电子技术实验 2 实验报告

学号：2206113602
班级：信息 005
姓名：王靳朝

8 数字系统设计

一 题目描述

实验要求利用芯片和七段数码管制作一个简易的数字钟，六个七段数码管分别显示时、分、秒。

二 实验原理

本次实验采用自顶向下的设计方法，主要思路如下：

自顶向下的设计方法中，主要有一下几个模块：产生 1Hz 时钟信号的分频器，根据时钟信号分别产生时、分、秒的数字钟，由时钟信号执行选数码管功能的译码器，以及根据一码结果进行的扫描电路和七段数码管驱动电路。

1. 首先利用分频器，将 7Pin 产生的 24MHz 晶振转换为 1Hz 的时钟信号。
2. 根据时钟信号，时、分、秒设计各自的两个四位二进制数的进制转化。当秒低位为 9 时，须向上进位；特别的如果当秒达到 59 时，下一次时钟信号来高低位均需制为 0。以此类推，考虑分、时的进位条件，利用 if 语句完成 Verilog 语句。输出即为时、分、秒的高低位。
3. 根据时钟信号，在译码器部分分别设计控制具体哪一盏灯亮，在共阴极接法下，0 有效，1 无效。译码器输出结果选择当前时刻亮的灯。
4. 在扫描部分，控制扫描频率，在分频器部分设计出 1kHz 的扫描信号接入。根据译码器的结果驱动当前的灯亮灭。
5. 子模块完成后，将各个模块连接。首先是产生 1Hz 的时钟信号和 1kHz 的扫描信号。其次时钟信号送入进制转换电路和译码电路。进制转换电路产生的 6 个 4 为 2 进制输出结果送给扫描部分，同时扫描部分接入时钟信号和扫描信号。得到的结果时七段数码管的驱动，译码器的输出作为选管依据。

三 实验过程

产生时钟信号和扫描信号代码如下：

```
module clk24M(
    input               clk_in,
    output reg          clk_out1,  //输出1kHz频率信号，用于扫描
    output reg          clk_out2   //输出1Hz频率信号，用于计时
);

reg [14:0] count_1;
reg [9:0] count_2;

always @(posedge clk_in) begin
    if(count_1 == 14'd12000) begin
        count_1<=14'd0;
    end
    else begin
        count_1 <= count_1 + 14'd1;
    end
end

always @(posedge clk_out1) begin
    if(count_2 == 9'd500) begin
        count_2<=9'd0;
    end
    else begin
        count_2 <= count_2 + 9'd1;
    end
end


always @(posedge clk_in) begin
    if(count_1 == 14'd0)begin
        clk_out1 <= ~clk_out1;
    end
    else begin
        clk_out1 <= clk_out1;
    end
end

always @(posedge clk_out1) begin
    if(count_2 == 9'd0)begin
        clk_out2 <= ~clk_out2;
    end
    else begin
        clk_out2 <= clk_out2;
    end
end

endmodule
```

进位转换部分代码如下：

```
module shuzizhong(
    input  clk,

    output reg [3:0] s_low,
    output reg [3:0] s_high,
    output reg [3:0] m_low,
    output reg [3:0] m_high,
    output reg [3:0] h_low,
    output reg [3:0] h_high
);
```

```
//秒计数器
always @(posedge clk) begin
    if(s_low == 4'd9) begin
        s_low <= 4'd0;
    end
    else begin
        s_low <= s_low + 4'd1;
    end
end
always @(posedge clk) begin
    if((s_low == 4'd9) && (s_high < 4'd5)) begin
        s_high <= s_high + 4'd1;
    end
    else if((s_low == 4'd9) && (s_high  ==4'd5)) begin
        s_high <= 4'd0;
    end
    else if(s_high > 4'd5) begin
        s_high <= 4'd0;
    end
    else begin
        s_high <= s_high;
    end
end
```

```
//分计数器
always @(posedge clk) begin
    if((s_high == 4'd5) && (s_low == 4'd9)) begin
        if(m_low == 4'd9) begin
            m_low <= 4'd0;
        end
        else begin
            m_low <= m_low + 4'd1;
        end
    end
    else begin
        m_low <= m_low;
    end
end
always @(posedge clk) begin
    if((s_high == 4'd5) && (s_low == 4'd9)) begin
        if((m_low == 4'd9) && (m_high < 4'd5)) begin
            m_high <= m_high + 4'd1;
        end
        else if((m_low == 4'd9) && (m_high  ==4'd5)) begin
            m_high <= 4'd0;
        end
        else begin
            m_high <= m_high;
        end
    end
    else begin
        m_high <= m_high;
    end
end
```

```
//时计数器
always @(posedge clk) begin
    if((m_high == 4'd5) && (m_low == 4'd9)) begin
        if((s_high == 4'd5) && (s_low == 4'd9)) begin
            if((h_low == 4'd9) && (h_high < 4'd2)) begin
                h_low <= 4'd0;
            end
            else if((h_low == 4'd3) && (h_high == 4'd2)) begin
                h_low <= 4'd0;
            end
            else begin
                h_low <= h_low + 4'd1;
            end
        end
        else begin
            h_low <= h_low;
        endDS
    end
    else begin
        h_low <= h_low;
    end
end
```

```
always @(posedge clk) begin
    if((m_high == 4'd5) && (m_low == 4'd9)) begin
        if((s_high == 4'd5) && (s_low == 4'd9)) begin
            if((h_low == 4'd3) && (h_high  ==4'd2))  begin
                h_high <= 4'd0;
            end
            else if((h_low == 4'd9) && (h_high < 4'd2))begin
                h_high <= h_high + 4'd1;
            end
            else begin
                h_high <= h_high;
            end
        end
        else begin
            h_high <= h_high;
        end
    end
    else begin
        h_high <= h_high;
    end
end


endmodule
```

