

第十六届全国大学生  
智能汽车竞赛

# 技 术 报 告

学    校：北京理工大学

队伍名称：牛牛队

参赛队员：黄立群 詹天 高海智

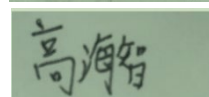
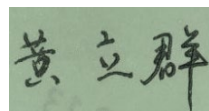
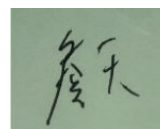
带队教师：冬雷

---

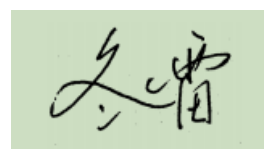
## 关于技术报告和研究论文使用授权的说明

本人完全了解全国大学生智能汽车竞赛关保留、使用技术报告和研究论文的规定，即：参赛作品著作权归参赛者本人，比赛组委会可以在相关主页上收录并公开参赛作品的设计方案、技术报告以及参赛模型车的视频、图像资料，并将相关内容编纂收录在组委会出版论文集中。

参赛队员签名：



带队教师签名：



日 期：2021 年 8 月 16 日

---

## 目录

第一章 概述 .....	1
第二章 智能车机械设计 .....	3
2.1 车模整体结构 .....	3
图 2.1 车模整体结构建模 .....	4
2.2 底板设计 .....	4
2.3 万向轮 .....	5
第三章 无线充电 .....	8
3.1 无线充电资料总结 .....	8
3.2 原理 .....	9
3.3 发射器 .....	9
3.3.1 基本电路 .....	9
3.3.2 功率限制 .....	10
3.4 接收部分 .....	10
3.4.1 “傻冲”方案 .....	10
3.4.2 线圈制作 .....	12
3.4.3 谐振电容的选择 .....	13
3.4.4 谐振电感 .....	13
3.4.5 整流 .....	13
第四章 硬件电路设计 .....	15
4.1 电源管理板 .....	15
4.2 驱动板 .....	17

---

4.3 编码器板 .....	18
4.4 控制板 .....	19
4.4.1 电源管理 .....	19
4.4.2 用户交互 .....	20
4.4.3 模拟量采样信号处理 .....	21
4.4.4 接口 .....	21
4.4.5 单片机主控 .....	22
第五章 软件设计 .....	24
5.1 运动感知 .....	24
5.1.1 主要思路 .....	24
5.1.2 灰度图二值化 .....	24
5.1.3 提取信标灯 .....	25
5.2 运动控制 .....	26
5.2.1 主要思路 .....	26
5.2.2 PID 控制 .....	27
第六章 开发工具及调试说明 .....	28
6.1 单片机型号介绍 .....	28
6.2 单片机开发工具 .....	28
第七章 总结与展望 .....	32
7.1 车模的具体参数 .....	32
7.2 比赛总结 .....	32
7.3 未来展望 .....	33
参考文献 .....	1

---

附录 A 主要代码.....	1
灰度图二值化代码 .....	1
提取信标 .....	3
运动方向外环 .....	6
运动方向环内环 .....	8
附录 B 电路原理图.....	10
控制板原理图 .....	10
电源板原理图 .....	11



## 第一章 概述

全国大学生智能汽车竞赛是以智能汽车为研究对象的创意性科技竞赛，面向全国大学生的一种具有探索性的工程实践活动，是教育部倡导的科技竞赛之一。竞赛以“立足培养、重在参与、鼓励探索、追求卓越”为指导思想，旨在促进高等学校素质教育。

它以设计制作在特定赛道上能自主行驶且具有优越性能的智能模型汽车这类复杂工程问题为任务，鼓励大学生组成团队，综合运用多学科知识，提出、分析、设计、开发并研究智能汽车的机械结构、电子线路、运动控制和开发与调试工具等问题，激发大学生从事工程技术开发和科学研究探索的兴趣和潜能，倡导理论联系实际、求真务实的学风和团队协作的人文精神。

无线节能信标组为第十六届智能车竞赛的新组别。其主要任务即是往届信标组与节能组的目标：车模运行的能源来自于无线接收线圈感应电流提供的电能，充电完成后进行寻灯和灭灯。与以往不同的是，本届智能车比赛不需要避障，只需要小车运行到灯罩上方即可。车模的比赛成绩由充电时间加上灭灯的时间来确定。

