

# Project Document

---

## 0. Teamwork

---

### 0.1 我们每个人贡献比例相同：

张渤昊(12312223): 细节完成与打磨, 子模块(gesture\_control, set\_gesture\_time, on\_off) 文档书写, 蜂鸣器调试.

李泽均(12311010): 外部设备(蓝牙 and 手机上微信小程序的使用), 子模块(bluetooth-related, audio)

张意恒(12311013): 顶层模块, idea贡献, 子模块(mode\_change, clock, reminder, seg\_related, debounce, bonetime, data\_select), 计划制作, 文档打磨

## 0.2 Planned Schedule:

- Week 11:
  - 小组讨论
  - 理解project
- Week 12, 13:
  - 完成项目基本功能
  - bonus部分讨论外设选择
  - 完成子模块 (led, set\_gesture\_time, mode\_change)
- Week 14:
  - 完成所有单个模块设计
  - 完成项目bonus
  - 上板测试并Debug

## 1. 系统功能列表

---

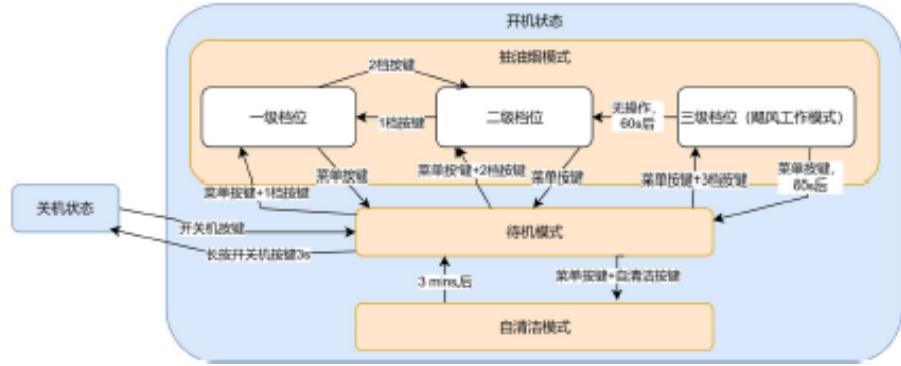
### 1.1 功能概述：

设计一个抽油烟机：抽油烟机分为开机状态和关机状态。开机状态下有三种工作模式：待机模式、抽油烟模式和自清洁模式。抽油烟模式下分为三档：1档、2档、3档（飓风工作档位）。抽油烟机通过自清洁模式对抽油烟机进行清洗，并在长时间、多次数工作后发出清洗提醒。

在开机状态下，能通过按下switch[7] (开机键) 随时开启关闭照明功能，还能持续显示时间；

在待机模式下，可以分别进入抽油烟模式和自清洁模式，还可以手动设置时间。

用户可以通过设置，在规定时间内实现特殊手势开关机（详细部分在bonus板块有介绍）。（下图为抽油烟机不同模式切换图）设计一个抽油烟机：抽油烟机分为开机状态和关机状态。开机状态下有三种工作模式：待机模式、抽油烟模式和自清洁模式。



抽油烟模式下分为三档：1档、2档、3档（飓风工作档位）。

抽油烟机通过按下 **switch[6]** 自清洁模式 对抽油烟机进行清洗，并在长时间、多次数工作后发出清洗提醒。

在开机状态下，能随时开启关闭照明功能，还能持续显示时间；在待机模式下，可以分别进入抽油烟模式和自清洁模式，还可以手动设置时间。用户可以通过设置，在规定时间内实现特殊手势开关机。具体功能如下

### 1.1.1 参数：

- (1) 出厂设置：以下描述中所有涉及的时间（清洁智能提醒相关的抽油烟工作累积时长10小时，自清洁的15s工作时间、飓风模式下的12s倒计时、开关机手势模拟的5s有效时间）均为出厂设置（为配合项目测试需要）
- (2) 可通过 **init** 按键让这些参数恢复到出厂设置的值。
- (3) 可通过按下 **switch[0], switch[1], switch[2]** 分别设置累计设置提醒时间的上限，手势开关的，有效时间（具体参考“辅助功能”中的“高级设置”项目）

### 1.1.2 开关机：

开机：短按开关机键实现开机操作，开机后抽油烟机完成初始化。

抽油烟机初始化的表现：当前时间为0小时0分0秒，如果查询抽油烟机，按下 **switch[3]** 看累计的使用时间，查询结果为0

关机：长按 3s 开关机键实现关机操作，关机后所有开关按键都失效。

### 1.1.3 模式切换：

- a) 抽油烟机开机后进入待机模式，待机模式下先按 **button[4]** 菜单键 准备切换模式，然后通过不同按键进入不同模式（抽油烟模式（按下 **button[0]** 1档、**button[1]** 2档、**button[2]** 3档）、**button[4]** 自清洁模式），比如待机模式中先按菜单键，再按1档键，即可进入抽油烟的1档工作模式。

从不同的工作模式返回待机模式：

- 抽油烟的1档和2档工作模式都可以通过按菜单键后直接返回待机模式；
- 抽油烟的3档工作模块是在按菜单键后开始12s倒计时，倒计时到期再返回待机模式；
- 抽油烟的自清洁模式在开始工作后开启3mins倒计时，倒计时到期后再返回待机模式；
- 抽油烟的自清洁模式在开始工作后开启3mins倒计时，倒计时到期后再返回待机模式；

### 1.1.4 抽油烟功能：

待机模式下，按动菜单键进入模式切换，再按动档位按键（1档键、2档键、3档键）进入抽油烟模式下的不同风力档位。进入任何档位的抽油烟工作模式时即开启累积计时，结束抽油烟模式时停止计时，存储开机后抽油烟的累积工作时长。该时长将用于自清洁相关的智能提醒功能。

