浙江大学

电 子 电 路 系 统 综 合 实 验
实 验 报 告

课程名称： 电子电路系统综合实验

组　　员：

专　　业： 电子科学与技术

实验名称： 基于 PWM 实现景观灯控制器的设计与研究

组　　号： 4 号

学　　号：

指导教师： 楼东武 陈鹏飞 崔宁

2022 年 7 月 8 日

目录

1 实验目的和要求

1.1 实验目的

（1）了解 LED 的特性，学习并掌握 LED 的驱动方式。
（2）学习并掌握 PWM 的原理，探索研究用 PWM 方式实现 LED 亮度调节的方式。
（3）学习探索单片机输出 3 路 PWM，驱动 RGB LED 的具体实现方式。
（4）学习利用蓝牙模块，将三色灯与手机进行通信，并利用手机控制三色灯。
（5）学习利用光敏电阻和 AD 采样来实现三色灯的自动开启和关闭。
（6）学习并设置故障检测模块的实现方式，并将结果通过光电信号呈现出来。

1.2 实验要求

（1）完成系统功能设计和电路设计。
（2）完成原理图绘制和 PCB 版图绘制。
（3）通过软件编程对单片机进行控制并实现设计功能。
（4）安装调试后能够实现相应的功能，并撰写最终实验报告。

2 实验原理

2.1 硬件方面

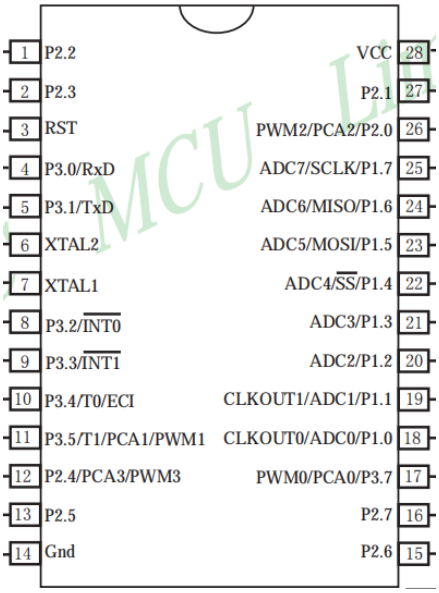
2.1.1 集成电路 STC12C5616AD 介绍

（1）主要性能

- 高速：1 个时钟/机器周期，增强型 8051 内核，速度比普通 8051 快 8～12 倍
- 宽电压：5.5～3.3V
- 增加外部掉电检测电路，可在掉电时，及时将数据保存进 EEPROM,正常工作时无需操作 EEP
- 工作频率：0～35MHz，相当于普通 8051：0～420MHz
- 时钟：外部晶体或内部 RC 振荡器可选，在 ISP 下载编程用户程序时设置
- 1280 字节片内 RAM 数据存储器
- 芯片内 EEPROM 功能,擦写次数 10 万次以上
- ISP / IAP，在系统可编程/在应用可编程,无需编程器/仿真器
- 8 通道,10 位高速 ADC，速度可达 25 万次/秒,2 路 PWM 还可当 2 路 D/A 使用
- 2 通道捕获/比较单元（PWM/PCA/CCP），也可用来实现 2 个定时器或 2 个外部中断
- 硬件看门狗
- 高速 SPI 串行通信端口
- 全双工异步串行口，兼容普通 8051 的串口
- 先进的指令集结构，兼容普通 8051 指令集，有硬件乘法/除法指令

- 通用 I/O 口，复位后为： 准双向口/弱上拉可设置成四种模式：准双向口/弱上拉，推挽/强上拉，仅为输入/高阻，开漏每个 I/O 口驱动能力均可达到 20mA，但整个芯片最大不得超过 100mA

（2）STC12C5616AD 的引脚图



2.1.2 MC34063 芯片介绍

（1）结构组成

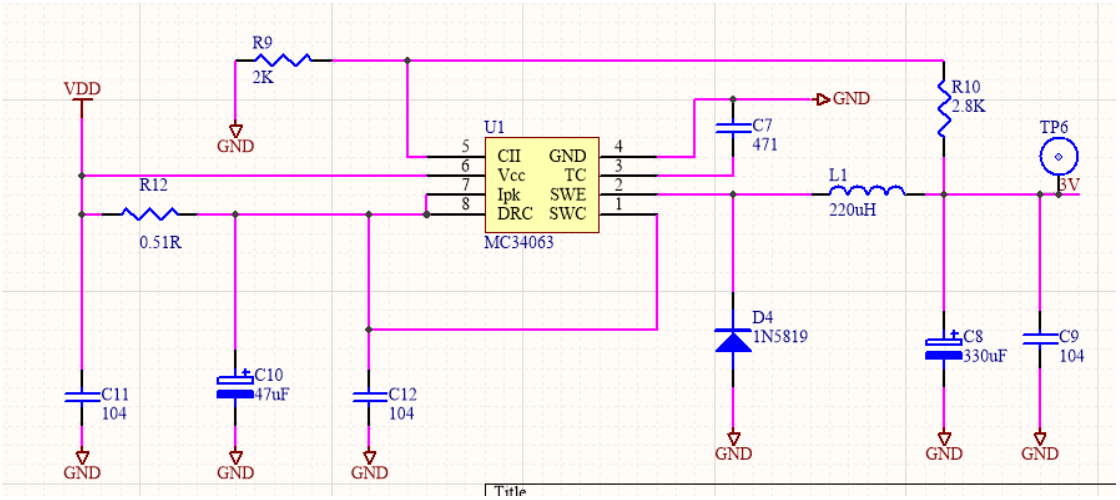
MC34063 是一种开关型高效 DC/DC 变换集成电路，该器件本身包含了 DC/DC 变换器所需要的主要功能的单片控制电路且价格便宜。其内设置有大电流的电源开关，34063 能够控制的开关电流达到 1.5A；它的内部含有具有温度补偿的基准电压源、比较器、具有限电流电路的占空比可控的振荡器、R—S 触发器和大电流输出开关管等。参考电压源是温度补偿的带隙基准源，振荡器的振荡频率有 3 脚的外接定时电容决定；开关晶体管由比较器反向输入端与振荡器相连的逻辑控制线路置成 ON，并由与振荡器输出同步的下一个脉冲设置成 OFF。该电路是在低静态电流典型的降压电路上，用开关变压器取代自感线圈实现的。利用开关变压器以获取隔离直流电源的能量供给。

（2）工作原理

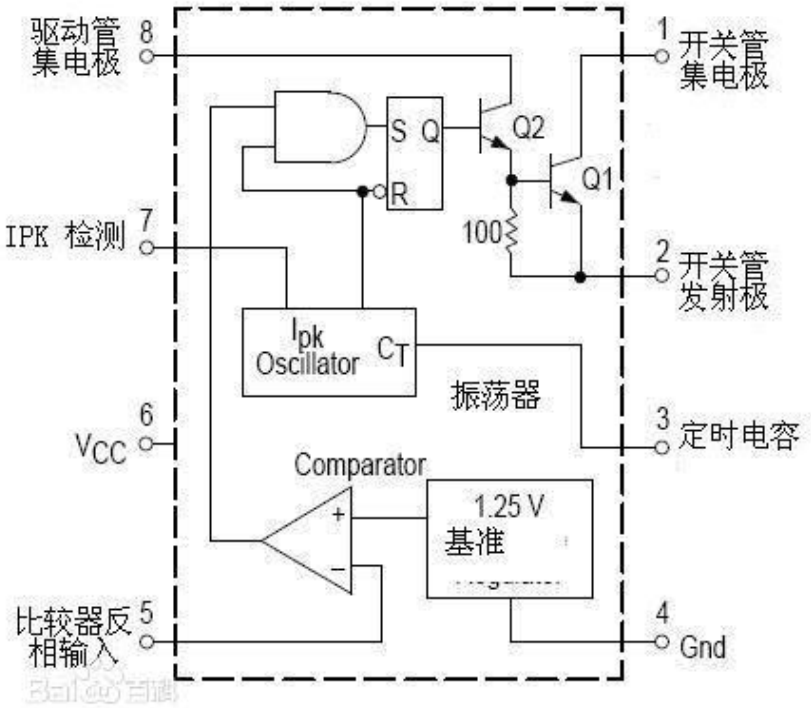
降压电路工作原理：

- 比较器的反相输入端(脚 5)通过外接分压电阻 R9、R10 监视输出电压 。其中，输出电压 Uo=1.25(1+ R10/R9)由公式可知输出电压 。仅与 R9、R10 数值有关，因 1.25V 为基准电压，恒定不变。若 R9、R10 阻值稳定，Uo 亦稳定。
- 脚 5 电压与内部基准电压 1.25V 同时送入内部比较器进行电压比较。当脚 5 的电压值低于内部基准电压(1.25V)时，比较器输出为跳变电压，开启 R—S 触发器的 S 脚控制门，R—S 触发器在内部振荡器的驱动下，Q 端为“1”状态(高电平)，驱动管 T2 导通，开关管 T1 亦导通，使输入电压 Ui 向输出滤波器电容 Co 充电以提高 U。，达到自动控制 U。稳定的作用。

