多功能数字时钟实验报告

1. 团队成员及分工

Lab_3 11612601　何志成　　蜂鸣器，闹钟与整点报时
Lab_2 11712742　郑云　　分频器
Lab_2 11612618　朱家明　　时钟控制与显示
Debug 与整合工作由我们一起完成

2. 功能介绍

2.1. 基本功能

① 正常显示时间
② 能通过拨码开关调时、调分、调秒
③ 可以整点报时

2.2. 附加功能

① 新增闹钟功能，并支持可视化设置闹钟时间（通过数码管显示）
② 整点报时和闹钟功能都增加了开关控制，用户可以灵活选择是否使用每个功能（更人性化）

3. 使用说明

如下图所示，我们用六个拨码开关作为系统的六个输入控制信号，六个七段数码管和蜂鸣器作为我们的输出组件。下面简要介绍输入及输出，详细信息可以参见 4、5 部分模块的详细信息。

输入：

1. Y9（key_h）：调时开关
2. W9（key_m）：调分开关
3. Y7（key_s）：秒针清零开关
4. Y8（key_al）：模式切换，当开关打开时进入设置闹钟时间模式，这时可以通过调时与调分两个开关变更闹钟时间，用户可

以通过数码管可视化设定闹钟，并且不会影响计时电路正常运行。开关关闭时即为正常时间显示模式，此时可以通过调时调分秒清零三个开关对计时电路进行设定。

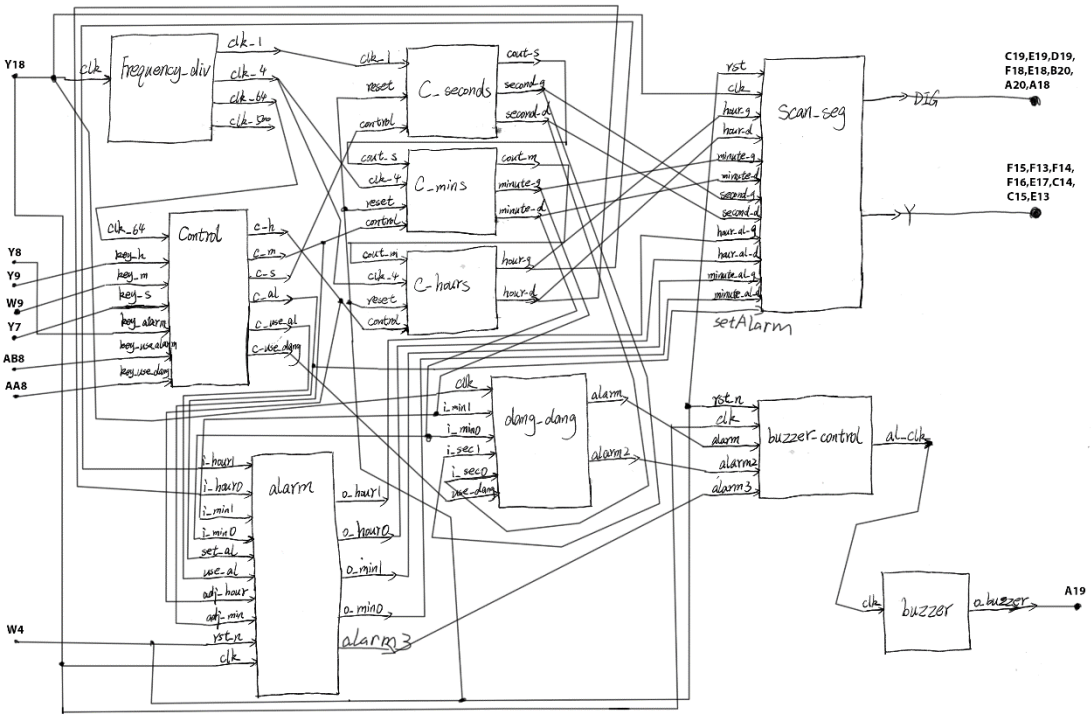
5. AB8（key_use_al）：闹钟开关，为 0 即不使用闹钟功能，为 1 则使用。
6. AA8（key_use_dang）： 整点报时开关，为 0 即不使用整点报时功能，为 1 则使用。

