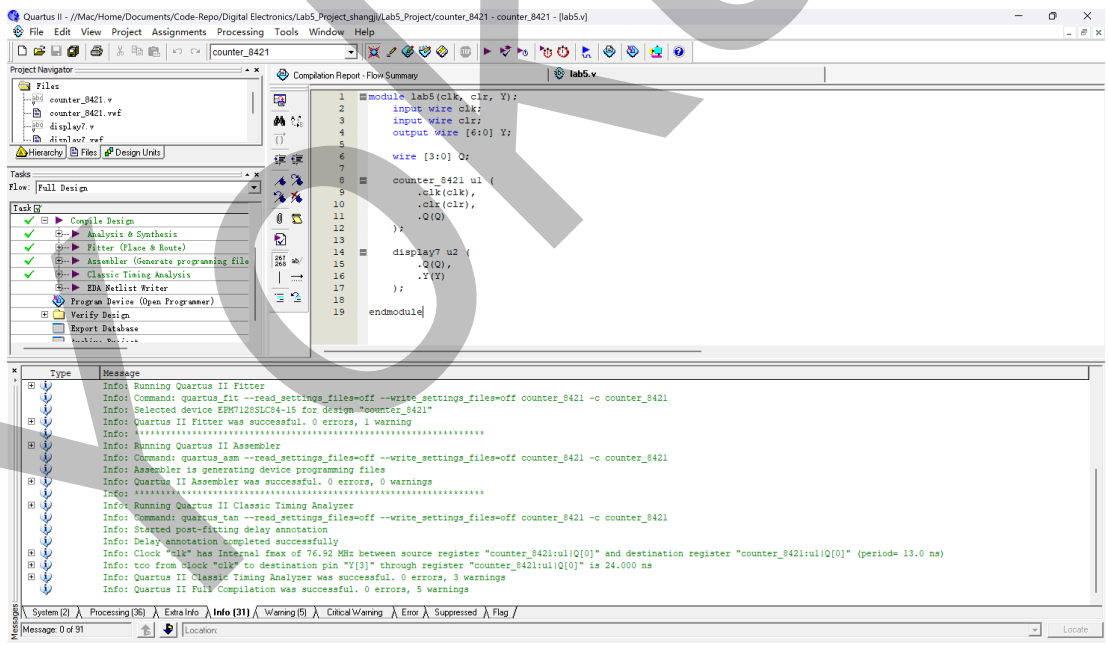
二、仿真结果：（前一段为计数器从 0 开始依次计数到最大的数码管的编码，后一段包含异步复位信号）



三、分配管脚图：

	Node Name	Direction	Location	I/O Bank	VREF Group	Reserved	Group	PCB layer
1	TCK	Input						
2	TDI	Input						
3	TDO	Output						
4	TMS	Input						
5	Y[6]	Output	PIN_51				Y[6..0]	
6	Y[5]	Output	PIN_50				Y[6..0]	
7	Y[4]	Output	PIN_49				Y[6..0]	
8	Y[3]	Output	PIN_48				Y[6..0]	
9	Y[2]	Output	PIN_46				Y[6..0]	
10	Y[1]	Output	PIN_45				Y[6..0]	
11	Y[0]	Output	PIN_44				Y[6..0]	
12	clk	Input	PIN_60					

四、综合编译结果：



四、 实验总结

通过实验五，熟悉了 Verilog 编程语言语法以及 Quartus2 软件的设计与仿真，熟悉了分配引脚以及将程序烧录到硬件上的全流程。

教训是扁平电缆方向要插对且插紧！插紧！插紧！

实验六 在系统编程提高篇

一、 实验目的

① 熟练掌握基本 verilog 语言的语法；

② 综合设计一个略复杂的数字电路；

二、 实验器件和设备

- TEC8 数字电路实验系统 1 台
- QUARTUS II 软件

三、 实验过程及结果

任务一：用 verilog 语言设计一个模 60 计数器

子任务 1.1：设计模 60 计数器

1. 源代码

```
module counter_60(clk, clr, method, Q, co);
    input wire clk;
    input wire clr;
    input wire method;

    output reg [5:0] Q;
    output reg co;

    always @(posedge clk, negedge clr)
        begin
            if (!clr)
                begin
                    Q <= 6'd0;
                    co <= 1'b0;
                end
            else
                begin
                    if (method)
                        begin
                            if (Q == 6'd59)
                                begin
                                    Q <= 6'd0;
                                    co <= 1'b1;
                                end
                            else
                                begin
                                    Q <= Q + 1;
                                    co <= 1'b0;
                                end
                        end
                    else
                        begin
                            if (Q == 6'd0)
                                begin
                                    Q <= 6'd59;
                                    co <= 1'b1;
                                end
                            else
                                begin
                                    Q <= Q - 1;
                                    co <= 1'b0;
                                end
                        end
                end
        end
endmodule
```

2. 引脚及功能分析：

- clk，时钟信号（clk，PIN_55）；
- clr，异步清 0 端（clr，PIN_1）；
- method，模式选择（*正计时 or 倒计时均可实现！*），（K0，PIN_54）；