- 当脚 5 的电压值高于内部基准电压(1.25V)时，R—S 触发器的 S 脚控制门被封锁，Q 端为“0”状态(低电平)，T2 截止，T1 亦截止。
- 振荡器的 Ipk 输入(脚 7)用于监视开关管 T1 的峰值电流，以控制振荡器的脉冲输出到 R—S 触发器的 Q 端。
- 脚 3 外接振荡器所需要的定时电容 CT 电容值的大小决定振荡器频率的高低，亦决定开关管 T1 的通断时间。



（3）内部结构图



MC34063 内部结构图

（4）MC34063 的引脚：

1 脚：开关管 T1 集电极引出端；

2 脚：开关管 T1 发射极引出端；

3 脚：定时电容 CT 的接线端，调节电容 Ct 的电容值可以使工作频率在 100—100KHz 之间变化；

4 脚：GND：

5 脚：电压比较器反相输入端同时也是电压输出取样端，使外接电阻精度不低于 1%的精度电阻；

6 脚：Vcc；

7 脚：负载峰值电流取样端，6.7 脚之间的电压超过 300mV 时芯片启动内部过流保护电路，起到过流保护的作用；

8 脚：驱动管 T2 的集电极引出端。

（5）电路主要应用

MC34063 大电流降压变化器电路、大电流升压变换器电路；

MC34063 反向变换器电路；

MC34063 降压变化器电路、升压变换器电路。

2.2 软件方面

2.2.1 I/O 口各种不同的工作模式及配置介绍

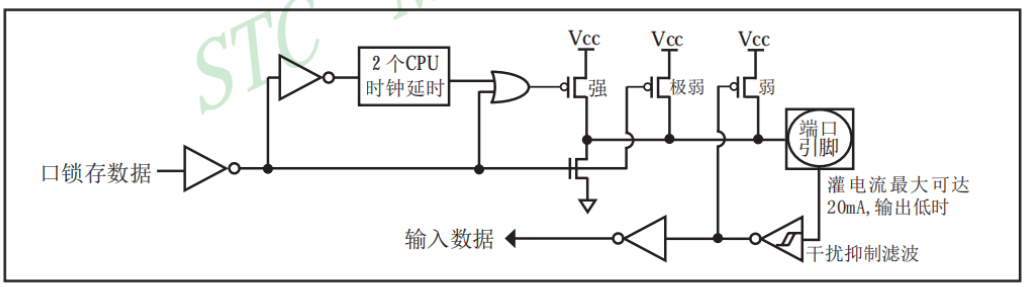
（1）四种模式介绍

单片机的 P1-P3 各个端口均有如下四种工作模式：

P1M0 [7：0]	P1M1 [7：0]	I/O 口模式（P1.x 如做A/D使用，需先将其设置成开漏或高阻输入）
0	0	准双向口(传统8051 I/O 口模式，弱上拉）， 灌电流可达20mA，拉电流为230μA， 由于制造误差， 实际为250uA～ 150uA
0	1	推挽输出（ 强上拉输出， 可达20mA, 要加限流电阻）
1	0	高阻输入（电流既不能流入也不能流出），如果该I/O口需作为A/D使用，可选此模式
1	1	开漏(Open Drain)，如果该I/O口需作为A/D使用，可选此模式

1）准双向口（弱上拉）输出

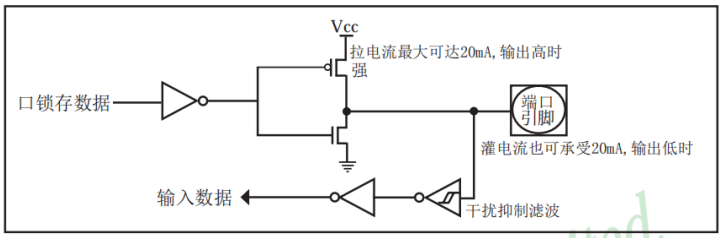
准双向口输出类型可用作输出和输入功能而不需重新配置端口输出状态。这是因为当端口输出为 1 时驱动能力很弱，允许外部装置将其拉低。当引脚输出为低时，它的驱动能力很强可吸收相当大的电流。准双向口有 3 个上拉晶体管适应不同的需要。在 3 个上拉晶体管中，有 1 个上拉晶体管称为“弱上拉”，第 2 个上拉晶体管，称为“极弱上拉”，第 3 个上拉晶体管称为“强上拉”。



准双口输出

2）强推挽输出

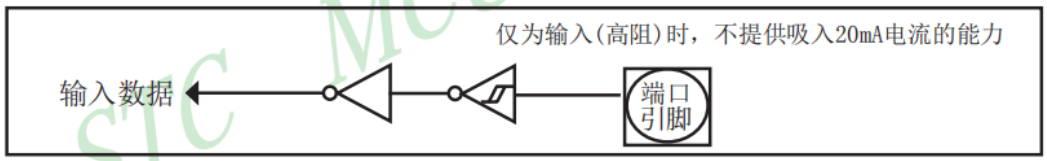
强推挽输出配置的下拉结构与开漏输出以及准双向口的下拉结构相同，但当锁存器为 1 时提供持续的强上拉。推挽模式一般用于需要更大驱动电流的情况。



强推挽输出

3）高阻输入

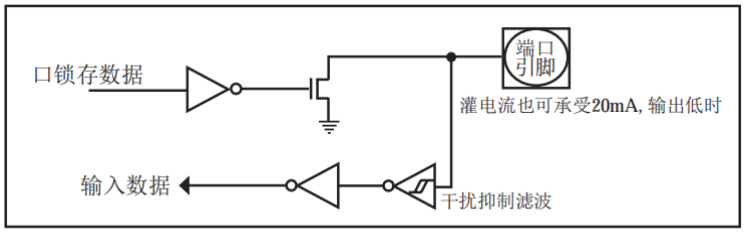
输入口带有一个施密特触发输入以及一个干扰抑制电路。



仅为输入(高阻)模式

4）开漏输出

当端口锁存器为 0 时，开漏输出关闭所有上拉晶体管。当作为一个逻辑输出时，这种配置方式必须有外部上拉，一般通过电阻外接到 Vcc。如果外部有上拉电阻，开漏的 I/0 口还可读外部状态，即此时被配置为开漏模式的 I/0 口还可作为输入 I/0 口。这种方式的下拉与准双向口相同。开漏端口带有一个施密特触发输入以及一个干扰抑制电路。



开漏输出(如外部有上拉电阻，也可读)

（2）模式配置

下图为用到的对采样口的寄存地址的配置，用 str P1M0 = 0x91 和 str P1M1 = 0x92 来完成地址配置，接下去则是用 P1M0 = 0x48 和 P1M1 = 0x4F 来设置 P1 口的输出模式。

P1M0 register (不可位寻址)

SFR name	Address	bit	B7	B6	B5	B4	B3	B2	B1	B0
P1M0	91H	name	P1M0.7	P1M0.6	P1M0.5	P1M0.4	P1M0.3	P1M0.2	P1M0.1	P1M0.0

P1M1 register (不可位寻址)

SFR name	Address	bit	B7	B6	B5	B4	B3	B2	B1	B0
P1M1	92H	name	P1M1.7	P1M1.6	P1M1.5	P1M1.4	P1M1.3	P1M1.2	P1M1.1	P1M1.0

2.2.2 定时器设置

（1）相关寄存器

符号	描述	地址	位地址及其符号 MSB LSB								复位值
TCON	定时器控制寄存器	88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	0000 0000B
TMOD	定时器模式寄存器	89H	GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0	0000 0000B
TL0	Timer Low 0	8AH									0000 0000B
TL1	Timer Low 1	8BH									0000 0000B
TH0	Timer High 0	8CH									0000 0000B
TH1	Timer High 1	8DH									0000 0000B
AUXR	辅助寄存器	8EH	T0x12	T1x12	UART_M0x6	EADCI	ESPI	ELVDI	-	-	0000 00xxB
WAKE_CLKO	时钟输出和掉电唤醒寄存器	8FH	PCAWAKEUP	RXD_PIN_IE	T1_PIN_IE	T0_PIN_IE	-	-	T1CLKO	T0CLKO	0000 xx00B