为了尽可能地减少比赛时间，就需要对比赛的规则与任务进行分析。在节能组中，考察的是对无线电能有效的接受和利用，并鼓励设计良好的机械结构以减小机械能力损耗。因此，“开源节流”成为了节能组一个很重要的目标。若对目标细化，便可以分成三个子目标，即实现轻便灵活的车模制作，实现快速无线充电，控制算法实现高效能源利用及高速运行。这三个方面对应了机械，电路，控制，分别由组内的三位同学合作完成。

由于本届比赛中有灯罩，相当于往届比赛中的坡道。若要实现车模的稳定运行，需要控制算法与机械结构相互配合。机械结构决定了车模的性能上限，而控制算法决定了车模能够发挥出的性能。这一方面将在机械及软件两章进行论述。

对于无线充电部分，根据往届的数据与资料，有多种方案可以选择。为了尽可能地提高充电效率并实现一定的补电能力，提高系统的灵活性与鲁棒性，我们使用了“LCC”恒流充电的方案，实现了高效充电和补电。这一部分将在无线充电一章进行阐述。

对于控制部分，我们对小车进行了初步数学建模，设计了控制系统的结构。实现小车的快速运行，这一部分将在软件及调试两章进行论述。

根据上述内容，将技术报告共分为三个部分。首先，报告在第一章对比赛的背景与报告结构进行介绍。之后，报告将分为机械、无线充电、电路，软件四部分分别进行论述。之后，在第七章，报告将对智能车的开发工具，调试所遇到的问题进行说明。最后在第八章，报告将对此次比赛进行总结。



## 第二章 智能车机械设计

机械结构作为智能汽车的实现基础之一，起着系统平台的作用。一个稳定、结实、精确、合理的机械结构能在一定程度上简化控制算法。我们对机械结构的要求是：可靠、稳定、轻便。

### 2.1 车模整体结构

为了减少能耗，减轻重量，车模只使用两个直流电机作为执行元件，再加上一个万向轮作为从动轮，构成三轮车的结构。出于稳定性要求，为了车模能在加速时不发生从动轮抬起的翘头现象，车模的重心设计在靠近两个主动轮和一个从动轮构成的三角形的形心处，并且尽量接近地面。与此同时，由于信标灯灯罩突起，车模底盘不能设计的过低，否则会出现车模停在信标灯上主动轮悬空失控的情况。

车模整体结构以两根碳杆为骨架，辅以 3D 打印的连接件加以固定，并作为电路板、传感器和电机的安装平台。这样的结构强度高，重量极轻，零件全部用来受力和承重，几乎没有冗余的零件和结构。

