共享单车物联网锁

成员姓名：邱永骋
龚治乔
郑秋里

摘要

本设计以 STM32 单片机为控制芯片，设计制作的一款物联网智能锁。它采用了 SIM800C 作为通信模块，以阿里云服务器作为智能锁与 Android 的通信中介，用 MySQL 数据库保存锁的状态和各种操作标志位，以 PHP 及 HTML 网页作为访问数据库的媒介，网页运行于 Apache 服务器中。手机 APP 创建了查询及开锁两个按钮，并建立对应的点击事件监听，当按钮被点击时，通过访问相应的 HTML、PHP 网页完成对数据库查询及修改功能，并建立一个单独的线程，通过访问相应功能网页来执行关锁完成的通知任务。在待机过程中，STM32 不停地访问查询数据库里的开锁标志位，当手机通过 APP 使数据库里的开锁标志改变时，STM32 会即刻查询到该变化，然后控制电机来开启马蹄锁，并可用微动开关来检测电动开锁以及手动关锁是否完成。

关键词：STM32，SIM800C，MySQL 数据库，PHP，多线程，web 网页，Android

1 方案论证

1.1 联网方式的比较与选择

通信模块采用 SIM800C，服务器采用阿里云服务器。如何让 STM32 联网，并且能与手机 APP 产生交互，我们思考了两个方案。

方案一：通过 TCP 服务，以阿里云服务器为通信中介，作为服务端，STM32 和 APP 作为客户端，直接连接服务器中的 MySQL 数据库，以此进行交互。

方案二：通过 HTTP 服务，让 STM32 单片机和手机 APP 访问 PHP 网页，通过功能网页查询或修改数据库的值，以此进行交互。

经比较，我们决定采用第二种方案，原因是第二种方案，比较简洁，实现过程相比于第一种更加方便。而且 SIM800C 模块通过 AT 指令直接使用 HTTP GET 可以很方便的提取出网页中我们需要查询的值，而且网页存于服务器中，方便于修改，如果日后需要相应变化的改动，不必更新 Android APP 及智能锁，直接修改网页即可。该方案的短板是速度不是很快，在 APP 上完成开锁按钮动作后，锁需要几秒钟才能反应开锁操作。

1.2 锁设计中的电机驱动器的比较与选择

在锁的研究设计过程中，我们发现直流电机不能被单片机直接驱动，需要一个电机驱动器。市场上现在最流行的直流电机驱动应该就是 L298N 和 TB6612 了。所以关于电机驱动我们有两个选择。

方案一：采用 L298N 驱动直流电机，L298N 有很强的驱动能力并且有过电流保护能力，当电机出现卡死时，可以保护电路和电机。

方案二：采用 TB6612 驱动直流电机，TB6612 的性能比较好而且体积很小。

最终我们决定采用第一种方案，使用 L298N 来驱动直流电机，因为我们对 L298N 这个模块比较熟悉，相对的对 TB6612 比较陌生。L298N 使用起来非常方便。但 L298N 也有不足之处，就是它的功耗较大，而且它的体积没有 TB6612 那么小巧，在锁的组装时会占很大空间。

1.3 Android 设计的比较与选择

我们小组 3 人都是非计算机专业，以前没接触过 Android，因此 Android APP 的设计，是一步步尝试摸索出来的，前后也出现过几种方案。

方案一：使用一个 TextView 文本框作为显示界面，用两个按钮完成用户交互功能，并建立对应的点击事件监听来完成查询和开锁功能，APP 直接访问 MySQL 数据库，并在文本框中显示相应的内容。

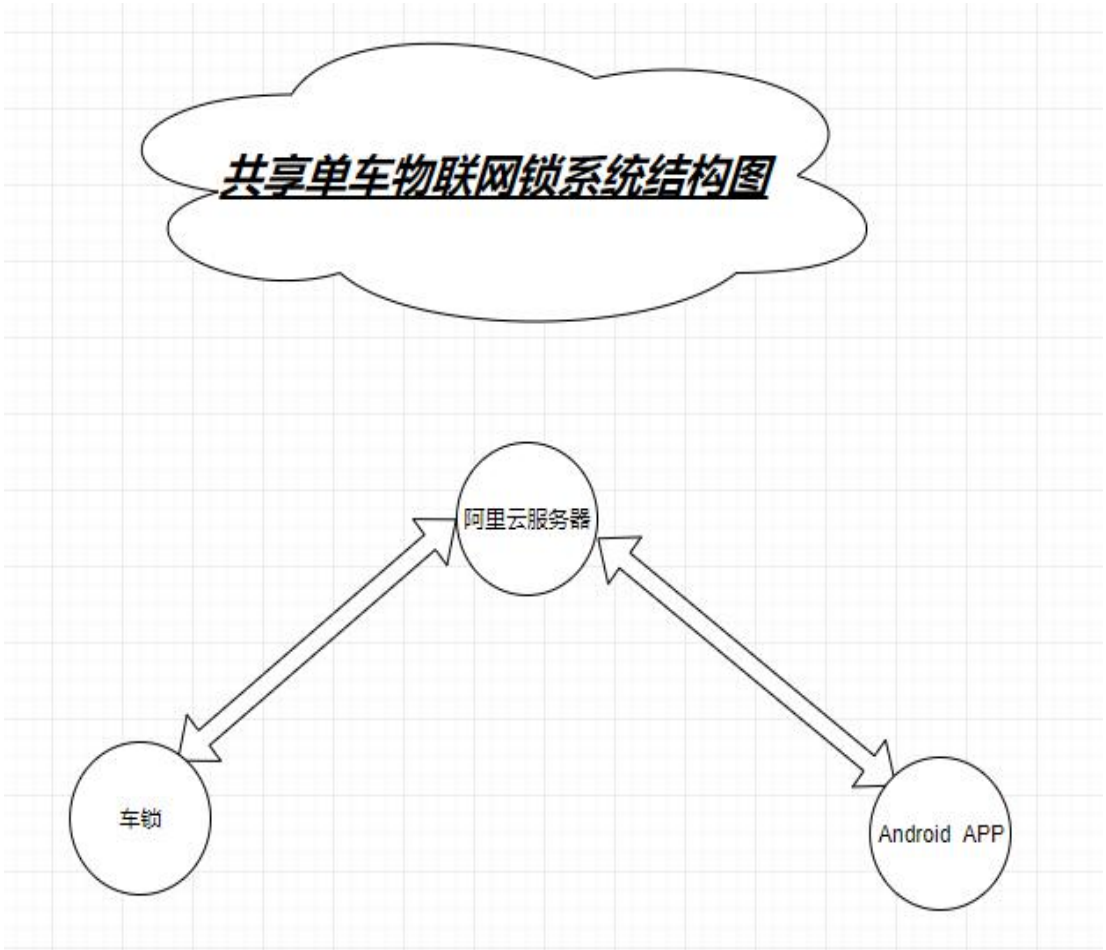
方案二：使用一个 webview 网页视图作为显示界面，用两个网页来连接数据库并显示相应界面，每个网页中都内嵌有自动刷新查询功能，当手动关锁完成时，通过 JavaScript 语言生成一个 alert 弹窗来通知关锁完成。