输出：

1. 时间显示由六个七段数码管完成
2. 整点报时和闹钟报时功能由蜂鸣器完成

Name	Direction	Neg Diff Pair	Package Pin	Fixed	Bank	I/O Std	Vcco	Vref	Drive Strength
All ports (25)									
DIG (8)	OUT			☑	16	LVCMOS33*	3.300		12
DIG[7]	OUT		A18	☑	16	LVCMOS33*	3.300		12
DIG[6]	OUT		A20	☑	16	LVCMOS33*	3.300		12
DIG[5]	OUT		B20	☑	16	LVCMOS33*	3.300		12
DIG[4]	OUT		E18	☑	16	LVCMOS33*	3.300		12
DIG[3]	OUT		F18	☑	16	LVCMOS33*	3.300		12
DIG[2]	OUT		D19	☑	16	LVCMOS33*	3.300		12
DIG[1]	OUT		E19	☑	16	LVCMOS33*	3.300		12
DIG[0]	OUT		C19	☑	16	LVCMOS33*	3.300		12
Y (8)	OUT			☑	16	LVCMOS33*	3.300		12
Y[7]	OUT		E13	☑	16	LVCMOS33*	3.300		12
Y[6]	OUT		C15	☑	16	LVCMOS33*	3.300		12
Y[5]	OUT		C14	☑	16	LVCMOS33*	3.300		12
Y[4]	OUT		E17	☑	16	LVCMOS33*	3.300		12
Y[3]	OUT		F16	☑	16	LVCMOS33*	3.300		12
Y[2]	OUT		F14	☑	16	LVCMOS33*	3.300		12
Y[1]	OUT		F13	☑	16	LVCMOS33*	3.300		12
Y[0]	OUT		F15	☑	16	LVCMOS33*	3.300		12
Scalar ports (9)									
clk	IN		Y18	☑	14	LVCMOS33*	3.300		
key_al	IN		Y8	☑	34	LVCMOS33*	3.300		
key_h	IN		Y9	☑	34	LVCMOS33*	3.300		
key_m	IN		W9	☑	34	LVCMOS33*	3.300		
key_s	IN		Y7	☑	34	LVCMOS33*	3.300		
key_use_al	IN		AB8	☑	34	LVCMOS33*	3.300		
key_use_dang	IN		AA8	☑	34	LVCMOS33*	3.300		
o_buzzer	OUT		A19	☑	16	LVCMOS33*	3.300		12
rst_n	IN		W4	☑	34	LVCMOS33*	3.300		

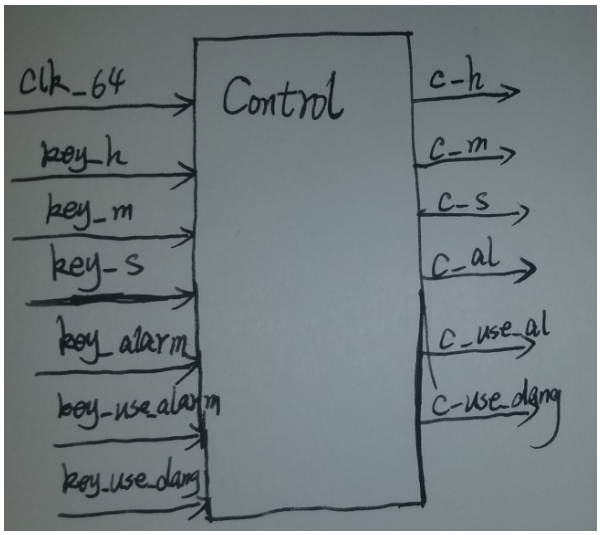
4. 模块总览



5. 子模块详情

5.1. 控制模块

5.1.1. 端口



Input:

clk_64: 64Hz 的时钟信号

key_h: 调时开关，打开时小时迅速递增

key_m: 调分开关，打开时分钟迅速增加

key_s: 调秒开关，打开时秒清零

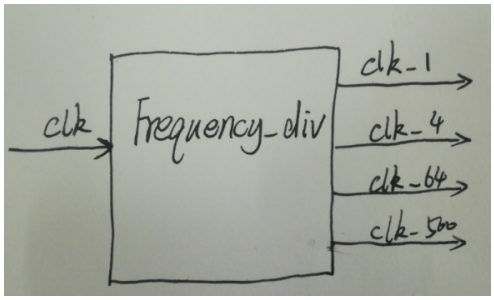
key_alarm: 设置闹钟开关，打开时可设置闹钟时间

key_use_alarm: 闹钟开关
key_use_dang: 整点报时开关

Output：对应 input 的六个控制开关，存储控制信号

5.2. 分频模块

5.2.1. 端口



Input:

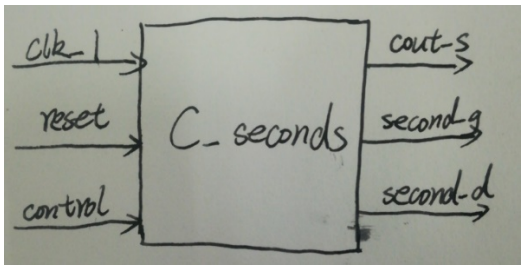
clk: fpga 开发板自带的 100MHz 的时钟

Output：

clk_1: 1Hz 时钟信号
clk_4: 4Hz 时钟信号
clk_64: 64Hz 时钟信号
clk_500: 500Hz 时钟信号

5.3. 秒计数器

5.3.1. 端口



Input:

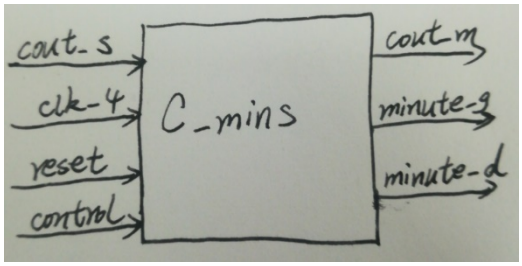
clk_1: 1Hz 时钟
reset: reset 信号
control: 调秒控制信号

Output：

cout_s: 秒进位，1bit 位宽
second_g: 秒的十位，4bit 位宽
second_d: 秒的个位，4bit 位宽

5.4. 分计数器

5.4.1. 端口



Input:

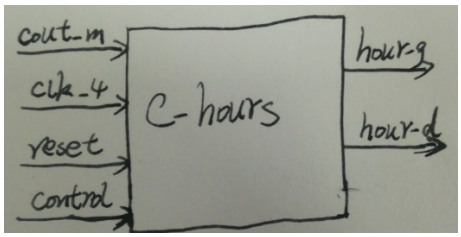
cout_s: 秒进位作为分钟模块的驱动时钟
clk_4: 4Hz 时钟，用来调分时迅速递增
reset: reset 信号
control: 调分控制信号

Output:

cout_m: 分进位，1bit 位宽
minute_g: 分的十位，4bit 位宽
minute_d: 分的个位，4bit 位宽

5.5. 时计数器

5.5.1. 端口



Input:

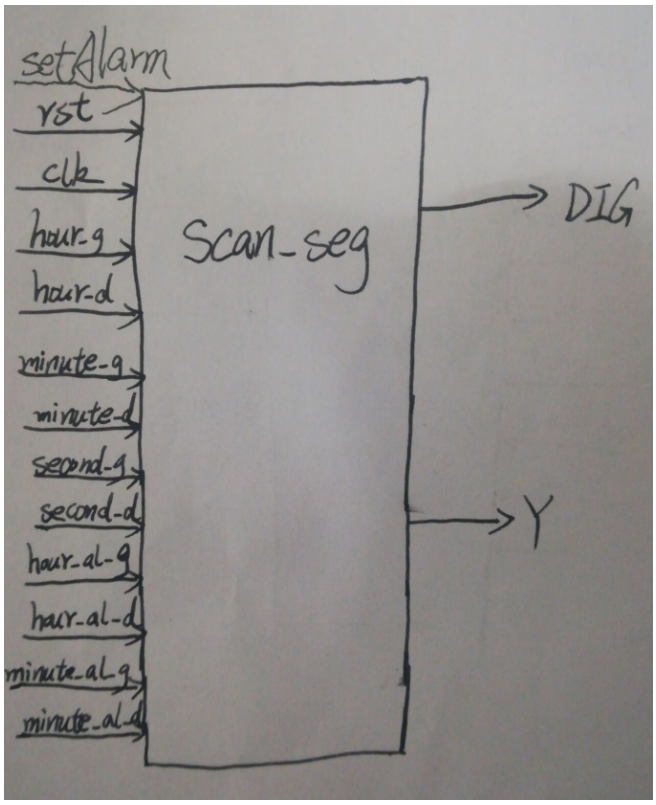
cout_m: 分进位作为小时模块的驱动时钟
clk_4: 4Hz 时钟，用来调时时迅速递增
reset: reset 信号
control: 调时控制信号