其中，比较重要的符号有：

1）TF1/TF0：定时器/计数器 T1/T0 的溢出标志。

2）TR1/TR0：定时器 T1 的运行控制位。TR1=1 时就允许 T1 开始计数，TR1=0 时禁止 T1 计数。当 GATE (TMOD.7)=1，TR1=1 且 INT1 输入高电平时，才允许 T1 计数。

3）GATE：TMOD.7 控制定时器 1,置 1 时只有在 INT1 脚为高及 TR1 控制位置 1 时才可打开定时器/计数器 1；TMOD.3 同理。

4）M1、M0：定时器/计数器 1/0 模式选择。

5)T0x12/T1x12：定时器 0/1 速度控制位。

（2）中断设置

1）中断触发：定时器 0/1 溢出。

2）与定时器中断相关的寄存器

SFR name	Address	bit	B7	B6	B5	B4	B3	B2	B1	B0
IE	A8H	name	EA	EPCA_LVD	EADC_SPI	ES	ET1	EX1	ET0	EX0

EA：CPU 的总中断允许控制位，EA=1，CPU 开放中断，EA=0，CPU 屏蔽所有的中断申请。

ET1/ET0：定时/计数器 T1/T0 的溢出中断允许位。ET1=1，允许 T1 中断；ET1=0，禁止 T1 中断。ET0 同理。

3）C 语言编程和查询次序

```
void Timer0_Rountine(void) interrupt 1;
void Timer1_Rountine(void) interrupt 3;
```

2.2.3 串口通信设置

相关寄存器

符号	描述	地址	位地址及符号 MSB LSB								复位值
AUXR	Auxiliary register	8EH	T0x12	T1x12	UART_M0x6	EADCI	ESPI	ELVDI	-	-	0000 00xxB
SCON	Serial Control	98H	SM0/FE	SM1	SM2	REN	TB8	RB8	TI	RI	0000 0000B
SBUF	Serial Buffer	99H									xxxx xxxxB
PCON	Power Control	87H	SMOD	SMOD0	LVDF	POF	GF1	GF0	PD	IDL	0011 0000B
IE	Interrupt Enable	A8H	EA	EPCA_LVD	EADC_SPI	ES	ET1	EX1	ET0	EX0	0000 0000B
IP	Interrupt Priority Low	B8H	-	PPCA_LVD	PADC_LVD	PS	PT1	PX1	PT0	PX0	x000 0000B
IPH	Interrupt Priority High	B7H	-	PPCA_LVDH	PADC_LVDH	PSH	PT1H	PX1H	PT0H	PX0H	x000 0000B

其中，比较重要的符号有：

1）REN：允许/禁止串行接收控制位。由软件置位 REN，即 REN=1 为允许串行接收状态，可启动串行接收器 RxD，开始接收信息。软件复位 REN，即 REN=0，则禁止接收。

2）TI：发送中断请求标志位。在方式 0，当串行发送数据第 8 位结束时，由内部硬件自动置位，即 TI=1，向主机请求中断，响应中断后 TI 必须用软件清零，即 TI=0。在其他方式中，则在停止位开始发送时由内部硬件置位，即 TI=1，响应中断后 TI 必须用软件清零。

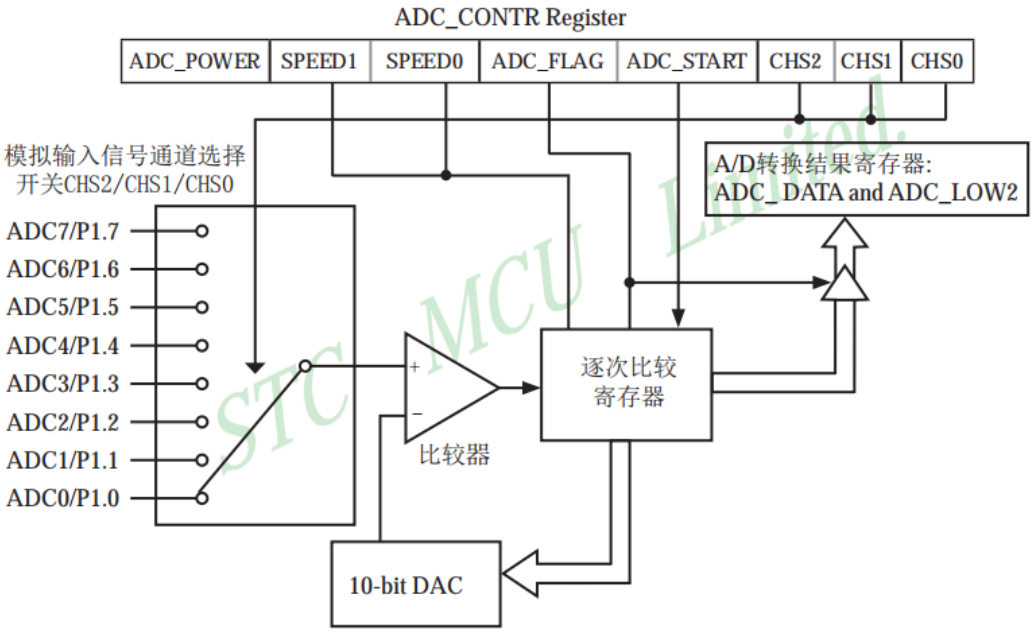
3）RI：接收中断请求标志位。在方式 0，当串行接收到第 8 位结束时由内部硬件自动置位 RI=1，向主机请求中断，响应中断后 RI 必须用软件清零，即 RI=0。在其他方式中，串行接收到停止位的中间时刻由内部硬件置位，即 RI=1，响应中断后 RI 必须由软件清零。

4）ES：串行口中断允许位，ES=1，允许串行口中断，ES=0，禁止串行口中断。

5）SBUF（串行口数据缓冲寄存器）：串行口缓冲寄存器的地址是 99H，实际是 2 个缓冲器，写 SBUF 的操作完成待发送数据的加载，读 SBUF 的操作可获得已接收到的数据。两个操作分别对应两个不同的寄存器，1 个是只写寄存器，1 个是只读寄存器。

2.2.4 A/D 转换器设置

（1）单片机 ADC(A/D 转换器)的结构如下图所示



（2）ADC 控制寄存器 ADC_CONTR

ADC_CONTR：ADC控制寄存器

SFR name	Address	bit	B7	B6	B5	B4	B3	B2	B1	B0
ADC_CONTR	C5H	name	ADC_POWER	SPEED1	SPEED0	ADC_FLAG	ADC_START	CHS2	CHS1	CHS0

ADC_POWER：ADC 电源控制位。

0：关闭 转换器电源；

1：打开 转换器电源。

启动 A/D 转换前一定要确保打开 A/D 电源。初次打开内部 A/D 转换模拟电源，需适当延时，等内部模拟电源稳定后，再启动 AD 转换。

SPEED1，SPEED0：模数转换器转换速度控制位

SPEED1	SPEED0	A/D转换所需时间
1	1	90个时钟周期转换一次，CPU工作频率27MHz时，A/D转换速度约300KHz（=27MHz÷90）
1	0	540个时钟周期转换一次
0	1	810个时钟周期转换一次
0	0	1080个时钟周期转换一次

ADC_FLAG：模数转换器转换结束标志位，当 A/D 转换完成后，ADC_FLAG=1,要由软件清 0。不管是 A/D 转换完成后由该位申请产生中断，还是由软件查询该标志位 A/D 转换是否结束，当 A/D 转换完成后，ADC FLAG=1，一定要软件清 0。

ADC_START：模数转换器转换启动控制位，设置为“1”时，开始转换，转换结束后为 0。

CHS2/CHS1/CHS0：模拟输入通道选择，CHS2/CHS1/CHS0

CHS2	CHS1	CHS0	Analog Channel Select（模拟输入通道选择）
0	0	0	选择 P1.0 作为A/D输入来用
0	0	1	选择 P1.1 作为A/D输入来用
0	1	0	选择 P1.2 作为A/D输入来用
0	1	1	选择 P1.3 作为A/D输入来用
1	0	0	选择 P1.4 作为A/D输入来用
1	0	1	选择 P1.5 作为A/D输入来用
1	1	0	选择 P1.6 作为A/D输入来用
1	1	1	选择 P1.7 作为A/D输入来用