车模整体结构如图所示。

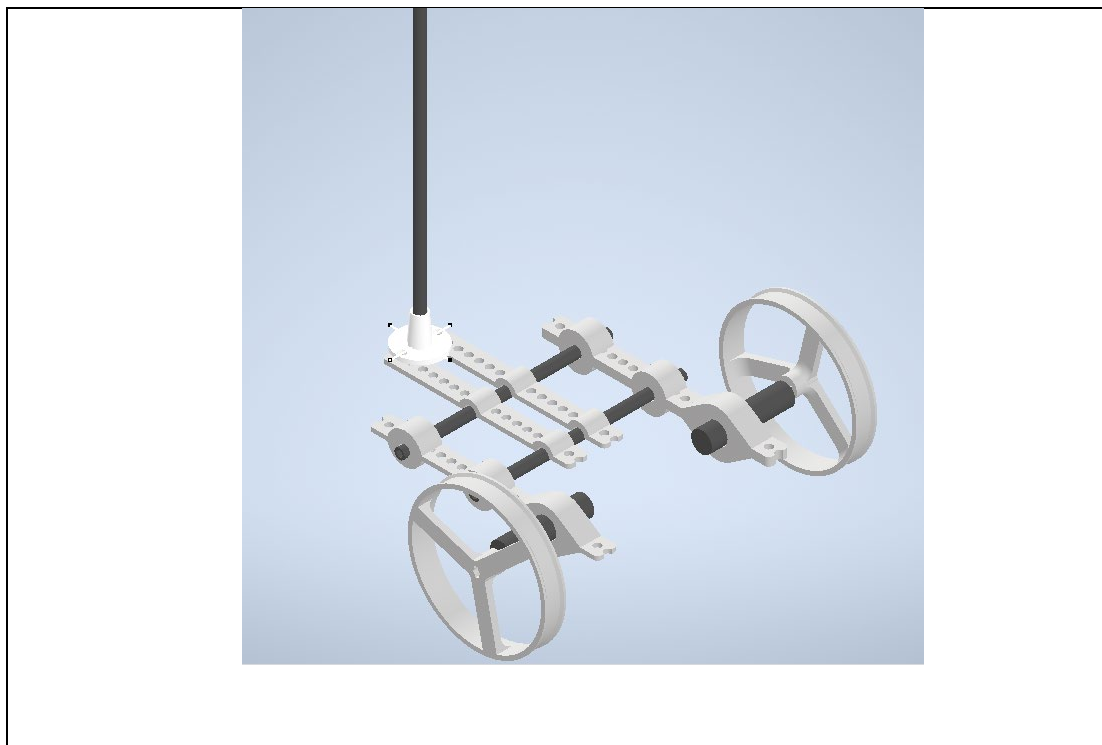


图 2.1 车模整体结构建模

## 2.2 底板设计

底盘使用 3D 打印制作，实际上就是连接两根主要承重的碳杆的连接件。其上设计了许多固定孔，用来固定电路版和传感器。

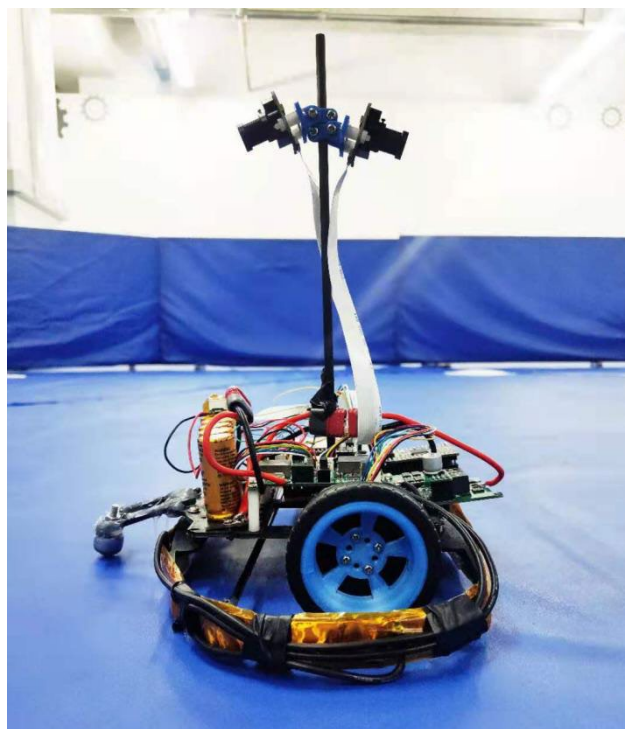
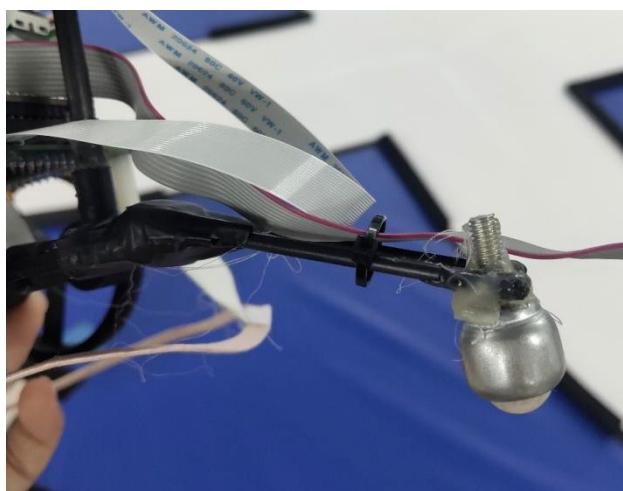


图 2.2 车模实物

## 2.3 万向轮

图片 2.3 万向轮



万向轮用碳杆和热熔胶固定在车的前部，采用了最轻的结构来减少车重。

## 2.4 轮毂设计

轮毂采用 3D 打印制作，设计成较窄并且镂空部分较大，这样能减小轮子的转动惯量，为车模提供更好的动态性能，并减轻车重。轮胎使用的是 C 车轮胎，经裁剪后，其轮径合适、摩擦系数大。

