目录

[1. 项目简介 1](#_Toc80730899)

[2. 总体设计 1](#_Toc80730900)

[2.1 硬件设计 1](#_Toc80730901)

[2.2 软件设计 2](#_Toc80730902)

[2.2.1 LCD12864显示模块 2](#_Toc80730903)

[2.2.2 按键扫描模块 2](#_Toc80730904)

[2.2.3 通信模块 2](#_Toc80730905)

[2.2.4 游戏效果模块 3](#_Toc80730906)

[3. 详细设计 3](#_Toc80730907)

[3.1 硬件设计 3](#_Toc80730908)

[3.1.1 主机的通信 3](#_Toc80730909)

[3.1.2 LCD1284的显示 4](#_Toc80730910)

[3.1.3 键盘输入扫描 4](#_Toc80730911)

[3.1.4 数码管和蜂鸣器 5](#_Toc80730912)

[3.2 软件设计 5](#_Toc80730913)

[3.2.1 游戏菜单 5](#_Toc80730914)

[3.2.2 游戏运行 6](#_Toc80730915)

[3.2.3 按键扫描 8](#_Toc80730916)

[3.2.4 双机通信 10](#_Toc80730917)

[3.2.5 EEPROM读写 11](#_Toc80730918)

[3.2.6 游戏效果 12](#_Toc80730919)

[4. 数据结构与数据处理 12](#_Toc80730920)

[4.1 LCD坐标 12](#_Toc80730921)

[4.2 防御塔坐标 13](#_Toc80730922)

[5. 亮点设计 14](#_Toc80730923)

[5.1 软件亮点 14](#_Toc80730924)

[5.2 硬件亮点 16](#_Toc80730925)

[6. 系统兼容性说明 16](#_Toc80730926)

[7. 总结与体会 16](#_Toc80730927)

# 项目简介

1. 项目概述：以51单片机为开发平台的放置类塔防游戏机，使用C语言编写程序，通过Proteus 8运行
2. 项目功能：用户在Proteus 8运行该项目，可以获得放置类塔防游戏体验。用户通过控制七个不同功能的按键，在LCD屏上进行放置类塔防游戏操作
3. 项目硬件构成：AT89C51单片机、LCD12864、24C02C、一位数码管、无源蜂鸣器、电容、不同阻值的电阻、排阻、整流二极管
4. 项目软件构成：基于Proteus 8的硬件仿真、使用Keil 5编译的C语言程序
5. 项目优点：具备市面上常见的放置类塔防游戏的基本功能，具有多野怪、多防御塔，多地图，不同的功能道具，并配备有游戏音乐和游戏暂停功能
6. 项目缺点：由于Proteus 8本身的不稳定性，项目运行过程中偶尔会出现LCD花屏、卡死的现象

# 总体设计

## 硬件设计

1. AT89C51单片机：使用两块AT89C51单片机组成该项目的硬件主体，其中一块单片机作为主机运行游戏，负责游戏界面、游戏信息的显示；另一块单片机作为遥控器，负责接受按键的输入，并把接收到的信息传给主机单片机，实现用户对游戏的操作。其中，两块单片机之间通过硬件UART进行异步串行通信
2. LCD12864：使用一块LCD12864显示模块，用于该项目的主要显示界面，包括游戏菜单、游戏运行界面、大部分游戏信息
3. EEPROM存储芯片24C02C：使用一块EEPROM存储芯片对用户的游戏信息进行断电保存，单片机与EEPROM之间通过I2C通信协议进行读写操作
4. 一位LED数码管：一块一位LED数码管用于当前游戏关卡得分的显示，在当前游戏关卡结束后，排行榜的游戏分数会进行更新
5. 无源蜂鸣器：一个无源蜂鸣器，在游戏开始、结束时分别播放不同的音乐以提示用户开始、结束游戏，给予用户更好的游戏体验
6. 十六引脚排阻：使用排阻拉主高机单片机P0 口的电平，将P0口作为普通IO口使用，输出电平信号控制LCD12864的显示
7. 电阻、电容、二极管、按键：通过合理的设计与布局，使单片机可以准确接收到七个不同按键发送的信号

## 软件设计

### LCD12864显示模块

1. LCD12864硬件配套显示函数
2. 开始菜单界面：在LCD上绘制开始游戏、排行榜、操作说明三个选项，并绘制出箭头光标，用户通过按键控制光标选中不同选项执行不同功能
3. 关卡选择界面：在LCD上绘制关卡一到六等六个关卡，并绘制出箭头光标，用户通过按键控制光标选中不同关卡开始游戏
4. 游戏运行界面：通过游戏动画函数在LCD上绘制野怪行进的动画，并绘制出十字光标，用户通过按键控制光标移动到游戏界面的不同位置，选中不同功能。进行防御塔的放置、移除和道具的使用，按键的响应通过遥控器和外部中断0的共同配合完成，这是游戏的主体部分
5. 排行榜界面：进入排行榜时，主机对EEPROM进行读取数据，并通过LCD显示出来
6. 操作说明界面：在LCD上绘制简要的游戏操作说明

### 按键扫描模块

#### 遥控器单片机

按键信号发送到遥控器，遥控器通过UART通信不断向主机发送其接收到的按键信号

#### 主机单片机

1. 菜单按键扫描：直接对收到的按键信号进行处理并做出响应
2. 游戏界面按键扫描：当接收到遥控器单片机发送来的信号时，触发外部中断0，打断正在运行的游戏动画，对按键信号进行处理并做出响应