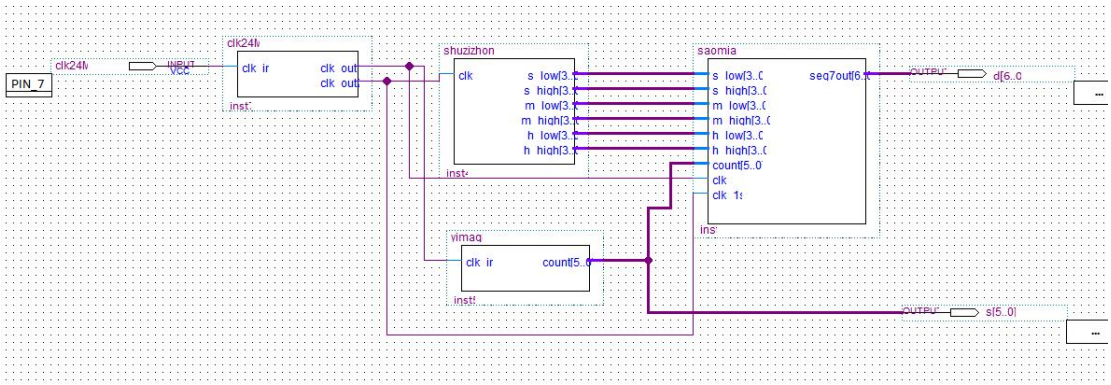
译码器部分代码如下：

```
module yimaqi(
    input           clk_in,
    output reg[5:0] count
);

always @(posedge clk_in) begin
    if(count == 6'b111110) begin
        count <= 6'b111101;
    end
    else if(count == 6'b111101) begin
        count <= 6'b111011;
    end
    else if(count == 6'b111011) begin
        count <= 6'b110111;
    end
    else if(count == 6'b110111) begin
        count <= 6'b101111;
    end
    else if(count == 6'b101111) begin
        count <= 6'b011111;
    end
    else if(count == 6'b011111) begin
        count <= 6'b111110;
    end
    else begin
        count <= 6'b111110;
    end
end


endmodule
```