- 风力一级档位：按动1档键，抽油烟机进入一级档位工作，按动菜单键切换回待机模式
- 风力二级档位：按动2档键，抽油烟机进入二级档位工作，按动菜单键切换回待机模式
- 风力一级档位与风力二级档位抽油烟模式下，可以通过按动档位按键相互进行切换
- 风力三级档位（飓风工作模式）：按动3档键，抽油烟机进入三级档位工作，工作模式的12s倒计时，倒计时结束后，自动进入二级档位继续工作。飓风工作模式的12s倒计时未结束时，如按动菜单键进行强制待机，则启动另一个12s倒计时。结束后返回待机模式。
- 颶风模式在每次开机后只能使用一次，如果多次按3档键位，则除第一次外的其他按3档键位的操作无效。

### 1.1.5 自清洁功能：

- a) 自清洁模式：
- i. 仅能通过待机模式进入自清洁模式。
  - ii. 待机模式下，按动菜单键，再按动自清洁按键进入自清洁模式，同时开启15s倒计时，倒计时结束后，提醒自清洁完成，自动进入待机模式。

### 1.1.6 辅助功能：

- a) 照明功能：开机后任何模式下都可以开启或者关闭照明功能
- b) 时间功能：
- i. 时间显示：开机后动态显示当前时间（小时、分钟、秒）
  - ii. 时间设置：待机模式下，通过输入设置时、分调整时间（例如可通过bonus部分的蓝牙）
- c) 智能提醒：抽油烟工作模式下会统计抽油烟工作的累计时长，当该工作时长达到指定数值（默认时长为10小时），在待机模式下抽油烟机将通过输出设备即文档output部分中的提醒led，提醒用户进行手动清洗（可在待机状态下通过switch[6]实现）或者开启自清洁。每次完成自清洁后抽油烟的累计工作时长自动清零，完成手动清洁后可通过手动开关对该统计值清零。
- d) 高级设置：在待机模式下可以对部分参数进行重新设置，设置方法如上文所示。
- 1) 设置触发智能提醒的使用时长的上限，设置后生效，如果不设置则使用出厂时默认设置。  
设置手势开关的有效时间（单位s），比如默认为5s，用户可以调整为7s（更长）或者2s（更短）的时间。方法是：打开switch[1],按下button 0即可实现将时间加1

## 2.系统使用说明（含系统的输入、输出端口的说明）

### 2.1 系统的输入、输出端口

✓  button (5)	IN			
✓ button[4]	IN	R17	▼	
✓ button[3]	IN	U4	▼	
✓ button[2]	IN	R11	▼	
✓ button[1]	IN	R15	▼	
✓ button[0]	IN	V1	▼	
✓  modes (5)	OUT			
✓ modes[4]	OUT	J5	▼	
✓ modes[3]	OUT	K6	▼	
✓ modes[2]	OUT	L1	▼	
✓ modes[1]	OUT	M1	▼	
✓ modes[0]	OUT	K3	▼	
✓  seg2 (8)	OUT			
✓ seg2[7]	OUT	D5	▼	
✓ seg2[6]	OUT	B2	▼	
✓ seg2[5]	OUT	B3	▼	
✓ seg2[4]	OUT	A1	▼	
✓ seg2[3]	OUT	B1	▼	
✓ seg2[2]	OUT	A3	▼	
✓ seg2[1]	OUT	A4	▼	
✓ seg2[0]	OUT	B4	▼	
✓  seg1 (8)	OUT			
✓ seg1[7]	OUT	H2	▼	
✓ seg1[6]	OUT	D2	▼	
✓ seg1[5]	OUT	E2	▼	
✓ seg1[4]	OUT	F3	▼	
✓ seg1[3]	OUT	F4	▼	
✓ seg1[2]	OUT	D3	▼	
✓ seg1[1]	OUT	E3	▼	
✓ seg1[0]	OUT	D4	▼	
✓  tub_control (8)	OUT			
✓ tub_control[7]	OUT	G2	▼	
✓ tub_control[6]	OUT	C2	▼	
✓ tub_control[5]	OUT	C1	▼	
✓ tub_control[4]	OUT	H1	▼	
✓ tub_control[3]	OUT	G1	▼	
✓ tub_control[2]	OUT	F1	▼	
✓ tub_control[1]	OUT	E1	▼	
✓ tub_control[0]	OUT	G6	▼	

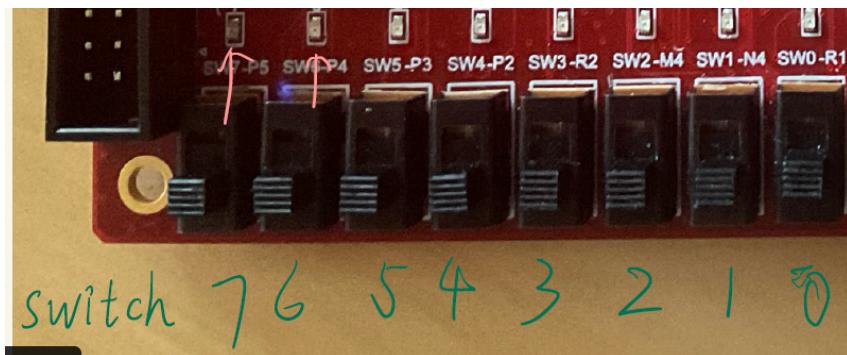
Scalar ports (17)			
bt_master_slave	OUT	C16	▼
bt_pw_on	OUT	D18	▼
bt_RST_n	OUT	M2	▼
bt_sw	OUT	E18	▼
bt_sw_hw	OUT	H15	▼
clk	IN	P17	▼
init	IN	U2	▼
lb_sel_pin	IN	R3	▼
light	OUT	F6	▼
music	OUT	H17	▼
reminder_led	OUT	G4	▼
rst	IN	P15	▼
rst_pin	IN	T5	▼
rxd_pin	IN	L3	▼
sw_pin	IN	U3	▼
t	OUT	T1	▼
txd_pin	OUT	N2	▼

## 2.1.1 Input

1. clk: 系统时钟信号

2. rst: 重置按键

3. switch[7:0]:



不同开关作用：

```

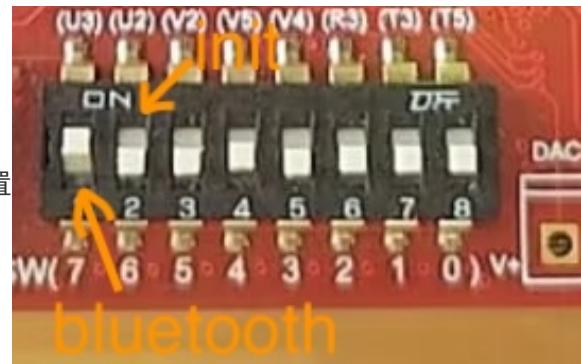
0: Set the clock
1: Set gesture switch
2: Set a ceiling
3: Running time
4: Turn on gesture switch
5: Enable Bluetooth time
6: Manual cleaning
7: LED light
    
```

4. button[4:0]: (关于switch, button在不同情况下的使用以及实现效果参见使用说明部分)

5. rxd\_pin: 接收串行数据（蓝牙UART 数据）输入。

6. lb\_sel\_pin: 选择是否启用回环模式（Loopback Mode）。在回环模式下，蓝牙模块会将接收到的数据直接发送回。（本项目中未使用）

7. sw\_pin: 蓝牙模式开关

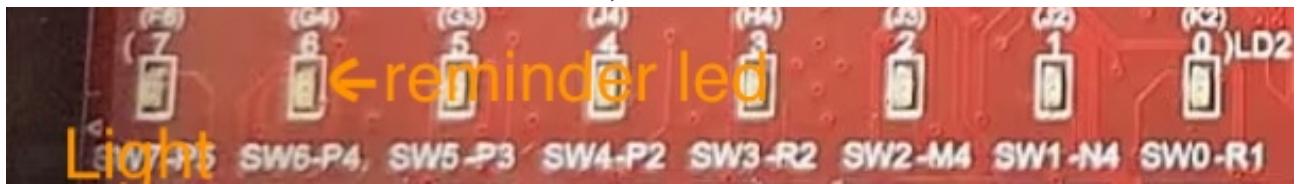


8. init: 在开机之后，开启init并按下button[1]会出厂设置

9. 余下输出，如bt\_pw\_on, bt\_master\_slave 本项目中未用到相关功能

## 2.1.2 Output

1. t: 设置为常量, 0, 使蜂鸣器能够发声
2. music: 1bit常量, 控制蜂鸣器何时作响
3. light: 反映照明功能的灯: 开机后任何模式下都可以开启或者关闭照明功能
4. reminder led: 达到设定时间后 在待机状态下亮起, 提示需要清洁



5. modes: 按位显示，不同modes的表示如图：



```

5'b00000, //Power Off
5'b00001, //standby
= 5'b00010, //Settings
= 5'b00011, //inquire
5'b00100, //menu
5'b00101, //First gear
5'b00110, //Second gear
5'b00111, //Third gear
5'b01000, //self-cleaning
5'b01001: //Third gear return
    
```

6. seg\_data, seg\_data2: 左, 右 晶体管 展示.

7. tub\_control: 流动灯，从而使视觉上看到同时在亮的八个晶体管

8. txd\_pin: 发送串行数据 (UART 数据) 输出。将数据从主设备发送到蓝牙模块.

9. bt\_sw: 蓝牙模式指示灯 蓝牙模式开启后，会进入闪烁状态，连接上设备后转为常亮

## 2.2 系统具体使用说明

### 2.2.1 关机模式下：

- 按动button[3]开机

- 在打开switch[1]的状态下，可通过手势开关开机

## 2.2.2 待机模式下：

- 长按button[3]关机
- 在打开switch[1]的状态下，可通过手势开关开机
- 按动button[4]进入菜单模式
- 打开init开关可以按动button[1]恢复出厂设置
- 可以通过bluetooth开关启用蓝牙输入
- 可以通过switch[7]打开照明
- 可以通过switch[6]进行手动清洁
- 可以通过switch[5]显示和启用蓝牙输入
- 可以通过switch[3]查看运行时间
- 可以通过switch[2]设置提醒上限时间
- 可以通过switch[1]设置手势开关时间
- 可以通过switch[0]设置时钟时间

## 2.2.3 菜单模式下：

- 长按button[3]关机
- 在打开switch[1]的状态下，可通过手势开关开机
- 按动button[4]进入菜单模式
- 按动button[0]进入一档模式
- 按动button[1]进入二档模式
- 按动button[2]进入三档模式（飓风模式）

## 2.2.4 一档模式下：

- 长按button[3]关机
- 在打开switch[1]的状态下，可通过手势开关开机
- 按动button[1]进入二档模式
- 按动button[4]返回待机模式

## 2.2.5 二档模式下：

- 长按button[3]关机
- 在打开switch[1]的状态下，可通过手势开关开机
- 按动button[0]进入一档模式
- 按动button[4]返回待机模式