### 通信模块

1. 主机和遥控器：使用硬件UART进行异步串行通信，遥控器接收按键的输入，并把按键信号发送给主机，主机在需要执行功能的时候接收遥控器发送来的按键信号，进行处理并做出响应
2. 主机和EEPROM：使用软件模拟I2C通信协议，单片机作主机，EEPROM作从机，在需要的时候由主机对EEPROM进行读写操作，用于断电保存用户的游戏信息

### 游戏效果模块

1. 游戏得分：设置单独的数码管显示模块，把在游戏进行的时候，把游戏得分实时传送到数码管显示模块，实时更新游戏得分并显示在数码管上
2. 游戏音乐：设置单独的蜂鸣器音乐模块，通过51单片机的定时器1模拟PWM波输出，在游戏开始和结束的时候分别播放不同的音乐，提升用户的游戏体验

# 详细设计

## 硬件设计

### 主机的通信

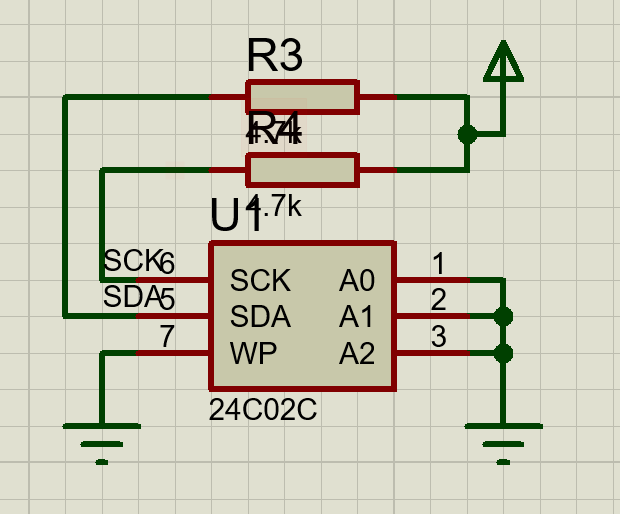
1. 主机和遥控器的通信：使用硬件UART进行异步串行通信，在硬件上直接将两个单片机对应引脚（RXD、TXD）连接，具体配置在软件实现
2. 主机和EEPROM的通信：任意选取主机的两个普通IO口分别作为串行数据线SDA和串行时钟线SCL，将主机和24C02C对应SDA、SCK引脚连接，并对24C02C另外的四个引脚进行接线，确定24C02C的从机地址。然后使用软件模拟IIC进行主从通信，实现主机对24C02C的数据读写

图 1 24C02C的接线

### LCD1284的显示

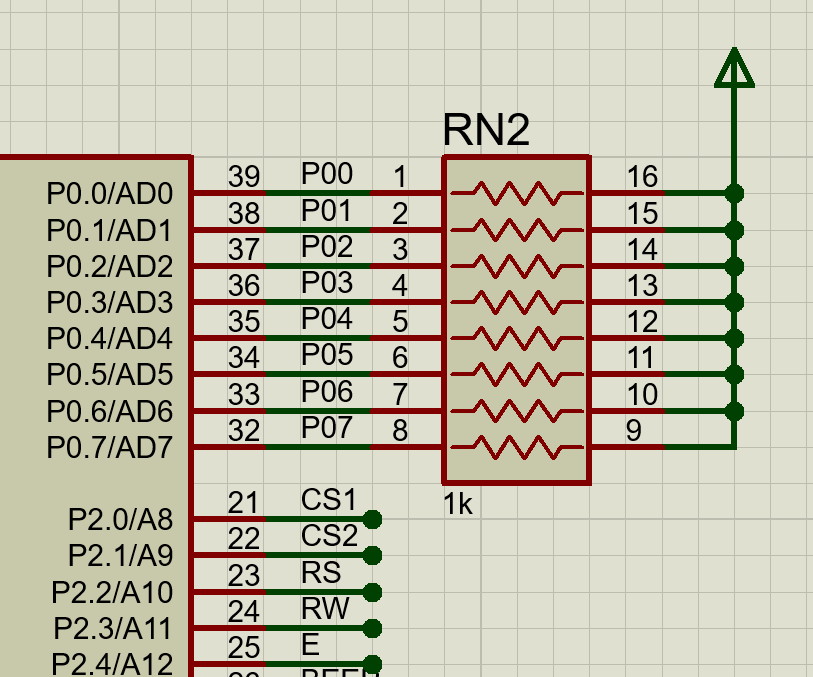
使用主机的P0口和P2口控制LCD12864的显示，其中P0口作为普通IO口使用时需要接上拉电阻，阻值为1k

图 2 控制LCD12864显示的IO口接线

### 键盘输入扫描

由于程序在软件上使用了主机的外部中断0来响应按键的输入，以完成对游戏的操作，所以在按键输入信号给遥控器P1口接收的同时，要引出一条线接上主机的P3.2口（外部中断0）。并且通过合理的电路设计，使得该线的数据传输不受到按键输入信号的影响且不会影响到按键的输入信号