子任务 1.2：将输出传送给数码管

```
module display_7(Q, Y1, Y2);
    input [5:0] Q;

    output [3:0] Y1;

    output [3:0] Y2;

    assign Y1 = Q / 10;
    assign Y2 = Q % 10;
endmodule
```

1. 源代码

2. 引脚及功能分析：

- Q，输入，接模 60 计数器的 6 位输出；
- Y1，计数的十位，输出给数码管 LG3（PIN_35、PIN_36、PIN_17、PIN_18）；
- Y2，计数的个位，输出个数码管 LG2（PIN_37、PIN_39、PIN_40、PIN_41）；

顶层设计：

1. 源代码

```
module lab6_counter60(clk, clr, method, Y1, Y2);
    input wire clk;
    input wire clr;
    input wire method;
    output wire [3:0] Y1;
    output wire [3:0] Y2;

    wire [5:0] Q;
    wire co;

    counter_60 u_counter_60 (
        .clk(clk),
        .clr(clr),
        .method(method),
        .Q(Q),
        .co(co)
    );

    display_7 u_display_7 (
        .Q(Q),
        .Y1(Y1),
        .Y2(Y2)
    );

endmodule
```

2. 实现功能及设计思路总结：

- 实现了秒表（即模 60 计时器）的基础功能；
- ***支持正计时和倒计时的切换***；
- 设计思路上，先设计模 60 计数器，在将计数传递给两个数码管，分别显示十位和个位（不需要译码，数码管的 4 个引脚分别对应四位二进制数）

任务二：用 verilog 语言设计一个电子琴；

子任务 2.1：按键对应

1. 源代码

```
module key_to_freq(key, m);
    input [7:0] key;
    output reg [15:0] m;

    always @(*) begin
        case (key)
            8'b00000001: m = 16'd1908;
            8'b00000010: m = 16'd1701;
            8'b00000100: m = 16'd1515;
            8'b00001000: m = 16'd1433;
            8'b00010000: m = 16'd1276;
            8'b00100000: m = 16'd1136;
            8'b01000000: m = 16'd1012;
            8'b10000000: m = 16'd956;
            default: m = 16'd0;
        endcase
    end
endmodule
```

2. 引脚及功能如下：

- key，对应 8 个电平开关输入（K0～K7，PIN）；
- m，按键对应频率，传递给模 m 计数器；

子任务 2.2：设计模 m 计数器

1. 源代码

```
module m_counter(clk, clr, m, out);
    input clk;
    input clr;
    input [15:0] m;
    output reg out;

    reg [15:0] count;

    always @(posedge clk, negedge clr) begin
        if (!clr) begin
            count <= 16'd0;
            out <= 0;
        end
        else if (m != 16'd0) begin
            if (count == m - 1) begin
                count <= 16'd0;
                out <= ~out;
            end
            else begin
                count <= count + 1;
            end
        end
    end
endmodule
```

2. 引脚及功能说明：

- clk，时钟信号，接 1MHZ 时钟源（PIN_55）；
- clr，异步清 0 信号（PIN_1）；
- m，频率；
- out，输出的分频，传递给二分频；

子任务 2.3：二分频

1. 源代码

```
module divide_2(clk_in, clr, clk_out);
    input clk_in;
    input clr;
    output reg clk_out;

    always @(posedge clk_in, negedge clr) begin
        if (!clr)
            clk_out <= 0;
        else
            clk_out <= ~clk_out;
    end
endmodule
```

2. 引脚及功能如下：

- clk_in，来自计数器的信号；
- clr，异步清 0 信号；
- clk_out，输出波形，传递给扬声器（PIN_52）；

顶层设计：

1. 源代码

```
module piano(clk, clr, key, speak
    input clk;
    input clr;
    input [7:0] key;
    output speaker;

    wire [15:0] m;
    wire m_clk;
    wire speaker_clk;

    key_to_freq key_to_freq_inst
        .key(key),
        .m(m)
    );

    m_counter m_counter_inst (
        .clk(clk),
        .clr(clr),
        .m(m),
        .out(m_clk)
    );

    divide_2 divide_2_inst (
        .clk_in(m_clk),
        .clr(clr),
        .clk_out(speaker_clk)
    );

    assign speaker = speaker_clk;
endmodule
```

2. 实现功能及设计思路：

- 电子琴的 8 个音弹奏；
- 设计思路上，由于驱动扬声器的必须是方波，所以要输出音符，则先 m 分频，再 2 分频。不需要使用奇数分频。

任务三：用 verilog 语言设计一个寄存器堆；

子任务 3.1：设计 8 位寄存器 R0-R3，负责存储数据

1. 源代码

```
module register_8(clk, clr, D, Q, en, CRW);
    input clk;
    input wire clr;
    input wire en;
    input wire CRW;
    input wire[7: 0] D;

    output reg[7: 0] Q;

    always @(posedge clk, negedge clr)
        begin
            if (!clr)
                Q <= 8'b0;
            else if (!en && CRW)
                Q <= D;
            else
                Q <= Q;
        end
endmodule
```