（3）A/D 转换结果寄存器 ADC_DATA、ADC_LOW2

Mnemonic	Add	Name	B7	B6	B5	B4	B3	B2	B1	B0
ADC_DATA	C6h	A/D转换结果寄存器，全部8位有效，为10位A/D转换结果的高8位	-	-	-	-	-	-	-	-
ADC_LOW2	BEh	A/D转换结果寄存器，只有低2位有效，为10位A/D转换结果的低2位	x	x	x	x	x	x	-	-

得到完整的 ADC 转换结果

$$res:(ADC\_DATA[7:0],ADC\_LOW2[1:0])=1024\times\frac{Vin}{Vcc}$$

2.2.5 LCD1602 设置

（1）LCD1602 引脚功能表

编号	符号	引脚说明	标号	符号	引脚说明
1	VSS	电源地	9	D2|	数据
2	VDD	电源正极	10	D3	数据
3	VL	液晶显示偏压	11	D4	数据
4	RS	数据/命令选择	12	D5	数据
5	R/W	读/写选择	13	D6	数据
6	E	使能信号	14	D7	数据
7	D0	数据	15	BLA	背光源正极
8	D1	数据	16	BLK	背光源负极

各引脚功能如下：

引脚 1：VSS 为电源地

引脚 2：VDD 接 5V 正电源

引脚 3：VL 为液晶显示屏对比度调整端，接正电源时对比度最弱，接地时对比度最高，对比度过高时会产生“鬼影”现象，使用时可以通过一个 10kQ 的电位器调整其对比度。

引脚 4：RS 为寄存器选择脚，高电平时选择数据寄存器、低电平时选择指令寄存器。

引脚 5：R/W 为读/写信号线，高电平时进行读操作，低电平时进行写操作。当 RS 和 R/W 共同为低电平时可以写入指令或显示地址；当 RS 为低电平，R/W 为高电平时，可以读忙信号；当 RS 为高电平，R/W 为低电平时，可以写入数据。

引脚 6：E 端为使能端，当 E 端由高电平跳变为低电平时，液晶模块执行命令。

引脚 7~14：D0~D7 为 8 位双向数据线。

引脚 15：背光源正极。

引脚 16：背光源负极。

（2）LCD1602 指令集

序号	指令	RS	R/W	D7	D6	D5	D4	D3	D2	D1	D0
1	清屏	0	0	0	0	0	0	0	0	0	1
2	光标复位	0	0	0	0	0	0	0	0	1	x
3	输入方式设置	0	0	0	0	0	0	0	1	I/D	S
4	显示开关控制	0	0	0	0	0	0	1	D	C	B
5	光标或字符移位控制	0	0	0	0	0	1	S/C	R/L	x	x
6	功能设置	0	0	0	0	1	DL	N	F	x	x
7	字符发生存储器地址设置	0	0	0	1	字符发生存储器地址					
8	数据存储器地址设置	0	0	1	显示数据存储器地址						
9	读忙标志或地址	0	1	BF	计数器地址						
10	写入数据至CGRAM或DDRAM	1	0	要写入的数据内容							
11	从CGRAM或DDRAM中读取数据	1	1	读取的数据内容							

指令 1：清屏。指令码 01H，光标复位到地址 00H。

指令 2：光标复位。光标复位到地址 00H。

指令 3：输入方式设置。其中，I/D 表示光标的移动方向，高电平右移，低电平左移；S 表示显示屏上所有文字是否左移或右移，高电平表示有效，低电平表示无效。

指令 4：显示开关控制。其中，D 用于控制整体显示的开与关，高电平表示开显示，低电平表示关显示；C 用于控制光标的开与关，高电平表示有光标，低电平表示无光标；B 用于控制光标是否闪烁，高电平闪烁，低电平不闪烁。

指令 5：光标或字符移位控制。其中，S/C 表示在高电平时移动显示的文字，低电平时移动光标。

指令 6：功能设置命令。其中，DL 表示在高电平时为 8 位总线，低电平时为 4 位总线；N 表示在低电平时为单行显示，高电平时双行显示；F 表示在低电平时显示 5×7 的点阵字符，高电平时显示 5×10 的点阵字符。

指令 7：字符发生器 RAM 地址设置。
指令 8：DDRAM 地址设置。
指令 9：读忙信号和光标地址。其中，BF 为忙标志位，高电平表示忙，此时模块不能接收命令或数据，如果为低电平则表示不忙。
指令 10：写数据。
指令 11：读数据。

3 功能设计

3.1 主要功能

根据实验目的和要求，我们设计了如下五个主要功能：

- RGB 三色调色、调亮度
- 使用手机定时开关控制 LED 灯
- 根据环境光开关 RGB 景观灯
- 通过蓝牙，使用手机自定义调光
- 系统故障时能够主动报警

3.2 功能细化

3.2.1 RGB 三色调色、调亮度

（1）使单片机输出 3 路 PWM 信号，通过分别改变占空比来控制 RGB 三色的亮度。
（2）不同亮度的三原色混合即可实现不同的颜色的调色，同时改变三色的亮度即可改变三色灯的整体亮度。
（3）可以实现呼吸灯，即灯光由灭到亮，再由亮到灭，并同时以一个灭亮灭的时间周期进行 8 种颜色的循环。
（4）可以在 LCD 屏上显示当前的 RGB 三色的 PWM 占空比。

3.2.2 使用手机定时开关控制 LED 灯

配置蓝牙通信，在手机上可以修改、设定景观灯的开启时间和关闭时间，并在 LCD 上显示相应的倒计时时间。

3.2.3 根据环境光改变 LED 灯亮度

利用光敏电阻，设计电路分压使得光线暗时，LED 灯亮；光线亮时，LED 熄灭，并在 LCD 上显示当前光敏分压电阻的采样电压。

3.2.4 通过蓝牙，使用手机自定义调光

配置蓝牙通信，在手机上可以自定义景观灯的颜色、亮度，可以依次对三种三原色进行 PWM 占空比的调节，从而进行亮度调节，实现自由调节想要的颜色。

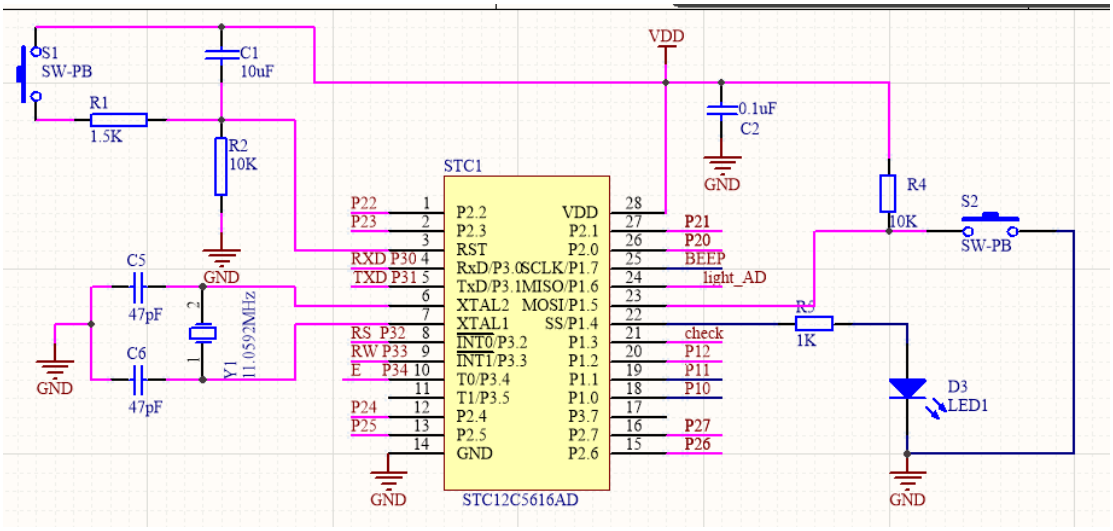
3.2.5 系统故障时能够主动报警

监测采样电阻的电压，当上电后三色灯任意一个 LED 灯发生故障导致断路时，系统会发出声光警报，且在 LCD 屏上显示“！”予以警示。

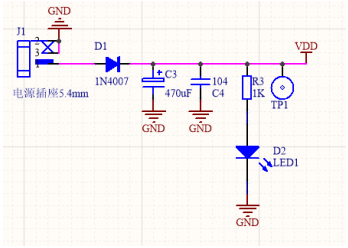
4 电路设计

4.1 单片机模块

为了启用复位键，我们在 reset 处连接了一个按键，并设计了相关的电路，其中与按键的限流电阻并联的电容值为 10μF，分压电阻则为 10KΩ。6、7 脚接的是晶振，来产生一个固定的时间周期。考虑到需要用到 AD 采样来进行故障检测采样和光敏分压电阻的采样，我们将 AD 采样的 P1.0-P1.7 这 8 个通道进行了合理的分配，在保留了测试用的按钮和指示灯（在 23 脚则是连了一个按键，22 脚则是接了一个发光二极管作为指示灯）外，将 25 脚选做蜂鸣器的控制端，利用 24、21 脚的 AD 采样功能来对所需电压进行一个采样。



