实验六 多功能电子时钟设计与实现 实验报告

小组成员	负责分工	备注
杨昊杰	闹钟、报警、功能集成、debug	以组为单位，每组一份项目报告
宋京润	存储、24 小时计时、报告撰写、debug	
张前锋	秒表、时间设定、仿真测试、debug	

摘要

在电子计数产品高度普及的时代，电子设计自动化计数得到广泛地应用与发展。电子设计自动化计数的主流采用“自顶向下”的设计方法，允许设计者在设计早期阶段发现设计问题，并为设计的自动化提供便利。

本设计小组采用“自顶向下”的设计方法对多功能电子时钟系统展开设计，所得系统集成 24 小时计时、校时、闹钟、秒表功能，并允许用户手动选择工作模式、使能或复位系统，具有较完善的用户交互机制。

为实现项目设计目标，设计小组首先对各项功能单独进行实现，分别设计了系统时钟生成模块、计时功能控制模块、计时信号生成模块、LED 指示模块、计时信号存储模块、定时报警触发模块、定时报警声音模块、数码管显示模块，并在顶层模块中例化调用上述功能单元，实现了不同工作模式与用户交互系统的集成。上述模块采用 Verlog HDL 语言编写，软件平台为 Xilinx 公司开发的 Vivado 设计套件，硬件平台为电子科技大学自主研发的 FPGA 开发板 xc7335tft256-1。

经过前期调研、中期设计、后期实现，设计小组按照项目设计要求成功集成了各项功能模块，经编写、仿真、综合确认设计无误后结合硬件平台进行物理实现，硬件测试功能优越地满足了实验项目要求，本次多功能电子时钟系统设计与实现成功。

第1章 引言

1.1 项目研究的主要内容

1.1.1 项目设计要求

采用自顶向下的设计方法，完成多功能数字时钟的设计与实现，功能包括：24 小时计时、校时、闹钟、秒表等

1.1.2 项目设计指标

1） 完成时、分、秒的计数，数码管显示格式“00-00-00”，以 24 小时循环计时；

2） 具备清零、时间调节功能；

3） 具备闹钟设置和定时报警功能；

4） 具有秒表功能；

1.2 项目研究的关键技术与需求资源

1.2.1 项目关键技术

实验项目通过自顶向下的设计方法实现电子设计自动化（Electronic Design Automation，EDA），选择 Verilog 为实验项目的硬件描述语言，最终通过 FPGA 方式对实验项目进行最终物理实现。

自顶向下（TOP-DOWN）的设计方法是从系统硬件的高层次抽象描述向底层物理描述转换的设计方法，设计过程分为行为级描述、RTL 描述、逻辑综合、物理实现四个阶段。自顶向下的设计方法具有两类特点：首先，该方法允许在设计四个阶段分别进行仿真，方便设计者在设计早期阶段解决设计问题；其次，该方法具备“自动化”特点，可采用 EDA 工具自动完成综合及实现。凭借自身具备的独有优势，自顶向下的设计方法已经成为现代电子设计中的主流方法。

Verilog，即 Verilog HDL，是一种用于算法级、行为级、结构级、门级等多种抽象设计层次的数字系统建模的硬件描述语言。Verilog 易学易用，允许设计者在一个电路模型内进行不同抽象层次描述，实现层次化设计，并可在相同描述中显式地进行时序建模，并受到大多数厂商、综合工具的支持。

FPGA 是 EDA 设计内容最终物理实现的方式之一，指由加工厂完成了集成电路工艺制造和封装测试的可编程集成电路成品。设计者可采用 EDA 软件将设计的目标文件烧录入 FPGA，改变其内部配置，从而实现目标功能。

1.2.2 项目需求资源

根据实验项目要求，所得多功能电子时钟系统应具备 24 小时计时功能、时间调节功能、闹钟设置及定时报警功能、秒表功能，且项目系统允许清零及时间调节。对上述目标功能进行分析，可知实验项目需求资源如下：

1） 输入晶振电路，用以获取时钟输入；
2） 按键开关电路，用以实现按键输入；
3） 拨码开关电路，用以实现功能转换；
4） LED 指示电路，用以指示当前功能或当前数位；
5） 数码显示电路，用以显示当前计时；
6） 蜂鸣器电路，用以实现报警功能；

项目硬件平台选取应结合上述项目需求资源进行考虑，以尽可能涵盖上述资源需求。

第2章 实验项目方案设计

2.1 项目系统设计原理

项目系统整体设计原理立足于实验项目五“电子秒表设计与实现”，并在功能上做出扩充。其计数通过计数器实现，其数据存储通过寄存器实现，其数据显示通过动态扫描实现。

针对计数，考虑到项目系统集成 24 小时计时功能与秒表功能，故系统内应包括模 10 计数器、模 4 计数器、模 5 计数器、模 3 计数器。不同计数器间通过异步级联方式实现时钟系统的多位计数功能。

针对存储，一方面不同工作模式下计时数据需妥善存储，另一方面 24 小时计时时应实现多次存储。针对上述情况，分别在计时信号生成模块、计时数据存储模声明多个数组以存储数据。

针对显示，考虑到项目系统不同数位数据来源不同，故采用数码管的动态扫描方法，利用高频次的时钟信号依次选中数码管的不同数位，并在各数位分别赋予不同来源的计时数据。

同时，考虑到项目系统设计到多种工作模式的切换，需要对用户指令进行直接处理。因此，项目系统中应包含用户按键指令的消抖模块，使用寄存器消抖方式获得稳定的用户按键指令，并依据按键指令实现不同工作模式间切换。

2.2 项目系统设计方案及模块组成

项目系统采用自顶而下设计方法，其设计模块可分为三个层次，即顶层模块、一级子模块、二级子模块。

顶层模块 top 位居系统设计最顶层，其作用在于调用各一级子模块，并指定各子模块的模块名及端口连接关系。本设计实验中，顶层模块 top 调用了时钟信号生成模块 timecreator、计数功能控制模块 cntcontrol、计数信号生成模块 cntmaker、计数数据存储模块 save、系统状态指示模块 ledrun、定时报警触发模块 clocktriger、定时报警声音模块 buzz、数码管显示模块 segmaker，共 8 处一级子模块。

一级子模块中，计数功能控制模块 cntcontrol、计数信号生成模块又由二级子模块组成。计数功能控制模块 cntcontrol 调用按键消抖模块 ajxd 以得到稳

定无抖动的按键控制信号；计数信号生成模块 cntmaker 调用模值不同计数器模块实现异步级联计数并输出各位计数信号值。

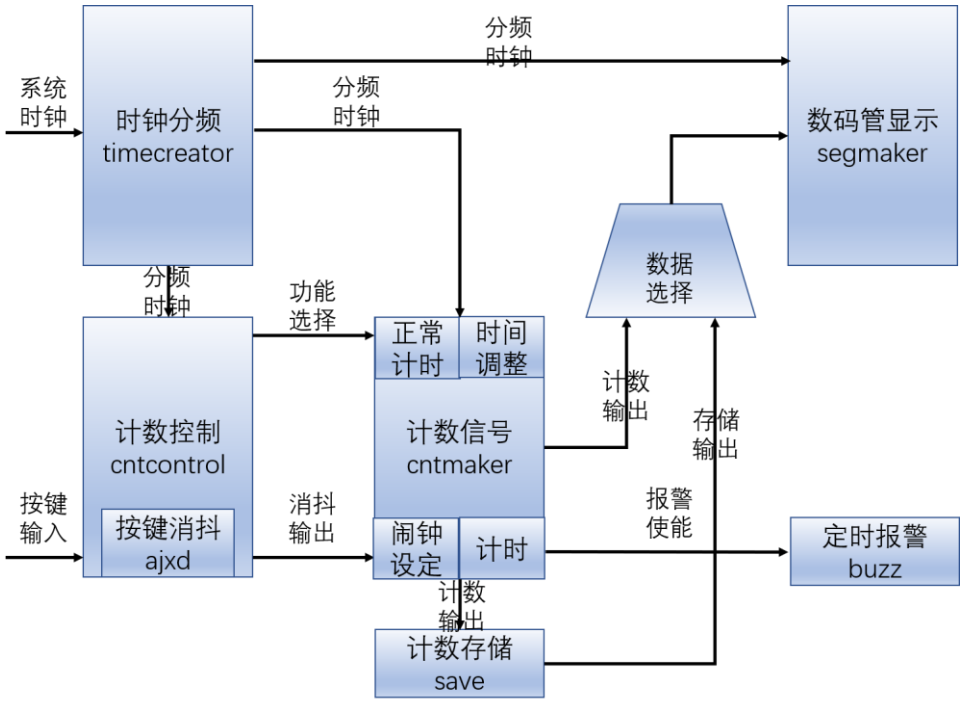


图 1：项目系统原理框图

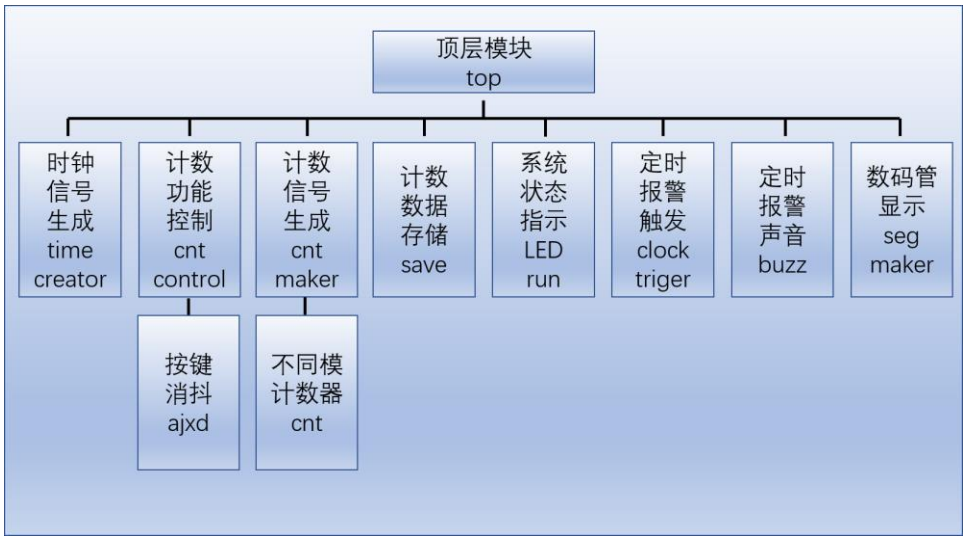


图 2：系统模块组成

第3章 实验项目设计平台简介

3.1 软件平台功能简介

实验项目软件平台采用 Xilinx 公司开发的 Vivado 设计套件，该软件提供高度集成的设计环境和新一代从系统到 IC 级的工具，均建立在共享的可扩展数据模型和通用调试环境基础上，实现了各类可编程技术的结合。

Vivado 在高层次设计功能、实现功能、验证功能方面均具有一定优势。在高层次设计方面，Vivado 设计套件可以根据高抽象度的行为描述产生对应电路；在验证功能方面，Vivado 设计套件采用用于快速综合和验证的设计，极大提高了模块和系统的仿真速度，并提高了硬件协仿真性能；在实现功能方面，Vivado 设计套件采用层次化器件编辑器和布局规划器，提高实现速度。同时增量式流程允许对于工程的修改只需对设计的部分重新实现就能够快速处理，提高了实现性能。

3.2 硬件平台基本结构及提供资源

实验项目硬件平台采用电子科大自制 FPGA 开发板。开发板主芯片采用 Artix-7、xc7a35tftg256-1，封装采用 FTG256，配备接口包括 USB2.0 接口、VGA 接口以及 4 个 24 座 Pmod 接口。

硬件平台主要为设计者提供五类模块资源，包括输入拨码开关电路、输入按键开关电路、输入晶振电路、输出 LED 指示灯电路、数码管电路。对于拨码开关电路，提供 12 个拨码开关，开关向上为高电平，开关向下为低电平；对于输出按键开关电路，提供 4*4 矩阵键盘，按键按下为低电平，按键常态为高电平；对于输入晶振电路，连接到板上 D4 管脚，作为板上主时钟提供系统 50MHz 的时钟源；对于输出 LED 指示灯电路，提供 12 个 LED 灯，在高电平下点亮，在低电平下熄灭；对于数码管电路，提供 6 位、8 段数码管，数码管接法为共阴极，位码低有效，段码高有效。

第4章 实验项目单元模块电路设计

4.1 时钟信号生成模块 timecreator

4.1.1 模块描述

时钟信号生成模块用于接收开发板板上自带输入晶振电路的时钟脉冲，并通过计数分频手段实现不同分频时钟信号的输出。

实验设计中模块命名为 timercreator，对应模块例化名称为 tc。模块输入端口接收 50MHz 时钟信号 clk50mhz、使能信号 en、复位信号 rst_s。模块输出端口输出分频后 10KHz 时钟信号 clk10khz、100Hz 时钟信号 clk100hz、1Hz 时钟信号 clk1hz、5Hz 时钟信号 clk5hz。

模块具体设计源程序关键语句节选如下：

```
1. module timecreator(
2.     input clk50mhz,
3.     input en,
```

```
    input rst_s,
    output reg clk10khz,
    output reg clk100hz,
    output reg clk1hz,
    output reg clk5hz
    );
    reg [27:0] cnt1,cnt2,cnt3,cnt4;
    always@(posedge clk50mhz)
    begin
        if(~en)
        begin
        cnt1<=28'd0;clk10khz<=1'b0;
        end
        else
        begin
        if(cnt1<2499)
        cnt1<=cnt1+1;
        else
        begin
        clk10khz<=~clk10khz;
        cnt1<=28'd0;
        end
        end
    end

       always@(posedge clk50mhz)
    begin
        if(~en)
        begin
            cnt2<=28'd0;clk100hz<=1'b0;
        end
        else
        begin
        if(cnt2<249999)
        begin
        cnt2<=cnt2+1;clk100hz<=1'b0;
        end
        else
        begin
        clk100hz<=1;
        cnt2<=28'd0;

        end
        end
```

```
    end
       always@(posedge clk50mhz)
    begin
        if(~en)
        begin
            cnt3<=28'd0;clk1hz<=1'b0;
        end
        else
        begin
        if(cnt3<24999999)
        cnt3<=cnt3+1;
        else
        begin
        clk1hz<=~clk1hz;
        cnt3<=28'd0;
        end
        end
    end
    always@(posedge clk50mhz)
    begin
        if(~en)
        begin
        cnt4<=28'd0;clk5hz<=1'b0;
        end
        else
        begin
        if(cnt4<4999999)
        cnt4<=cnt4+1;
        else
        begin
        clk5hz<=~clk5hz;
        cnt4<=28'd0;
        end
        end
    end
endmodule
```