Output:

hour_g: 小时的十位，4bit 位宽
hour_d: 小时的个位，4bit 位宽

5.6. 显示模块

5.6.1. 端口



Input:

setAlarm: 设置闹钟开关信号，为零则正常显示时间，为 1 则为设置闹钟时间模式（此时并不会影响时间的正常计数）

clk: fpga 开发板自带的 100MHz 的时钟

rst: reset 信号

hour_g: 时的十位，4bit 位宽

hour_d: 时的个位，4bit 位宽

minute_g: 分的十位，4bit 位宽

minute_d: 分的个位，4bit 位宽

second_g: 秒的十位，4bit 位宽

second_d: 秒的个位，4bit 位宽

hour_al_g: 闹钟时的十位，4bit 位宽

hour_al_d: 闹钟时的个位，4bit 位宽

minute_al_g: 闹钟分的十位，4bit 位宽

minute_al_d: 闹钟分的个位，4bit 位宽

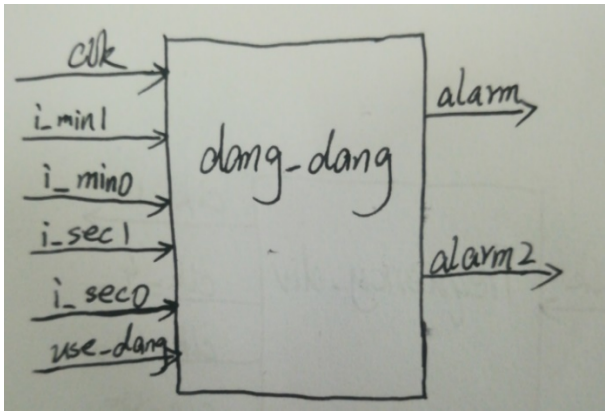
Output：

DIG: 七段数码管控制信号，8bit 位宽

Y: 七段数码管使能信号，8bit 位宽

5.7. 报时模块

5.7.1. 端口



Input:

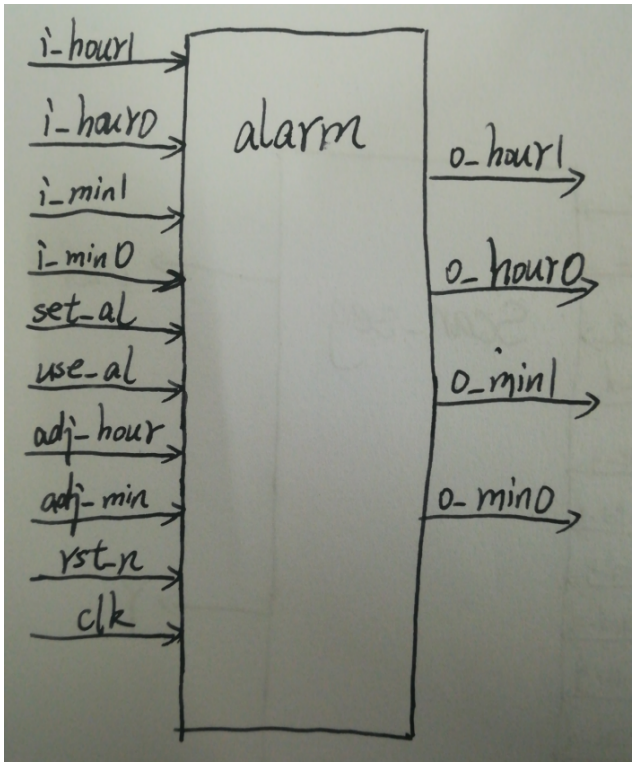
clk: fpga 开发板自带的 100MHz 的时钟
i_min1: 当前时间分的十位，4bit 位宽
i_min0: 当前时间分的个位，4bit 位宽
i_sec1: 当前时间秒的十位，4bit 位宽
i_sec0: 当前时间秒的个位，4bit 位宽
use_dang: 整点报时开关