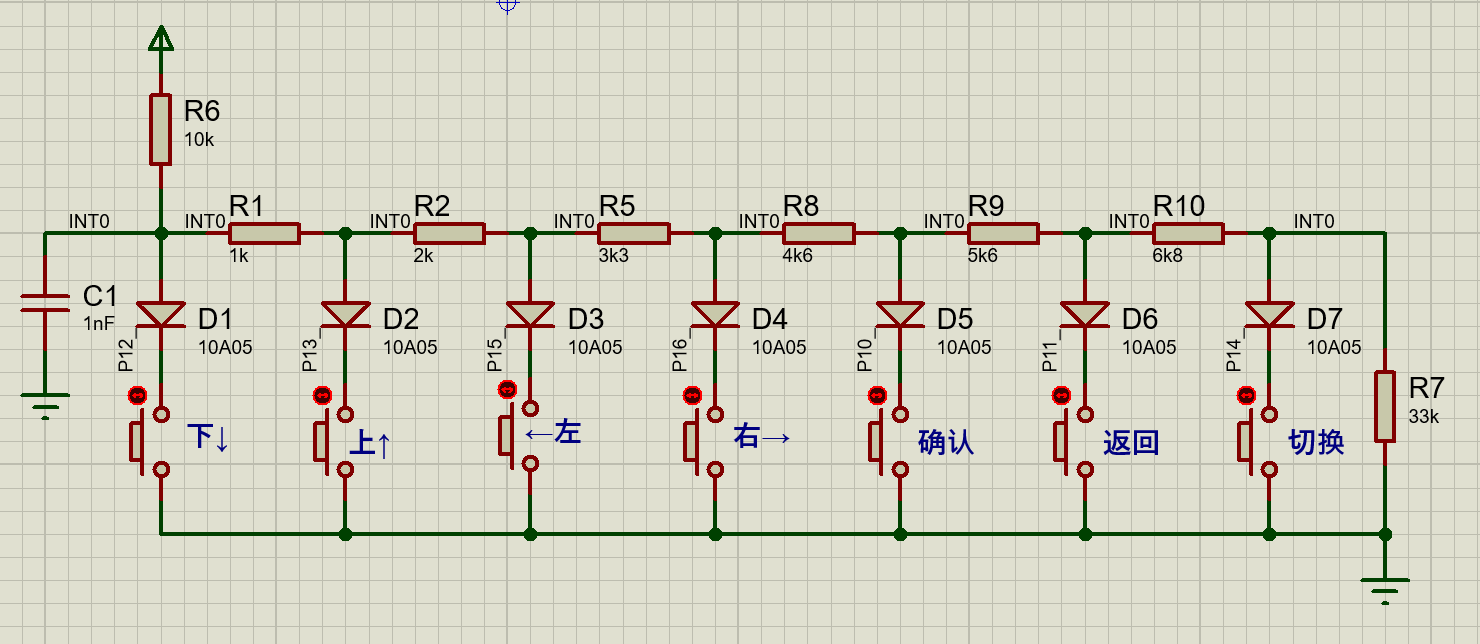


图 3 按键的电路设计

通过整流二极管和不同电容电阻的合理布置，当某个按键被按下时，除了遥控器P1口接收到的按键信号外，还有一个多余的INT0信号发送到主机的P3.2口，且该信号不会反过来影响到遥控器P1口的按键信号接收

### 数码管和蜂鸣器

在主机还有多余的IO口中引出一组IO口控制一位数码管的显示，引出一个IO口控无源制蜂鸣器的音乐播放。由于要使蜂鸣器能够播放音乐，即不间断地发出不同音调的蜂鸣声，所以选用无源蜂鸣器而非有源蜂鸣器

## 软件设计

### 游戏菜单

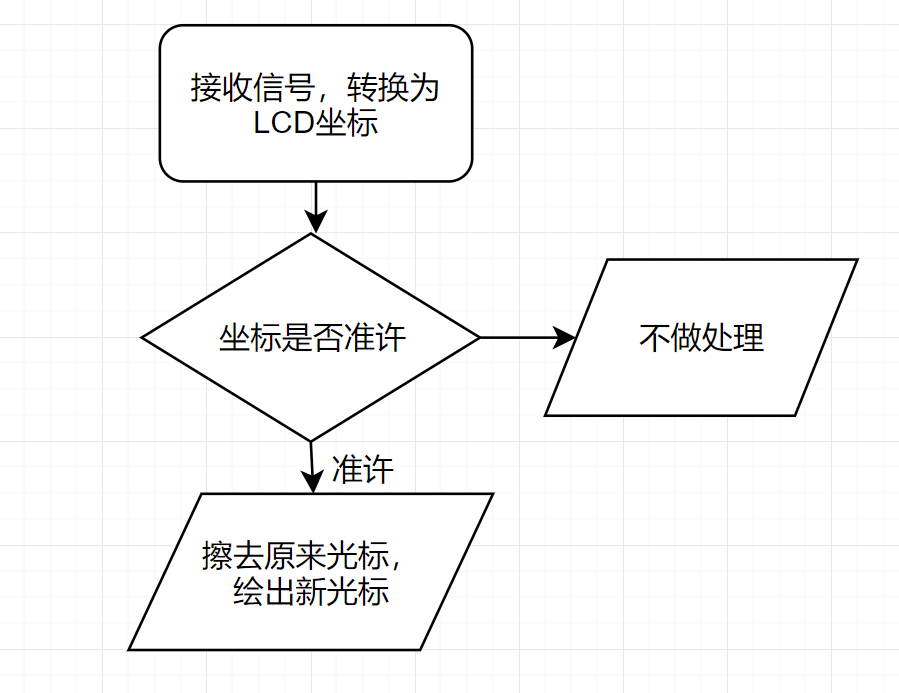
1. 硬件配套模块：LCD12864功能函数
2. 开机动画：在LCD上简单地绘制图案，放在程序的开头，当程序运行时，LCD屏上首先运行的便是开机动画，而按下任意按键之后，该动画便跳过，进入正真的主程序——主菜单
3. 主菜单：由三个选项组成：开始游戏、排行榜和操作说明，外加一个箭头光标。用户通过按下不同的按键可以控制光标在这三个选项之间移动，或选中进入对应的功能，程序跳转到对应的子程序执行。光标的移动选中原理如下图