4.1.2 模块仿真

1） 仿真程序

```
module timecreatortestbench;
/*
input clk50mhz,
    input en,
```

```
    input rst_s,
    output reg clk10khz,
    output reg clk100hz,
    output reg clk1hz,
    output reg clk5hz
*/
reg clk50mhz=1'b0;
reg en;
reg rst_s=1'b0;
wire clk10khz;
wire clk100hz;
wire clk1hz;
wire clk5hz;
timecreator u1(
clk50mhz,
en,
rst_s,
clk10khz,
clk100hz,
clk1hz,
clk5hz
);
initial
begin
en=1'b0;
#100
en=1'b1;
end
always#10 clk50mhz=~clk50mhz;


endmodule
```

2） 仿真波形

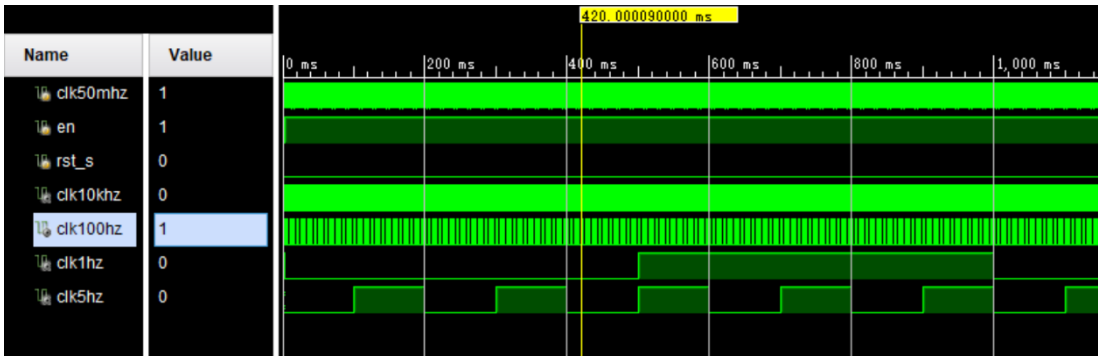


图 3：timecreator 模块测试波形

由上述仿真波形可知，对于 50MHz 的开发板晶振电路输入信号，时钟信号生成模块成功对其进行不同模值的计数分频，并依次输出了 10KHz、100Hz、1Hz、5Hz 的时钟信号。模块设计成功。

4.2 计数功能控制模块 cntcontrol

4.2.1 模块描述

计数功能控制模块用于根据用户按键输入生成对应功能选择信号，决定多功能电子时钟系统工作模式；计数功能控制模块为消除机械键盘抖动影响、获取稳定的用户输入，又调用二级子模块进行按键消抖。

实验设计中计数功能控制模块命名为 cntcontrol，对应模块例化名称为 c1. 模块输入端口接收按键输入 btn_in、50MHz 时钟脉冲 clk50mhz、使能输入 en。模块输出端口输出消抖后按键输入 btnout、系统功能选择信号 mode；实验设计中按键消抖模块命名为 ajxd，对应模块例化名称为 ajxd1。模块输入端口接收 50MHz 时钟脉冲 clk50mhz、按键输入信号 btn_in、复位信号 rst_n。模块输出端口输出消抖后按键输入信号 btn_out。

计数功能控制模块具体设计源程序关键语句节选如下：

```
1. module cntcontrol(
2.     input [3:0] btnin,
3.     input clk50mhz,
4.     input en,
5.     output [3:0] btnout,
6.     output reg [3:0] mode
7.     );
8.     wire [3:0] btn_out;
9.     assign btnout = btn_out;
10.    ajxd ajxd1(
11.    .clk50mhz(clk50mhz),
12.    .btn_in(btnin),//3 控制模式 2 和 1 控制左右选位  0 表示调节
   按钮
13.    .btn_out(btn_out),
14.    .rst_n(en)
15.    );
16.    always@(posedge btn_out[3] or negedge en)
17.    begin
18.        if(~en)
19.            mode<=4'b0001;
20.        else
21.        begin
22.                mode[3:1]<=mode[2:0];
```

```
                mode[0]<=mode[3];
        end
    end

endmodule
```

按键消抖模块具体设计源程序关键语句节选如下：

```
module ajxd(
    input clk50mhz,
    input  [3:0] btn_in,
    output [3:0] btn_out,
    input rst_n
    );

reg clk_20ms = 0;
reg [31:0] temp = 0;
always@(posedge clk50mhz)
begin
    if(rst_n == 0)
    begin
        temp <= 0;
        clk_20ms <= 0;
    end
    else if(temp == 999999)
        begin
            temp <= 0;
            clk_20ms <= 1;
        end
        else if(temp < 999999)
            begin
                clk_20ms <= 0;
                temp <= temp+1;
            end
end

reg [3:0] btn0;
reg [3:0] btn1;
reg [3:0] btn2;
initial
begin
btn0=0;
btn1=0;
btn2=0;
end
```

```
38.
39.always@(posedge clk_20ms)
40.begin
41.btn0<=btn_in;
42.btn1<=btn0;
43.btn2<=btn1;
44.end
45.
46.assign btn_out=((btn0&btn1)&(~btn2) | (btn0&btn1&btn2) | ((
   ~btn0)&btn1&btn2));
47.
48.endmodule
```

4.2.2 模块仿真

考虑到计数功能控制模块主要用于消抖用户输入指令、控制系统功能选择，其功能抽象程度较高，难以通过仿真测试直观展示，故该模块的验证通过后续部分的硬件测试证明。

4.3 计数信号生成模块 cntmaker

4.3.1 模块描述

计数信号生成模块用于在多功能电子时钟系统的不同模式选择下生成对应计数信号，并为数码管显示模块提供输入数据。模块具体实现过程中需调用计数器不同模值计数器进行异步级联。

实验设计中技术信号产生模块命名为 cntmaker，对应模块例化名称为 cm1。模块输入端口接收 100Hz 时钟信号 clk100hz、1Hz 时钟信号 clk1hz、使能信号 en、停止信号 stop、功能选择信号 mode、消抖后按键输入信号 btnout。模块输出端口输出数位指示信号 sel1、各位计数信号 cnt1~cnt6。

实验设计中不同模值计数器模块命名为 cnt1~cnt6，对应模块例化名称为 cnt1~cnt6。各模块输入端口接收不同分频时钟脉冲 clk1、clk2、按键输入信号 button、功能选择信号 mode、数位指示信号 sel、使能信号 en、复位信号 rst_s。模块输出端口输出计数输出信号 cntout、进位输出信号 upper1、upper2。

计数信号生成模块具体设计源程序关键语句节选如下：

```
1. module cntmaker(
2.     input clk100hz,
3.     input clk1hz,
4.     input en,
5.     input stop,
```

```
    input rst_s,
    input [3:0] mode,
    input [3:0] btnout,
    output [5:0] sel1,
    output [3:0] cnt_1,cnt_2,cnt_3,cnt_4,cnt_5,cnt_6
    );
    reg [5:0] sel;
    wire cout1,cout2,cout3,cout4,cout5,flag,flag1;
    wire sout1,sout2,sout3,sout4,sout5,flag2;
    assign sel1=sel;
    assign flag=flag1|flag2;
    always@(posedge btnout[1] or negedge en)
    begin
        if(~en)
            sel<=6'b000001;
        else
        begin
                sel[5:1]<=sel[4:0];
                sel[0]<=sel[5];
        end
    end
    cnt1 cnt1
    (
    .clk1(clk1hz),
    .clk2(clk100hz),
    .en(en),
    .stop(stop),
    .mode(mode),
    .rst_s(rst_s),
    .sel(sel[0]),
    .button(btnout[0]),
    .cntout(cnt_1),
    .upper1(cout1),
    .upper2(sout1)
    );
    cnt2 cnt2
    (
    .clk1(cout1),
    .clk2(sout1),
    .en(en),
    .rst_s(rst_s),
    .stop(stop),
    .mode(mode),
    .sel(sel[1]),
```

```
    .button(btnout[0]),
    .cntout(cnt_2),
    .upper1(cout2),
    .upper2(sout2)
    );
    cnt3 cnt3
    (
    .clk1(cout2),
    .clk2(sout2),
    .en(en),
    .stop(stop),
    .rst_s(rst_s),
    .mode(mode),
    .sel(sel[2]),
    .button(btnout[0]),
    .cntout(cnt_3),
    .upper1(cout3),
    .upper2(sout3)
    );
        cnt4 cnt4
    (
    .clk1(cout3),
    .clk2(sout3),
    .en(en),
    .stop(stop),
    .mode(mode),
    .rst_s(rst_s),
    .sel(sel[3]),
    .button(btnout[0]),
    .cntout(cnt_4),
    .upper1(cout4),
    .upper2(sout4)
    );
    cnt5 cnt5
    (
    .clk1(cout4),
    .clk2(sout4),
    .en(en),
    .flag(flag),
    .rst_s(rst_s),
    .stop(stop),
    .mode(mode),
    .sel(sel[4]),
    .button(btnout[0]),
```

```
    .cntout(cnt_5),
    .upper1(cout5),
    .upper2(sout5)
    );
    cnt6 cnt6
    (
       .clk1(cout5),
       .clk2(sout5),
       .en(en),
       .stop(stop),
       .mode(mode),
       .rst_s(rst_s),
       .sel(sel[5]),
       .button(btnout[0]),
       .cntout(cnt_6),
       .upper1(flag1),
       .upper2(flag2)
       );
endmodule
```

单个计数器代码如下：

```
module cnt1(
    input clk1,clk2,
    input stop,
    input en,
    input [3:0] mode,
    input sel,
    input button,
    input rst_s,
    output reg [3:0] cntout,
    output reg upper1,upper2
    );
    reg [3:0] cnttemp1,cnttemp2,cnttemp3;
    reg clk;
    always@(clk1,button)
    begin
        case(mode)
        4'b0001:clk<=clk1;
        4'b0010:clk<=button;
        default:clk<=clk1;
        endcase
    end
    always@(mode)
    begin
        case(mode)
```

```
        4'b0100:cntout<=cnttemp2;
        4'b1000:cntout<=cnttemp3;
        default:cntout<=cnttemp1;
        endcase
    end
    always@(negedge clk or negedge en)//时钟与调整时间
    begin
        if(~en)
        begin
            upper1<=1'b0;cnttemp1<=4'd0;
        end
        else
        begin
            if(stop)
            ;
            else
                begin
                    case(mode)
                    4'b0010:begin
                        if(sel)
                        begin
                                upper1<=1'b0;
                                if(cnttemp1==4'd9) begin cn
   ttemp1<=4'd0; end
                                else begin cnttemp1<=cnttem
   p1+1;end
                        end
                    end

                    default:begin
                                if(cnttemp1==4'd9) begin cn
   ttemp1<=4'd0; upper1<=1'b1;end
                                else begin cnttemp1<=cnttem
   p1+1;upper1<=1'b0; end
                    end
                    endcase
                end
          end
    end
    always@(posedge button or negedge en)
     begin
        if(~en)
        begin
            cnttemp2<=4'd0;
```

```
65.         end
66.         else
67.         begin
68.             if(stop)
69.             ;
70.             else
71.                 begin
72.                     case(mode)
73.                     4'b0100:begin
74.                                     if(sel)

75.                                     begin

76.                                             if(cnttemp2==4'd9)
   begin cnttemp2<=4'd0; end
77.                                             else begin cnttemp2
   <=cnttemp2+1; end
78.
79.                                     end

80.                     end
81.
82.                     default:;
83.                     endcase
84.                 end
85.           end
86.     end
87.     always@(posedge clk2 or negedge en)
88.      begin
89.         if(~en)
90.         begin
91.             cnttemp3<=4'd0;upper2<=0;
92.         end
93.         else
94.         begin
95.         if(rst_s) begin cnttemp3=4'd0;upper2=1'd0;end
96.         else
97.         begin
98.             if(stop)
99.             ;
100.              else
101.                 begin
102.                     case(mode)
103.                     4'b1000:begin
```

```
                                        if(cnttemp3==4'd9) be
   gin cnttemp3<=4'd0; upper2<=1'b1;end
                                        else begin cnttemp3<=
   cnttemp3+1;upper2<=1'b0; end
                                    end

                        default:;
                        endcase
                    end
                    end
                end
        end
  endmodule
```

4.3.2 模块仿真

1） 仿真程序

```
module clock_testbench;
/*
input clk100hz,
    input clk1hz,
    input en,
    input stop,
    input rst_s,
    input [3:0] mode,
    input [3:0] btnout,
    output [5:0] sel1,
    output [3:0] cnt_1,cnt_2,cnt_3,cnt_4,cnt_5,cnt_6
*/
reg clk100hz;
reg clk1hz;
reg en;
reg stop;
reg rst_s;
reg[3:0] mode;
reg [3:0] btnout;
wire [5:0] sell;
wire  [3:0] cnt_1,cnt_2,cnt_3,cnt_4,cnt_5,cnt_6;
cntmaker u0(
clk100hz,clk1hz,en,stop,rst_s,
mode,btnout,sell,
cnt_1,cnt_2,cnt_3,cnt_4,cnt_5,cnt_6
```

```
);
initial
begin
clk100hz=1'b0;
clk1hz=1'b0;
en=1'b0;
stop=1'b0;
rst_s=1'b0;
mode=4'b0000;
btnout=1'b0;
#10000
en=1'b1;
mode=4'b0001;
end
always #10 clk100hz=~clk100hz;

always #1000 clk1hz=~clk1hz;
endmodule
```