4.2 电源模块

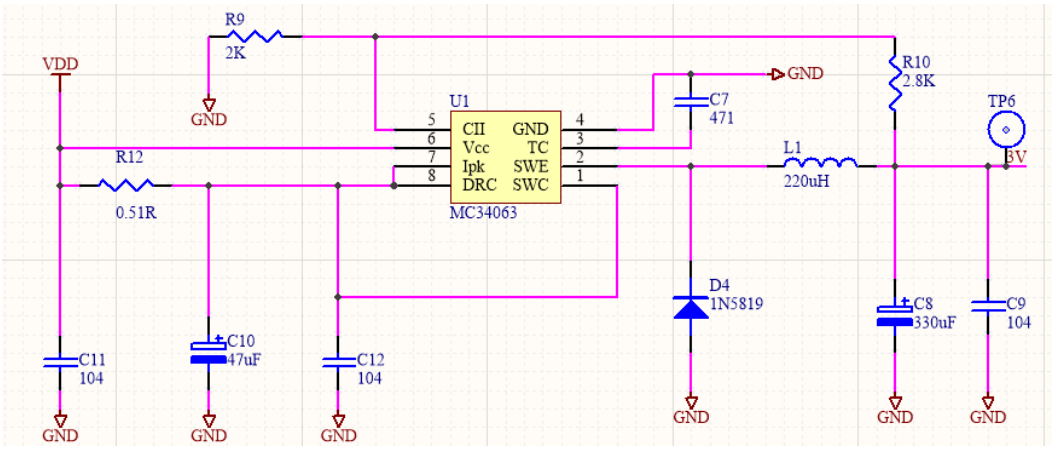


我们在原有的电路基础上增加了一个 D2 发光二极管的工作指示灯，观察是否上电成功。

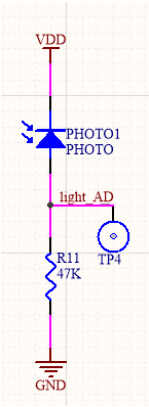
4.3 5V-3V 的 DC-DC 转换模块

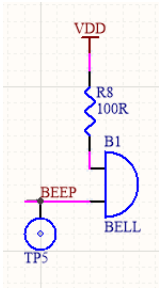
此次设计的主要设计要求是设计用 MC34063 芯片制作降压型变换电源，将 5V 输入电压转换成 3V 的输出电压，我们需根据设计要求对相关参数进行调整。

参考老师提供的 12V-5V 的 DC-DC 转换电路，根据电压计算公式 Uo=1.25(1+R10/R9)，我们可以算得 R10 等于 2.8KΩ，R9 等于 2KΩ，而 2.8KΩ则可由两个 5.6KΩ的电阻并联得到。



4.4 光敏电阻模块





光敏电阻的分压电阻：

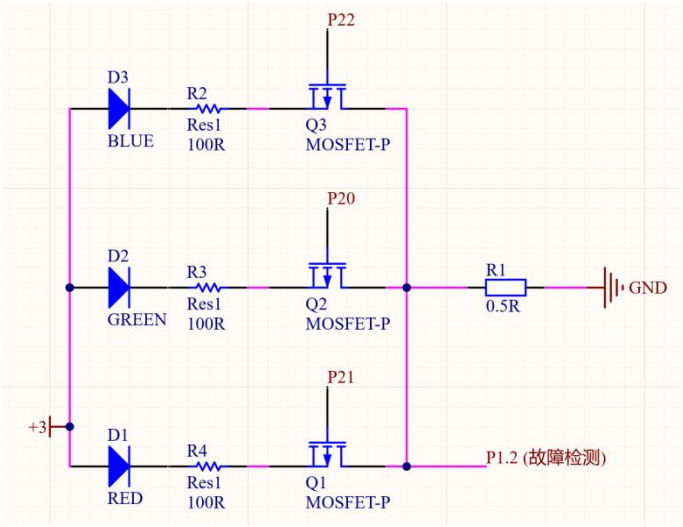
通过面包板，我们采用遮光调试的方法，检测出光敏电阻和额定电阻分压处的电压波动幅度大小。为了能够在单片机 AD 检测端能够较明显的区分光照强度，我们采用 47KΩ 的电阻，因为此时全遮光和不遮光的电压变化幅度大。

4.5 蜂鸣器模块

通过查阅资料，选择了合适的限流电阻 100Ω 来保护蜂鸣器。

4.6 三色灯模块

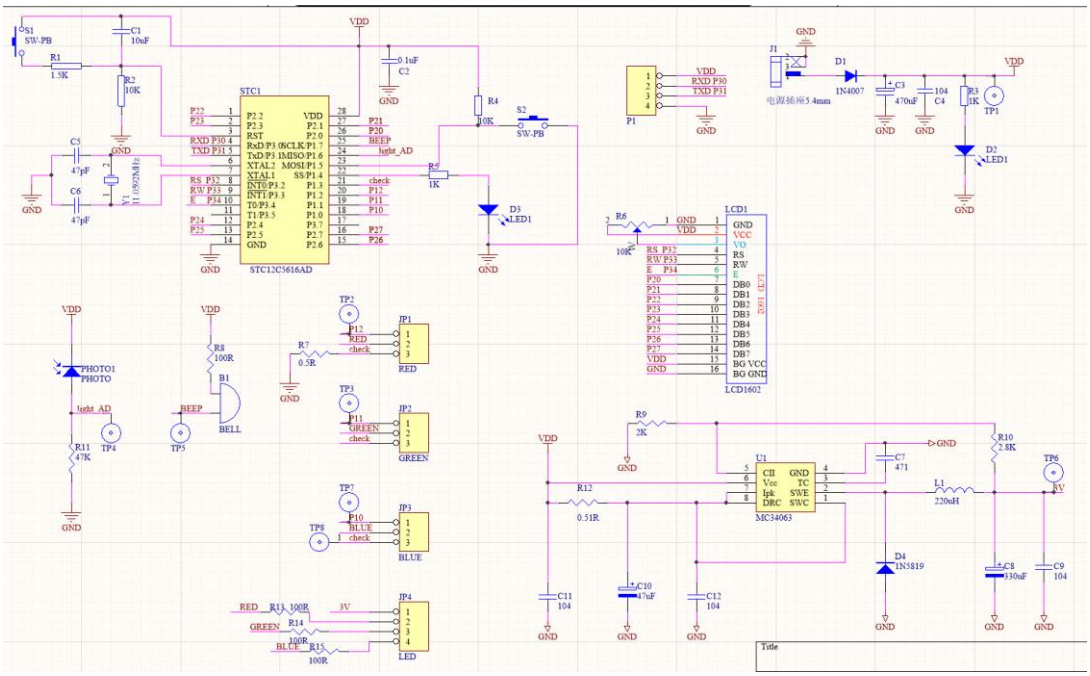
通过限流电阻来保护 LED 灯，并通过 CMOS 管来实现 LED 的控制，R1 作为采样，可分路来判断该分路的 LED 灯是否有断路现象，若有问题则反馈给单片机。为了满足单片机的精度，我们保证采样电压在正常工作时有 15-20mV，故选择 R1 为 0.5Ω 。