图 4 菜单光标移动程序流程图

1. 排行榜和操作说明：这两部分都是直接在LCD上静态显示，在主菜单选中对应功能并确认之后既可进入对应的LCD显示程序。唯一的不同点在于排行榜需要每次都从EEPROM中读取最新的数据进行显示，而操作说明则是原本便写好放在单片机中的，对于EEPROM读取部分在3.2.5有详细描述
2. 关卡选择界面：该部分实现方法与主菜单相同，都是静态文字显示+动态光标移动，选择对应选项并确认既可进入对应子程序，不多赘述

### 游戏运行

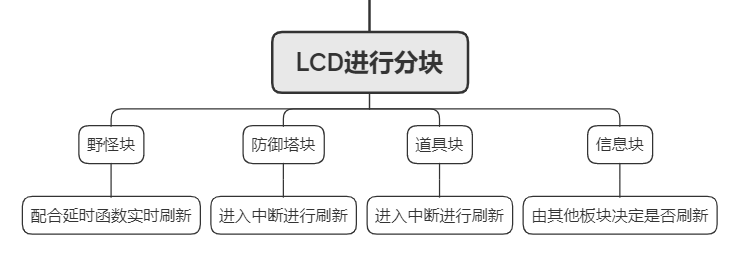
该部分是整个项目的核心，通过把整个LCD屏进行划分为4\*8个小块，在每个小块上单独编程，再对整个LCD进行合并，通过不断修改每个小块的显示内容，配合以短暂的延时函数，达到动态显示的效果

图 5 LCD分块编程

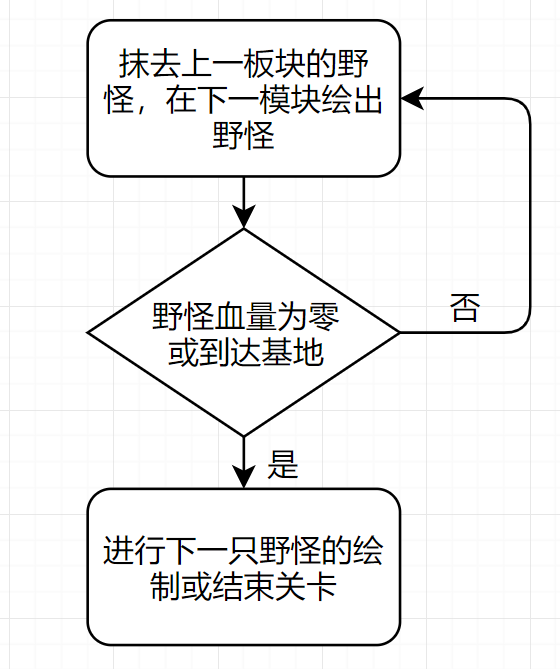
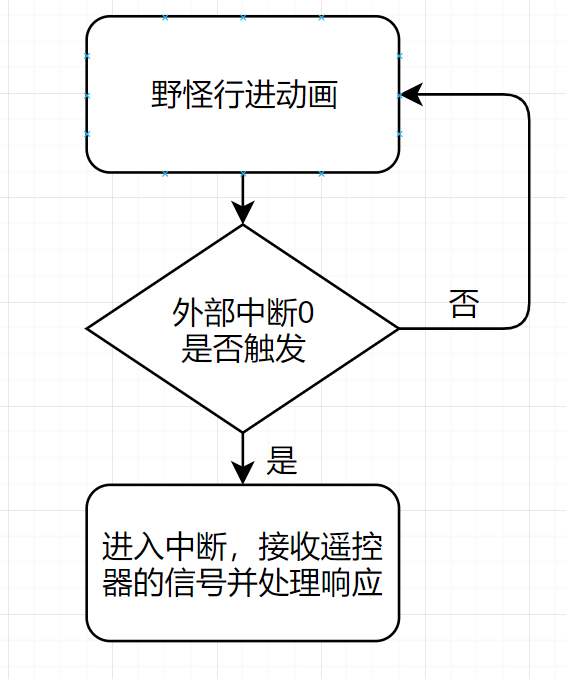
1. 野怪块：根据一定的坐标规律，实时修改两个板块的显示内容为野怪或空白，配合延时函数，达到野怪动画行进的效果。绘制不同的野怪、不同的野怪行进录像，只需要调用对应的函数。一个关卡由不同的该模块组成。
2. 防御塔块：该模块包含防御塔的放置、移除，以及光标的移动。模块由中断触发，只有外部中断0接收到中断信号时，即有遥控器有按键按下，程序进入中断，接收来自遥控器的信号并处理和响应