方案三：用两个 webview 作为显示界面，一个主显示界面用来显示查询和开锁内容，另一个 webview 用来显示关锁完成通知网页，并创建一个线程来控制它。

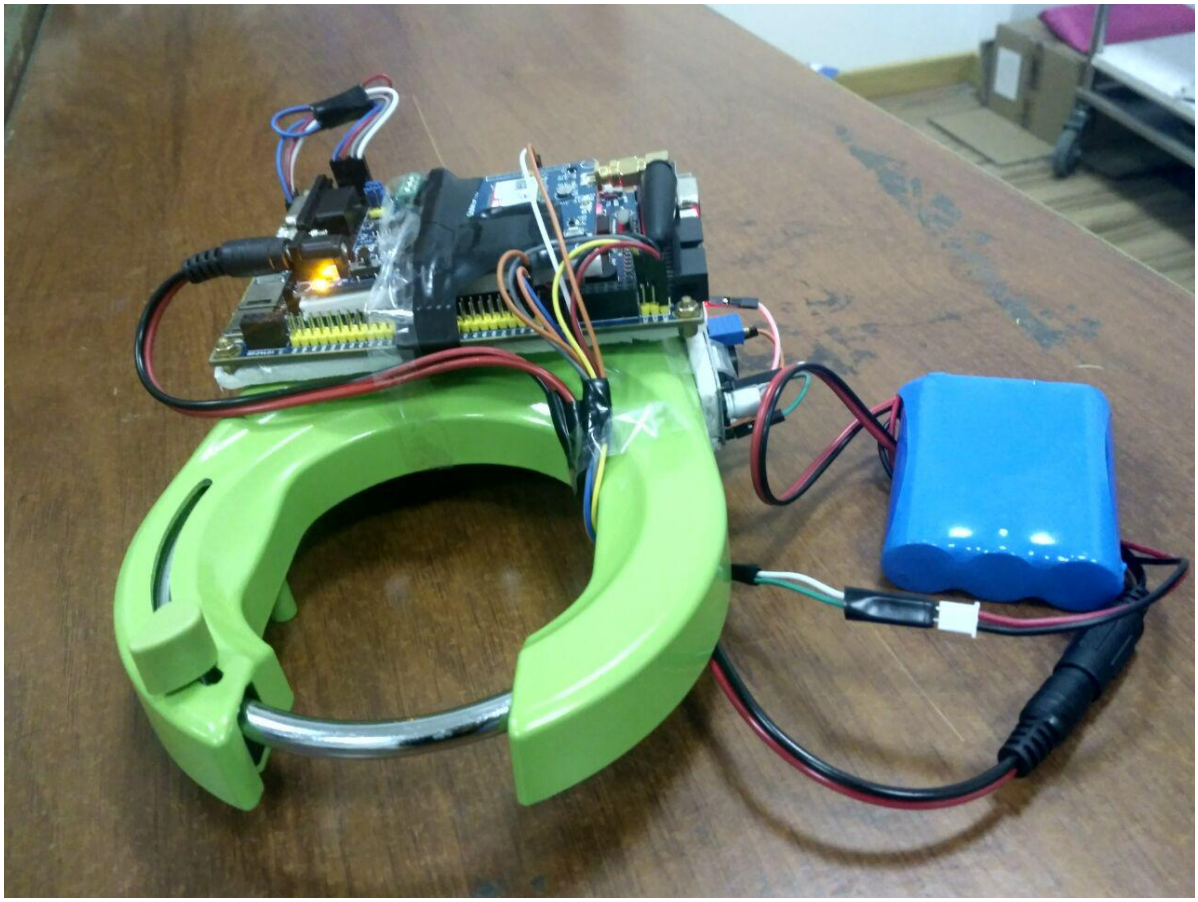
经过多次尝试，我们选择了第三种方案，因为 APP 直接访问 MySQL 数据库很困难，JavaScript 脚本在 webview 中无法正常运行，而第三种方案实现起来相对简单，且网页修改方便。

2 详细设计

共享单车物联网锁系统在设计上主要分为了硬件和软件两个部分，特别在软件方面，我们下足了功夫，把它分为阿里云云服务器、Android APP 和主控芯片程序，云服务器作为智能锁与 APP 的通信中介，如下是该系统结构图及实物图。