2） 仿真波形

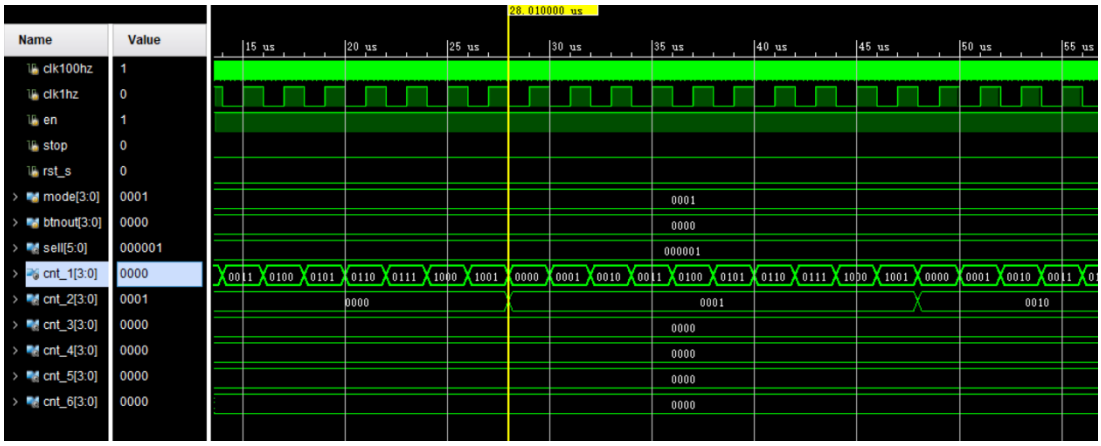


图 4：cntmaker 模块测试波形-秒

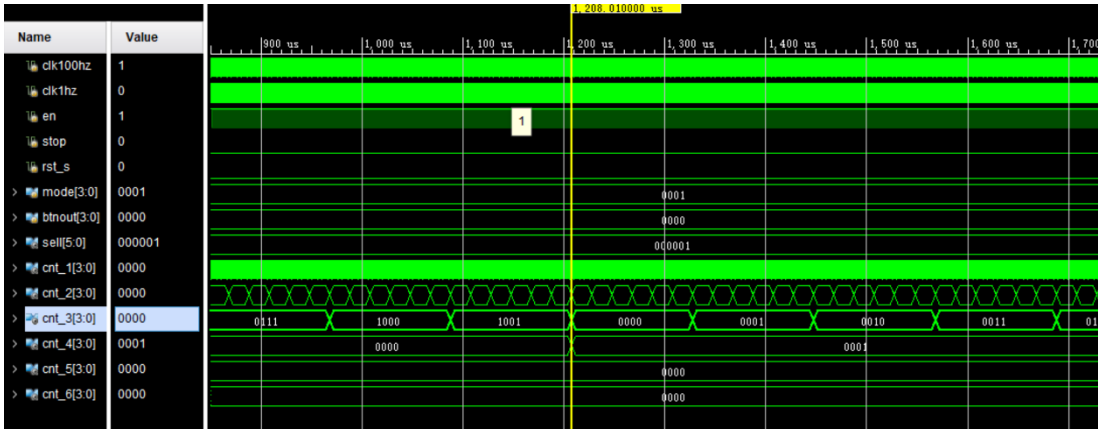


图 5：cntmaker 模块测试波形-分

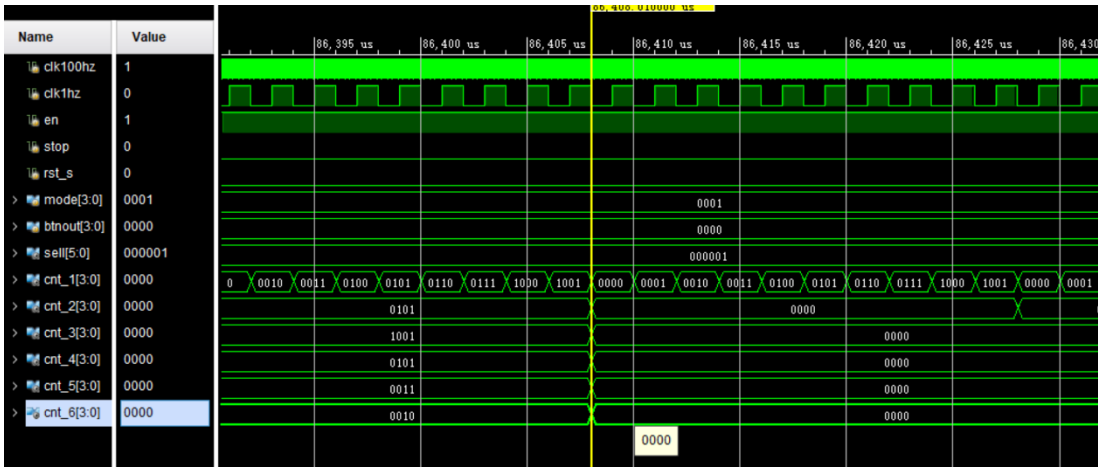


图 6：cntmaker 模块测试波形-时

由上述测试波形可见，在正常 24 小时计数模式下，多功能电子时钟系统计数功能正常运行，计数进位、计数时长均符合要求。模块设计成功。

4.4 计数数据存储模块 save

4.4.1 模块描述

计数数据存储模块用于在多功能电子时钟系统进行 24 小时正常计时时存储当前计时数据，并作为数码管显示模块数据来源之一。

实验设计中技术信号产生模块命名为 save，对应模块例化名称为 s1。模块输入端口接收使能信号 en、读取使能信号 readen、被存储 24 位计数数据 cntin、存读按键输入 button、复位信号 rst；模块输出端口输出被读取 24 位计时数据 cntout。

计数数据存储模块具体设计源程序关键语句节选如下：

```
1. module save(
2.     input en,//链接 mode[3]
```

```
    input [23:0] cntin,
    input button,//链接 bout3
    input readen,//链接一个拨码开关
    input rst,//链接一个拨码开关
     output reg [23:0] cntout
    );
    reg [23:0] savecnt [7:0];
    reg [2:0] i,j;
    always@(posedge button)
    begin
        if(en)
        begin
            if(rst)
                begin
                savecnt[7] = 24'd0;
                savecnt[6] = 24'd0;
                savecnt[5] = 24'd0;
                savecnt[4] = 24'd0;
                savecnt[3] = 24'd0;
                savecnt[2] = 24'd0;
                savecnt[1] = 24'd0;
                savecnt[0] = 24'd0;
                i= 3'd0;j=3'd0;
                end
                else
                begin
                    if(~readen)
                    begin
                            i<=i+1;
                            case(i)
                            3'd0:savecnt[0]=cntin;
                            3'd1:savecnt[1]=cntin;
                            3'd2:savecnt[2]=cntin;
                            3'd3:savecnt[3]=cntin;
                            3'd4:savecnt[4]=cntin;
                            3'd5:savecnt[5]=cntin;
                            3'd6:savecnt[6]=cntin;
                            3'd7:savecnt[7]=cntin;
                            default:;
                            endcase
                    end
                    else
                    begin
                            j<=j+1;
```

```
                                case(j)
                                3'd0:cntout=savecnt[0];
                                3'd1:cntout=savecnt[1];
                                3'd2:cntout=savecnt[2];
                                3'd3:cntout=savecnt[3];
                                3'd4:cntout=savecnt[4];
                                3'd5:cntout=savecnt[5];
                                3'd6:cntout=savecnt[6];
                                3'd7:cntout=savecnt[7];
                                default:;
                                endcase
                        end
                    end
                end
            end
endmodule
```

4.4.2 模块仿真

考虑到模块功能在于存储 24 小时计数工作模式下不同时刻下计时数据，涉及到多组数组数据展示，难以通过仿真测试方法简明直观表现，故该部分测试通过后续硬件测试证明。

4.5 系统状态指示模块 ledrun

4.5.1 模块描述

系统状态指示模块用于指示当前多功能电子时钟系统功能模式或当前时间调整位数。

实验设计中系统状态指示模块命名为 ledrun，对应模块例化名称为 lr1。模块输入端口接收 4 位当前功能模式选择信号 mode、6 位当前数位选择信号 sel；模块输出端口输出 10 位 LED 灯驱动信号 led。

系统状态指示模块具体设计源程序关键语句节选如下：

```
module ledrun(
    input [3:0] mode,
    input [5:0] sel,
    output [9:0] led
    );
    assign led[3:0]=mode;
    assign led[9:4]=sel;
```

```
endmodule
```

4.5.2 模块仿真

考虑到模块功能用于驱动 LED 灯指示当前工作模式或当前选择数位，功能抽象程度较高，难以通过仿真测试方法进行简明直观展示，故选择在后续硬件测试中对模块功能进行检验。

4.6 定时报警触发模块 clocktrigger

4.6.1 模块描述

定时报警触发模块用于判断当前计时值是否达到闹钟设定值，并生成报警使能信号作为蜂鸣器电路输入。

实验设计中定时报警触发模块命名为 clocktrigger，对应模块例化名称为 ctl。模块输入端口接收模块使能信号 en、时钟信号 clk、复位信号 rst、24 位当前计数输入 cnt；模块输出端口输出报警使能信号 beepen。

计数信号生成模块具体设计源程序关键语句节选如下：

```
module clocktriger(
    input en,
    input [23:0] cnt,
    input clk,
    input rst,
    output reg beepen
    );
    reg [23:0] set;
    always@(posedge clk)
    begin
        if(~rst)
            set<=24'd0;
        else
        begin
            if(en)
                set<=cnt;
             else
             begin
                if(cnt==set)
                    begin
                    beepen <= 1;
                    end
```

```
                else beepen<=0;
            end
        end
    end
endmodule
```

4.6.2 模块仿真

1） 仿真程序

```
module clocktri_test;
reg en;
reg [23:0] cnt;
reg clk;
reg rst;
wire beenpen;
clocktriger u0(
en,cnt,clk,rst,beenpen
);
initial
begin
rst=0;
en=0;
cnt=0;
clk=0;
#100
rst=1;
cnt=9;
en=1;
#100
cnt=0;
en=0;
end
always#10 clk=~clk;
always#500 cnt=cnt+1;
endmodule
```

2） 仿真波形

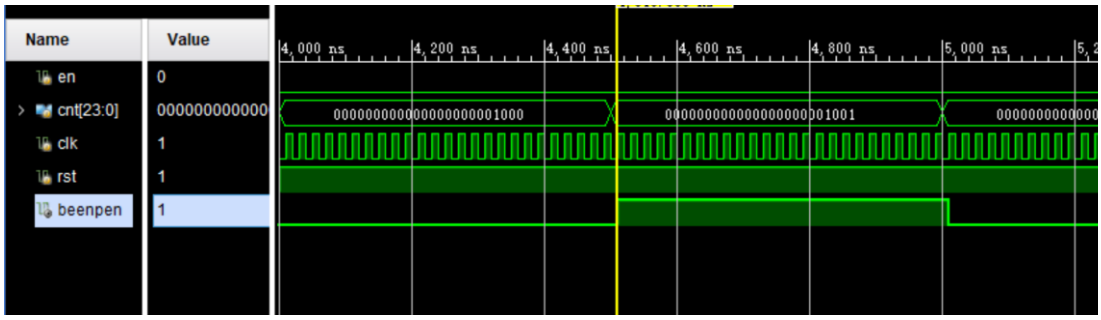


图 7：clocktrigger 模块测试波形

从模块仿真测试波形图可以看出，看模块计时达到设定值后，定时报警触发信号 beepen 被置为有效，用以使能定时报警声音模块使其进行蜂鸣报警。模块设计成功。

4.7 定时报警声音模块 buzz

4.7.1 模块描述

定时报警声音模块用于对报警使能信号做出反应，并根据预设频率驱动开发板蜂鸣器电路报警。

实验设计中定时报警声音模块命名为 buzz，对应模块例化名称为 b1。模块输入端口接收报警使能信号 beepen、不同频率时钟信号 clk50mhz 和 clk1hz；模块输出端口输出蜂鸣器驱动信号 b_eep。

计数信号生成模块具体设计源程序关键语句节选如下：

```
1. module buzz(
2.      input beepen,
3.      output b_eep,
4.      input clk50mhz,
5.      input clk1hz
6.      );
7.      reg beep = 0;
8.      reg [2:0] timecnt;
9.      reg timecnten;
10.     reg [31:0] cnt=32'd0;
11.     parameter      MIN_DO = 18'd190800,//(50_000_000/262)
12.                    MIN_RE = 18'd170050,//(50_000_000/294)
13.                    MIN_MI = 18'd151500,//(50_000_000/330)
14.                    MIN_FA = 18'd143250,//(50_000_000/349)
15.                    MIN_SO = 18'd127550,//(50_000_000/392)
16.                    MIN_LA = 18'd113600,//(50_000_000/440)
17.                    MIN_XI = 18'd101200;//(50_000_000/494)
18.//中音
```

```
    parameter      MID_DO = 17'd95600,//(50_000_000/523)
                    MID_RE = 17'd85150,//(50_000_000/587)
                    MID_MI = 17'd75850,//(50_000_000/659)
                    MID_FA = 17'd71600,//(50_000_000/698)
                    MID_SO = 17'd63750,//(50_000_000/784)
                    MID_LA = 17'd56800,//(50_000_000/880)
                    MID_XI = 17'd50600;//(50_000_000/988)
//高音
    parameter  MAX_DO = 16'd47755,//(50_000_000/1047)
                MAX_RE = 16'd42553,//(50_000_000/1175)
                MAX_MI = 16'd37907,//(50_000_000/1319)
                MAX_FA = 16'd35790,//(50_000_000/1397)
                MAX_SO = 16'd31887,//(50_000_000/1568)
                MAX_LA = 16'd28409,//(50_000_000/1760)
                MAX_XI = 16'd25419;//(50_000_000/1967)
    parameter  TIME_300ms = 24'd14_999_999,//300ms,半拍
                TIME_500ms = 25'd24_999_999;//500ms,一拍
    always@(posedge beepen or posedge clk1hz)
    begin
        if(beepen == 1)
        timecnten<=1;
        else timecnten<=0;
    end
    always@(posedge clk1hz or posedge timecnten)
    begin
    if(timecnten==1) timecnt<=3'd7;
    else if(timecnt!=0) timecnt<=timecnt-1;
    else ;
    end
    always@(posedge clk50mhz)
    begin


        if(timecnt==0) ;
        else
        begin
            case(timecnt)
            3'd7:begin

            if(cnt>= MIN_LA)  begin cnt<=32'd0;beep<=~beep;
   end
            else cnt<=cnt+32'd1;
            end
```

```
            3'd6:begin
                
            if(cnt>=  MIN_XI)  begin cnt<=32'd0;beep<=~beep
;end
            else cnt<=cnt+32'd1;
            end
            3'd5:begin
                
            if(cnt>=  MID_DO)  begin cnt<=32'd0;beep<=~beep
;end
            else cnt<=cnt+32'd1;
            end
            3'd4:begin
                
            if(cnt>=  MID_RE)  begin cnt<=32'd0;beep<=~beep
;end
            else cnt<=cnt+32'd1;
            end
            3'd3:begin
                
            if(cnt>= MID_MI)  begin cnt<=32'd0;beep<=~beep;
end
            else cnt<=cnt+32'd1;
            end
            3'd2:begin
                
            if(cnt>= MID_MI)  begin cnt<=32'd0;beep<=~beep;
end
            else cnt<=cnt+32'd1;
            end
            3'd1:begin
                
            if(cnt>=  MID_XI)  begin cnt<=32'd0;beep<=~beep
;end
            else cnt<=cnt+32'd1;
            end
            3'd0:begin
                
            if(cnt>=  MID_LA)  begin cnt<=32'd0;beep<=~beep
;end
            else cnt<=cnt+32'd1;
            end
            default:;
            endcase
```

```
99.          end
100.       end
101.       assign b_eep=beep;
102.  endmodule
```