图 6 防御塔、光标处理

图 7 野怪行进动画

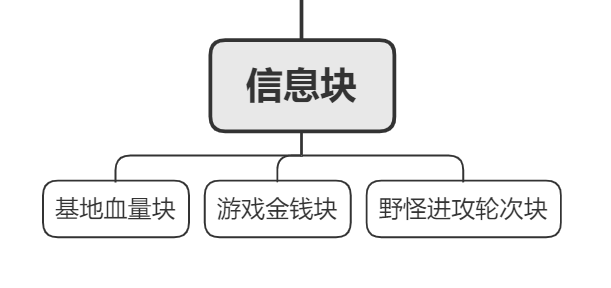
1. 道具块：道具块实现原理与防御塔块基本相同，都是进入同一个中断服务函数进行信号的处理和响应
2. 信息块：该板块分为两个部分，一个是静态的信息标识区域，一个是信息量变动的动态区域，该板块的刷新由整个其他三个板块决定。该板块根据显示信息不同又可分为基地血量块、游戏金钱块和野怪进攻轮次块，每块各有两个小块，一个静态块和一个动态块

图 8 信息块再细分

### 按键扫描

该模块是整个项目的次核心，如果没有这个次核心而只有核心，该项目将不是一个放置类塔防游戏，而只是一个可以播放动画的显示器

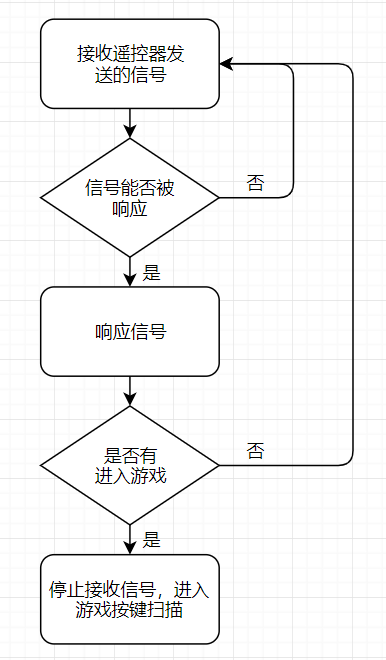
1. 主菜单扫描按键：该项目在硬件上配备了七个实体按键，而在未进入游戏之前，即主菜单和关卡选择界面，实际可用的按键只有四个，分别为：上、下、确认、返回。该部分的按键扫描较为简单，主机通过循环接收遥控器发来的信号，对信号进行处理和判断，若是有用信号，则做出响应
2. 游戏按键扫描：该部分不同于主菜单的按键扫描，由于已经进入游戏，游戏界面绘制函数已经在运行，此时若想在LCD上修改内容，只有先进入中断，停止游戏界面的绘制，转而对用户的按键输入进行响应，接收来自遥控器的信号并处理响应。由于使用外部中断0，是最高级别的中断，可以确保无论在任何时候，用户的按键输入都可以被遥控器接收并发送给主机响应，并且游戏界面也不会有明显的停顿感，游戏动画可以流畅地继续

图 9 主菜单按键扫描

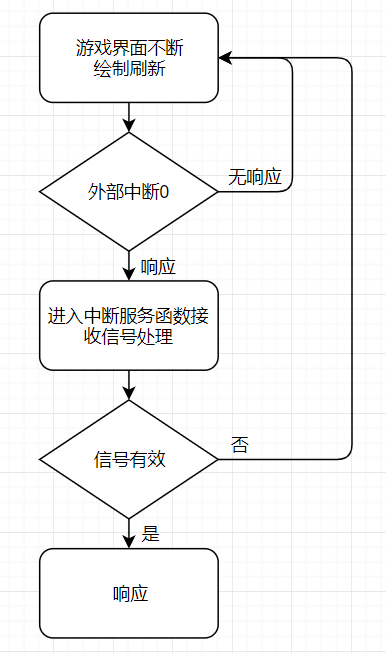


图 10 游戏按键扫描

### 双机通信

主机单片机和遥控器单片机之间的通信方式为硬件UART异步串行通信。使用该通信的好处为所需接口少，只需要两个接口既可完成，且在软件设计方面代码编写简单。

在硬件上，七个按键直接与遥控器单片机的一组IO口相连接，因此，在软件方面，遥控器单片机只需要不断地把该组IO口的信号发送给主机，而主机只需要在需要接收的时候调用接收信号函数，就可以实时接收到来自遥控器的按键信号，并通过对该信号的处理和响应，控制LCD的显示，达到用户操作游戏的目的