1 系统结构图



2 作品实物图

2.1 云服务器

根据题目的要求，我们的云服务器选择了阿里云，安装了 Apache 服务器来运行 HTML，PHP 功能网页，用 MySQL 数据库存储锁的状态及智能锁与 App 通信所需的标志位。数据库结构如下图所示，其中 id 为锁的编号，state 为当前锁的状态，flag 为是否需要电动开锁的标志位，notice 为手机 APP 是否需要显示手动关锁完成的标志位。

id	state	flag	notice
1	0	1	0
2	1	0	0
3	1	0	0
4	0	0	0
5	0	0	0
6	0	0	0
7	0	0	0
8	0	0	0
9	0	0	0

3 数据库结构图

针对本系统所需的功能，我们用 HTML、CSS、JavaScript 及 PHP 等语言写了 6 个

不同的网页，每个网页完成一个对应的查询修改数据库及显示信息功能。其中由智能锁访问的有 3 个 PHP 网页，分别为 flag、open 和 close。flag 网页用于查询数据库中开锁标志位，flag 是否为 1，即是否需要电动开锁；open 网页在开锁完成后进行访问，用于将数据库中锁的状态改为开，并置 0 开锁标志位 flag，防止重复开锁；close 网页在手动关锁完成后访问，用于将数据库中锁的状态改为关，并将手动关锁通知位 notice 置 1。

用于 Android APP 访问的有 3 个 HTML、CSS 和 PHP 语言嵌套的网页，分别为 notice、query_android 和 open_android。notice 用于查询是否有手动关锁完成，如果有则在网页中显示“关锁已完成”，并将数据库中通知标志位改为 0，防止重复显示，否则显示空白；query_android 网页用于查询当前锁的状态，将数据库中锁的状态显示出来；open_android 网页用于手机控制开锁，访问它后会将数据库中的开锁标志位 flag 置 1。

2.2 智能锁系统设计

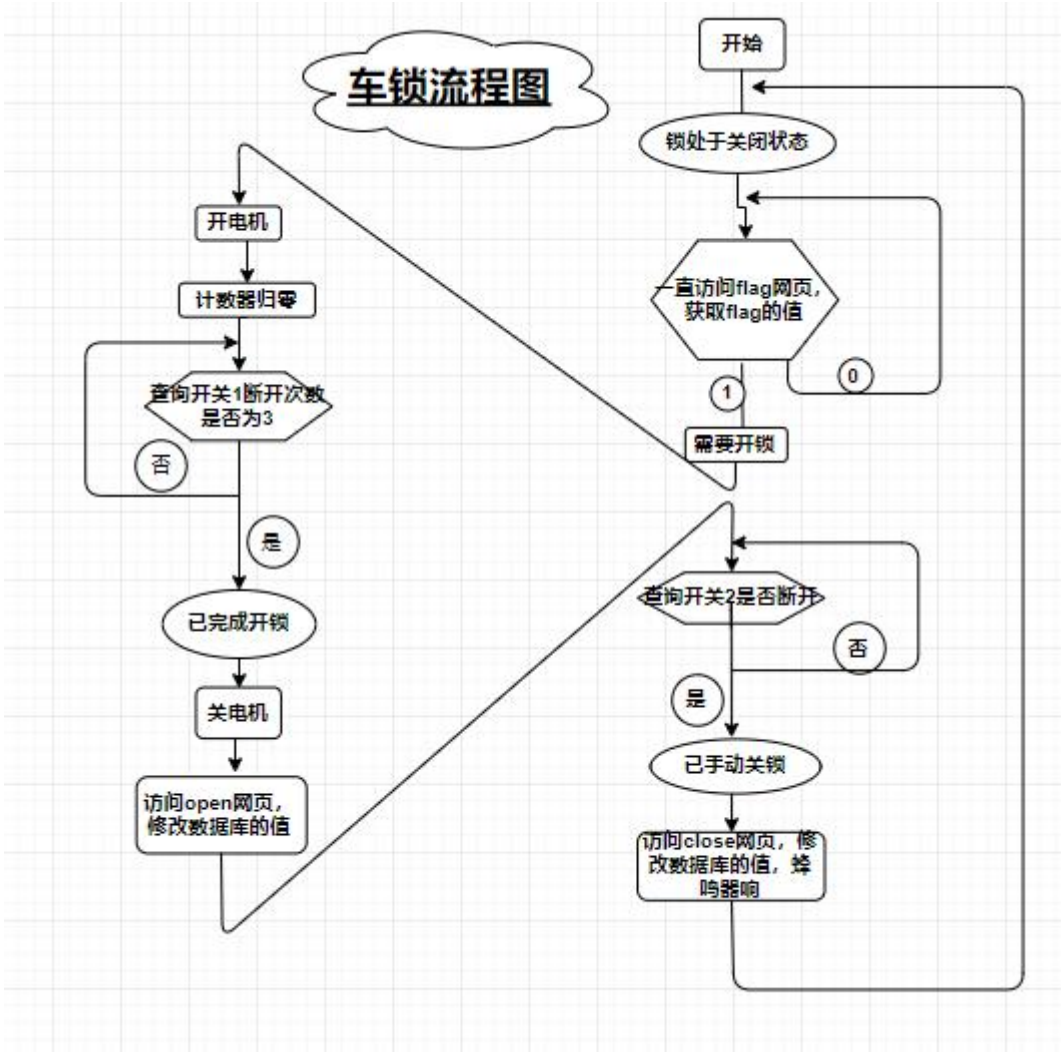
我们的锁硬件是从网上买的，控制过程中进行了一些修改。锁舌为一环形铁棒，其上有一个缺口并在一端连接一个弹簧，用一个卡口控制锁的开关。当锁关闭时，只要打开卡口，锁舌就会被弹簧拉开，完成开锁，手动关锁时，人力拉动弹簧，当锁舌到达适当位置时，卡口会自动卡住，关锁完成。