## 2.2.6 三档模式下：

- 长按button[3]关机

- 在打开switch[1]的状态下，可通过手势开关开机
- 按动button[4]进入倒计时，之后返回待机模式
- 如不进行操作，倒计时结束后自动返回二档模式

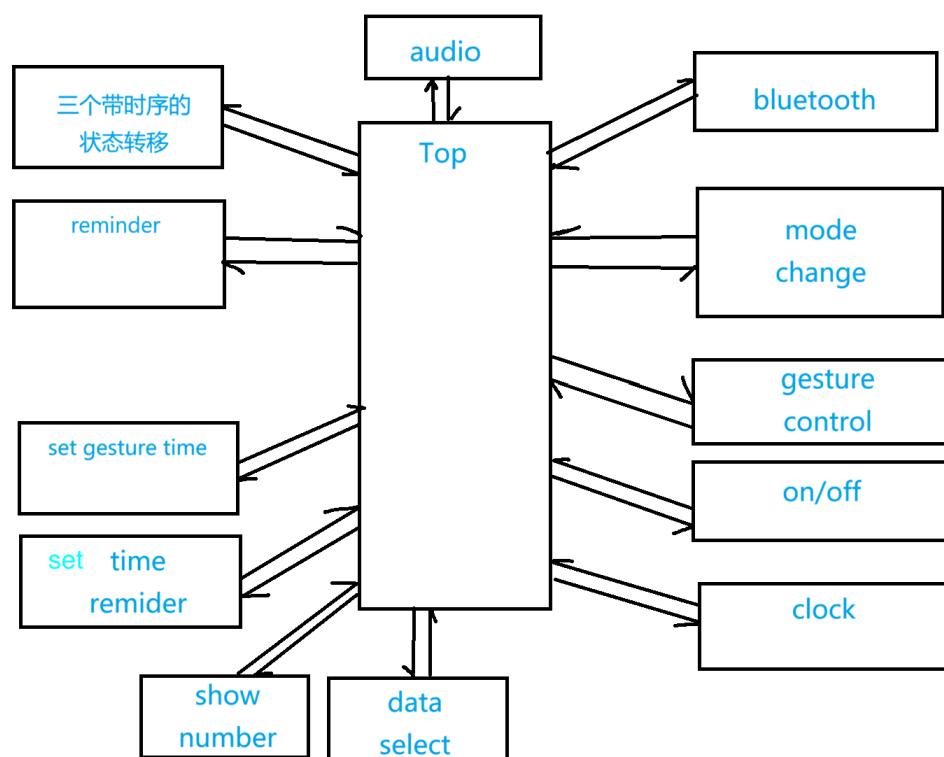
### 2.2.7 自清洁模式下：

- 倒计时结束后自动返回待机模式

## 2.系统结构说明

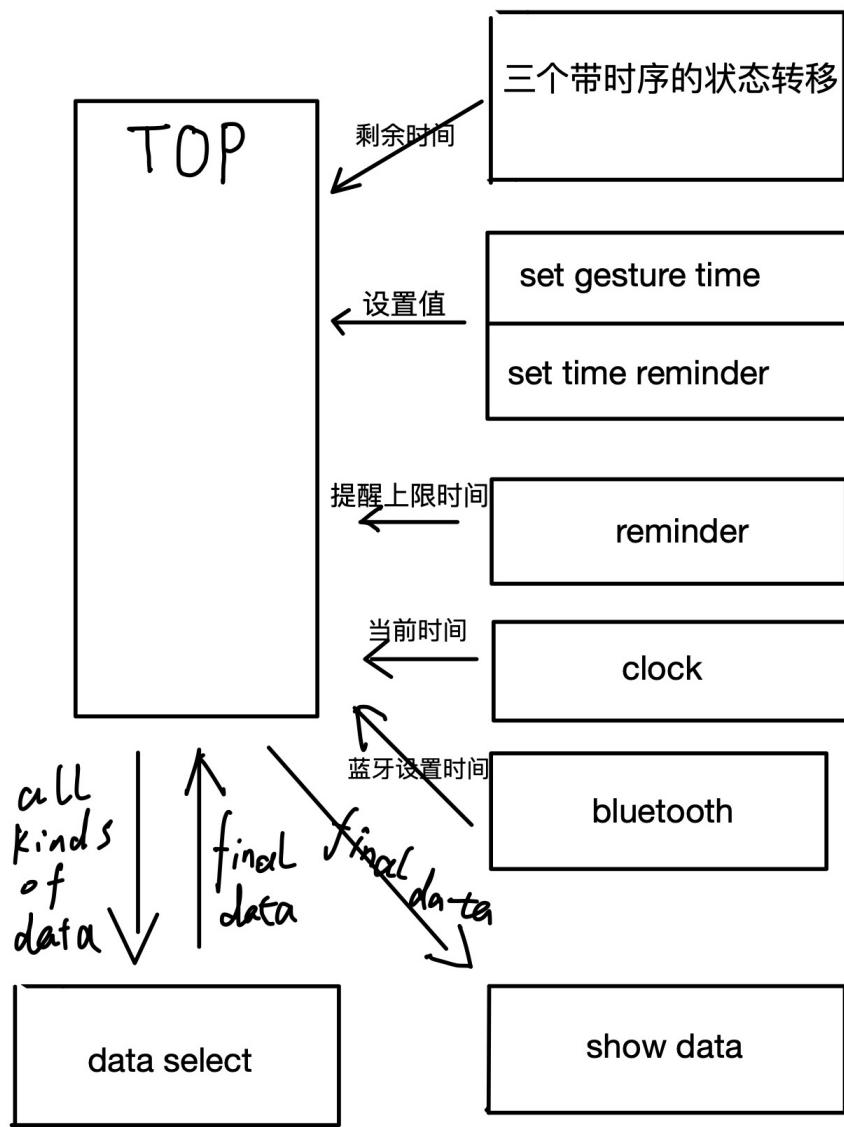
### 2.1 总体结构

以top module 作为主体，子模块之间的交互都需要经过top module.