扫描电路代码如下：

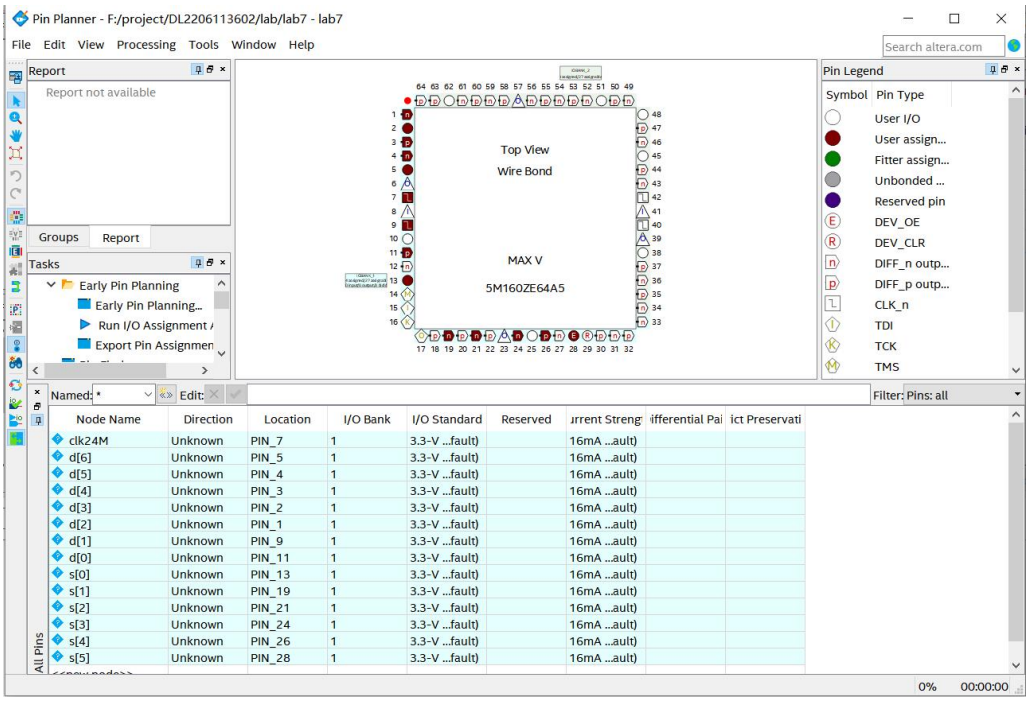
```
module saomiao(
        input [3:0] s_low,
        input [3:0] s_high,
        input [3:0] m_low,
        input [3:0] m_high,
        input [3:0] h_low,
        input [3:0] h_high,
        input [5:0] count,
        input clk,
        input clk_1s,
        output reg[6:0] seg7out
);

    reg[3:0] now;

always @(*) begin
    if(count == 6'b111110) begin
        now <= s_low;
    end
    else if(count == 6'b111101) begin
        now <= s_high;
    end
    else if(count == 6'b111011) begin
        now <= m_low;
    end
    else if(count == 6'b110111) begin
        now <= m_high;
    end
    else if(count == 6'b101111) begin
        now <= h_low;
    end
    else begin
        now <= h_high;
    end
    //数码管显像（高有效）

end

always @(*) begin
    case(now)
        4'd0:  seg7out <= 7'b1111110;
        4'd1:  seg7out <= 7'b0110000;
        4'd2:  seg7out <= 7'b1101101;
        4'd3:  seg7out <= 7'b1111001;
        4'd4:  seg7out <= 7'b0110011;
        4'd5:  seg7out <= 7'b1011011;
        4'd6:  seg7out <= 7'b1011111;
        4'd7:  seg7out <= 7'b1110000;
        4'd8:  seg7out <= 7'b1111111;
        4'd9:  seg7out <= 7'b1111011;
    endcase
end


endmodule
```

将子模块生成符号文件，连接得到 bdf 文件如下：



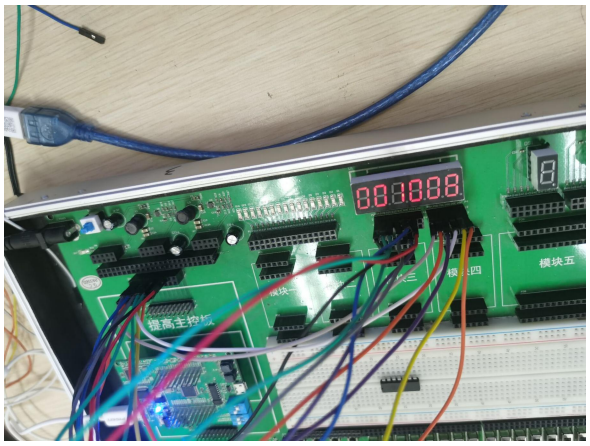
全部编译通过后分配管脚，7Pin 分配时钟信号即可，另外连接前确定高低位。管脚分

配结果如下，分配管脚之后再进行编译，并下载至芯片中。



四 实验结果

由于 24MHz 时钟信号过快无法进行仿真，故直接进行下载连接验证：



五 总结

1. 描述数字系统的设计方法

数字系统的设计方法主要有两种，分别是自底向上的设计方法和自顶向下的设计方法。

自底向上的电路设计是将各个子模块的功能制作和验证完成后，将不同模块进行组装达到最终目的。自顶向下的电路设计与前者相反，先分析顶层模块，在分析构成顶层模块必要的底层，最后制作、验证。

2. 本次实验中遇到的问题及解决方法

实验中主要遇到以下问题：在设计扫描电路的过程中对高低有效不清楚，连线时高低位

不明确。主要发现方法是当电路连接无误后数码管亮灭不正确，检查之后发现高低位设置错误。

3. 这门实验课程的学习体验和建议

本实验课程主要培养我运用知识解决问题的能力。课程主要学习的是利用所学的数字逻辑电路中的器件，通过电路图或者 Verilog 语句实现电路设计、搭建和验证的过程。在学习过程中，能够体会到理论知识在事件中的运用，能够对数字逻辑器件例如译码器、分频器等有更加清楚的认识。另外课程锻炼了我发现问题和解决问题的能力，在面对 Verilog 中未出现过的错误是能够自己查找资料解决，面对下载时出现的问题也能够采用合适的方法解决。

在实验过程中我的感受是时间有点紧张，尽管已经做了课前预习，也已经完成了一部分的代码，但是由于数字系统的设计需要进行验证，验证时出现的问题可能需要花费大量的时间去排查，因此 2 两小时的实验时间感觉不太够。