而卡口部分则由弹簧和电机共同控制，卡口在弹簧作用下有卡住的趋势，锁舌一旦转到缺口处便自动卡住，当需要开锁时，电机会拉开卡口，锁自动打开。卡口处还有两个微动开关，一个用于检测当前卡口的状态，是开还是关，另一个用于检测电机转动的角度，每转 120 度会被触发一次，每次开锁需要触发三次，即转 360 度。

整个系统由一个 12v 电池提供电源，它分两路为系统供电，一路接到 L298n 驱动模块上，用于驱动电机，并为单片机供电，另一路通过一个稳压模块降到 7.4v 后为 SIM800C 供电。

主控芯片采用的是 STM32 单片机，主要使用到了串口、定时器、外部中断、DMA 等外设。

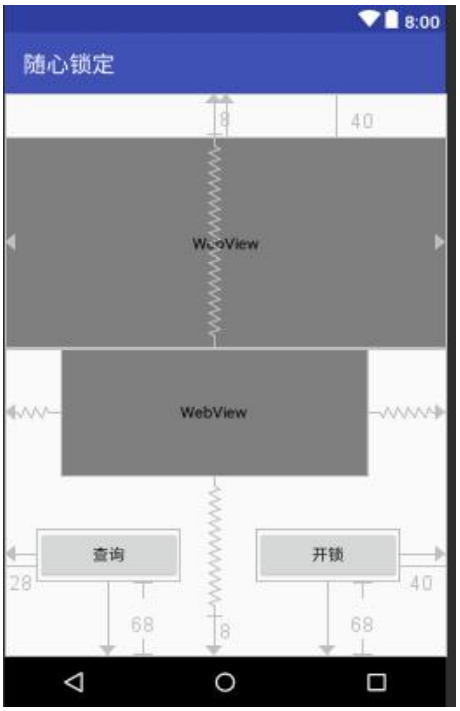
因为我们的通信模块是 SIM800C 模块，单片机可以通过串口来给 SIM800C 模块发送 AT 指令，从而控制 SIM800C。串口发送采用了 DMA 方式来为 CPU 减负。由于每一次 AT 指令都有一个返回指令，所以同时使用了串口接收功能，通过定时器中断的方式来检查返回的是否为同一个字符串。STM32 操作 SIM800C 模块最主要的是我们封装了一个 Send_Command 函数，通过这个函数来发送 AT 指令，并检查是否接收到了正确的返回信息，如果没有接收到正确的返回信息，那就通过再次发送这条指令，直到接受到正确的返回信息，才继续执行下一条程序。单片机联网通信是通过这个函数发送 AT 指令开启 HTTP 服务实现的。外部中断的使用是用来检测锁是否被上锁，还有电机是否转了合适的角度。如下这张图是整个主控芯片程序的流程图