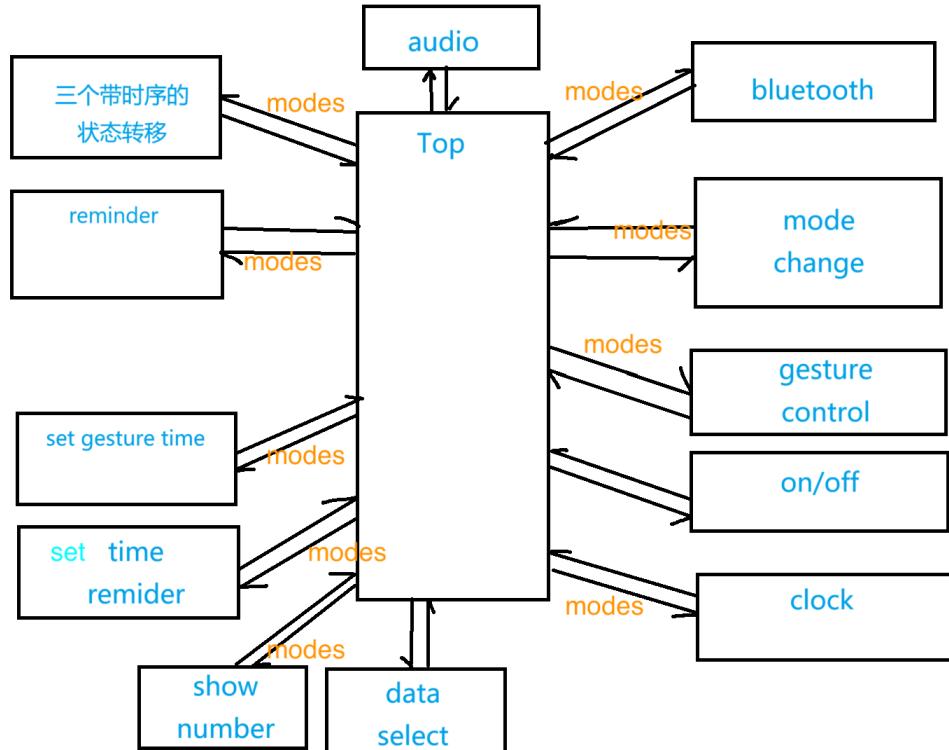


### 2.2 主要信号的传递

#### 2.2.1 data信号(均为32bits, 按一定格式存储)



## 2.2.2 modes信号



### 3.子模块功能说明：输入、输出端口说明、子模块功能说明

#### 3.1 具体子模块实现

##### 3.1.1 module debounce

实现按键的消抖功能

输入的input : clk 系统时钟

run\_stop 控制系统是否正常工作

input\_in 从顶层模块进入的按键

输出的output: key\_out 消抖后的按键

##### 3.1.2 module bonetime

实现效果：信号上升沿的检测

输入的：clk 系统时钟

x\_in输入的信号

输出的 x\_out/输出的信号

##### 3.1.3 module getsture\_control\_on

实现效果：手势开关实现开机（先按左键开启5s倒计时，在5s内再按右键）

输入的： clk/ 系统时钟 reset/重置 button1 /按键1即左键  
button2 /按键2即右键 modes/从顶层模块输入的模式状态  
gesture\_time/从顶层模块输入的手势时间（配合set\_gesture\_time使用）

输出的： on/控制系统开关

### 3.1.4 module getgesture\_control\_off

实现效果：手势开关实现关机（先按右键开启5s倒计时，在5s内再按左键为关机操作）

输入的： clk/ 系统时钟 reset/重置 button1 /按键1即左键  
button2 /按键2即右键 modes/从顶层模块输入的模式状态  
gesture\_time/从顶层模块输入的手势时间（配合set\_gesture\_time使用）

输出的： on/控制系统开关

### 3.1.5 module clock

输入信号 clk: 时钟信号，控制逻辑触发。

rst: 异步复位信号，高电平复位。

mode: 模式选择，控制模块的操作行为。

switch: 开关信号，用于触发特定的功能。

bin: 数据输入，用于控制时间调节模式。

输出信号: data: 用于输出当前时间的BCD格式数据（31:28为小时十位，27:24为小时个位，19:16为分钟十位，15:12为分钟个位，7:4为秒十位，3:0为秒个位）。

实现效果

timer\_cnt: 一个计数器，用于控制秒级时间增量（假设时钟频率为100 MHz，每1秒递增）。

seconds, minutes, hours: 分别用于存储当前的秒、分钟和小时。

who: 在时间调节模式下，用于选择当前修改的时间单位（小时、分钟或秒）。

功能描述

(1) 时间调节模式 (Mode=00001)

当mode == 5'b00001且switch[0] == 1时，进入时间调节模式：

如果bin[1]为高电平：

变量who递增，用于切换调节的时间单位（小时、分钟、秒）。

who值在0~2之间循环。

如果bin为5'b00001，具体调节逻辑如下：

who == 0时，小时递增。

who == 1时，分钟递增。

who == 2时，秒递增。

调节过程中，小时、分钟和秒都有溢出复位机制（24小时制，60分钟制，60秒制）。

#### (2) 时间计时模式 (Mode=00000)

在mode == 5'b00000时，模块重置时间计数器(timer\_cnt)和时间值（小时、分钟、秒）。

#### (3) 计时功能

每经过timer\_cnt >= 100\_000\_000（假设时钟频率为100 MHz），秒数递增：

当秒数达到60时，清零并让分钟数递增。

当分钟数达到60时，清零并让小时数递增。

当小时数达到24时，清零。

#### (4) 时间显示

使用always @(\*)块，根据小时、分钟和秒的值将时间转换为BCD格式存储到data中：(We will use this rule in this document later)

data[31:28]存储小时的十位数。

data[27:24]存储小时的个位数。

data[19:16]存储分钟的十位数。

data[15:12]存储分钟的个位数。

data[7:4]存储秒的十位数。

data[3:0]存储秒的个位数。

用04'hf作为分隔符显示（冒号）。

### 3.1.6 module data\_select

实现效果：将传入顶层模块的不同data（用于显示时间）根据传入的modes变量进行选择 最终的数据，并传入顶层模块 data\_final

输入输出：传入不同的32位相同格式的数据，并根据模式选择最终显示data

```
input          clk      ,
input          rst      ,
//传入当前modes和switch
input [4:0]    modes   ,
input [7:0]    switch  ,
//传入不同data，因为本项目data的命名相同，可去对应模块查看相应data含义
input [31:0]   data_clock  ,
input [31:0]   data_set_gt  ,
input [31:0]   data_set_r   ,
input [31:0]   data_time   ,
```

```

input [31:0]      data_cmc      ,
input [31:0]      data_three    ,
input [31:0]      data_three_return ,
input [31:0]      data_bluetooth   ,
//传出最终显示data
output reg[31:0] data_final

```

### 3.1.7 module show\_number

主要功能是接收32位数据（根据上文规则），将其转换为数码管可显示的段码并通过扫描动态显示。并且实现了时钟分频，将输入时钟 clk 分频为 500Hz，便于驱动数码管的动态扫描 (clk\_500hz)。

输入输出：

```

input          clk      ,
input          rst      ,
input [31:0]   data      , //需要显示的数据
output reg[7:0] seg_data  ,
output reg[7:0] seg_data2 ,
output reg[7:0] seg_cs

```

### 3.1.8 module setReminder

实现效果：

1. 初始化/复位，即当reset为低电平或init为高时，count，时分秒都被重置0
2. 计数增量，增量规则与clock相同

输入输出：

```

input          clk      ,
input          rst      ,
input [4:0]    mode     ,
input [7:0]    switch   ,
input [4:0]    bin      ,
input          init      , //初始信号，为1'b1时值为初始设置值
output reg [31:0] data

```

### 3.1.9 module set\_gesture\_time

实现效果是

1. 初始化/复位，即当reset为低电平或init为高时，count被重置为 (4'b0101)
2. 计数增量，即在enable信号为高电平的时候，检查mode=5'b00001且switch == 8'b0000\_0010 并且addition为高

如果满足条件，那么count增加，当count ==15时，重置为0.