## 2.5 传动与动力单元

选用海太机电公司的直流电机 HT-S-3505 及配套减速箱，其转矩和转速合适、效率高。主动轮的轮距约 160mm。

## 2.6 传感器安装

两个 MT9V034 摄像头用一根碳杆固定在底盘上并和底盘保持约 150mm 的高度，二者镜头朝向一前一后。

陀螺仪安装在底盘上，靠近重心的位置，这样可以避免引入正反馈。

## 2.7 线圈安装

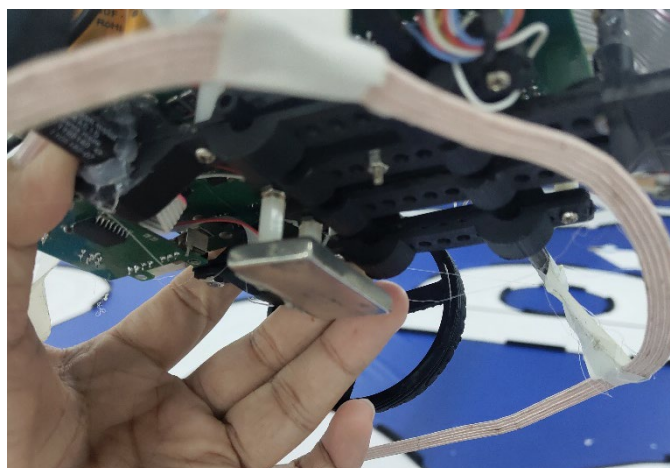
线圈采用多股纱包线绕制，直径 200mm，绕制 4 圈。接收线圈与信标灯的发送线圈直径近似，能够减少漏磁，提高效率。线圈通过线和胶带悬吊在底盘下，这样能尽量增大接收效率，同时又不会影响车模行驶。

## 2.8 电路板和超级电容组安装

电路板通过尼龙柱安装在底盘上，尼龙柱把电路版稍微架高，可以避免涡流对电路的影响。超级电容组用胶枪固定在电机上方，可以调整重心使其接近主动轮，增大正压力从而增大轮胎摩擦力。

## 2.9 磁标安装

磁标采用长方形的钕磁铁，其磁性大，能够良好触发霍尔传感器。磁标通过尼龙柱安装在底盘下方，其安装高度不宜过低或过高。过低容易在车模行驶过程中与信标灯接触，造成车模主动轮悬空；过高容易导致无法触发信标灯上的霍尔传感器。最终经试验，磁标安装在距地面 10mm 左右时效果较好。



图片 2.4 磁标安装

## 第三章 无线充电

### 3.1 无线充电资料总结

在卓晴老师公众号(TsinghuaJoking)中关于无线节能组的相关推文如下，在这些文章中介绍了无线充电的结构与设计，可供同学参考。

《无线信标功能初步测试》

《火中取栗》

《再谈火中取栗》

《无线信标使用说明》

《节能车模》

《如何把大象装进冰箱》

《无线充电》

《电子负载》

《智能永动车》

## 3.2 原理

无线输电（wireless power transfer, WPT）最早由物理学家尼古拉·特斯拉提出，其主要依据是电磁感应原理。在十六届无线节能组中，主要有以下几种拓扑结构