4　智能锁系统流程图

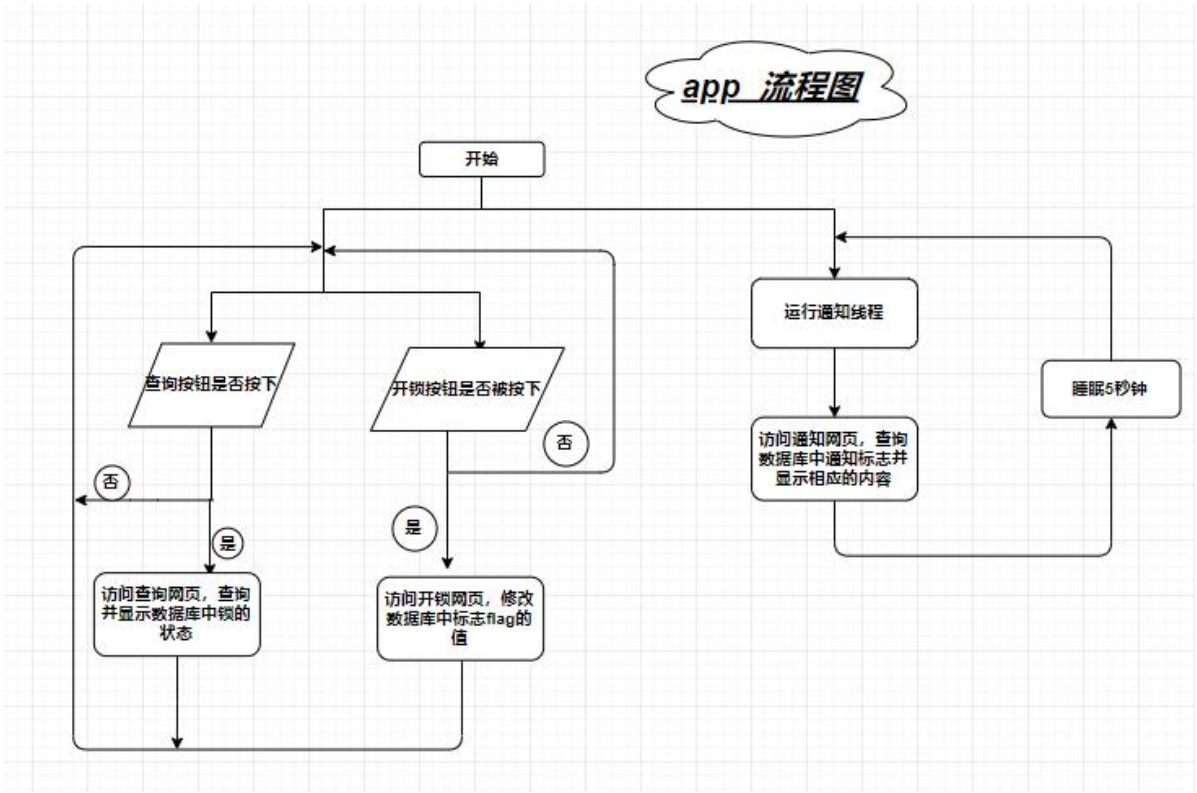
2.3　Android　APP

我们的安卓软件 UI 界面设计如下图所示，上面一个 webview 网页视图作为主要的显示界面，用来显示查询，开锁网页；另一个 webview 为辅显示界面，专门用来显示手动开锁完成的网页，当不需要时它会显示为空白，使用户察觉不到它的存在，只有需要它时才会显示信息；下面的两个按钮为用户交互控件，用来完成查询和开锁功能。



5 Android APP UI 界面设计图

功能部分主要是对这两个按钮建立对点击的事件进行监听，并定义内部类并重写onClick()方法，在主网页视图中完成对相应功能网页的访问及显示。此外，再创建一个线程，专门用来完成手动关锁完成通知的任务，每隔 5 秒钟运行一次该线程，并在辅网页视图中显示通知网页的内容。APP 的流程如下图所示



6 Android APP 流程图

APP 的执行效果如下图所示。

7/8 Android APP 界面 效果图

3 调试

3.1 云服务器及 Android APP

安装完 Apache 服务器及 MySQL 数据库后，我们开始了功能网页的编写，由于该功能不是很复杂，所以很快便编写完毕。但 Android 编写过程就复杂得多，我们是从零开始学习与摸索。

最开始想到的方案是用两个按钮用于用户交互，实现查询及开锁功能，用一个 TextView 文本框来显示信息，内部直接连接 MySQL 数据库，但是连接数据库的环节出现了问题，连续几天都没能解决。于是决定更换方案，用一个 webview 网页视图来显示信息，通过网页来访问数据库，最终该方案成功实现。还有一个功能就是要通知手动关锁完成，刚开始想到的方法是将相应的 JavaScript 脚本嵌套在查询和开锁两个网页中，自动刷新查询，当关锁完成时，通过 alert 弹窗来通知，结果 JavaScript 脚本在 Android 中无法正常运行。于是想到用另一个 webview 来显示通知网页，并新建了一个线程用来完成通知功能，每 5 秒运行一次，来访问刷新 notice 网页。

3.2 智能锁系统

对于智能锁的通信，刚开始想到的是学习 TCP/IP 协议，根据协议来编写通信程序，但了解 SIM800c 后，发现它里面有很全面的 AT 指令，覆盖了它所有的功能，根本不需要自己从头开始写。买来 SIM800c 后，调试过程也是一波三折，刚开始时总是出现意外情况，手机卡大小不合适，经常会有接触不良、跳线帽连接错误和天线出问题等意外，浪费了不少时间，几经周折之后，SIM800c 模块终于调试完成。

对于车锁硬件，我们发现：车锁马达虽小，但单片机引脚输出的功率还是不能驱动它，于是使用 L298n 来驱动它。最后为了统一电源，用一个稳压模块将电压降到 7.4v 后再为 SIM800c 供电。调试完成后，我们将各导线加固了下，把导线的连接处用锡焊了起来，并缠上绝缘胶，最后将系统各部分固定起来，防止拉扯导致接触不良或掉线。

4 小结

经过了一个多月的付出，我们团队克服了种种出人意料的阻碍，完成了这个共享单车物联网锁的设计，整个项目做下来并不简单，无论是锁的联网通信，还是 Android APP 的设计都需要花很多心思去攻克难题。

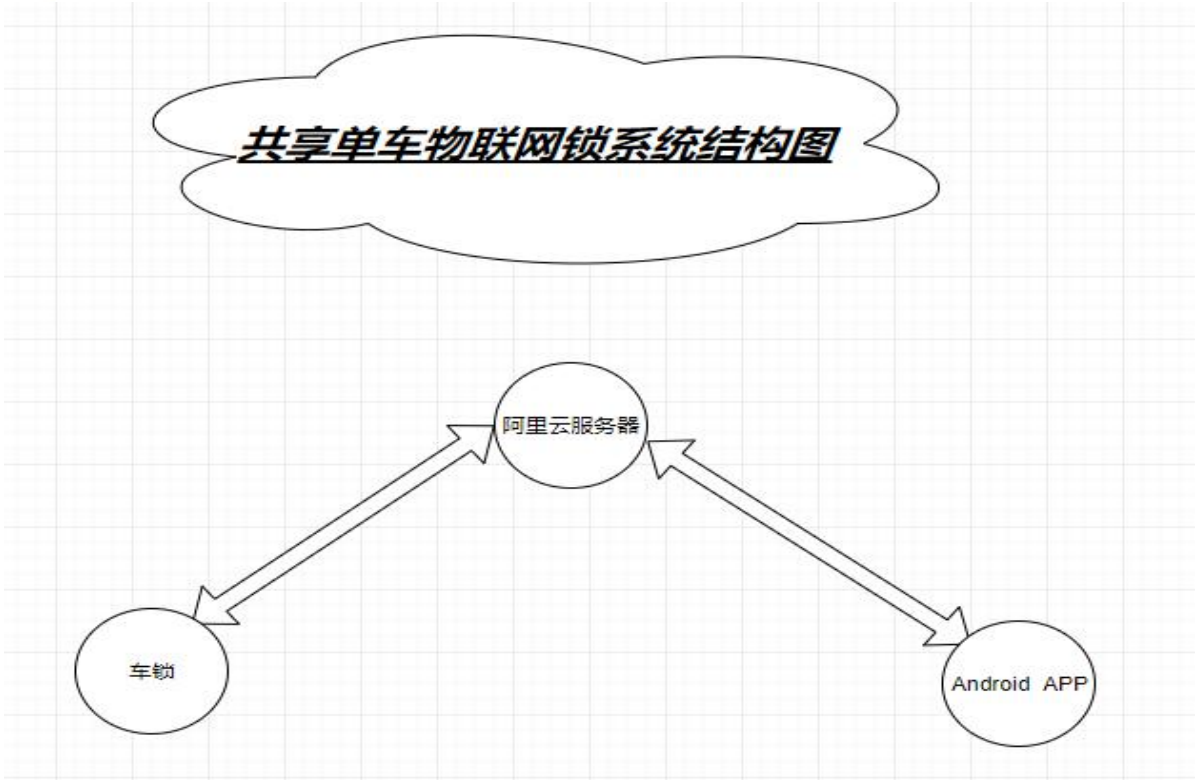
尽管有的处理方式对于这个作品来说并不是最优的选择，但是我们还是尽力把它做到最好。不管是 Android APP 的设计、智能锁系统设计、驱动电机的选择、联网方式等部分都凝结了我们团队的智慧和心血。调试完成之后，我们把各个部分组装在一起，看看整体是否能运行。特别是当它第一次成功实现我们的要求后，我们那种激动、