输入输出:

```
input          clk      ,
input          reset     ,
input [4:0]    mode      ,
input [7:0]    switch    ,
input          enable     ,
input          addition   ,
input          init      , //初始信号，为1'b1时值为初始设置值
output reg [31:0] data    , //传入的设置值
output reg [3:0]  count
```

### 3.1.10 module counter

主要功能是一个通用的计数器模块，支持计时（Countup）和倒计时（Countdown）两种模式

输入输出:

```
input clk,
input rst,
input enable,
input reverse, //1'b1为正计时，1'b0为倒计时
input [31:0] start, //倒计时开始时间，正计时时可随便填写
//时分秒分别传出
output reg [6:0]seconds,
output reg [6:0]minutes,
output reg [5:0]hours
```

### 3.1.11 module three\_return\_Counter

基于counter 实现倒数并在结束后传出信号用于进行状态转移。

输入输出:

```
input          clk      ,
input          rst      ,
input [4:0]    mode      ,
output reg    signal    , //传出结束信号
output reg [31:0] data //传出格式化时间，用于显示
```

### 3.1.12 module three\_Counter

基于counter 实现倒数并在结束后传出信号用于进行状态转移。

输入输出:

```
input          clk      ,
input          rst      ,
input [4:0]    mode     ,
output reg    signal   , //传出结束信号
output reg [31:0] data //传出格式化时间，用于显示
```

### 3.1.13 module clean\_mode\_counter

基于counter 实现倒数并在结束后传出信号用于进行状态转移。

输入输出:

```
input          clk      ,
input          rst      ,
input [4:0]    mode     ,
output reg    signal   , //传出结束信号
output reg [31:0] data //传出格式化时间，用于显示
```