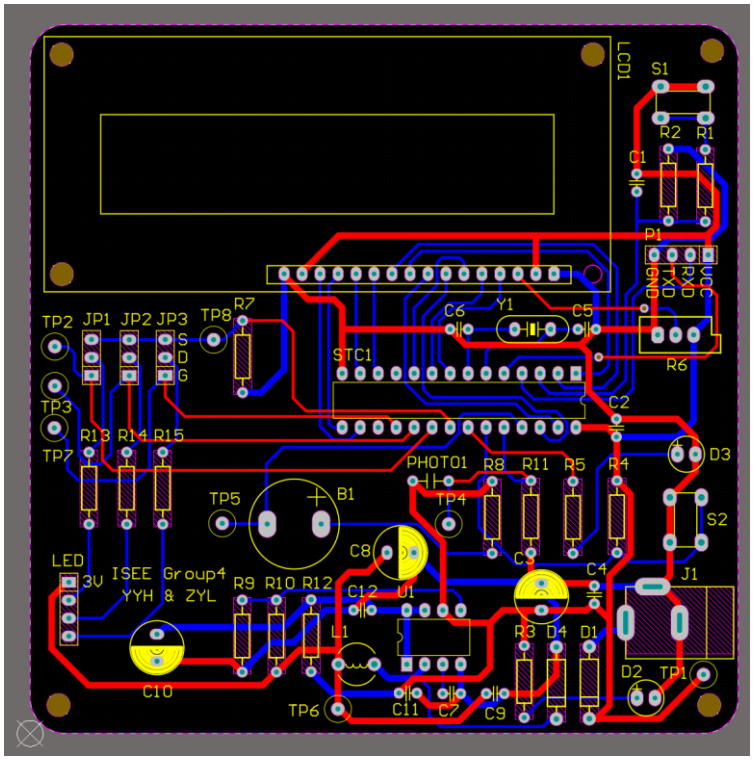
5 项目设计结果

5.1 硬件设计

5.1.1 原理图

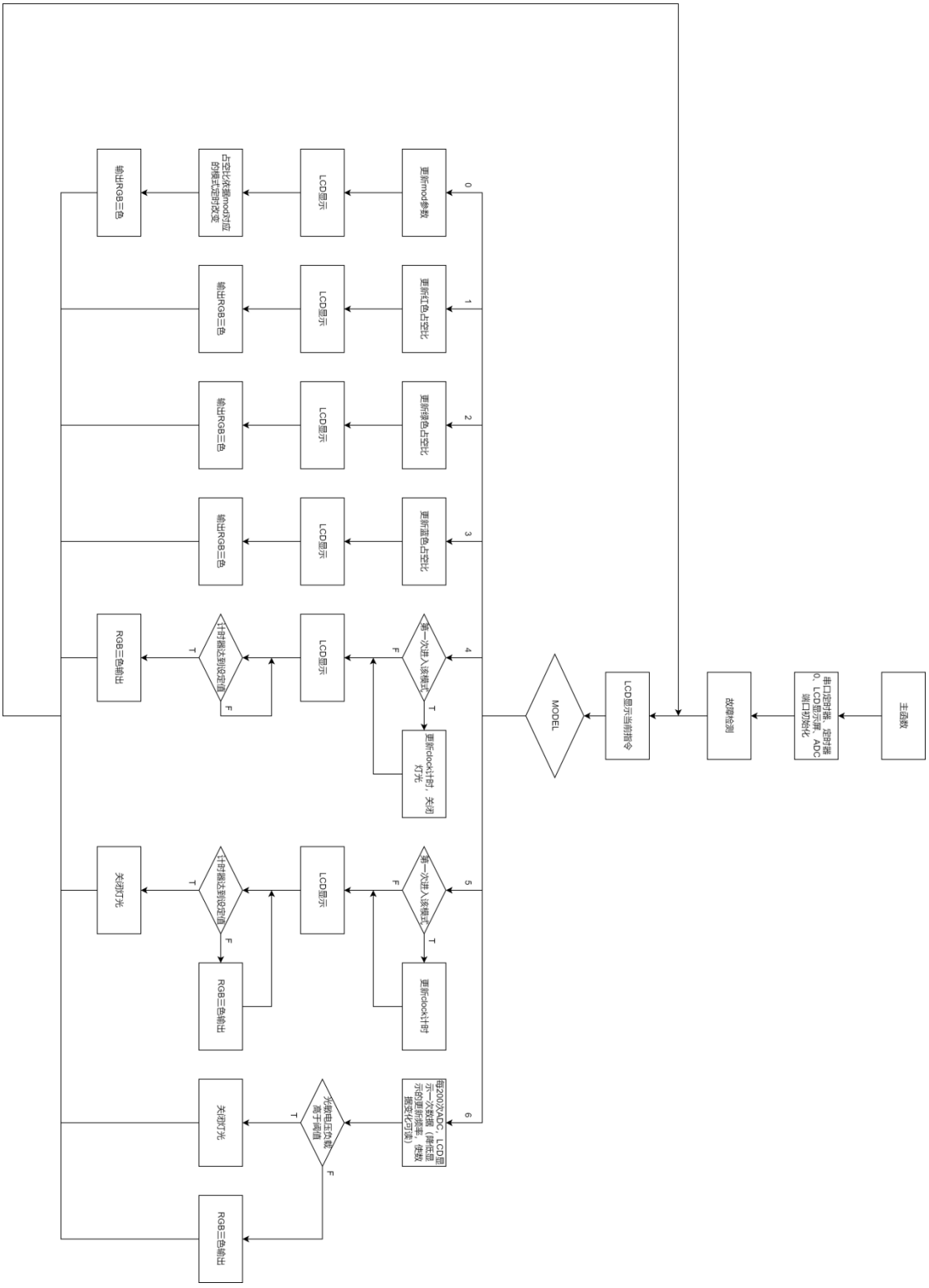


5.1.2 PCB

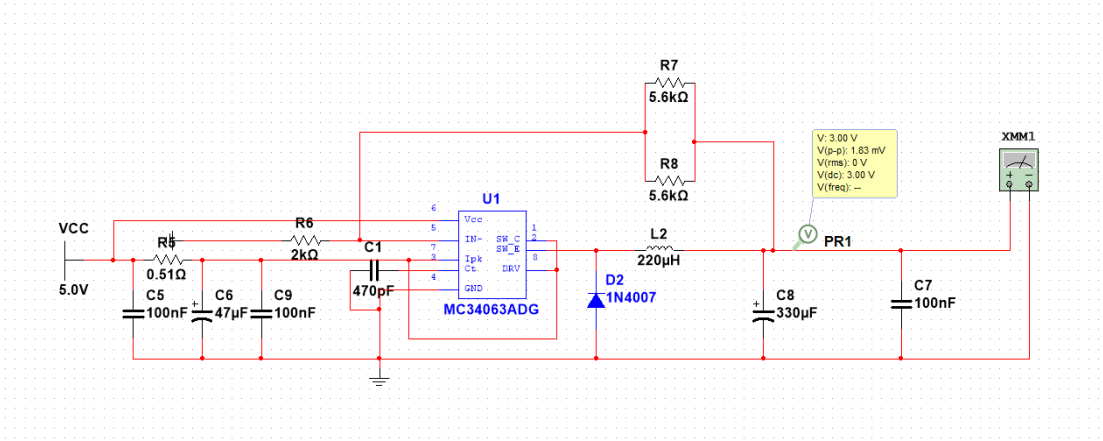


5.2 软件设计

整体功能实现流程如下图所示：



6 电路仿真



此为 DC-DC 降压电路部分的仿真，通过改变 R6、R7、R8 的电阻阻值，可以调整电路的输出电压值。经仿真可得，此电路完成了 5V 到 3V 的降压功能。

7 实验调试

7.1 硬件电路组装调试

7.1.1 原单片机电路下的调试与电路设计

（1）问题 1：红灯过亮，三色的亮度不匹配

起初，给三色灯的三路 LED 的输入端都置 1 时，我们发现三色灯并没有呈现理想的白色，且颜色过于刺眼；而当切换成 R、G、B 三种纯色灯时，又可以明显感受到三色之间的亮度差异，其中红色亮度最大，蓝色亮度最小。故我们给三者分别加了 100Ω 的限流电阻，此时在面包板和原单片机的电路上三色灯亮度合适。

（2）问题 2：红灯会有明显的延迟，且在没有置 1 时亮起

原因：单片机 I/O 口驱动方式不合适。在我们的调试过程中，会发现红色不同于另外两种颜色，有着明显的延迟，且在置 0 时也会受驱动亮起，因而会打乱呼吸灯的亮度和颜色的呈现。通过查询数据手册，我们发现单片机的 I/O 口输出存在着四种模式，如下表所示。其中，推挽电路，主要作用是增强高电平驱动能力，为外部设备提供大电流，同时提供真正的低电平(0V)，增强其低电平驱动能力，避免以小电压驱动红灯；但推挽输出的一个缺点是，如果当两个推挽输出结构相连在一起，一个输出高电平，同时另一个输出低电平时，整个通路上电阻很小，会发生短路，进而可能造成端口的损害，所以推挽输出不能实现“线与”；推旋输出是用两个晶体管或者场效应管构成的推挽电路，这个电路的特点就是输出电阻小，所以能够驱动大的负载，从而能够使得单片机管脚直接驱动发光二极管、蜂鸣器甚至更小阻抗的负载。故我们对 LED 灯的单片机控制口进行了强推挽设置，在所需的 I/O 口将对应的 P1M0 和 P1M1 的对应位分别改为 0 和 1。

P1口设定〈P1.7，P1.6，P1.5，P1.4，P1.3，P1.2，P1.1，P1.0口〉(P1口地址：90H)