4.7.2 模块仿真

1） 仿真程序

```
1. module testbuzz;
2.      reg beepen;
3.      wire beep;
4.      reg clk50mhz;
5.      reg clk1hz;
6.      buzz b1(
7.      .beepen(beepen),
8.      .b_eep(beep),
9.      .clk50mhz(clk50mhz),
10.     .clk1hz(clk1hz)
11.     );
12.     initial begin
13.     clk50mhz=0;clk1hz=0;beepen=0;
14.     #1000
15.     beepen = 1;
16.     end
17.
18.     always begin
19.     #1 clk50mhz=~clk50mhz;
20.     end
21.      always begin
22.     #50000000 clk1hz=~clk1hz;
23.     end
24.endmodule
```

2） 仿真波形

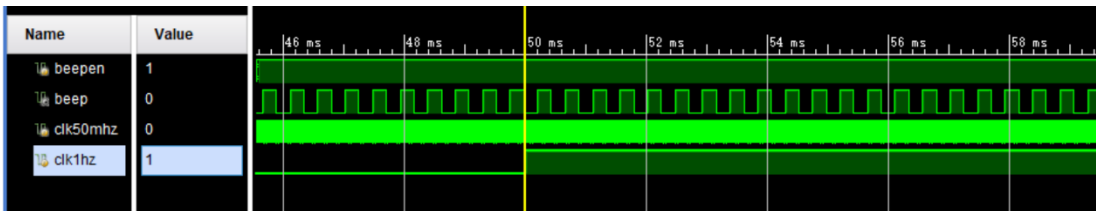


图 8：buzz 模块测试波形

由定时报警声音模块 buzz 的测试波形可以看出，在蜂鸣器使能信号有效后，

模块输出设定频率的蜂鸣器驱动信号，可驱动开发板上无源蜂鸣器电路发出报警声。模块设计成功。

4.8 数码管显示模块 segmaker

4.8.1 模块描述

数码管显示模块用于接收不同来源数据，并采用动态扫描方式驱动开发板数码管显示电路显示当前数据输入。

实验设计中数码管显示模块命名为 segmaker，对应模块例化名称为 sm1。模块输入端口接收 4 位不同位输入数据 cnt1~cnt6、分频后时钟信号 clk10khz；模块输出端口输出 6 位数码管位选信号、8 位数码管段选信号。

计数信号生成模块具体设计源程序关键语句节选如下：

```
1. module segmaker(
2.     input [3:0] cnt1,cnt2,cnt3,cnt4,cnt5,cnt6,
3.     input clk10khz,
4.     output reg [5:0] dig,//低有效
5.     output reg [7:0] seg
6.     );
7.     reg [2:0] flag;
8.     initial flag=3'b000;
9.     always@(posedge clk10khz)
10.    begin
11.    if(flag==3'd5)
12.    flag<=3'd0;
13.    else
14.    flag<=flag+1;
15.    end
16.    always@(flag)
17.    begin
18.    case(flag)
19.    3'd0:
20.    begin
21.        dig<=6'b111110;
22.     case(cnt1)
23.  4'h0: seg=8'h3f;// DP,GFEDCBA
24.  4'h1: seg=8'h06;
25.  4'h2: seg=8'h5b;
26.  4'h3: seg=8'h4f;
27.  4'h4: seg=8'h66;
28.  4'h5: seg=8'h6d;
29.  4'h6: seg=8'h7d;
```

```
4'h7: seg=8'h07;
4'h8: seg=8'h7f;
4'h9: seg=8'h6f;
default: seg=0;
endcase
  end
  3'b1:
  begin
       dig<=6'b111101;
      case(cnt2)
4'h0: seg=8'h3f;// DP,GFEDCBA
4'h1: seg=8'h06;
4'h2: seg=8'h5b;
4'h3: seg=8'h4f;
4'h4: seg=8'h66;
4'h5: seg=8'h6d;
4'h6: seg=8'h7d;
4'h7: seg=8'h07;
4'h8: seg=8'h7f;
4'h9: seg=8'h6f;
default: seg=0;
endcase
  end
  3'd2:
  begin
   dig<=6'b111011;
   case(cnt3)
4'h0: seg=8'hbf;// DP,GFEDCBA
4'h1: seg=8'h86;
4'h2: seg=8'hdb;
4'h3: seg=8'hcf;
4'h4: seg=8'he6;
4'h5: seg=8'hed;
4'h6: seg=8'hfd;
4'h7: seg=8'h87;
4'h8: seg=8'hff;
4'h9: seg=8'hef;
default: seg=0;
endcase
  end
  3'd3:
  begin
   dig<=6'b110111;
   case(cnt4)
```

```
74.  4'h0: seg=8'h3f;// DP,GFEDCBA
75.  4'h1: seg=8'h06;
76.  4'h2: seg=8'h5b;
77.  4'h3: seg=8'h4f;
78.  4'h4: seg=8'h66;
79.  4'h5: seg=8'h6d;
80.  4'h6: seg=8'h7d;
81.  4'h7: seg=8'h07;
82.  4'h8: seg=8'h7f;
83.  4'h9: seg=8'h6f;
84.  default: seg=0;
85.  endcase
86.    end
87.    3'd4:
88.    begin
89.     dig<=6'b101111;
90.    case(cnt5)
91.  4'h0: seg=8'hbf;// DP,GFEDCBA
92.  4'h1: seg=8'h86;
93.  4'h2: seg=8'hdb;
94.  4'h3: seg=8'hcf;
95.  4'h4: seg=8'he6;
96.  4'h5: seg=8'hed;
97.  4'h6: seg=8'hfd;
98.  4'h7: seg=8'h87;
99.  4'h8: seg=8'hff;
100.    4'h9: seg=8'hef;
101.    default: seg=0;
102.    endcase
103.      end
104.      3'd5:
105.      begin
106.       dig<=6'b011111;
107.      case(cnt6)
108.    4'h0: seg=8'h3f;// DP,GFEDCBA
109.    4'h1: seg=8'h06;
110.    4'h2: seg=8'h5b;
111.    4'h3: seg=8'h4f;
112.    4'h4: seg=8'h66;
113.    4'h5: seg=8'h6d;
114.    4'h6: seg=8'h7d;
115.    4'h7: seg=8'h07;
116.    4'h8: seg=8'h7f;
117.    4'h9: seg=8'h6f;
```

```
118.    default: seg=0;
119.    endcase
120.      end
121.      default:;
122.      endcase
123.      end
124.  endmodule
```

4.8.2 模块仿真

考虑到模块采用动态扫描显示方式，涉及的数据展示来源多、切换快，且由于数码管段选编码的缘故抽象度高，故难以通过软件仿真方式直观表明测试结果。因此，小组选择通过后续硬件平台测试证明模块功能实现。

第5章 实验项目系统电路设计

5.1 系统设计

多功能电子时钟系统整体应具备 24 小时计时功能、计时时间调整功能、定时闹钟设定及报警功能、倒计时秒表功能。

系统单项功能通过各单元模块实现，单项功能的集成及嵌入需要在顶层模块调用各单元模块并分配各端口数据流动。整体来看，系统输入端口接收用户控制信号及系统时钟信号，包括用户使能信号 en、用户复位信号 rst_s、用户停止信号 stop、用户存储使能信号 readen、4 位用户按键输入信号 btn_in；同时，系统输出端口输出硬件平台各电路模块驱动信号，包括 10 位 LED 指示信号 led、6 位数码管显示位选信号 dig、8 位数码管显示段选信号 seg、蜂鸣器模块使能信号 beepen、蜂鸣器模块驱动信号 beep。

系统顶层模块设计源代码为：

```
1. module top(
2. input clk50mhz,
3.     input en,
4.     input stop,
5.     input readen,
6.     input rst_s,
7.     input  [3:0] btn_in,
8.     output [9:0] led,
9.     output [5:0] dig,//低有效
10.    output [7:0] seg,
11.    output beepen,
12.    output beep
```

```
13.     );
14.     wire clk10khz;
15.     wire clk100hz;
16.     wire clk1hz;
17.     wire clk5hz;
18.     wire [3:0] cnt1,cnt2,cnt3,cnt4,cnt5,cnt6;//记录生成的时间
19.     wire [23:0] cnts;//记录 save 的时间
20.     wire [3:0] cnt1r,cnt2r,cnt3r,cnt4r,cnt5r,cnt6r;//送入显
    示模块的时间
21.     wire [3:0] mode;//0001 24 小时计时  0010 调时间  0100 闹钟
    设置  1000 秒表
22.     wire [3:0] btnout;
23.     wire [5:0] sel;
24.     wire [23:0] tempcnt;
25.     timecreator tc(
26.     .clk50mhz(clk50mhz),
27.     .en(en),
28.     .clk10khz(clk10khz),
29.     .clk100hz(clk100hz),
30.     .clk1hz(clk1hz),
31.     .clk5hz(clk5hz)
32.     );
33.     cntcontrol c1(
34.     .mode(mode),//00 24 小时计时  01 调时间  10 闹钟设置  11 秒
    表
35.     .en(en),
36.     .clk50mhz(clk50mhz),
37.     .btnin(btn_in),
38.     .btnout(btnout)//3 控制模式 2 和 1 控制左右选位  0 表示调节
    按钮
39.     );
40.     ledrun lr1(
41.     .mode(mode),
42.     .sel(sel),
43.     .led(led)
44.     );
45.     cntmaker cm1(
46.     .clk100hz(clk100hz),
47.     .clk1hz(clk1hz),
48.     .mode(mode),
49.     .en(en),
50.     .rst_s(rst_s),
51.     .stop(stop),
52.     .sel1(sel),
```

```
    .btnout(btnout),
   .cnt_1(cnt1),.cnt_2(cnt2),.cnt_3(cnt3),.cnt_4(cnt4),.cnt_5(cnt5),.cnt_6(cnt6)
    );

    assign tempcnt = {cnt6,cnt5,cnt4,cnt3,cnt2,cnt1};
    clocktriger ct1(
    .en(mode[2]),
    .cnt(tempcnt),
    .clk(clk100hz),
    .rst(en),
    .beepen(beepen)
    );
    buzz b1(
    .beepen(beepen),
    .b_eep(beep),
    .clk50mhz(clk50mhz),
    .clk1hz(clk5hz)
    );
    save s1(
    .en(mode[3]),//链接 mode[3]
    .cntin(tempcnt),
    .button(btnout[0]),//链接 bout0
    .readen(readen),//链接一个拨码开关
    .rst(rst_s),//链接一个拨码开关
    .cntout(cnts)
    );
    assign cnt1r=(mode[3]&&readen)?cnts[3:0]:cnt1;
    assign cnt2r=(mode[3]&&readen)?cnts[7:4]:cnt2;
    assign cnt3r=(mode[3]&&readen)?cnts[11:8]:cnt3;
    assign cnt4r=(mode[3]&&readen)?cnts[15:12]:cnt4;
    assign cnt5r=(mode[3]&&readen)?cnts[19:16]:cnt5;
    assign cnt6r=(mode[3]&&readen)?cnts[23:20]:cnt6;
    segmaker sm1(
    .cnt1(cnt1r),.cnt2(cnt2r),.cnt3(cnt3r),.cnt4(cnt4r),.cnt5(cnt5r),.cnt6(cnt6r),
    .clk10khz(clk10khz),
    .dig(dig),//低有效
    .seg(seg)
    );
endmodule
```