### 3.1.14 module audio

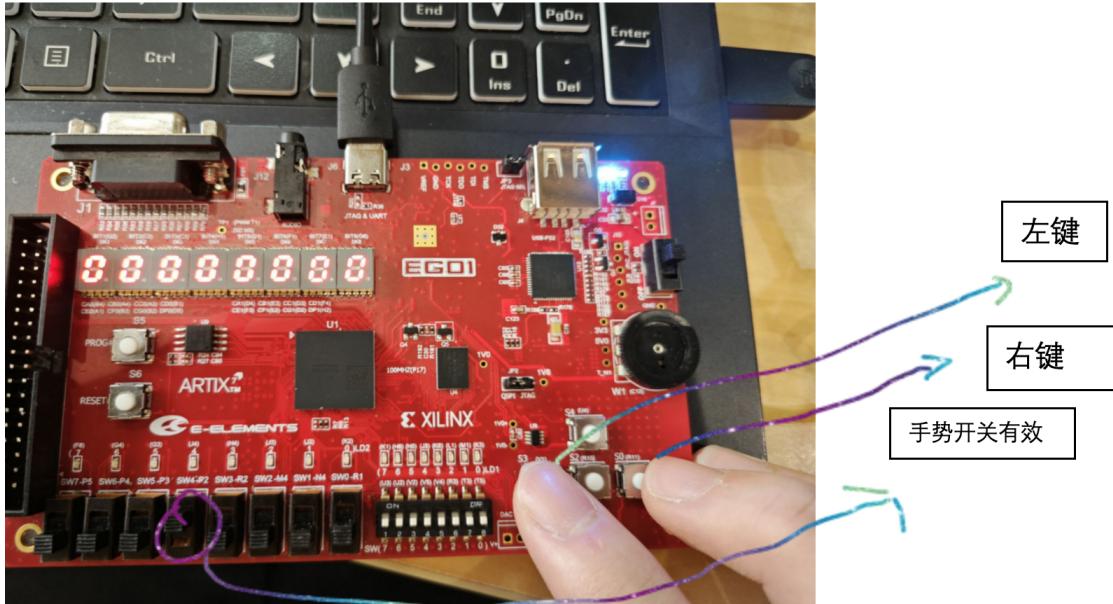
控制蜂鸣器输出的模块，根据传入的mode不同，播放声音

输入输出:

```
input clk,
input wire[4:0] mode,
output reg[0:0] music = 0
```

## 4.bonus 的实现说明

### 4.1 用按键操作模拟手势开关实现开关机功能



实现效果举例：

先按下左键，五秒内按下右键（R2开启即手势开关有效）

先按下左键，五秒内按下右键（R2关闭即手势开关无效）

## 4.2 蓝牙输入、输出

### 4.2.1 通过蓝牙，微信小程序显示当前状态：

首先微信小程序打开“HCBLE串口助手”，

再在待机状态下拨动如图所示拨码开关

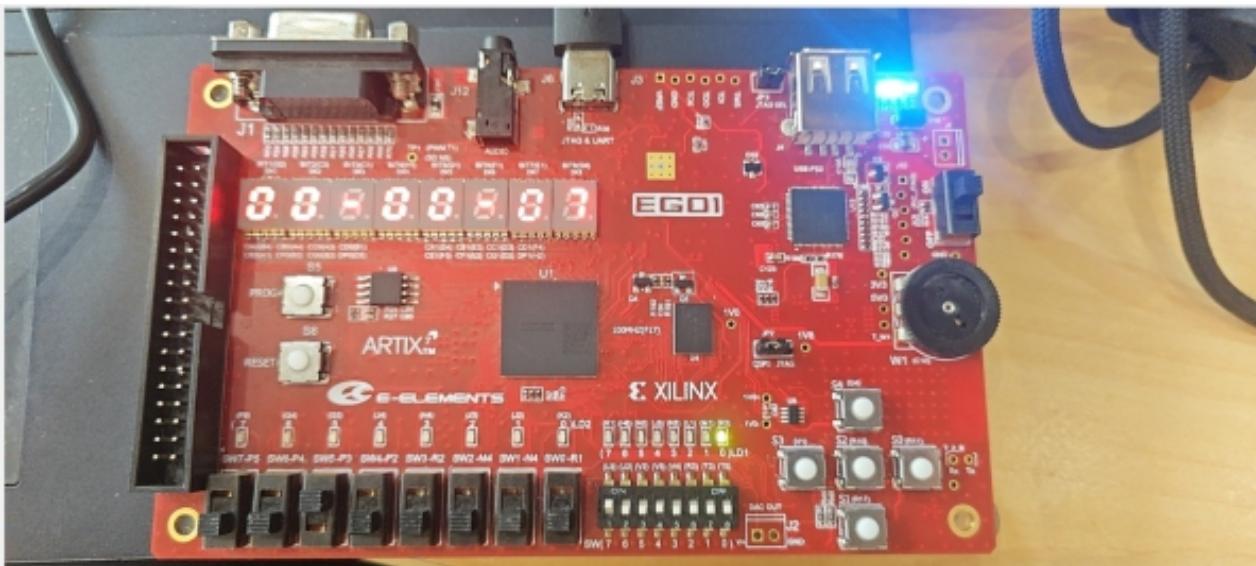
接着在微信小程序输入/DD，点发送

微信小程序接受状态（如图显示待机状态）

### 4.2.2 使用拨码开关、按键开关以外的输入设备（方便用户操作）

我们同样打开蓝牙，重复上述操作后，拨下P3（设置手势开关时间），按规则发送指令：/S 1 hh-mm-ss 设置智能提醒时间

例如如图（发送了/S 1 00-00-07）



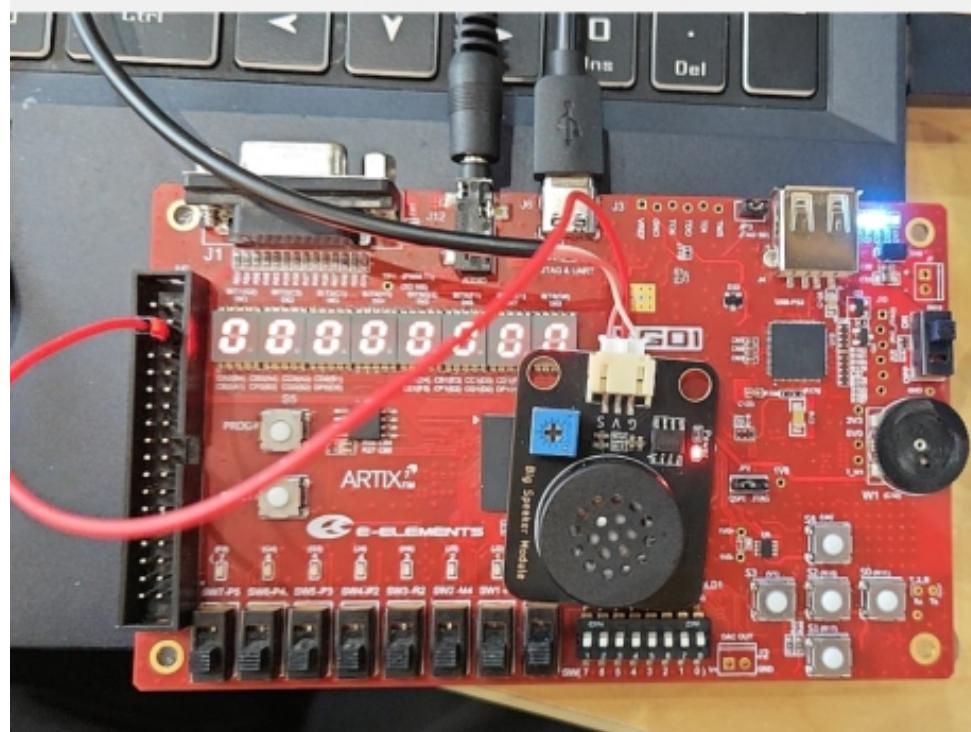
#### 4.2.3 原理：

Uart\_rx和Uart\_tx模块负责与串行设备进行数据传输，外部设备传入的指令被逐字符接收，根据ascii码表转为对应的2进制流。Cmd\_parse模块负责解析通过通信接口接收到的 ASCII 字符命令，提取命令参数，根据命令设置硬件状态或发送响应。resp\_gen模块负责对命令生成响应，并将其发送到字符 FIFO。最后字符由uart\_tx模块发送至终端。其中模块整体框架借鉴自互联网，本项目中对Cmd\_parse和resp\_gen模块进行修改，重新编写了如何对传入及传出数据进行处理的部分。

### 4.3 蜂鸣器提示音

#### 4.3.1 使用说明：

蜂鸣器提示（1, 2, 3档各有提示音）如图所示进行连线，即可实现效果。



#### 4.3.2 原理：

Audio模块根据输入的模式，选择不同的音符序列并控制生成的音符频率（对应音调）以及节奏，使用状态机控制音符的切换和播放逻辑，并在模式变化时重置当前播放状态。其中，音符的频率由参数指定，表示每个音符所需的时钟周期数。各个模式对应的音符序列通过 melody 参数硬编码，通过时钟周期计数器控制音符输出的频率和持续时间。蜂鸣器模块由本项目独立完成。