热血的心情实在难以言表，是非常有成就感和满足感的。在成功运行后，我们一起玩了好久这把锁，从手机开锁到手动关锁还有提示音我们反复做一系列操作，以至它趋于稳定。不过，这个作品同样有很多不足的地方，比如功耗问题和反应的速度问题，因为是采用的 HTTP 访问网页的方式，所以在速度方面确实是慢了一点。还有功耗问题，至今我们也找不到合适的解决办法。

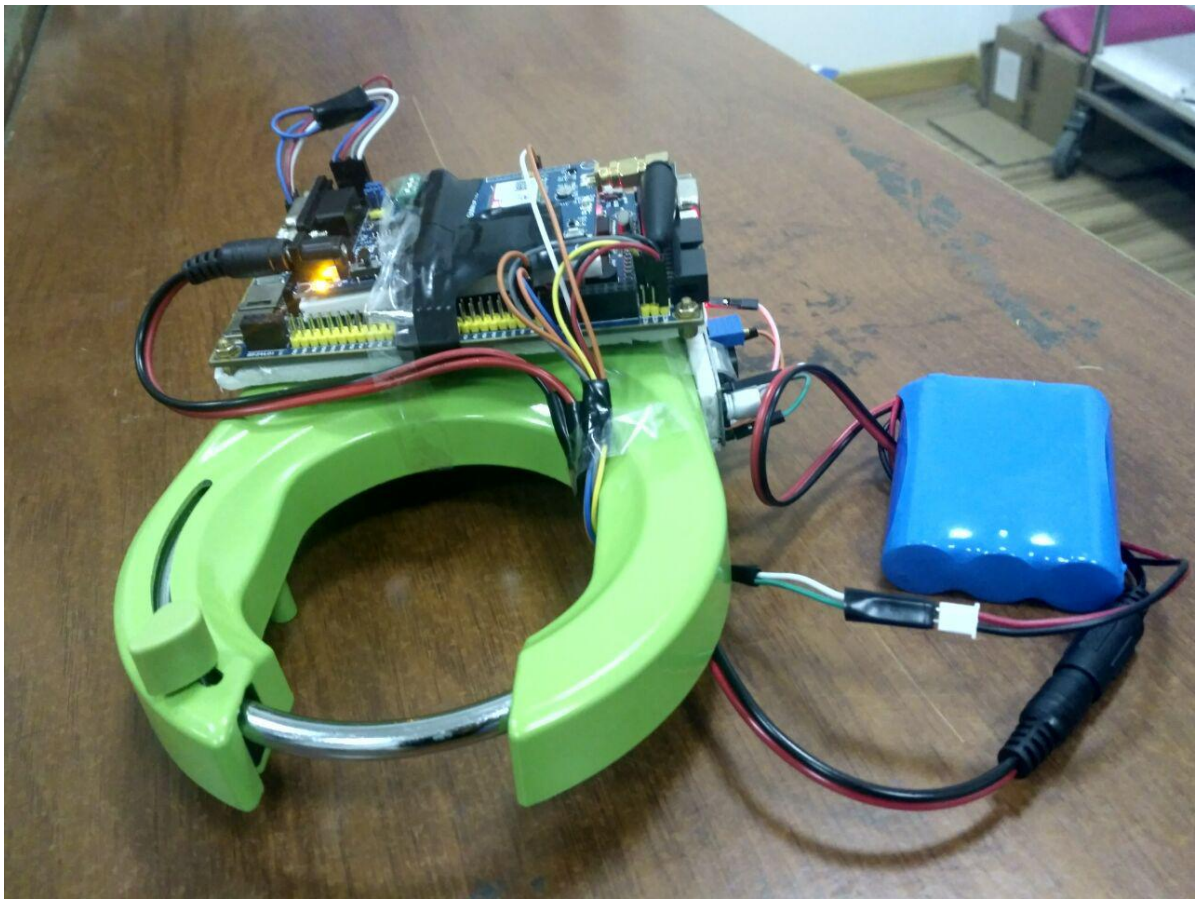
毫无疑问，在本次参赛制作的过程中，我们的专业知识得到了丰富，理论与实践的能力也提高不少。虽然在一些问题的处理上，我们确实存在短板。所以，在以后还有很多东西需要我们去学习去提升。我相信，以后的我们会有更强的能力设计出更好的作品。