对系统顶层模块进行 RTL 分析，可得系统施密特图如下：

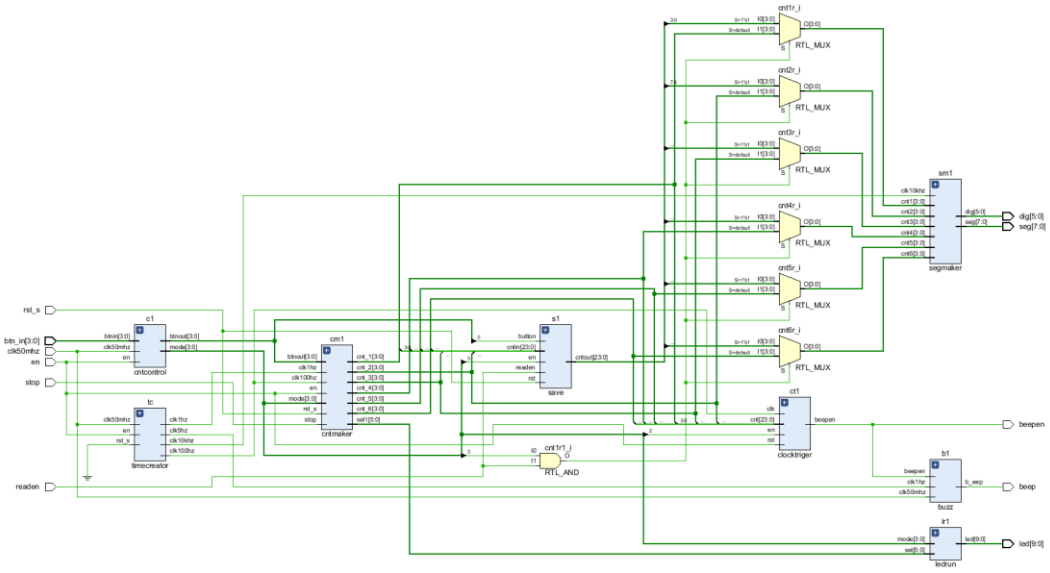


图 9：多功能电子时钟系统施密特图

考虑到系统顶层模块用于集成系统各项功能，难以通过仿真手段验证，故选择通过在后续章节中下载至硬件平台中进行验证。

第6章 实验项目系统设计实现

6.1 系统管脚适配表

为物理实现项目系统、在硬件平台验证多功能电子时钟系统功能，需依据硬件平台管脚对系统管脚做出适配。系统管脚约束代码如下：

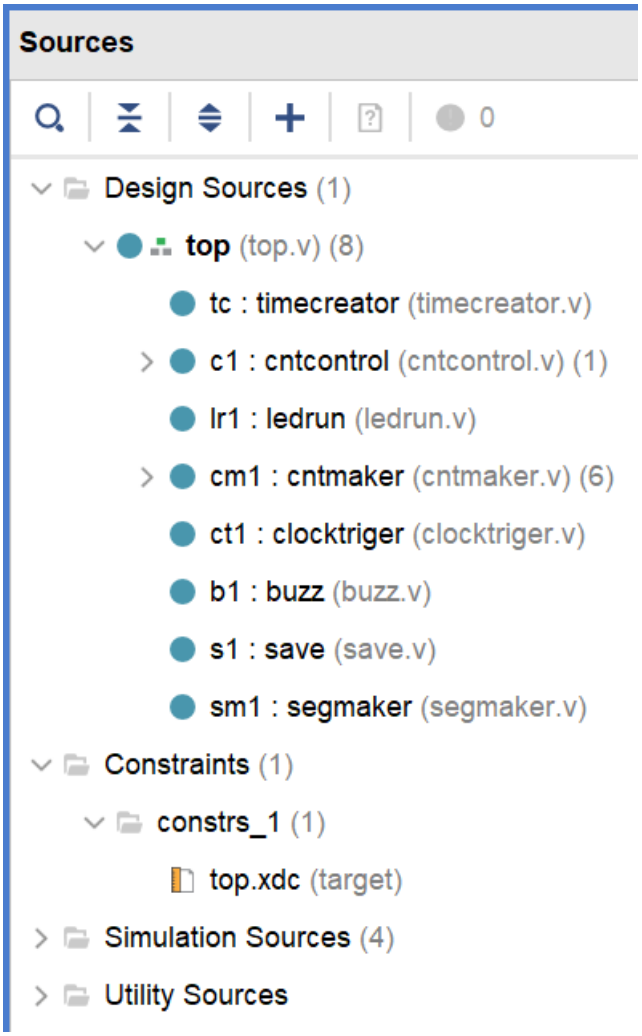
```
1. set_property IOSTANDARD LVCMOS33 [get_ports clk50mhz]
2. set_property PACKAGE_PIN D4 [get_ports clk50mhz]
3. set_property PACKAGE_PIN T9 [get_ports en]
4. set_property IOSTANDARD LVCMOS33 [get_ports en]
5. set_property PACKAGE_PIN E3 [get_ports {led[9]}]
6. set_property PACKAGE_PIN H3 [get_ports {led[8]}]
7. set_property PACKAGE_PIN G5 [get_ports {led[7]}]
8. set_property PACKAGE_PIN R1 [get_ports {led[6]}]
9. set_property PACKAGE_PIN T2 [get_ports {led[5]}]
10.set_property PACKAGE_PIN T3 [get_ports {led[4]}]
11.set_property PACKAGE_PIN P9 [get_ports {led[0]}]
12.set_property PACKAGE_PIN R8 [get_ports {led[1]}]
13.set_property PACKAGE_PIN R7 [get_ports {led[2]}]
14.set_property PACKAGE_PIN T5 [get_ports {led[3]}]
15.set_property IOSTANDARD LVCMOS33 [get_ports {led[3]}]
16.set_property IOSTANDARD LVCMOS33 [get_ports {led[2]}]
17.set_property IOSTANDARD LVCMOS33 [get_ports {led[1]}]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports {led[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {led[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {led[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {led[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {led[7]}]
set_property IOSTANDARD LVCMOS33 [get_ports {led[8]}]
set_property IOSTANDARD LVCMOS33 [get_ports {led[9]}]
set_property IOSTANDARD LVCMOS33 [get_ports {dig[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {dig[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {dig[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {dig[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {dig[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {dig[0]}]

set_property PACKAGE_PIN N11 [get_ports {dig[5]}]
set_property PACKAGE_PIN N14 [get_ports {dig[4]}]
set_property PACKAGE_PIN N13 [get_ports {dig[3]}]
set_property PACKAGE_PIN M12 [get_ports {dig[2]}]
set_property PACKAGE_PIN H13 [get_ports {dig[1]}]
set_property PACKAGE_PIN G12 [get_ports {dig[0]}]
set_property PACKAGE_PIN T10 [get_ports {btn_in[3]}]
set_property PACKAGE_PIN R11 [get_ports {btn_in[2]}]
set_property PACKAGE_PIN T12 [get_ports {btn_in[1]}]
set_property PACKAGE_PIN R12 [get_ports {btn_in[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {btn_in[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {btn_in[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {btn_in[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {btn_in[0]}]
set_property PACKAGE_PIN T8 [get_ports stop]
set_property IOSTANDARD LVCMOS33 [get_ports stop]
set_property PACKAGE_PIN L13 [get_ports {seg[7]}]
set_property PACKAGE_PIN M14 [get_ports {seg[6]}]
set_property PACKAGE_PIN P13 [get_ports {seg[5]}]
set_property PACKAGE_PIN K12 [get_ports {seg[4]}]
set_property PACKAGE_PIN K13 [get_ports {seg[3]}]
set_property PACKAGE_PIN L14 [get_ports {seg[2]}]
set_property PACKAGE_PIN N12 [get_ports {seg[1]}]
set_property PACKAGE_PIN P11 [get_ports {seg[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[7]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[2]}]
```

```
set_property IOSTANDARD LVCMOS33 [get_ports {seg[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {seg[0]}]
set_property PACKAGE_PIN N6 [get_ports beepen]
set_property IOSTANDARD LVCMOS33 [get_ports beepen]
set_property PACKAGE_PIN L2 [get_ports beep]
set_property IOSTANDARD LVCMOS33 [get_ports beep]

set_property PACKAGE_PIN F3 [get_ports readen]
set_property PACKAGE_PIN H4 [get_ports rst_s]
set_property IOSTANDARD LVCMOS33 [get_ports rst_s]
set_property IOSTANDARD LVCMOS33 [get_ports readen]
```

6.2 系统编程文件

完成项目系统功能编写及集成后，系统整体编程文件列表如下：



6.3 硬件测试分析

接下来对项目系统进行硬件测试。下列硬件测试图中，左六位 LED 灯指示当前选中数位，右四位 LED 灯指示当前计数器工作模式，右 1 按键表示计数值加一，右 2 按键表示所选数位左移，右 3 按键表示所选数位右移，右 4 按键表示切换工作模式。0001 表示处于 24 小时计时、0010 表示当前处于时间调整、0100 表示当前处于闹钟设置、1000 表示当前处于秒表计时。

图 10 展示正常计数功能在硬件测试中表现。可以看出，当前功能指示为 0001，计时按照“时时-分分-秒秒”格式已达“21-41-32”。

图 11 展示时间调整功能在硬件测试中表现。可以看出，当前功能指示为 0010，测试者通过右 2、右 3 实现左右选位，通过右 1 实现该位数值调整，当前时间调整为“10-30-51”。

图 12 展示闹钟设置功能在硬件测试中表现。可以看出，当前功能指示为 0100，测试者通过右 2、右 3 实现左右选位，通过右 1 实现该位数值调整，当前闹钟时间设定为“00-11-12”。

图 13、图 14 展示秒表计时、计时存储、计时展示在硬件测试中表现。可以看出，当前功能指示为 1000，当前计时值为“00-07-98”。计时时，测试者可通过右 1 控制存储数据，按一次存储一次；读时时，测试者可通过右 1 展示数据，按一次展示一次。

图 10：24 小时计时

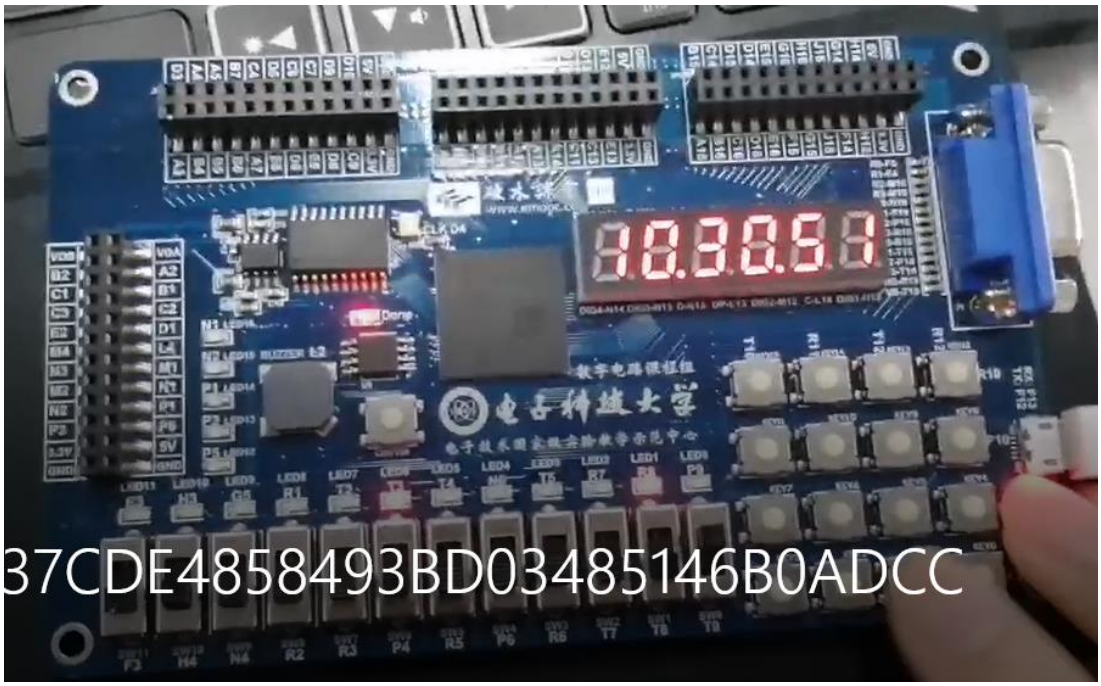


图 11：时间调整

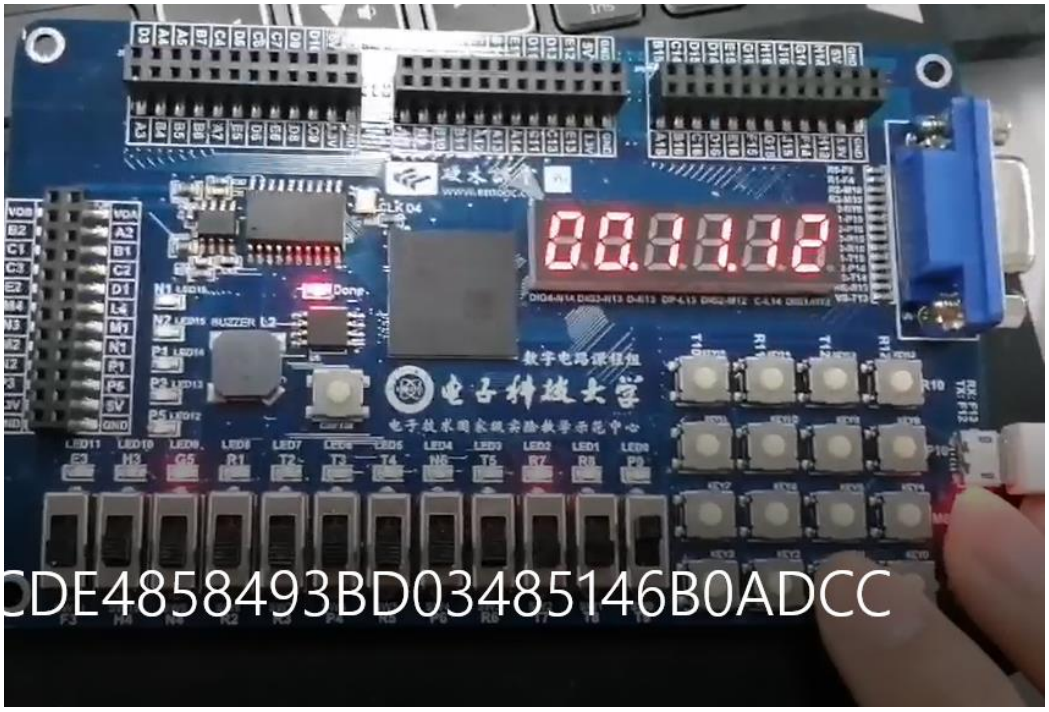


图 12：闹钟设定

图 13：秒表

图 14：读取数据

第7章 结束语

7.1 项目设计的重点与难点

多功能电子时钟系统设计过程中重难点集中于三方面。首先，如何实现计时数据的存储与展示功能;其次,如何使用最少案件资源实现多种功能切换;最后，如何实现自定义定时报警声音。