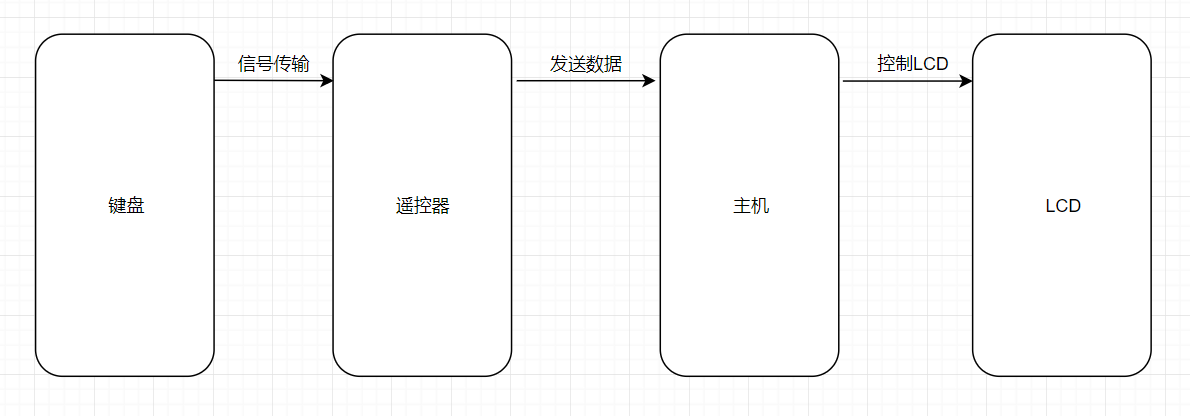


图 11 双机通信及数据传输

### EEPROM读写

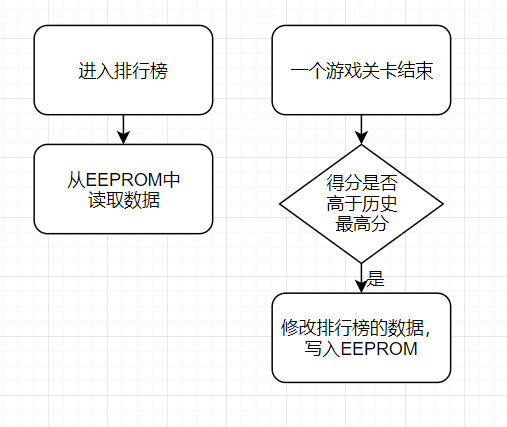
由于24C02C有硬件IIC，这里选择软件用模拟IIC主机的方式，将主机单片机作为主机，24C02C作为从机，由单片机对EEPROM进行读写。软件模拟IIC主机部分主要是模拟IIC的时序，使用延时函数做出IIC通信协议中的电平变化规定，分别模拟出起始信号、结束信号、应答信号，再在此的基础上编写对EEPROM的读写函数，最后调用该函数对EEPROM进行固定地址的读写。由于该项目需要用到的EEPROM的地址空间非常少，每次读写的数据也非常少（只有六组）。为了代码编写的简便，选择每次读写都进行全覆盖式地对固定的EEPROM的地址进行读写。

图 12 EEPROM读、写的两种情况

### 游戏效果

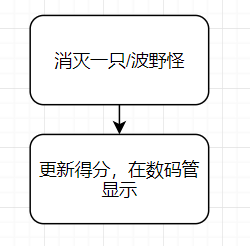
1. 数码管计分：在游戏运行的过程中，通过数码管实时把游戏关卡得分显示出来。游戏得分更新于两种情况：1，消灭一直野怪；2，消灭一波野怪。每关的游戏评分机制不一样，但是总共便是这两种情况才会对游戏得分进行更新统计

图 13 数码管计分更新

1. 蜂鸣器放音乐：在进入开始游戏和游戏结束时，通过调用蜂鸣器播放音乐对应的函数进行播放音乐。蜂鸣器播放音乐的软件实现：使用单片机的定时器1（低优先级），模拟PWM波的输出，实现无中断的微秒级别的延时，由于无源蜂鸣器的硬件特效，只要控制好PWM波的波形，就可以发出不同声调的蜂鸣声，控制每个音符的发声时长，从而演奏音乐

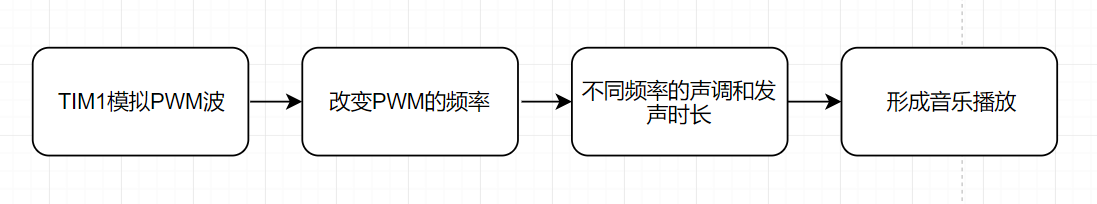


图 14 蜂鸣器播放音乐原理

# 数据结构与数据处理

## LCD坐标

LCD坐标：在整个项目的LCD显示中，无一例外的都用到了LCD坐标来记录光标、野怪、防御塔的位置，并以此坐标为依据在LCD上画出真正的图案。用户每次按下一个方向按键都是对当前光标的坐标的修改，而按下功能按键时则是对坐标的记录或是删除。在每次对坐标修改之后，都调用一次LCD显示函数在LCD上绘制相应的图案，以此达到用户操作游戏的目的。游戏动画效果也是一样的原理，只是不需要用户在按键上输入，对应的程序中有自动修改坐标的代码块