2. 引脚及功能如下：

- clk，时钟信号，大家共用（clk，PIN_55）；
- clr，异步清 0 端，低电平有效（clr，PIN_1）；
- en，使能信号，来自 2-4 译码器，任何一个时刻只有一个寄存器使能有效；
- CRW，写信号，共用 1 个，来自电平开关（K8，PIN_73）；
- D，数据，来自电平开关 K0～K7；
- Q，数据输出

子任务 3.2：设计 2-4 译码器，负责产生寄存器的使能信号

1. 源代码

```
module decode2_4(A, en_decode, Y);
    input wire [1:0] A;
    input wire en_decode;

    output reg [3:0] Y;

    always @(*)
        begin
            if (!en_decode)
                begin
                    Y = 4'b1111;
                end
            else
                begin
                    case(A)
                        2'b00: Y = 4'b1110;
                        2'b01: Y = 4'b1101;
                        2'b10: Y = 4'b1011;
                        2'b11: Y = 4'b0111;
                        default: Y = 4'b1111;
                    endcase
                end
        end
endmodule
```

2. 引脚及功能如下：

- A，2 位地址（A_write），来自电平开关 K9~K10（PIN_70、PIN_69）；
- en_decode，使能信号，低电平有效（K13，PIN_65）；
- Y，4 位输出，给寄存器的使能信号（分别给 reg1～4）；

子任务 3.3：设计 4 选 1 选择器，负责选择一路数据输出

1. 源代码

```
module mux_4 (A, D0, D1, D2, D3, Y);
    input wire [1:0] A;
    input wire [7:0] D0, D1, D2, D3;

    output reg [7:0] Y;
    always @(*) begin
        case (A)
            2'b00: Y = D0;
            2'b01: Y = D1;
            2'b10: Y = D2;
            2'b11: Y = D3;
            default: Y = 8'b0;
        endcase
    end
endmodule
```

2. 引脚及各功能如下：

- A，2 位地址（A_read）,决定选择哪个寄存器（K11～K12，PIN_68、PIN_67）；
- D0、D1、D2、D3，4 个 8 位数据输入，来自四个寄存器；
- Y，8 位数据输出；

子任务 3.4：设计三态门，负责控制数据的输出

1. 源代码

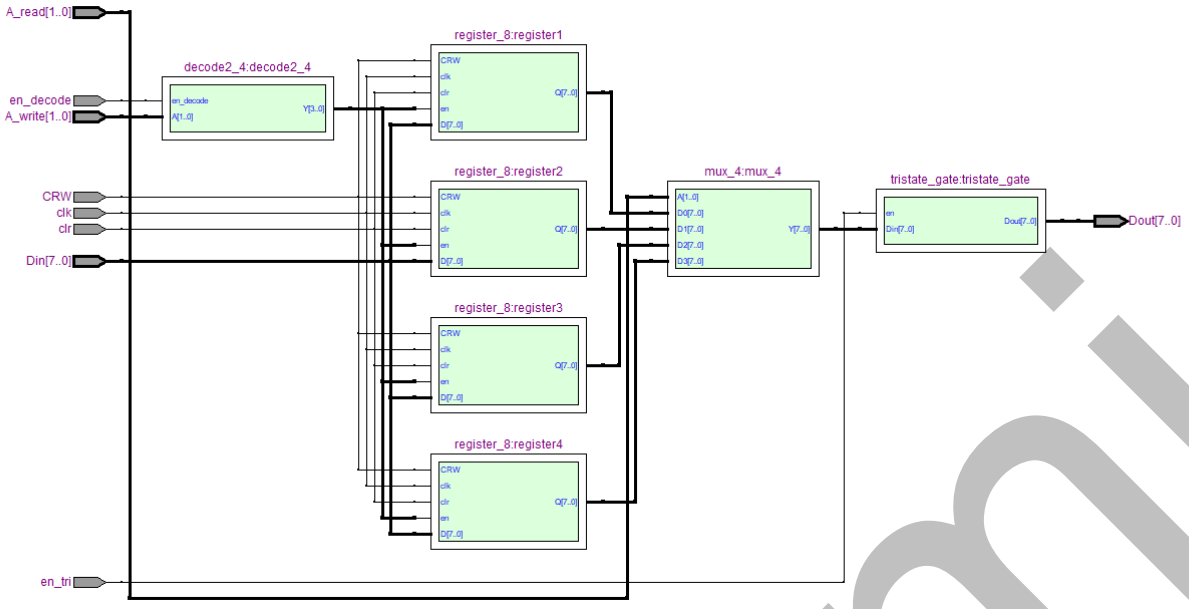
```
module tristate_gate(Din, en, Dout);
    input wire [7:0] Din;
    input wire en;

    output wire [7:0] Dout;

    assign Dout = !en ? Din : 8'bz;
endmodule
```

2. 引脚及各功能如下：

- Din，数据输入，来自数据选择器；
- Dout，数据输出，给 8 个 LED 灯；
- en，使能信号 en_tri，低电平有效，高电平时三态门高阻态（K14，PIN_64）。

顶层设计：

1. 网表文件（源代码较长）



2. 实现功能及设计思路：

- 寄存器堆的实现；
- 先设计基础模块，再通过顶层文件整合，将所有部件连接起来；