针对重难点一，设计小组对于数据的存储与展示分别采取不同方法实现。数据存储时，设计小组选择在模块内声明多个 24 位寄存器以实现当前计时的多次存储；数据展示时，设计小组通过数据选择方式实现不同情况下数码管显示电路接收不同数据来源并展示。通过上述手段，设计小组最终解决计时数据的存储与展示功能。

针对重难点二,设计小组选择将多种功能模式集成嵌入于同一计数信号生成模块内，将模式选择指示信号 mode[3:0]设置为独热码形式并通过同一按键的按动实现独热码的移位。小组随后将 mode[3:0]信号作为 always 块的敏感信号，在 mode[3:0]信号变化后结合判断语句实现不同功能。通过上述手段，设计小组可通过单个案件实现四种工作模式的切换，极大节省了按键资源。

针对重点三，设计小组经过对蜂鸣器电路调研发现，开发板上自带蜂鸣器电路属于无源蜂鸣器，其内部不含有振荡源，因此小组选择在定时报警声音模块内对报警时振荡信号进行设定。小组成员对 Do、Re、Mi、Fa、Sol、La、Si 的对应低音、中音、高音频率针对开发板自带时钟频率进行分频，得出各音调对应频率信号。采用上述信号驱动蜂鸣器电路，即可实现定时报警声音的自定义功能。

7.2 项目设计过程中的主要问题与处理方法

多功能电子时钟系统设计过程中，设计小组主要问题在于，如何在多种工作模式切换中保持后台时钟信号的连续性、如何降低项目设计成品的功耗。

问题一出现于小组设计早期,小组成员尝试采用一组寄存器存储不同工作模式下计时数据。在此设计模式下，仅当前工作模式下的计时数据会随时钟改变，即各项功能未成功嵌入于整体系统内,功能间无法实现并行运行。针对上述问题，小组成员经讨论决定在计数信号生成模块内采取多组寄存器分别存储不同工作模式下的计时数据，并通过当前 mode[3:0]信号决定输出哪种功能对应寄存器数据被输出。

问题而出现于小组设计后期，在小组设计功能基本完成、硬件测试基本通过后受到 Vivado 软件的系统警告，声称小组设计系统功耗过高。小组成员讨论分析后认为，功耗过高的可能原因在于设计中存在可能的并行驱动所有 IO 输出的问题。为解决上述系统警告，小组成员尝试降低时钟占空比、降低数码管动态显示扫描频率等方法，但功耗降低并不明显。通过与实验项目指导教师陈学英老师讨论，小组成员认为可能是 Vivado 软件平台的功耗计算算法存在漏洞，故小组并未在此问题继续深究。

7.3 收获与改进意见

多功能电子时钟系统设计过程中，小组对于实际项目设计中软件、硬件均有了更加深入的了解。软件方面，电子时钟系统不同功能的实现与集成对于“自顶而下”设计方法的理解提出了较高要求，且单个功能的实现对于 Veriloge HDL

语言的运用做出了校验，整体提高了设计小组的软件设计能力；硬件方面，对于项目系统硬件资源占用、项目系统功耗的优化过程促使小组成员深入了解硬件平台的资源构成、电路组成及驱动原理，深化了小组成员对于硬件平台的认知水准。总体而言，本次多功能电子时钟设计项目极大程度上提高了小组成员 EDA 设计能力。

考虑到多功能电子时钟系统需求多种功能模块的集成与嵌入，整体实现难度较高，小组成员建议实验课时安排更多、更分散，从而留给实验参与者更多时间调研项目设计相关知识、深入了解实验原理。

附录

顶层模块代码：

```
1. module top(
2. input clk50mhz,
3.     input en,
4.     input stop,
5.     input readen,
6.     input rst_s,
7.     input  [3:0] btn_in,
8.     output [9:0] led,
9.     output [5:0] dig,//低有效
10.    output [7:0] seg,
11.    output beepen,
12.    output beep
13.    );
14.    wire clk10khz;
15.    wire clk100hz;
16.    wire clk1hz;
17.    wire clk5hz;
18.    wire [3:0] cnt1,cnt2,cnt3,cnt4,cnt5,cnt6;//记录生成的时间
19.    wire [23:0] cnts;//记录 save 的时间
20.    wire [3:0] cnt1r,cnt2r,cnt3r,cnt4r,cnt5r,cnt6r;//送入显
   示模块的时间
21.    wire [3:0] mode;//0001 24 小时计时  0010 调时间  0100 闹钟
   设置  1000 秒表
22.    wire [3:0] btnout;
23.    wire [5:0] sel;
24.    wire [23:0] tempcnt;
25.    timecreator tc(
26.    .clk50mhz(clk50mhz),
27.    .en(en),
28.    .clk10khz(clk10khz),
29.    .clk100hz(clk100hz),
30.    .clk1hz(clk1hz),
31.    .clk5hz(clk5hz)
32.    );
33.    cntcontrol c1(
34.    .mode(mode),//00 24 小时计时  01 调时间  10 闹钟设置  11 秒
   表
35.    .en(en),
36.    .clk50mhz(clk50mhz),
37.    .btnin(btn_in),
```

```
38.     .btnout(btnout)//3 控制模式 2 和 1 控制左右选位  0 表示调节
    按钮
39.     );
40.     ledrun lr1(
41.     .mode(mode),
42.     .sel(sel),
43.     .led(led)
44.     );
45.     cntmaker cm1(
46.     .clk100hz(clk100hz),
47.     .clk1hz(clk1hz),
48.     .mode(mode),
49.     .en(en),
50.     .rst_s(rst_s),
51.     .stop(stop),
52.     .sel1(sel),
53.     .btnout(btnout),
54.    .cnt_1(cnt1),.cnt_2(cnt2),.cnt_3(cnt3),.cnt_4(cnt4),.cnt
    _5(cnt5),.cnt_6(cnt6)
55.     );
56.
57.     assign tempcnt = {cnt6,cnt5,cnt4,cnt3,cnt2,cnt1};
58.     clocktriger ct1(
59.     .en(mode[2]),
60.     .cnt(tempcnt),
61.     .clk(clk100hz),
62.     .rst(en),
63.     .beepen(beepen)
64.     );
65.     buzz b1(
66.     .beepen(beepen),
67.     .b_eep(beep),
68.     .clk50mhz(clk50mhz),
69.     .clk1hz(clk5hz)
70.     );
71.     save s1(
72.     .en(mode[3]),//链接 mode[3]
73.     .cntin(tempcnt),
74.     .button(btnout[0]),//链接 bout0
75.     .readen(readen),//链接一个拨码开关
76.     .rst(rst_s),//链接一个拨码开关
77.     .cntout(cnts)
78.     );
79.     assign cnt1r=(mode[3]&&readen)?cnts[3:0]:cnt1;
```

```
    assign cnt2r=(mode[3]&&readen)?cnts[7:4]:cnt2;
    assign cnt3r=(mode[3]&&readen)?cnts[11:8]:cnt3;
    assign cnt4r=(mode[3]&&readen)?cnts[15:12]:cnt4;
    assign cnt5r=(mode[3]&&readen)?cnts[19:16]:cnt5;
    assign cnt6r=(mode[3]&&readen)?cnts[23:20]:cnt6;
    segmaker sm1(
    .cnt1(cnt1r),.cnt2(cnt2r),.cnt3(cnt3r),.cnt4(cnt4r),.cnt5(cnt5r),.cnt6(cnt6r),
    .clk10khz(clk10khz),
    .dig(dig),//低有效
    .seg(seg)
    );
endmodule
```

分频模块代码：

```
module timecreator(
    input clk50mhz,
    input en,
    input rst_s,
    output reg clk10khz,
    output reg clk100hz,
    output reg clk1hz,
    output reg clk5hz
    );
    reg [27:0] cnt1,cnt2,cnt3,cnt4;
    always@(posedge clk50mhz)
    begin
        if(~en)
        begin
        cnt1<=28'd0;clk10khz<=1'b0;
        end
        else
        begin
        if(cnt1<2499)
        cnt1<=cnt1+1;
        else
        begin
        clk10khz<=~clk10khz;
        cnt1<=28'd0;
        end
        end
    end

       always@(posedge clk50mhz)
    begin
```

```
31.         if(~en)
32.         begin
33.             cnt2<=28'd0;clk100hz<=1'b0;
34.         end
35.         else
36.         begin
37.         if(cnt2<249999)
38.         begin
39.         cnt2<=cnt2+1;clk100hz<=1'b0;
40.         end
41.         else
42.         begin
43.         clk100hz<=1;
44.         cnt2<=28'd0;
45.
46.         end
47.         end
48.     end
49.        always@(posedge clk50mhz)
50.     begin
51.         if(~en)
52.         begin
53.             cnt3<=28'd0;clk1hz<=1'b0;
54.         end
55.         else
56.         begin
57.         if(cnt3<24999999)
58.         cnt3<=cnt3+1;
59.         else
60.         begin
61.         clk1hz<=~clk1hz;
62.         cnt3<=28'd0;
63.         end
64.         end
65.     end
66.     always@(posedge clk50mhz)
67.     begin
68.         if(~en)
69.         begin
70.         cnt4<=28'd0;clk5hz<=1'b0;
71.         end
72.         else
73.         begin
74.         if(cnt4<4999999)
```

```
        cnt4<=cnt4+1;
        else
        begin
        clk5hz<=~clk5hz;
        cnt4<=28'd0;
        end
        end
    end
endmodule
```

计时控制模块代码：

```
module cntcontrol(
    input [3:0] btnin,
    input clk50mhz,
    input en,
    output [3:0] btnout,
    output reg [3:0] mode
    );
    wire [3:0] btn_out;
    assign btnout = btn_out;
    ajxd ajxd1(
    .clk50mhz(clk50mhz),
    .btn_in(btnin),//3 控制模式 2 和 1 控制左右选位  0 表示调节按钮
    .btn_out(btn_out),
    .rst_n(en)
    );
    always@(posedge btn_out[3] or negedge en)
    begin
        if(~en)
            mode<=4'b0001;
        else
        begin
                mode[3:1]<=mode[2:0];
                mode[0]<=mode[3];
        end
    end

endmodule
```

按键消抖代码：

```
module ajxd(
    input clk50mhz,
    input  [3:0] btn_in,
    output [3:0] btn_out,
    input rst_n
```

```
    );

reg clk_20ms = 0;
reg [31:0] temp = 0;
always@(posedge clk50mhz)
begin
    if(rst_n == 0)
    begin
        temp <= 0;
        clk_20ms <= 0;
    end
    else if(temp == 999999)
        begin
            temp <= 0;
            clk_20ms <= 1;
        end
        else if(temp < 999999)
            begin
                clk_20ms <= 0;
                temp <= temp+1;
            end
end

reg [3:0] btn0;
reg [3:0] btn1;
reg [3:0] btn2;
initial
begin
btn0=0;
btn1=0;
btn2=0;
end

always@(posedge clk_20ms)
begin
btn0<=btn_in;
btn1<=btn0;
btn2<=btn1;
end

assign btn_out=((btn0&btn1)&(~btn2) | (btn0&btn1&btn2) | ((~btn0)&btn1&btn2));

endmodule
```

指示模块代码：

```
module ledrun(
    input [3:0] mode,
    input [5:0] sel,
    output [9:0] led
    );
    assign led[3:0]=mode;
    assign led[9:4]=sel;
endmodule
```

计时生成模块代码：

```
module cntmaker(
    input clk100hz,
    input clk1hz,
    input en,
    input stop,
    input rst_s,
    input [3:0] mode,
    input [3:0] btnout,
    output [5:0] sel1,
    output [3:0] cnt_1,cnt_2,cnt_3,cnt_4,cnt_5,cnt_6
    );
    reg [5:0] sel;
    wire cout1,cout2,cout3,cout4,cout5,flag,flag1;
    wire sout1,sout2,sout3,sout4,sout5,flag2;
    assign sel1=sel;
    assign flag=flag1|flag2;
    always@(posedge btnout[1] or negedge en)
    begin
        if(~en)
            sel<=6'b000001;
        else
        begin
                sel[5:1]<=sel[4:0];
                sel[0]<=sel[5];
        end
    end
    cnt1 cnt1
    (
    .clk1(clk1hz),
    .clk2(clk100hz),
    .en(en),
    .stop(stop),
    .mode(mode),
    .rst_s(rst_s),
```

```
35.     .sel(sel[0]),
36.     .button(btnout[0]),
37.     .cntout(cnt_1),
38.     .upper1(cout1),
39.     .upper2(sout1)
40.     );
41.     cnt2 cnt2
42.     (
43.     .clk1(cout1),
44.     .clk2(sout1),
45.     .en(en),
46.     .rst_s(rst_s),
47.     .stop(stop),
48.     .mode(mode),
49.     .sel(sel[1]),
50.     .button(btnout[0]),
51.     .cntout(cnt_2),
52.     .upper1(cout2),
53.     .upper2(sout2)
54.     );
55.     cnt3 cnt3
56.     (
57.     .clk1(cout2),
58.     .clk2(sout2),
59.     .en(en),
60.     .stop(stop),
61.     .rst_s(rst_s),
62.     .mode(mode),
63.     .sel(sel[2]),
64.     .button(btnout[0]),
65.     .cntout(cnt_3),
66.     .upper1(cout3),
67.     .upper2(sout3)
68.     );
69.         cnt4 cnt4
70.     (
71.     .clk1(cout3),
72.     .clk2(sout3),
73.     .en(en),
74.     .stop(stop),
75.     .mode(mode),
76.     .rst_s(rst_s),
77.     .sel(sel[3]),
78.     .button(btnout[0]),
```

```
    .cntout(cnt_4),
    .upper1(cout4),
    .upper2(sout4)
    );
    cnt5 cnt5
    (
    .clk1(cout4),
    .clk2(sout4),
    .en(en),
    .flag(flag),
    .rst_s(rst_s),
    .stop(stop),
    .mode(mode),
    .sel(sel[4]),
    .button(btnout[0]),
    .cntout(cnt_5),
    .upper1(cout5),
    .upper2(sout5)
    );
    cnt6 cnt6
    (
      .clk1(cout5),
      .clk2(sout5),
      .en(en),
      .stop(stop),
      .mode(mode),
      .rst_s(rst_s),
      .sel(sel[5]),
      .button(btnout[0]),
      .cntout(cnt_6),
      .upper1(flag1),
      .upper2(flag2)
      );
endmodule
```