1. LCC-LCC 式。这种方式指的是原边和副边采用 LCC 结构，这种结构的电路简单，可以直接对电容充电，接受效率高于 LCCS 方式，并且抗偏移特性好。

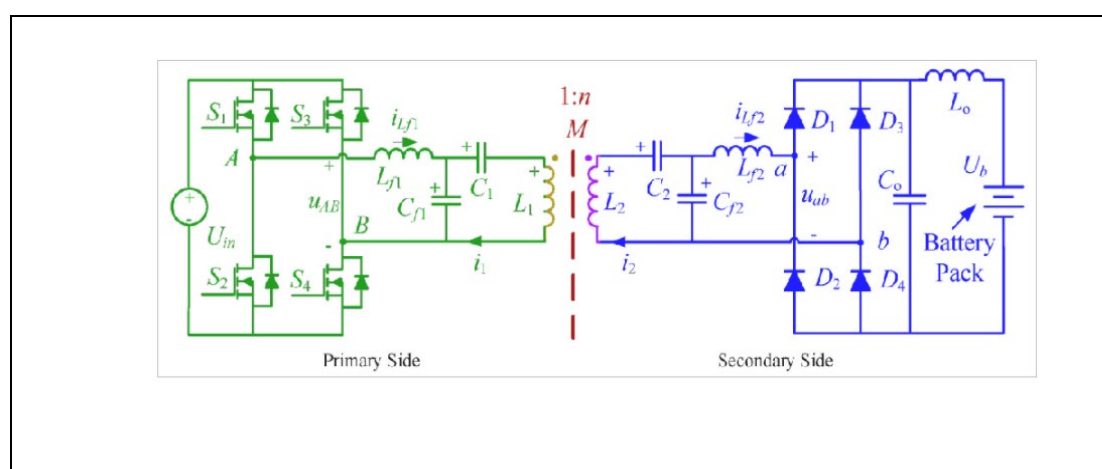


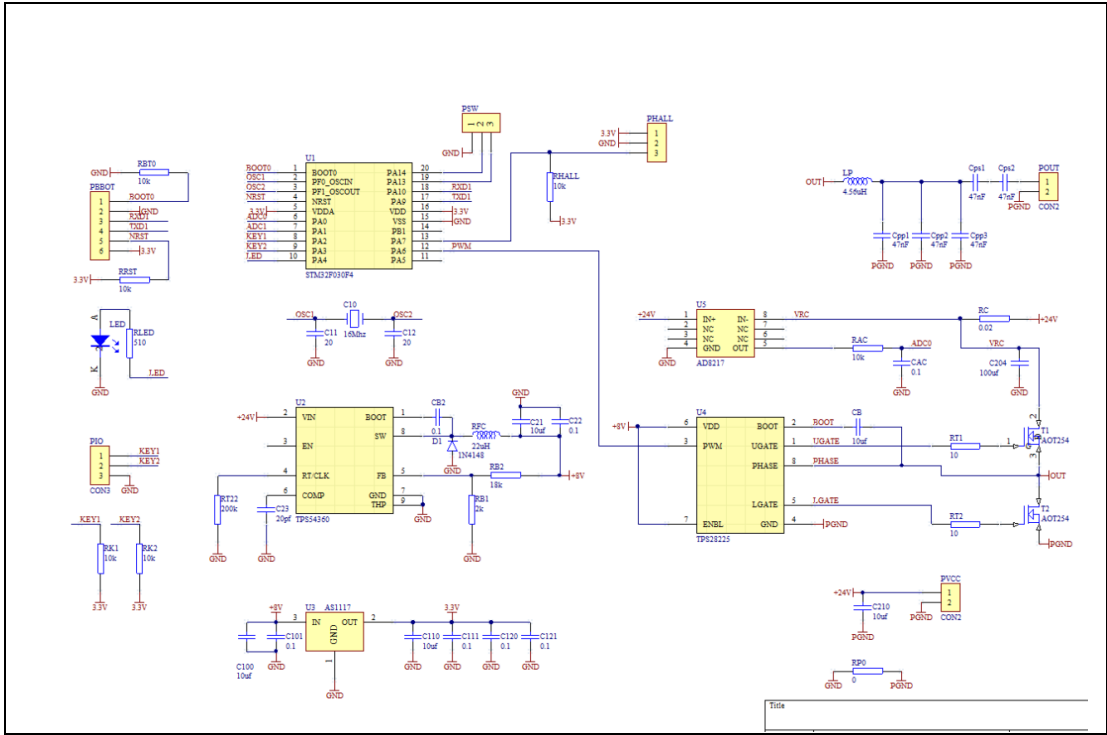
图 3.1 LCC-LCC 式拓扑

2. LCCS 方式指的是原边采用 LCC 结构，副边采用电感和谐振电容串联。这种方式结构更加简单，损耗低于 LCC-LCC 方式，但是作为傻充，效率低于后者，并且抗偏移特性明显不如前者。

## 3.3 发射器

### 3.3.1 基本电路

在第十六届智能车比赛中，发射器使用官方提供的发射器，不可更换。比赛所用的发射器为 E 类高频谐振功率放大器。基本电路原理及原理图如下<sup>[3]</sup>：