附录 A

本报告使用的图片



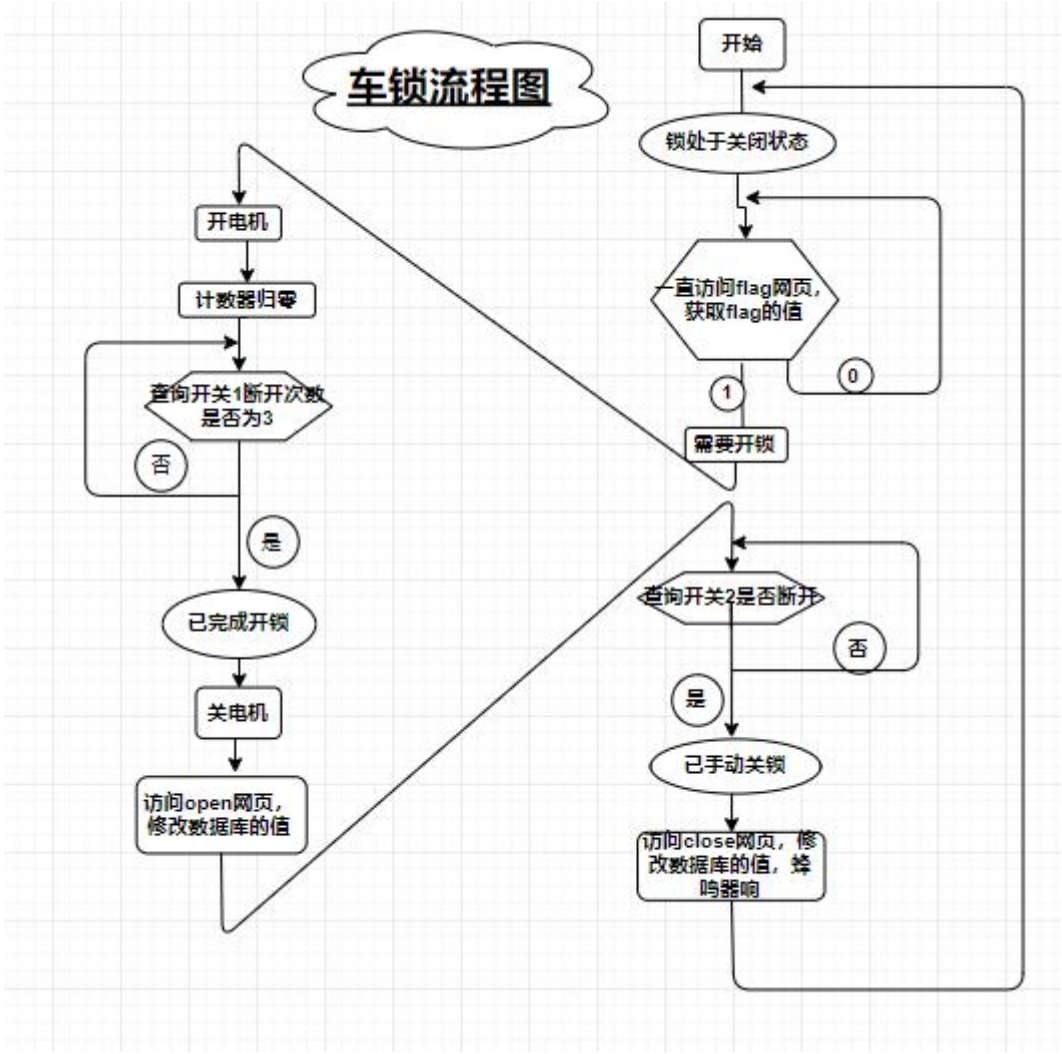
1



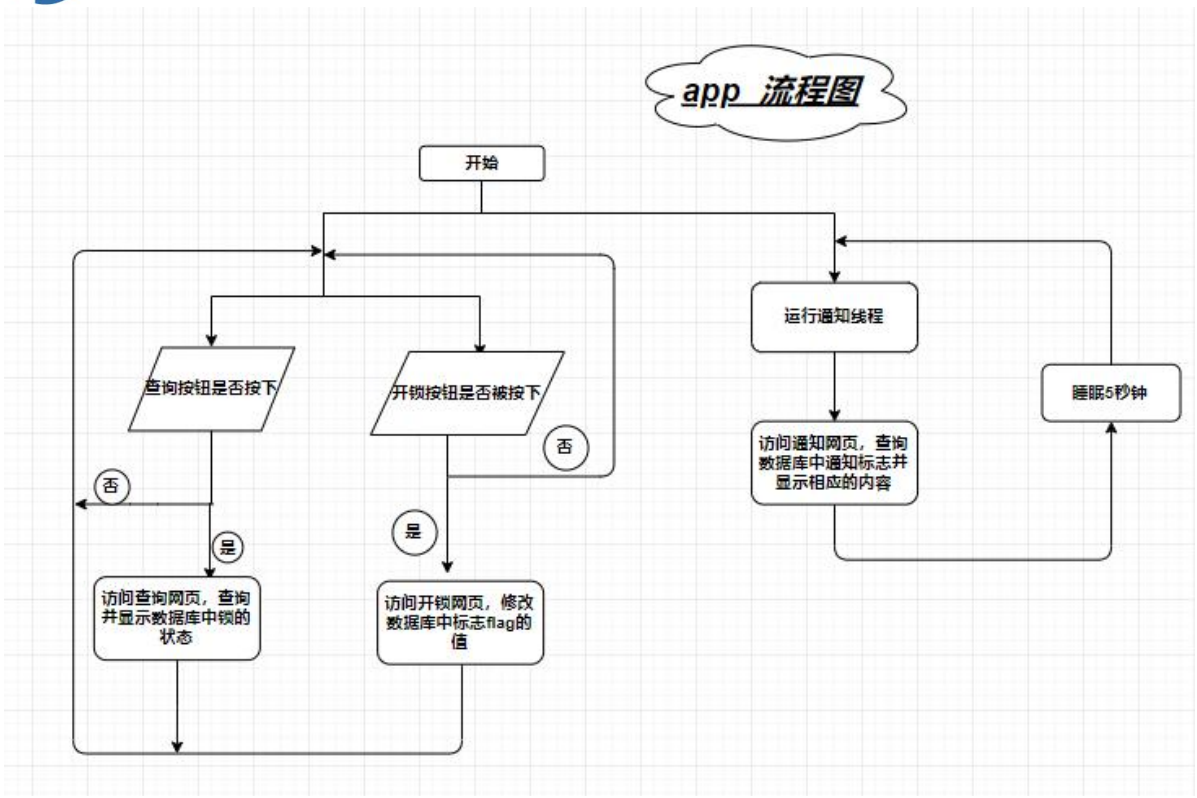
2

id	state	flag	notice
1	0	1	0
2	1	0	0
3	1	0	0
4	0	0	0
5	0	0	0
6	0	0	0
7	0	0	0
8	0	0	0
9	0	0	0
10	0	0	0
11	0	0	0
12	0	0	0
13	0	0	0
14	0	0	0

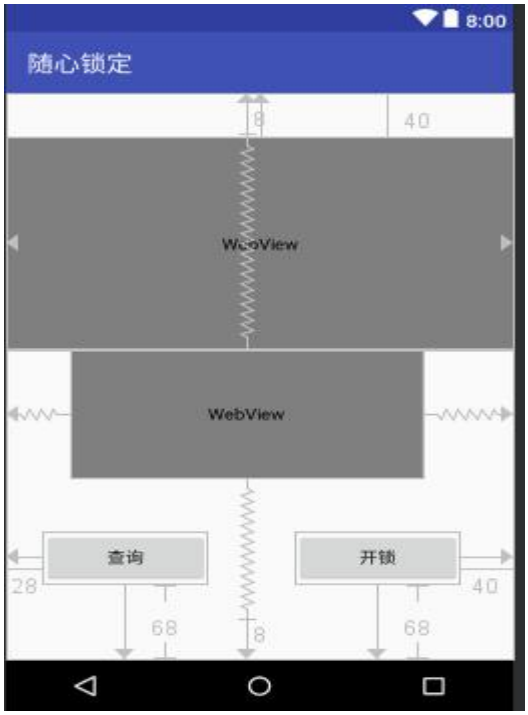
3


车锁流程图
开始
锁处于关闭状态
一直访问flag网页，获取flag的值
0
1
需要开锁
查询开关2是否断开
否
是
已手动关锁
访问close网页，修改数据库的值，蜂鸣器响
开电机
计数器归零
查询开关1断开次数是否为3
否
是
已完成开锁
关电机
访问open网页，修改数据库的值

4


app 流程图
开始
查询按钮是否按下
开锁按钮是否被按下
运行通知线程
睡眠5秒钟
否
否
是
是
访问查询网页，查询并显示数据库中锁的状态
访问开锁网页，修改数据库中标志flag的值
访问通知网页，查询数据库中通知标志并显示相应的内容

5


8:00
随心锁定
8
40
WebView
WebView
查询
开锁
28
40
68
8
68

6

附录 B

（1）stm32 控制程序主要部分

```
 web_ready();     //附着联网准备
TIM_SetCompare2(TIM3,450);     //L298n 控制 pwm，调整电机电压
while(1)
{
      if(lockflag==0)                    //当前锁为关闭状态，执行查询开锁功能
      {
            openflag = web_flag();   //查询数据库中开锁标志位 flag
             if(openflag==1)              //flag 为 1，即需要开锁
             {
                   open_lock();   //先开锁再访问 open 网页修改数据库中锁的状态并置 0 flag 标志
```

```
                delay_ms(1000);       //延时
            }
        }