计数器 1 代码：

```
module cnt1(
    input clk1,clk2,
    input stop,
    input en,
    input [3:0] mode,
    input sel,
    input button,
    input rst_s,
    output reg [3:0] cntout,
```

```
    output reg upper1,upper2
    );
    reg [3:0] cnttemp1,cnttemp2,cnttemp3;
    reg clk;
    always@(clk1,button)
    begin
        case(mode)
        4'b0001:clk<=clk1;
        4'b0010:clk<=button;
        default:clk<=clk1;
        endcase
    end
    always@(mode)
    begin
        case(mode)
        4'b0100:cntout<=cnttemp2;
        4'b1000:cntout<=cnttemp3;
        default:cntout<=cnttemp1;
        endcase
    end
    always@(negedge clk or negedge en)//时钟与调整时间
    begin
        if(~en)
        begin
            upper1<=1'b0;cnttemp1<=4'd0;
        end
        else
        begin
            if(stop)
            ;
            else
                begin
                    case(mode)
                    4'b0010:begin
                        if(sel)
                        begin
                                upper1<=1'b0;
                                if(cnttemp1==4'd9) begin cn
ttemp1<=4'd0; end
                                else begin cnttemp1<=cnttem
p1+1;end
                        end
                    end

```

```
52.                     default:begin
53.                                     if(cnttemp1==4'd9) begin cn
   ttemp1<=4'd0; upper1<=1'b1;end
54.                                     else begin cnttemp1<=cnttem
   p1+1;upper1<=1'b0; end
55.                     end
56.                     endcase
57.                 end
58.           end
59.     end
60.     always@(posedge button or negedge en)
61.      begin
62.         if(~en)
63.         begin
64.             cnttemp2<=4'd0;
65.         end
66.         else
67.         begin
68.             if(stop)
69.             ;
70.             else
71.                 begin
72.                     case(mode)
73.                     4'b0100:begin
74.                                     if(sel)

75.                                     begin

76.                                             if(cnttemp2==4'd9)
   begin cnttemp2<=4'd0; end
77.                                             else begin cnttemp2
   <=cnttemp2+1; end
78.
79.                                     end

80.                     end
81.
82.                     default:;
83.                     endcase
84.                 end
85.           end
86.     end
87.     always@(posedge clk2 or negedge en)
88.      begin
```

```
        if(~en)
        begin
            cnttemp3<=4'd0;upper2<=0;
        end
        else
        begin
        if(rst_s) begin cnttemp3=4'd0;upper2=1'd0;end
        else
        begin
            if(stop)
            ;
              else
                  begin
                      case(mode)
                      4'b1000:begin
                                              if(cnttemp3==4'd9) be
   gin cnttemp3<=4'd0; upper2<=1'b1;end
                                              else begin cnttemp3<=
   cnttemp3+1;upper2<=1'b0; end
                                      end

                      default:;
                      endcase
                  end
                  end
            end
     end
 endmodule
```

计数器 2 代码：

```
module cnt2(
    input clk1,clk2,
    input stop,
    input en,
    input [3:0] mode,
    input rst_s,
    input sel,
    input button,
    output reg [3:0] cntout,
    output reg upper1,upper2
    );
    reg [3:0] cnttemp1,cnttemp2,cnttemp3;
    reg clk;
    always@(clk1,button)
    begin
```

```
        case(mode)
        4'b0001:clk<=clk1;
        4'b0010:clk<=~button;
        default:clk<=clk1;
        endcase
    end
    always@(mode)
    begin
        case(mode)
        4'b0100:cntout<=cnttemp2;
        4'b1000:cntout<=cnttemp3;
        default:cntout<=cnttemp1;
        endcase
    end
    always@(posedge clk or negedge en)//时钟与调整时间
    begin
        if(~en)
        begin
            upper1<=1'b0;cnttemp1<=4'd0;
        end
        else
        begin
            if(stop)
            ;
            else
                begin
                    case(mode)
                    4'b0010:begin
                        if(sel)
                        begin
                                upper1<=1'b0;
                                if(cnttemp1==4'd5) begin cn
ttemp1<=4'd0; end
                                else begin cnttemp1<=cnttem
p1+1;end
                        end
                    end

                    default:begin
                                if(cnttemp1==4'd5) begin cn
ttemp1<=4'd0; upper1<=1'b1;end
                                else begin cnttemp1<=cnttem
p1+1;upper1<=1'b0; end
                    end
```

```
                    endcase
                end
          end
    end
    always@(posedge button or negedge en)
     begin
        if(~en)
        begin
            cnttemp2<=4'd0;
        end
        else
        begin
            if(stop)
            ;
            else
                begin
                    case(mode)
                    4'b0100:begin
                                    if(sel)

                                    begin

                                            if(cnttemp2==4'd5)
  begin cnttemp2<=4'd0; end
                                            else begin cnttemp2
  <=cnttemp2+1; end

                                    end

                    end

                    default:;
                    endcase
                end
          end
    end
    always@(posedge clk2 or negedge en or posedge rst_s)
     begin
        if(~en)
        begin
            cnttemp3<=4'd0;upper2<=0;
        end
        else
        begin
```

```
95.         if(rst_s) begin cnttemp3=4'd0;upper2=1'd0;end
96.         else

97.         begin
98.             if(stop)
99.             ;
100.              else
101.                  begin
102.                      case(mode)
103.                      4'b1000:begin
104.                                              if(cnttemp3==4'd9) b
    egin cnttemp3<=4'd0; upper2<=1'b1;end
105.                                             else begin cnttemp3<
    =cnttemp3+1;upper2<=1'b0; end
106.                      end
107.
108.                      default:;
109.                      endcase
110.                  end
111.            end
112.            end
113.      end
114.  endmodule
```

计数器 3 代码：

```
1. module cnt3(
2.     input clk1,clk2,
3.     input stop,
4.     input en,
5.     input [3:0] mode,
6.     input sel,
7.     input button,
8.     input rst_s,
9.     output reg [3:0] cntout,
10.    output reg upper1,upper2
11.    );
12.    reg [3:0] cnttemp1,cnttemp2,cnttemp3;
13.    reg clk;
14.    always@(clk1,button)
15.    begin
16.        case(mode)
17.        4'b0001:clk<=clk1;
18.        4'b0010:clk<=~button;
19.        default:clk<=clk1;
20.        endcase
```

```
    end
    always@(mode)
    begin
        case(mode)
        4'b0100:cntout<=cnttemp2;
        4'b1000:cntout<=cnttemp3;
        default:cntout<=cnttemp1;
        endcase
    end
    always@(posedge clk or negedge en)//时钟与调整时间
    begin
        if(~en)
        begin
            upper1<=1'b0;cnttemp1<=4'd0;
        end
        else
        begin
            if(stop)
            ;
            else
                begin
                    case(mode)
                    4'b0010:begin
                        if(sel)
                        begin
                                upper1<=1'b0;
                                if(cnttemp1==4'd9) begin cn
ttemp1<=4'd0; end
                                else begin cnttemp1<=cnttem
p1+1;end
                        end
                    end

                    default:begin
                                if(cnttemp1==4'd9) begin cn
ttemp1<=4'd0; upper1<=1'b1;end
                                else begin cnttemp1<=cnttem
p1+1;upper1<=1'b0; end
                    end
                    endcase
                end
          end
    end
    always@(posedge button or negedge en)
```

```
61.     begin
62.        if(~en)
63.        begin
64.            cnttemp2<=4'd0;
65.        end
66.        else
67.        begin
68.            if(stop)
69.            ;
70.            else
71.                begin
72.                    case(mode)
73.                    4'b0100:begin
74.                                    if(sel)

75.                                    begin

76.                                            if(cnttemp2==4'd9)
   begin cnttemp2<=4'd0; end
77.                                            else begin cnttemp2
   <=cnttemp2+1; end
78.
79.                                    end

80.                    end
81.
82.                    default:;
83.                    endcase
84.                end
85.          end
86.    end
87.    always@(posedge clk2 or negedge en or posedge rst_s)
88.     begin
89.        if(~en)
90.        begin
91.            cnttemp3<=4'd0;upper2<=0;
92.        end
93.        else
94.        begin
95.          if(rst_s) begin cnttemp3=4'd0;upper2=1'd0;end
96.          else
97.          begin
98.            if(stop)
99.            ;
```

```
            else
                begin
                    case(mode)
                    4'b1000:begin
                                    if(cnttemp3==4'd9) begin cnttemp3<=4'd0; upper2<=1'b1;end
                                    else begin cnttemp3<=cnttemp3+1;upper2<=1'b0; end
                    end
                    default:;
                    endcase
                end
            end
            end
        end
    endmodule
```

计数器 4 代码：

```
module cnt4(
    input clk1,clk2,
    input stop,
    input en,
    input [3:0] mode,
    input sel,
    input button,
    input rst_s,
    output reg [3:0] cntout,
    output reg upper1,upper2
    );
    reg [3:0] cnttemp1,cnttemp2,cnttemp3;
    reg clk;
    always@(clk1,button)
    begin
        case(mode)
        4'b0001:clk<=clk1;
        4'b0010:clk<=~button;
        default:clk<=clk1;
        endcase
    end
    always@(mode)
    begin
        case(mode)
        4'b0100:cntout<=cnttemp2;
        4'b1000:cntout<=cnttemp3;
        default:cntout<=cnttemp1;
```

```
28.         endcase
29.     end
30.     always@(posedge clk or negedge en)//时钟与调整时间
31.     begin
32.         if(~en)
33.         begin
34.             upper1<=1'b0;cnttemp1<=4'd0;
35.         end
36.         else
37.         begin
38.             if(stop)
39.             ;
40.             else
41.                 begin
42.                     case(mode)
43.                     4'b0010:begin
44.                         if(sel)
45.                         begin
46.                                 upper1<=1'b0;
47.                                 if(cnttemp1==4'd5) begin cn
   ttemp1<=4'd0; end
48.                                 else begin cnttemp1<=cnttem
   p1+1;end
49.                         end
50.                     end
51.
52.                     default:begin
53.                                 if(cnttemp1==4'd5) begin cn
   ttemp1<=4'd0; upper1<=1'b1;end
54.                                 else begin cnttemp1<=cnttem
   p1+1;upper1<=1'b0; end
55.                     end
56.                     endcase
57.                 end
58.           end
59.     end
60.     always@(posedge button or negedge en )
61.      begin
62.         if(~en)
63.         begin
64.             cnttemp2<=4'd0;
65.         end
66.         else
67.         begin
```

```
            if(stop)
            ;
            else
                begin
                    case(mode)
                    4'b0100:begin
                                    if(sel)

                                    begin

                                            if(cnttemp2==4'd5) begin cnttemp2<=4'd0; end
                                            else begin cnttemp2<=cnttemp2+1; end

                                    end

                    default:;
                    endcase
                end
          end
    end
    always@(posedge clk2 or negedge en or posedge rst_s)
     begin
        if(~en)
        begin
            cnttemp3<=4'd0;upper2<=0;
        end
        else
        begin
         if(rst_s) begin cnttemp3=4'd0;upper2=1'd0;end
         else
         begin
            if(stop)
            ;
              else
                  begin
                      case(mode)
                      4'b1000:begin
                                      if(cnttemp3==4'd5) begin cnttemp3<=4'd0; upper2<=1'b1;end
```

```
                                    else begin cnttemp3<=cntt
   emp3+1;upper2<=1'b0; end
                        end
                        default:;
                        endcase
                    end
            end
            end
        end
endmodule
```

计数器 5 代码：

```
module cnt5(
    input clk1,clk2,
    input stop,
    input en,
    input [3:0] mode,
    input sel,
    input button,
    input flag,
    input rst_s,
    output reg [3:0] cntout,
    output reg upper1,upper2
    );
    reg [3:0] cnttemp1,cnttemp2,cnttemp3;
    reg clk;
    always@(clk1,button)
    begin
        case(mode)
        4'b0001:clk<=clk1;
        4'b0010:clk<=~button;
        default:clk<=clk1;
        endcase
    end
    always@(mode)
    begin
        case(mode)
        4'b0100:cntout<=cnttemp2;
        4'b1000:cntout<=cnttemp3;
        default:cntout<=cnttemp1;
        endcase
    end
    always@(posedge clk or negedge en)//时钟与调整时间
    begin
        if(~en)
```

```
        begin
            upper1<=1'b0;cnttemp1<=4'd0;
        end
        else
        begin
            if(stop)
            ;
            else
                begin
                    case(mode)
                    4'b0010:begin
                     if(sel)

                     begin

                                if(cnttemp1==4'd9 && flag == 0
    ) begin cnttemp1<=4'd0; end
                                else if(cnttemp1 == 4'd3 && fl
   ag == 1)begin cnttemp1<=4'd0; end
                                else begin cnttemp1<=cnttemp1+
   1; end
                     end

                     end


                    default:begin
                                if(cnttemp1==4'd9 && flag == 0
    ) begin cnttemp1<=4'd0;upper1<=1; end
                                else if(cnttemp1 == 4'd3 && fl
   ag == 1)begin cnttemp1<=4'd0;upper1<=1; end
                                else begin cnttemp1<=cnttemp1+
   1;upper1<=0; end
                    end
                    endcase
                end
          end
    end
    always@(posedge button or negedge en)
     begin
        if(~en)
        begin
            cnttemp2<=4'd0;
        end
```

```
68.         else
69.         begin
70.             if(stop)
71.             ;
72.             else
73.                 begin
74.                     case(mode)
75.                     4'b0100:begin
76.                                 if(sel)

77.                                 begin

78.                                         if(cnttemp2==4'd9 &&
   flag == 0 ) begin cnttemp2<=4'd0; end
79.                                         else if(cnttemp2 == 4
   'd3 && flag == 1)begin cnttemp2<=4'd0; end
80.                                         else begin cnttemp2<=
   cnttemp2+1; end
81.                                 end

82.                     end
83.
84.                     default:;
85.                     endcase
86.                 end
87.           end
88.     end
89.     always@(posedge clk2 or negedge en or posedge rst_s)
90.      begin
91.         if(~en)
92.         begin
93.             cnttemp3<=4'd0;upper2<=0;
94.         end
95.         else
96.         begin
97.         if(rst_s) begin cnttemp3=4'd0;upper2=1'd0;end
98.         else
99.         begin
100.              if(stop)
101.              ;
102.              else
103.                  begin
104.                      case(mode)
105.                      4'b1000:begin
```

```
                                    if(cnttemp3==4'd5) begin cnttemp3<=4'd0; upper2<=1'b1;end
                                    else begin cnttemp3<=cnttemp3+1;upper2<=1'b0; end
                            end
                            default:;
                            endcase
                        end
                end
                end
        end
endmodule