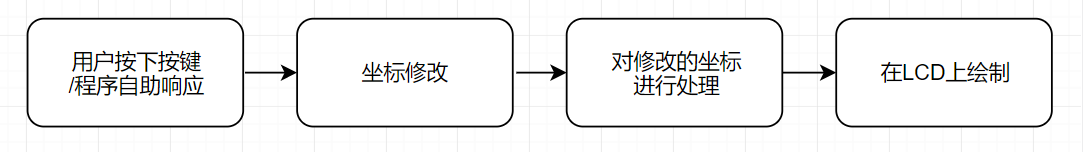


图 15 LCD显示的流程

## 防御塔坐标

防御塔的坐标：基本所有的坐标都是以全局变量的形式存在于程序之中，只有防御塔坐标是以一个结构体组的形式存在。一般的坐标包括s（半屏选择）、x（横轴八选一）、y（纵轴四选一），但由于防御塔类型不止一种，因此每组防御塔坐标里面都还要加入一个记录防御塔类型的标记量flag。在放置防御塔、移除防御塔、计算野怪收到防御塔的伤害时，可以直接调用整个结构体进行判断计算，这样做的好处是程序看起来会更简洁，代码执行效率也会更高

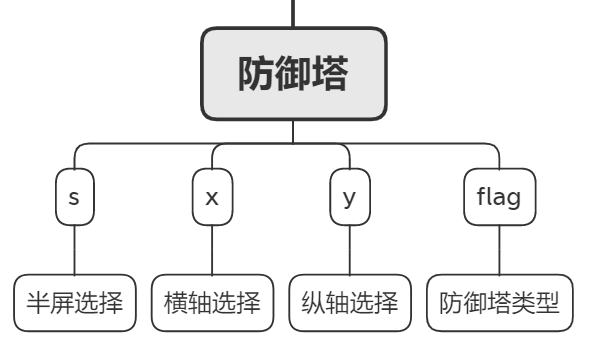


图 16 防御塔结构体的数据组成

# 亮点设计

## 软件亮点

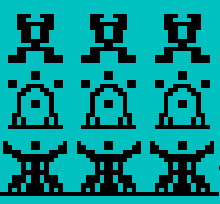
1. 防御塔类型：该项目设计有三种不同类型的防御塔，包括：两种攻击范围、两种攻击伤害和三种金钱值。

图 17 三种防御塔

三种防御塔对应所需金钱有一、二、三，攻击范围有一、三，攻击伤害有一、二

1. 防御塔伤害叠加：无论防御塔如何放置，无论野怪路线如何，只要进入到防御塔攻击范围内的野怪都会同时收到防御塔的攻击，若野怪同时处在多个防御塔的攻击范围内，那么野怪将会同时受到多个防御塔的攻击伤害，且伤害同时结算。这样做的好处是提升游戏的可玩性，用户对放防御塔的放置可以有多种设计，达到不同的效果
2. 防御塔可移除：用户可在道具栏中选择移除已放置的防御塔的道具

图 18 移除防御塔道具

用户选中该道具之后，将光标移动到已放置的防御塔上，点击确认，既可移除相应选中的防御塔（同时减去一定金钱），用户可再在该位置放置更高级别的防御塔（间接完成防御塔的升级）

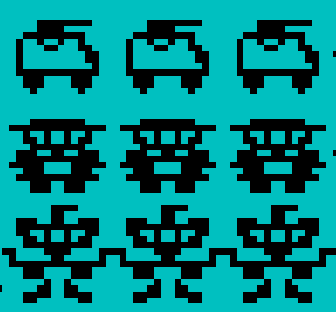
1. 野怪类型：该项目设计有三种不同类型的野怪，会在不同的关卡中出场，野怪血量值分别为二、四、六，不同野怪对基地的伤害值不同，分别为一、二

图 19 三种野怪

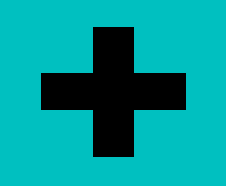
1. 野怪进攻波次：在同一关卡中，随着时间的推移，一波接一波的野怪数量会逐渐增多，野怪的级别更高，移动速度更快；在不同关卡的设计中，随着关卡的推移，野怪进攻的波次会逐渐增多，且数目、等级、速度都有一定的增加
2. 关卡计分机制：在不同关卡中计分机制不同，不同波次的分值也不同，有的是消灭一只野怪便得一分，有的是消灭整一波野怪才能得一分
3. 基地回血道具：用户可在道具栏中选择基地回血道具，确认之后基地血量将会增加，同时减去一定的金钱

图 20 基地回血道具

1. 游戏音乐：用户选中游戏关卡开始游戏和结束游戏时，都会播放一次音乐，且两次播放不同的音乐
2. 游戏暂停：用户在游戏中可随时按下返回键暂停游戏，按下确认键后继续游戏