射器线圈中产生一个相对稳定的磁场，并且由于没有超功率罚时，所以可以直接使用傻充方案充电。傻充方案分为 LCCS 拓扑和 LCC-LCC 拓扑，前者原理图如下：

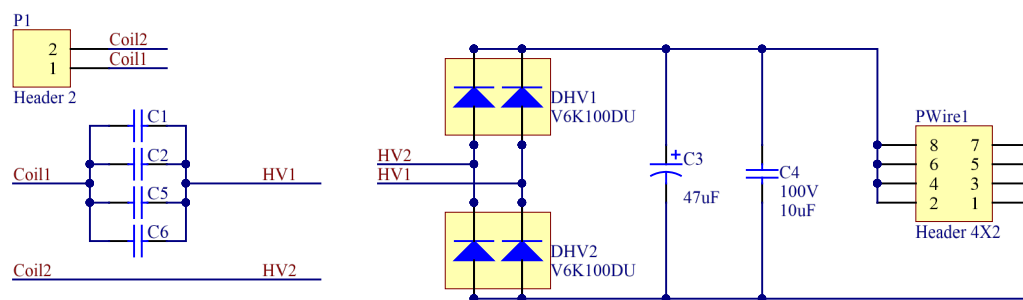


图 3.3 LCCS 傻冲方案原理图

后者原理图如下所示：

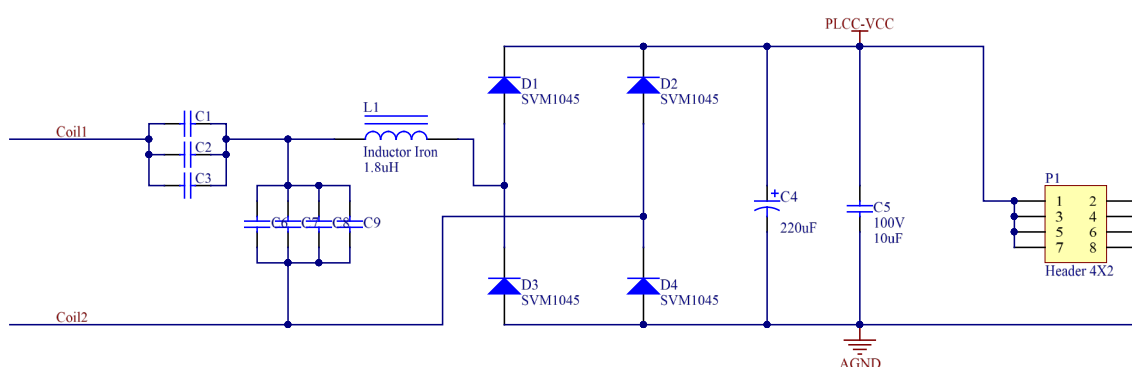


图 3.4 LCC-LCC 傻冲方案原理图

LCC 网络参数可以根据车友“火星大王”写的小程序计算得出。小程序链接如下：[http://www.juruoyun.top/jry\\_wb/jry\\_wb\\_small\\_application/index.php#sc\\_lcc](http://www.juruoyun.top/jry_wb/jry_wb_small_application/index.php#sc_lcc)

小程序页面外形如下所示：

返回蒟蒻云

小程序

dij模拟器	输出电流I0(A)	5
LCC反向计算	输入电压U0(V)	10
LCC计算器	接收线圈电感L0(μH)	13.79
位模式计算器	工作频率f0(KHZ)	150
信息查看	磁环电感系数AL(nH/N^2)	11
测长仪器	LCC基本电抗X0(Ω)	2.0000
测长仪器2	输出电感Ls(μH)	2.1221
	并联电容Cp(nF)	530.5165
	串联电容Cs(nF)	96.4859
	输出电感匝数N(仅参考，以实际测量为准)	13.8894

图 3.5 LCC 计算器页面

输入电压是接受线圈放置在发射线圈上的空载电压有效值，该有效值建议采用示波器测量，原因参考卓晴老师推文

<https://zhuoqing.blog.csdn.net/article/details/104097903> 都是高频惹的祸。

输出电流由用户决定，电感值使用 LCR 表测量

3.4.2 线圈制作

线圈可以使用漆包线、利兹线、铜箔等材料绕制。经过实验对比，漆包线和利兹线的接收效率相差不大，线圈电感量可以通过下式计算<sup>[4]</sup>，也可以通过专业仪器测得：