Output：

alarm: 用于整点报时的 1000Hz 时钟
alarm2: 用于整点报时的 2000Hz 时钟

5.8. 闹时模块

5.8.1. 端口



Input:

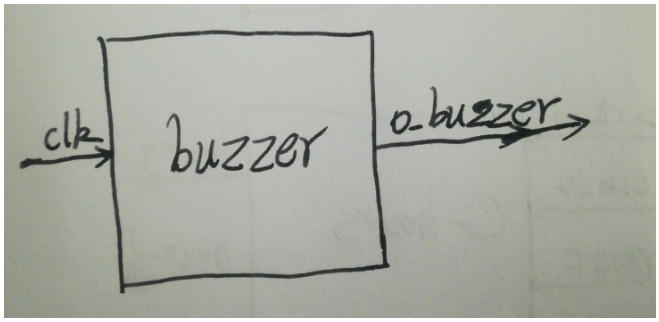
i_min1: 当前时间分的十位，4bit 位宽
i_min0: 当前时间分的个位，4bit 位宽
i_sec1: 当前时间秒的十位，4bit 位宽
i_sec0: 当前时间秒的个位，4bit 位宽
set_al: 设置闹钟开关，打开即进入设置闹钟模式
use_al: 闹钟开关
adj_hour: 调整闹钟小时控制信号
adj_min: 调整闹钟分钟控制信号
rst_n: reset 信号
clk: 4Hz 时钟信号

Output：

o_hour1: 设置的闹钟小时的十位，4bit 位宽
o_hour0: 设置的闹钟小时的个位，4bit 位宽
o_min1: 设置的闹钟分钟的十位，4bit 位宽
o_min0: 设置的闹钟分钟的个位，4bit 位宽

5.9. 蜂鸣器

5.9.1. 端口



Input:

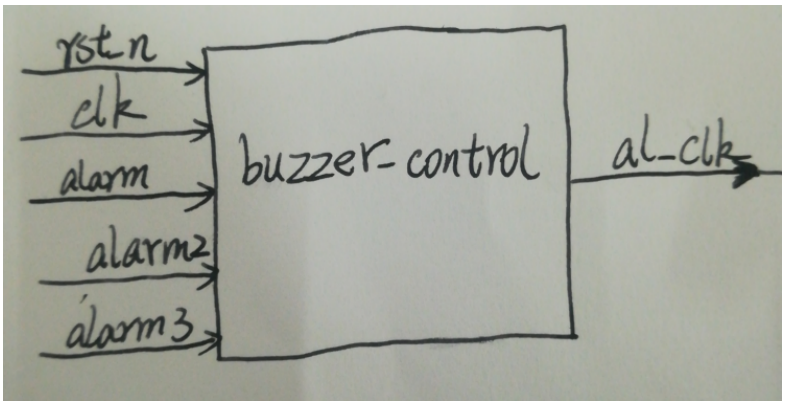
clk: 驱动时钟信号

Output：

o_buzzer: 蜂鸣器控制信号，输出到 A19

5.10. 蜂鸣器控制器

5.10.1. 端口



Input:

clk: fpga 开发板自带的 100MHz 的时钟

alarm: 用于整点报时的 1000Hz 时钟（每次满足条件时维持 1 秒）

alarm2: 用于整点报时的 2000Hz 时钟（每次满足条件时维持 1 秒）

alarm3: 用于闹钟的 2000Hz 时钟（每次满足条件时维持 1 分钟）

Output:

al_clk: 蜂鸣器控制时钟

6. 设计说明

如第 5 点所示，整个项目由十个子模块组成，以下是将所有模块组合起来的核心代码，各个子模块的代码详见源代码压缩包。

```
`timescale 1ns / 1ps

module digital_c(
  input key_h,key_m,key_s,key_al, key_use_al, key_use_dang,  //  时钟控制开关
  input clk,rst_n,   // 时钟和 reset 信号
  output o_buzzer,  // 蜂鸣器控制信号输出
  output [7:0] DIG,Y // 数码管控制信号输出
  );
  //  分频器，分出多个频率供其他模块使用
  wire o_clk1,o_clk4,o_clk64,o_clk500;
  clock_div u1(clk,rst_n,o_clk1,o_clk4,o_clk64,o_clk500);

  // 控制模块，读取输入
  wire c_h,c_m,c_s,c_al,c_use_al,c_use_dang;
  control
u2(o_clk64,key_h,key_m,key_s,key_al,key_use_al,key_use_dang,c_h,c_m,c_s,c_al,c_use_al,c_use_
dang);

  wire                                                    [3:0]
second_g,second_d,minute_g,minute_d,hour_g,hour_d,hour_g_al,hour_d_al,minute_g_al,minute_d_a
```

```
l;
  wire cout_s,cout_m;
  // 秒计数器
  C_seconds u3(o_clk1,rst_n,c_s,second_g,second_d,cout_s);

  // 分计数器
  C_mins u4(cout_s,o_clk4,c_al,rst_n,c_m,minute_g,minute_d,cout_m);

  // 时计数器
  C_hours u5(cout_m,o_clk4,c_al,rst_n,c_h,hour_g,hour_d);

  wire alarm, alarm2, alarm3;
  // 整点报时模块
  dang_dang u7(clk, minute_g, minute_d, second_g, second_d, c_use_dang, alarm, alarm2);

  // 闹钟模块
  alarm u8(hour_g,hour_d,minute_g,minute_d, c_al, c_use_al, c_h, c_m, rst_n, o_clk4, hour_g_al,
hour_d_al, minute_g_al, minute_d_al, alarm3);

  wire o_clkb;
  // 蜂鸣器控制模块
  buzzer_control u9(clk, rst_n, alarm, alarm2, alarm3, o_clkb);

  // 蜂鸣器驱动模块
  buzzer u0(o_clkb, o_buzzer);

  // 数码管显示模块
  scan_seg
u6( rst_n,clk,c_al,hour_g,hour_d,minute_g,minute_d,second_g,second_d,hour_g_al,hour_d_al,min
ute_g_al,minute_d_al,DIG,Y);

endmodule
```

7. 实验总结

7.1. 遇到的问题

1. 时分秒三个模块的进位与时钟的配置没弄明白，最开始这三个模块直接都用 1Hz 时钟，导致小时显示出错。经同学指点后才豁然开朗，解决了此问题。
2. 七段数码管的流水灯显示没弄懂，导致时间显示出错。
3. 在答辩时蜂鸣器无法使用的严重错误，经过长时间的 debug 发现竟然是蜂鸣器（A19）的 output 被误打成 input 导致

7.2. 创新点

人性化设计，整点报时和闹钟功能都增加了开关控制，用户可以灵活选择是否使用每个功能

7.3. 优化

1. 优化了蜂鸣器模块，当初不响是因为粗心将输出写成了输入。改正之后即可正常整点报时。
2. 新增闹钟功能，并能通过 setAlarm 控制信号，通过数码管实现可视化设置闹钟时间。
3. 闹钟模块与整点报时模块采用系统时钟，加快了反应速度

7.4. 经验总结

1. 时序逻辑电路不同于组合逻辑电路，使用的时非阻塞赋值，即先计算所有（RHS）右边表达式的值再完成对左边寄存器的赋值操作。这样就很好的解决了是我困惑很久的时分秒进位问题。
2. 流水灯显示的方法利用人眼的视觉停留，以极高的频率循环刷新每个数码管，既达到了显示效果，还节约了能源，是个非常赞的设计。
3. 小小的拼写错误也可能导致项目全崩，写代码时要特别小心
4. 绝对不能熬夜赶项目!!!