## 硬件亮点

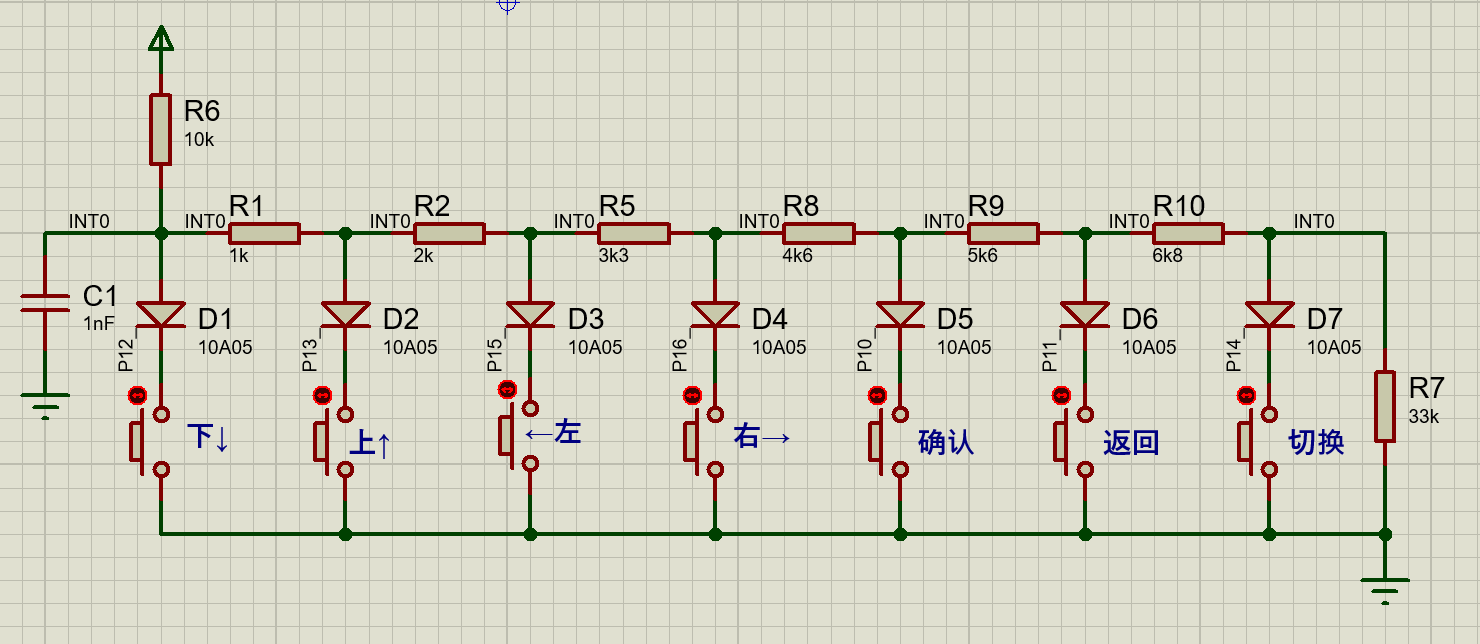
1. 键盘电路设计：使用电容、电阻和整流二极管，把七个按键联在一起，当某个按键按下时，不但遥控器单片机的一组IO口能接收到信号，还有额外的一个信号传给主机单片机，用来触发中断。合理的电路设计使该中断信号不会反过来影响到遥控器单片机IO口接受到的数据

图 21 键盘电路

1. LCD12864：选用LCD12864作为该项目的主要显示屏幕，128\*64的分辨率足够我们在上面绘制歌各种高辨识度的野怪和防御塔

# 系统兼容性说明

1. 环境配置：该项目的硬件部分是基于Proteus 8运行的仿真程序，用户需要在计算机上安装Proteus 8才能运行该项目。该项目的软件部分使用Keil 5进行编写和编译，以AT89C51为开发平台，该项目的程序大小不应超过该型号单片机的存储容量。
2. BUG说明：由于Proteus 8本身的不稳定性，该项目在运行过程中偶尔会有卡死现象，用户只需要重启Proteus 8的仿真即可，若游戏关卡已结束，分数统计会保存在外部存储器中

# 总结与体会

1. 软件：对不同函数进行封装很有必要，特别是像这样的一个大型项目时，把先完成的功能函数封装起来，放到不同的.c文件中，并为不同的.c文件建立一个.h头文件，这是单片机写程序比较特殊的地方，这样做的好处在于你可以对不同类型的函数做一个细分，而在需要调用到时，直接到对应的头文件中去查找即可。

单片机编程不同于控制台程序的另一点就是内存空间的大小，单片机是一个存储空间很小的系统，一般也就几KB大小的内存。这就需要我们在编写程序的时候对于变量的定义、存储空间的申请都要有精密的计算，确保自己申请的内存空间可以最大限度地被我们的程序所利用。在这儿，简单的节省空间的几个方法：对一个变量的重复赋值使用；在不同的函数中我们单独去定义变量申请空间；尽量使用code定义那些我们不需要去修改的变量

1. 硬件：嵌入式工程最大的不同之处在于，拿到手一个项目之后，首先思考的可能不是建立一个怎样的系统框架，而是先去搜索适合完成这个项目的硬件外设。比如，我要先找到LCD12864这样一块足以完成该项目要求的屏幕，才能开始着手去搭建一个系统框架，软件框架总是要建立在硬件基础之上，编写控制台程序时我们不用去考虑硬件的部分，而在单片机平台上，这是一个绕不过的点

另一个不同之处在于，对硬件的学习是很繁杂的，各种不同型号的外设让人眼花缭乱，我们很难把每一种外设的使用方法熟记于心，更多的时候，我们只是知道这样一个外设有什么样的功能，能用来干嘛，在真正需要到他的时候，再去深入学习他的使用方法。当然，一定的硬件基础知识积累是非常有必要的，可以让我们在学习使用这个过程变得更加简单，快速上手。

1. 软硬结合：对单片机开发的另一个重点在于，合理分配单片机的资源，比如IO口和外部中断、定时器、硬件UART等。在这个项目中，由于一个单片机的IO口是不足以驱动如此多的外设的，因此我们使用双机工作的方式，用两块单片机分工合作。而在定时器和外部中断资源的分配上，该项目使用了一个外部中断0和定时器1，分别实现不同的功能，合理的优先级设定使其工作时不会产生冲突