P1M0 [7：0]	P1M1 [7：0]	I/O 口模式（P1.x 如做A/D使用，需先将其设置成开漏或高阻输入）
0	0	准双向口(传统8051 I/O 口模式，弱上拉）， 灌电流可达20mA，拉电流为230μA， 由于制造误差，实际为250uA～ 150uA
0	1	推挽输出（强上拉输出，可达20mA, 要加限流电阻）
1	0	高阻输入（电流既不能流入也不能流出），如果该I/O口需作为A/D使用，可选此模式
1	1	开漏(Open Drain)，如果该I/O口需作为A/D使用，可选此模式

7.1.2 自己设计的电路调试和修改

（1）问题 1：故障采样电阻的 AD 采样为 0

在原有的电路设计中，我们通过搭建面包板调试出了合适的限流电阻 100Ω，且选择采样电阻时采用了计算的理论值 0.5Ω，该电阻实际过小，导致采样电阻上的分压达不到 AD 采样的精度。最终，通过测试不同的电阻搭配，降低限流电阻以提高单路电流，提高采样电阻以提高电阻分压。最后 R、G、B 三路采用不同的限流电阻(100Ω、50Ω、0Ω)，采样电阻改为 6Ω，实现准确的采样判断，且在发生故障时能产生较大的变化用以判断。

（2）问题 2：三色灯亮度不匹配，三路 LED 灯的单路通路时采样电阻的电压值相差过大

由于在调试中发现三路的采样电压相差过大、蓝色亮度太弱等问题，我们依次对三路限流电阻进行短路、并联较小的电阻等操作，最终以亮度最弱的蓝灯为参考，将蓝灯的限流电阻短路，红灯保持 100Ω 的限流电阻，绿灯则是并联上 56Ω 的电阻，从而将三路单通可得到的采样电压保持在差不多的大小，即 6-8mV。并将代码的 if 判断条件从 data==0 改为 data<15，即通过 LCD 显示三路的通路采样电压后，选择合适的电压阈值来作为判断条件，最终故障检测模块可以顺利进行。

```
if(data1 < 15)
    {
        EA = 0;//关闭全局中断
        //蓝牙发送报警
        while(1)
        {
            LCD1602_ShowStr(15,1,"!");  //LCD 显示感叹号
            D3 = 1;                     //灯闪烁，蜂鸣器间歇性报警
            BEEP = 1;
            delay(40000);
            D3 = 0;
            BEEP = 0;
            delay(40000);
        }
```

（3）问题 3：LCD 屏不显示

在 LCD 屏的调试过程中，我们发现在原 LCD 屏上能够显示“!”的代码，在刚组装好的电路上的新 LCD 屏上无法显示出来。最终，我们通过旋转调节屏幕显示字体的亮度的电位器，来将字体亮度调节到肉眼清晰可见的合适大小。

7.2 软件代码编写调试

（1）问题 1：定时器 0、定时器 1、ADC 冲突：

1）软件编写、调试方法

最初，我们将所有原来分开可行的模块整体到一起，结果程序直接崩溃，通过利用指示灯 D3，我们发现原程序无法正常执行定时器的中断程序，故通过对照数据手册反复查看 TMOD 等的参数赋值是否正确等方式来试图进行调试，但调试无果。

最终，我们选择改变原来的代码组合方式。最初是写好各个模块，调试成功后进行组合，发现不同代码中对使能关键字的处理可能有重复、冲突，最后导致定时器的中断函数无法正常运行。后来改变代码的组合方式，先保证两个定时器能够正常运行，再将基础代码中的具体操作改成需要的功能模块，逐渐添加、填充。添加一个部分就调试一个部分，最终程序能够正常运行。

2）定时器统一时间戳

在一个定时器内使用同一个计数器变量控制不同时间间隔。起初，我们在只进行呼吸灯轮换的只有 PWM 模块的代码里将原来的两个定时器整合为一个定时器时，采用了两个 time_count0++和 time_count1++来满足不同的计时需求，程序运行没有问题。但是将其加入已经包含定时器 1（串口通信的定时器）的代码时，发现程序中断出错，在将 time_count 时戳整合成一个后，即以需要的最大的时间长度为计数器变量循环，更短的时间循环通过对这个计数器变量取模操作实现的方式来满足需求，程序就运转自如了。

```
void timer0routine() interrupt 1                    //1ms
{
        static uint time_count;
        TH0 = 0XFC;
        TL0 = 0X67;
        time_count = (time_count + 1) % 1000;
        if (time_count % 2) {                       //2ms
                pwm_out_flag = 1;
        }
        if(time_count % 10 == 9){                   //10ms
                pwm_renew_flag = 1;
        }
        if(time_count == 999){                      //1s
                clock_flag = 1 ;
        }
}
```

（2）问题 2：RBG 多颜色循环呼吸灯无法正常运行，只能单色快速闪烁

此问题发生在添加 LCD 模块时，检查代码猜测可能是在 LCD 程序中为了使显示稳定而添加的过长的延时导致的（每次调用函数延时 2ms，定时器中断为 1ms 发生一次，PWM 是 2ms 输出一次）。考虑到单片机本身运行速度较快，将 LCD 模块内的延时改为两个'_nop_();'，缩短为两个机器周期。测试结果显示 LCD 能够正常使用，同时呼吸灯也能顺利运行。

```
void LCD1602_write_cmd(unsigned char cmd)     //写命令函数
{
    LcdWaitReady();
    LCD1602_RS = 0;
    LCD1602_RW = 0;
    LCD1602_DB = cmd;
    LCD1602_E   = 1;
    _nop_();_nop_();
    LCD1602_E   = 0;
}
```

（3）问题 3：LCD 显示频闪严重

在操作后增加两个机器周期的延时可以大幅降低频闪，增加了一个稳定的时间；而且每次更新前都进行清屏操作也会导致 LCD 显示频繁频闪，故改为每个模式初始化时对 LCD 清屏，此操作也同样可以大幅降低频闪；光敏分压电阻电压的采样值原来每次 while 循环都会执行一次，刷新过快也会导致 LCD 显示频闪严重，故我们选择每采样 200 次进行一次采样值的 LCD 显示刷新，此操作亦可大幅降低频闪。

```
    LCD1602_ShowStr(0,0,"clock:");
    _nop_();_nop_();
```

（4）问题 4：呼吸灯在某一阶段会卡顿一段时间，再继续运行

通过在代码的不同部分添加发光二极管的亮或灭来测试输出，了解代码具体在哪个部分失效、卡顿。最后发现当灯光卡顿时，定时器中断的计数部分发生故障，进而导致控制灯光的部分无法运行，整个程序会停滞，直到定时器的计数器再次正常运行。在尝试将整个循环周期减小时发现可以解决问题，精准定位到问题所在。然后再 LCD 屏上显示每次的 time_count 值，发现它并不能在设定的 0-60000 之间循环，而是会远超其值。而发生这种情况的原因是我们以 1ms 中断产生一个 1min 的计数器时需要的计数器的最大值达到了 60000，超过 int 类型的数据范围，而将数据类型改成 long 或者 unsigned int 则可以解决问题。

（5）除上述诸多问题之外，我们在软件调试中还存在一些逻辑上的不周到和冲突问题，如倒计时显示过程中从二位数跳转到个位数时，原来的显示个位的位置会保持为 0（即 1 会呈现成 10），或者 mod9 状态的各色循环却只在反复呈现同一个颜色等琐碎的问题，因篇幅所限，故不在此一一赘述。

8 实验结果

8.1 硬件方面

在完成电路组装后，我们还在调试过程中对电路进行了修改，最终能够配合我们的代码合理呈现较为理想的实现效果。

8.2 软件方面

通过软件代码的撰写，我们最终实现了我们预设的几项功能：

（1）故障检测和声、光、屏显报警：监测采样电阻的电压，当上电后三色灯任意一个 LED 灯发生故障导致断路时，系统会发出声光警报，且在 LCD 屏上显示“！”予以警示；若没有断路问题存在，则进入 LCD 的显示指令和初始的 8 色轮换呼吸灯模式。

（2）RGB 三色调色、调亮度：在模式 1、2、3 可以对 R、G、B 三色分别进行占空比的设置，10%为一档，通过三色的亮度改变可以实现颜色和亮度的自定义。而且在模式 0，我们可以实现 8 色的纯色呼吸灯和 8 色轮换的呼吸灯，并且可以在 LCD 屏上显示当前的 RGB 三色的 PWM 占空比。（模式之间的转换通过蓝牙传输指令控制）