        if(lockclose==1&&lockflag==1)          //锁处于打开状态且关锁开关被触发，即有手动关锁完成
            {
                beep();                  //蜂鸣器响
                lockclose=0;
              close_lock();              //执行关锁函数，修改数据库中锁的状态
            }
        delay_ms(1000);
    }
```

（2）手机开锁 open_android 网页程序

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
</head>
<body style="font-size:50px;">
 <div style="text-align: center;">
 <?php
$servername = "localhost";
$username = "xxx";
$password = "xxx";
$dbname = "xxx";
// 创建连接
$conn = new mysqli($servername, $username, $password, $dbname);     //连接数据库
// Check connection
if ($conn->connect_error) {
      die("连接失败，请重试");
}
$sql="select state from lock_ where id=1";
$result = $conn->query($sql);
$row = mysqli_fetch_array($result);
if($row['state'])          //如果当前锁的状态为开，则无需重复开锁
echo '<br>'.'已开锁';
else
{
    $sql = "update lock_ set flag=1 where id=1";
$result = $conn->query($sql);
```

```
if($result)      //如果锁的状态为关，则使开锁标志位 flag 置 1
echo '<br>'.'已开锁';
else
echo '<br>'.'失败，请重试！';
}
$conn->close();
?>
  </div>
</body>
</html>
```