```

计数器 6 代码：

```
module cnt6(
    input clk1,clk2,
    input stop,
    input en,
    input [3:0] mode,
    input sel,
    input button,
    input rst_s,
    output reg [3:0] cntout,
    output reg upper1,upper2
    );
    reg [3:0] cnttemp1,cnttemp2,cnttemp3;
    reg clk;
    always@(clk1,button)
    begin
        case(mode)
        4'b0001:clk<=clk1;
        4'b0010:clk<=~button;
        default:clk<=clk1;
        endcase
    end
    always@(mode)
    begin
        case(mode)
        4'b0100:cntout<=cnttemp2;
        4'b1000:cntout<=cnttemp3;
        default:cntout<=cnttemp1;
        endcase
    end
    always@(posedge clk or negedge en)//时钟与调整时间
```

```
    begin
        if(~en)
        begin
            upper1<=1'b0;cnttemp1<=4'd0;
        end
        else
        begin
            if(stop)
            ;
            else
                begin
                    case(mode)
                    4'b0010:begin
                        if(sel)
                        begin
                                if(cnttemp1==4'd2) begin cnttemp1<=4'd0; upper1<=1'b0;end
                                else if(cnttemp1==4'd1)begin cnttemp1<=cnttemp1+1;upper1<=1'b1;end
                                else begin cnttemp1<=cnttemp1+1;upper1<=1'b0; end
                        end
                    end

                    default:begin
                                if(cnttemp1==4'd2) begin cnttemp1<=4'd0; upper1<=1'b0;end
                                else if(cnttemp1==4'd1)begin cnttemp1<=cnttemp1+1;upper1<=1'b1;end
                                else begin cnttemp1<=cnttemp1+1;upper1<=1'b0; end
                    end
                    endcase
                end
          end
    end
    always@(posedge button or negedge en)
     begin
        if(~en)
        begin
            cnttemp2<=4'd0;
        end
        else
        begin
```

```
69.             if(stop)
70.             ;
71.             else
72.                 begin
73.                     case(mode)
74.                     4'b0100:begin
75.                                     if(sel)

76.                                     begin

77.                                     if(cnttemp2==4'd2) begin cn
   ttemp2<=4'd0; upper2<=1'b0;end
78.                                     else if(cnttemp2==4'd1)begi
   n cnttemp2<=cnttemp2+1;upper2<=1'b1;end
79.                                     else begin cnttemp2<=cnttem
   p2+1;upper2<=1'b0; end
80.                                     end

81.                     end
82.
83.                     default:;
84.                     endcase
85.                 end
86.           end
87.     end
88.     always@(posedge clk2 or negedge en or posedge rst_s)
89.      begin
90.         if(~en)
91.         begin
92.             cnttemp3<=4'd0;
93.         end
94.         else
95.         begin
96.         if(rst_s) begin cnttemp3=4'd0;end
97.         else
98.         begin
99.             if(stop)
100.              ;
101.             else
102.                 begin
103.                     case(mode)
104.                     4'b1000:begin
105.                                     if(cnttemp3==4'd5) begin
   cnttemp3<=4'd0; end
```

```
                                    else begin cnttemp3<=cntt
   emp3+1;end
                        end
                        default:;
                        endcase
                    end
            end
            end
        end
endmodule

```

报警触发代码：

```
module clocktriger(
    input en,
    input [23:0] cnt,
    input clk,
    input rst,
    output reg beepen
    );
    reg [23:0] set;
    always@(posedge clk)
    begin
        if(~rst)
            set<=24'd0;
        else
        begin
            if(en)
                set<=cnt;
             else
             begin
                if(cnt==set)
                    begin
                    beepen <= 1;
                    end
                else beepen<=0;
             end
        end
    end
endmodule
```

报警声音代码：

```
module buzz(
    input beepen,
    output b_eep,
    input clk50mhz,
```

```
    input clk1hz
    );
    reg beep = 0;
    reg [2:0] timecnt;
    reg timecnten;
    reg [31:0] cnt=32'd0;
    parameter      MIN_DO = 18'd190800,//(50_000_000/262)
                    MIN_RE = 18'd170050,//(50_000_000/294)
                    MIN_MI = 18'd151500,//(50_000_000/330)
                    MIN_FA = 18'd143250,//(50_000_000/349)
                    MIN_SO = 18'd127550,//(50_000_000/392)
                    MIN_LA = 18'd113600,//(50_000_000/440)
                    MIN_XI = 18'd101200;//(50_000_000/494)
//中音
    parameter      MID_DO = 17'd95600,//(50_000_000/523)
                    MID_RE = 17'd85150,//(50_000_000/587)
                    MID_MI = 17'd75850,//(50_000_000/659)
                    MID_FA = 17'd71600,//(50_000_000/698)
                    MID_SO = 17'd63750,//(50_000_000/784)
                    MID_LA = 17'd56800,//(50_000_000/880)
                    MID_XI = 17'd50600;//(50_000_000/988)
//高音
    parameter  MAX_DO = 16'd47755,//(50_000_000/1047)
                MAX_RE = 16'd42553,//(50_000_000/1175)
                MAX_MI = 16'd37907,//(50_000_000/1319)
                MAX_FA = 16'd35790,//(50_000_000/1397)
                MAX_SO = 16'd31887,//(50_000_000/1568)
                MAX_LA = 16'd28409,//(50_000_000/1760)
                MAX_XI = 16'd25419;//(50_000_000/1967)
    parameter  TIME_300ms = 24'd14_999_999,//300ms,半拍
                TIME_500ms = 25'd24_999_999;//500ms,一拍
    always@(posedge beepen or posedge clk1hz)
    begin
        if(beepen == 1)
        timecnten<=1;
        else timecnten<=0;
    end
    always@(posedge clk1hz or posedge timecnten)
    begin
    if(timecnten==1) timecnt<=3'd7;
    else if(timecnt!=0) timecnt<=timecnt-1;
    else ;
    end
    always@(posedge clk50mhz)
```

```
49.     begin
50.
51.
52.
53.         if(timecnt==0) ;
54.         else
55.         begin
56.             case(timecnt)
57.             3'd7:begin
58.
59.             if(cnt>= MIN_LA)  begin cnt<=32'd0;beep<=~beep;
    end
60.             else cnt<=cnt+32'd1;
61.             end
62.             3'd6:begin
63.
64.             if(cnt>=  MIN_XI)  begin cnt<=32'd0;beep<=~beep
    ;end
65.             else cnt<=cnt+32'd1;
66.             end
67.             3'd5:begin
68.
69.             if(cnt>=  MID_DO)  begin cnt<=32'd0;beep<=~beep
    ;end
70.             else cnt<=cnt+32'd1;
71.             end
72.             3'd4:begin
73.
74.             if(cnt>=  MID_RE)  begin cnt<=32'd0;beep<=~beep
    ;end
75.             else cnt<=cnt+32'd1;
76.             end
77.             3'd3:begin
78.
79.             if(cnt>= MID_MI)  begin cnt<=32'd0;beep<=~beep;
    end
80.             else cnt<=cnt+32'd1;
81.             end
82.             3'd2:begin
83.
84.             if(cnt>= MID_MI)  begin cnt<=32'd0;beep<=~beep;
    end
85.             else cnt<=cnt+32'd1;
86.             end
```

```
            3'd1:begin
            if(cnt>=  MID_XI)  begin cnt<=32'd0;beep<=~beep;end
            else cnt<=cnt+32'd1;
            end
            3'd0:begin
            if(cnt>=  MID_LA)  begin cnt<=32'd0;beep<=~beep;end
            else cnt<=cnt+32'd1;
            end
            default:;
            endcase
        end
    end
    assign b_eep=beep;
endmodule
```

计时存储代码：

```
module save(
    input en,//链接 mode[3]
    input [23:0] cntin,
    input button,//链接 bout3
    input readen,//链接一个拨码开关
    input rst,//链接一个拨码开关
     output reg [23:0] cntout
    );
    reg [23:0] savecnt [7:0];
    reg [2:0] i,j;
    always@(posedge button)
    begin
        if(en)
        begin
            if(rst)
                begin
                savecnt[7] = 24'd0;
                savecnt[6] = 24'd0;
                savecnt[5] = 24'd0;
                savecnt[4] = 24'd0;
                savecnt[3] = 24'd0;
                savecnt[2] = 24'd0;
                savecnt[1] = 24'd0;
                savecnt[0] = 24'd0;
                i= 3'd0;j=3'd0;
```

```
                end
                else
                begin
                    if(~readen)
                    begin
                            i<=i+1;
                            case(i)
                            3'd0:savecnt[0]=cntin;
                            3'd1:savecnt[1]=cntin;
                            3'd2:savecnt[2]=cntin;
                            3'd3:savecnt[3]=cntin;
                            3'd4:savecnt[4]=cntin;
                            3'd5:savecnt[5]=cntin;
                            3'd6:savecnt[6]=cntin;
                            3'd7:savecnt[7]=cntin;
                            default:;
                            endcase
                    end
                    else
                    begin
                            j<=j+1;
                            case(j)
                            3'd0:cntout=savecnt[0];
                            3'd1:cntout=savecnt[1];
                            3'd2:cntout=savecnt[2];
                            3'd3:cntout=savecnt[3];
                            3'd4:cntout=savecnt[4];
                            3'd5:cntout=savecnt[5];
                            3'd6:cntout=savecnt[6];
                            3'd7:cntout=savecnt[7];
                            default:;
                            endcase
                    end
                end
        end
    end
endmodule
```

数据显示代码：

```
module segmaker(
    input [3:0] cnt1,cnt2,cnt3,cnt4,cnt5,cnt6,
    input clk10khz,
    output reg [5:0] dig,//低有效
    output reg [7:0] seg
    );
```

```
    reg [2:0] flag;
    initial flag=3'b000;
    always@(posedge clk10khz)
    begin
    if(flag==3'd5)
    flag<=3'd0;
    else
    flag<=flag+1;
    end
    always@(flag)
    begin
    case(flag)
    3'd0:
    begin
        dig<=6'b111110;
     case(cnt1)
  4'h0: seg=8'h3f;// DP,GFEDCBA
  4'h1: seg=8'h06;
  4'h2: seg=8'h5b;
  4'h3: seg=8'h4f;
  4'h4: seg=8'h66;
  4'h5: seg=8'h6d;
  4'h6: seg=8'h7d;
  4'h7: seg=8'h07;
  4'h8: seg=8'h7f;
  4'h9: seg=8'h6f;
  default: seg=0;
  endcase
    end
    3'b1:
    begin
         dig<=6'b111101;
        case(cnt2)
  4'h0: seg=8'h3f;// DP,GFEDCBA
  4'h1: seg=8'h06;
  4'h2: seg=8'h5b;
  4'h3: seg=8'h4f;
  4'h4: seg=8'h66;
  4'h5: seg=8'h6d;
  4'h6: seg=8'h7d;
  4'h7: seg=8'h07;
  4'h8: seg=8'h7f;
  4'h9: seg=8'h6f;
  default: seg=0;
```

```
51.  endcase
52.    end
53.    3'd2:
54.    begin
55.     dig<=6'b111011;
56.     case(cnt3)
57.  4'h0: seg=8'hbf;// DP,GFEDCBA
58.  4'h1: seg=8'h86;
59.  4'h2: seg=8'hdb;
60.  4'h3: seg=8'hcf;
61.  4'h4: seg=8'he6;
62.  4'h5: seg=8'hed;
63.  4'h6: seg=8'hfd;
64.  4'h7: seg=8'h87;
65.  4'h8: seg=8'hff;
66.  4'h9: seg=8'hef;
67.  default: seg=0;
68.  endcase
69.    end
70.    3'd3:
71.    begin
72.     dig<=6'b110111;
73.     case(cnt4)
74.  4'h0: seg=8'h3f;// DP,GFEDCBA
75.  4'h1: seg=8'h06;
76.  4'h2: seg=8'h5b;
77.  4'h3: seg=8'h4f;
78.  4'h4: seg=8'h66;
79.  4'h5: seg=8'h6d;
80.  4'h6: seg=8'h7d;
81.  4'h7: seg=8'h07;
82.  4'h8: seg=8'h7f;
83.  4'h9: seg=8'h6f;
84.  default: seg=0;
85.  endcase
86.    end
87.    3'd4:
88.    begin
89.     dig<=6'b101111;
90.    case(cnt5)
91.  4'h0: seg=8'hbf;// DP,GFEDCBA
92.  4'h1: seg=8'h86;
93.  4'h2: seg=8'hdb;
94.  4'h3: seg=8'hcf;
```

```
95.  4'h4: seg=8'he6;
96.  4'h5: seg=8'hed;
97.  4'h6: seg=8'hfd;
98.  4'h7: seg=8'h87;
99.  4'h8: seg=8'hff;
100.    4'h9: seg=8'hef;
101.    default: seg=0;
102.    endcase
103.      end
104.      3'd5:
105.      begin
106.       dig<=6'b011111;
107.      case(cnt6)
108.    4'h0: seg=8'h3f;// DP,GFEDCBA
109.    4'h1: seg=8'h06;
110.    4'h2: seg=8'h5b;
111.    4'h3: seg=8'h4f;
112.    4'h4: seg=8'h66;
113.    4'h5: seg=8'h6d;
114.    4'h6: seg=8'h7d;
115.    4'h7: seg=8'h07;
116.    4'h8: seg=8'h7f;
117.    4'h9: seg=8'h6f;
118.    default: seg=0;
119.    endcase
120.      end
121.      default:;
122.      endcase
123.      end
124. endmodule
```