（3）使用手机定时开关控制 LED 灯：配置蓝牙通信，在手机上可以修改、设定景观灯的开启时间和关闭时间，并在 LCD 上显示相应的倒计时时间，其中模式 4 是定时开启，模式 5 是定时关闭。

（4）根据环境光开关 RGB 景观灯：模式 6 则是利用光敏电阻，通过电路分压控制三色灯在光线暗时灯亮，光线亮时灯灭，并在 LCD 上显示当前光敏分压电阻的采样电压。

8.3 与预期的对照

在功能方面完全实现了我们的预设功能，但是在电压转换方面会因为一些分压而偏离预设的转换后的 3V（略小），三色灯三路的采样电压并未调至一致而使部分颜色（如白色）不够纯净和符合，且在呼吸灯的最亮情况下会有一下闪动。整体来说，与预期成果相符度高。

9 心得体会和展望

个人总结

在本次短学期课程中，我收获良多。从系统板与面包板配合搭建初步电路、测试元器件参数开始，到自己绘制原理图、PCB 版图，最后编写程序调试，在短短 14 天里，我们实践了一个系统诞生的完整过程。

因为在之前的课程学习中已经接触过单片机，短学期开头的入门课程进行得非常顺利。之前对 keil 软件和单片机编程的基础使我们很快理解了选题的各部分功能，代码的思路也得以较快地形成一个大概。设计原理图时，我们参考老师给出的经典模型进行组合，并在面包板上做了简单的测试、调试，基本验证并实现了 PWM 控制 LED 灯的程序核心主体。绘制 PCB 时由于原理图上有一些小错误，反映到 PCB 板上会牵涉、干扰其他元器件。除此之外，在与老师的交流中，我更清楚地了解了一些“可以，但不够完美”的小缺陷，秉持着在能力

范围内做到最好的新年，我们选择重新绘制 PCB。在这反复修改中，我绘制 PCB 的技能得到了极大的锻炼，对功能区布局、连线的技巧等技能操作也更加熟练。同时，修改过程中我对此类错误的发生、纠正的过程也有了更深的理解，相信在之后的实践中能够有所进步。

之后的实验就开始有挫折。最初是电路中电阻的取值问题。我们焊接完成后发现，在最初选择某些电阻时只做了理论的仿真和计算，没有在实际电路中进行测量。最后 PCB 板载电源电压偏差、内部电路分压等多种原因导致故障检测部分的采样电阻取值不合适，于是我们重新焊接、飞线，再次调试硬件电路，最后使得我们的硬件不仅有良好的展示效果，而且和单片机、代码相匹配。

在软件调试部分，我吸取之前写代码的经验教训，在开始分工的时候就按照每个功能模块分别编写代码（如 ADC 模块，LCD 显示模块，PWM 控制模块等）。在分模块的同时也考虑了最后整体代码框架的组织和搭建，计划在各模块学习、调试，并最终实现后进行组合，预想其可以正常工作且易于分工。但实际进行时我们遇到了太多困难。最核心也最困难的就是定时器的问题。之前的编程经验使我们能够较快上手，但当初在样例程序上修改某些参数进行编写的方式使得我们对硬件控制部分的代码仅仅浅尝辄止，没有理解透彻。各个独立模块中使用了不同的定时器的控制方式，虽然它们各自能够正常工作，但以这种方式组合模块代码会导致单片机内部存在冲突。在并不稳定也不统一的代码基础中不断添加新的模块，最终导致我们的定时器中断函数整体失效，且由于代码体量过大，修改过程很可能会拆东墙，补西墙，不仅修改难以全面，而且几乎无法调试。不破不立，最后我们选择先调试好两个定时器，再将功能模块替换进去。每增加一部分功能就调试一部分，确保代码“地基”的正确稳定工作。在最后一天里，我们几乎从头开始重新写了一份近千行的代码，在这样难忘的过程里，我切实学到、学会了很多新的知识和技巧。最后的整体编写不仅解决了之前困扰我们一天多的问题，而且让我们对对方负责的模块也有了更深的了解，得以融会贯通，学到了更多的东西。经历了实验中期的绝望后，逐渐逼近正确结果时，我收获了极大的成就感和满足感。从逐渐学会看懂原先看不懂也难以静下心看懂的数据手册开始，切实地从全新文档中一步步搭建代码框架，并填充其血肉，我真切地体会到只有亲身实践才能对知识理解地更透彻。

每一个曾经的错误都会成为未来的基石。在这两周的课程实践中，我收获了很多，学到了很多，也将大学前一半的各种知识进行了总结与串联，使我对我的兴趣所在有了更清晰的了解，激发了我自主学习的动力。

个人总结

本次的短学期的实验课程的主要要求是让我们自己设计一个现实可行的项目，并对其进行电路设计、PCB 版图设计和功能的软件代码编写，是这几个学期所学知识点的一个综合运用。而在这次的实验课程中，实验课程内容不仅限于硬件部分，软件部分也有大量涉及。

在前半短学期课程中，我们着重于电路设计和 PCB 版图的绘制。首先，在确定好项目后，我和我的组员根据老师给出的要求，共同设定了我们想要达成的目标，也就是希望电路能够实现的功能。我们给出了 5 个主要功能，然后基于这 5 项功能划分了功能模块并设计相关的电路。其中，光敏电路、蜂鸣器电路和 LED 电路较易设计，但是 DC-DC 降压电路则由于涉及到了新的芯片而带来一些困难。通过查询资料和听取老师的对芯片内部电路的讲解，并参考 12V-5V 的降压电路，我们通过公式变换和基本要求，计算修改了相应的电阻阻值，

且电路仿真后表明结果正确。在将个模块连接在一起后，就形成了我们完整的电路原理图的设计。接下去，就是 PCB 版图绘制了。本次 PCB 版图绘制，我们将电路核心——单片机放置在 PCB 版图中央，其他模块则是在旁依次排开。其中值得注意的是，电源和外接接口应该放电路板边缘处，而 DC-DC 转换电路的电源输入输出口的连接线则是至少 40mil。比较不幸的是，在绘制过程中，我们发现 LCD 的输入数据段有被复用而会导致 LCD 失效，故重新修改了单片机的各个端口的分布。

到了后半课程，就是软件编写和电路调试阶段了，也就是最难的部分。由于之前有接触到相关的代码编写，所以在编写模块功能上，整体思路没有太大问题。但是因为之前是在既有代码上进行修改的，所以对于一些寄存器配置、定时器会冲突等硬件配置方面的问题了解不深，故在调试过程中出现了强烈的冲突和程序运行混乱。我们最初是分模块分别进行了代码编写，如 LCD 模块、AD 转换模块、PWM 输出和改变模块等等。经过查找资料和对数据手册的反复研读，我逐渐了解了每句代码的含义，在我主要负责的 LCD 模块和 PWM 输出模块等部分中，我从开始的研究别人的代码进行仿写到能够根据自己的想法增添功能代码。而在之前，我对于芯片等的了解主要是来自于老师的授课，而现在我也学会了阅读数据手册，进行更为严格的代码编写。

当然，在实验中我们也遇到了一些问题，比如采样电阻的理论计算值在实际运行中带来的分压不足的问题，对此我们进行了阻值调整；还有代码撰写中，PWM 的输出会受定时器等中断的冲突而停止运行，对此我们统一了时间戳；以及 AD、LCD、定时器等的冲突带来的一系列问题，最终我们通过一个个模式的叠加来进行逐次排查。

总之，这门课程富含探索性，能激发我们的探索热情。而且通过这门课，我重温了之前所学的电路知识，又一次利用 AD 软件画原理图、对元器件进行封装、布局布线等操作，对 PCB 版图的布线设计也是愈发的流畅；在巩固之前所学的 C 语言的基础上，也学会了阅读数据手册，别写硬件配置代码。新知识与旧知识的融合，再加上实践的方法，让我对这些知识的掌握更为深刻，对部分软件的使用更为熟练，同时动手能力也进一步提高了。至于展望，我认为通过这门课程，让我自己动手实现了一个小小的项目，让我学会了一些编写程序的基本操作和应该了解的知识点，同时进一步激发了我的创新和实践热情，对我未来的实践活动满怀期待。

10 小组分工

组员	组内分工	贡献比
	电路设计、开题 PPT、PCB 绘制、电路搭建、软硬件调试、代码撰写（整体框架搭建、AD 采样、蓝牙串口、定时器）、结题报告撰写	50%
	电路设计、开题报告、原理图绘制、电路搭建、软硬件调试、代码撰写（整体框架搭建、LCD 显示、PWM、定时器）、结题报告撰写	50%