## 5.项目总结（团队合作、开发及测试工作的总结

### 5.1 参数化

替换那些复数至关重要，特别是在需要设置大量“数据”（如手势时间）时，参数化在团队合作中尤为重要。

### 5.2 阻塞赋值与非阻塞赋值澄清

方面	阻塞赋值 (=)	非阻塞赋值 (<=)
执行顺序	顺序执行	并发执行
模拟更新	立即更新	延迟到时间步结束时更新
用例	组合逻辑	时序逻辑

组合逻辑使用阻塞赋值 (=)。

时序逻辑使用非阻塞赋值 (<=)，以避免竞态条件。

### 5.3 多驱动器注意事项

注意多驱动器问题，并重视设计中的黄色警告，因为这些可能指示潜在问题。

### 5.4 重置处理的一致性

注意团队成员之间在处理重置（posedge 或 negedge）时的一致性，以避免冲突并确保正确同步。

### 5.5 去抖动的重要性

去抖动至关重要，特别是在开关实现中，以减轻噪声并确保可靠的输入处理。（否则一次按下可能被多次检测）

#### 1. 测试时需谨慎

在测试时，我们应该小心可能的用例。例如，当我们测试蓝牙部分时，我们只尝试了包含十位数字的命令，如10-50-10。因此，当我们向老师检查时，未能通过所有测试。

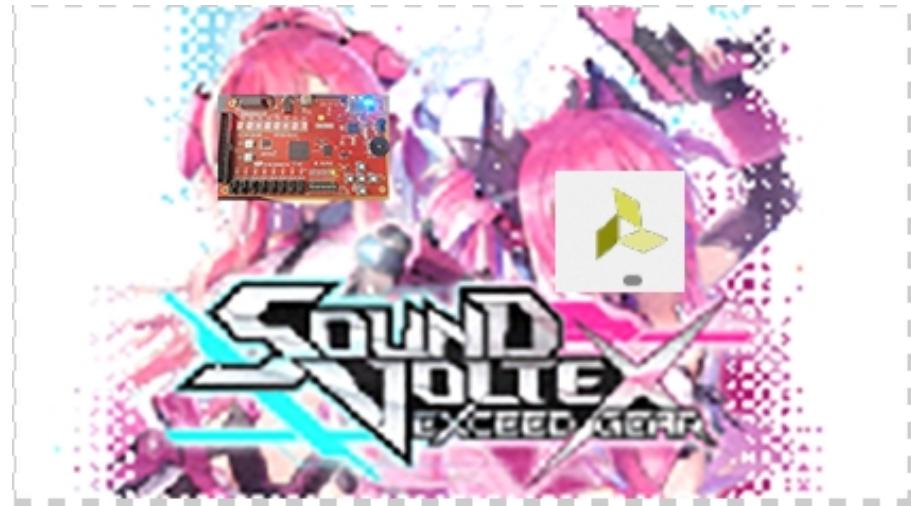
#### 2. 为每个部分设定合理的时间表和截止日期

虽然我们很早就对这个目标有了清晰的认识。但我们并没有为每个成员的任务设定明确的时间表。此外，我们可能在截止日期前过于匆忙，因此未能完美地完成这个项目。

## 6.假如你们团队负责project出题，请给出基于ego1可实现的project的想法和建议

### 6.1 原型介绍

低配版SOUND VOLTEX



**SOUND VOLTEX** (中文：音律炫动；简称：SDVX、ボルテ（）(VOLTE)) 是[科乐美](#)于2012年1月18日开始运营的[BEMANI](#)系列音乐游戏。[SOUND VOLTEX - 萌娘百科 万物皆可萌的百科全书](#)

游戏机样板

用户游玩，有四个按键和两个旋钮

进入游戏选择歌曲游玩

什么，玩不到最新最热？我们一起来用神奇的FPGA自己制作吧！

### 6.2 EGO1实现设计

#### 6.2.1 输入设备设计：

SDVX原版使用四个按钮和两个旋钮。EGO1板上有5个按键和8个拨码开关，可用于模拟这些输入设备。建议：

- 按钮输入：使用EGO1的5个按键中的4个，分别对应SDVX的四个按钮。
- 旋钮输入： EGO1的模拟电压输入（通过电位器）可用来模拟一个旋钮的功能。由于只有一个模拟输入，可考虑通过切换功能或增加外部硬件来实现第二个旋钮的输入。或者16个拨码开关模拟两个旋钮的旋转角度。
- 数码管用于显示游戏模式，进入歌曲游玩后显示得分

#### 6.2.2 显示输出：

EGO1配备了VGA接口，可用于连接显示器。建议：

- 图形显示：利用VGA接口显示游戏界面，包括下落的音符和判定线。由于FPGA资源有限，图形界面应尽量简化，例如使用基本的几何图形表示音符。

#### 6.2.3 用户进入待机模式，游玩模式

我们通过密码让用户可以进入游戏，即按下游戏机开关后游戏机进入待机模式，待机模式后通过拨动正确的左边拨码开关模拟游戏机密码，拨动正确后晶体管显示进入游戏模式，用户可以用其它四个按键选择歌曲。进入选择模式后左边八个拨码开关失效，右边的拨码开关选择游戏模式。注：用户在之后任何模式下都可以选择长按开关机键进行关机。

#### 6.2.4 用户可以选择听歌模式和游玩模式

我们可以通过右边的拨码开关选择听歌模式和游玩模式，

听歌模式：拨码开关显示LISTEN，此时用户选择歌曲，歌曲播放完一遍后循环播放，每当用户按下选歌按键后歌单立刻向下滚动并从新的被选择歌曲开始播放。

游玩模式：用户先选择歌曲，再通过右侧某个拨码开关进入这个歌曲的游玩模式。

#### 6.2.5 音符生成与显示逻辑

游戏逻辑：

- 按时间触发音符生成事件，根据音符数据调整其位置。
- 玩家输入与音符判定区域重叠时，检测输入时间差，反馈并更新分数，在晶体管显示。

具体实现：

- 16个Led灯分为两组，一组显示谱面（谱面组）。另一组是玩家输入（输入组）。
- 在谱面组led亮起时，玩家需要按下对应按键使输入组的对应led亮起
- 根据对应led亮起时间差大小可分perfect和good等等级，加上不同分数。如果时间差过大则判定为miss，不加分。
- 一局游戏结束后可根据总分显示不同等级如 S, A, B, C 等

注，最终实现中应至少包含一首歌曲的一部分，应该至少可以游玩