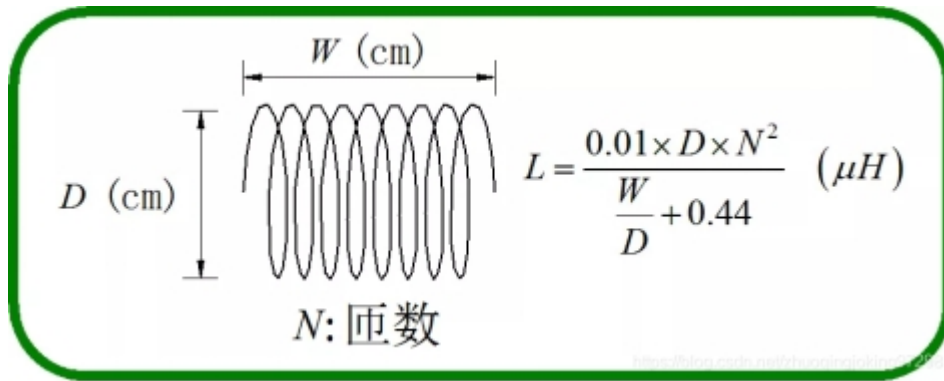


图 3.6 电感计算

### 3.4.3 谐振电容的选择

谐振电容使用 C0G (COG) 电容。这种电容温漂小，耐压高。在选择时，优先选用 ESR 较小的电容。

### 3.4.4 谐振电感

LCC 网络中存在串联电感，材料为铁氧体材料，使用利兹线绕制，并搭配 LCR 表测量。选型时注意材料性能，尽量减小损耗并不要让其磁饱和。这里推荐龙秋

### 3.4.5 整流

全桥整流效率低于半波整流[]，所以采用全桥整流网络。同步整流在发射器频率较低时理论上效率高于全波整流，但是同步整流控制器所需要的电压值很高，会带来安全隐患和 EMI 问题，所以不予考虑，肖特基二极管使用耐压值高，压降低的肖特基二极管。



## 第四章 硬件电路设计

电路分为功率部分与信号部分。功率部分包括 LCC 网络整流板（已经在上一章介绍），充电板，电源板，电机驱动板。信号部分由编码器板，主控板组成。

### 4.1 电源管理板

电源板直接与超级电容相连，为电机，主控和传感器提供电源。同时监测超级电容充电电流和电容电压，输出采样信号。

电源管理供电结构如下所示：

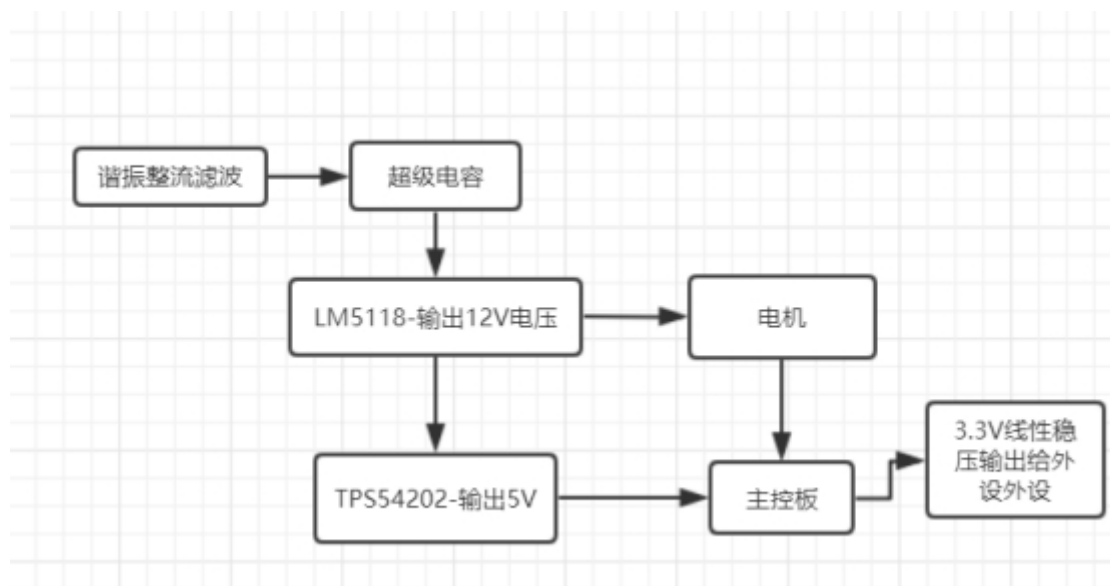


图 4.1 电源结构

LCC 傻充方案在空载时的输出电压极高，使用电子负载测量约为 42V 左右。当然，为了避免 LCC 空载，可以在电容充满电后马上离开发射器。而为了使小车能够适应不同的灯数，我们使用了耐压值为 16V 的超级电容，通过调整充电

阈值电压来灭不同的灯数。所以我们使用了耐压值为 75V 的 LM5118 升降压控制器从电容取电，这样可以承受 LCC 空载的电压，保证了电路安全。

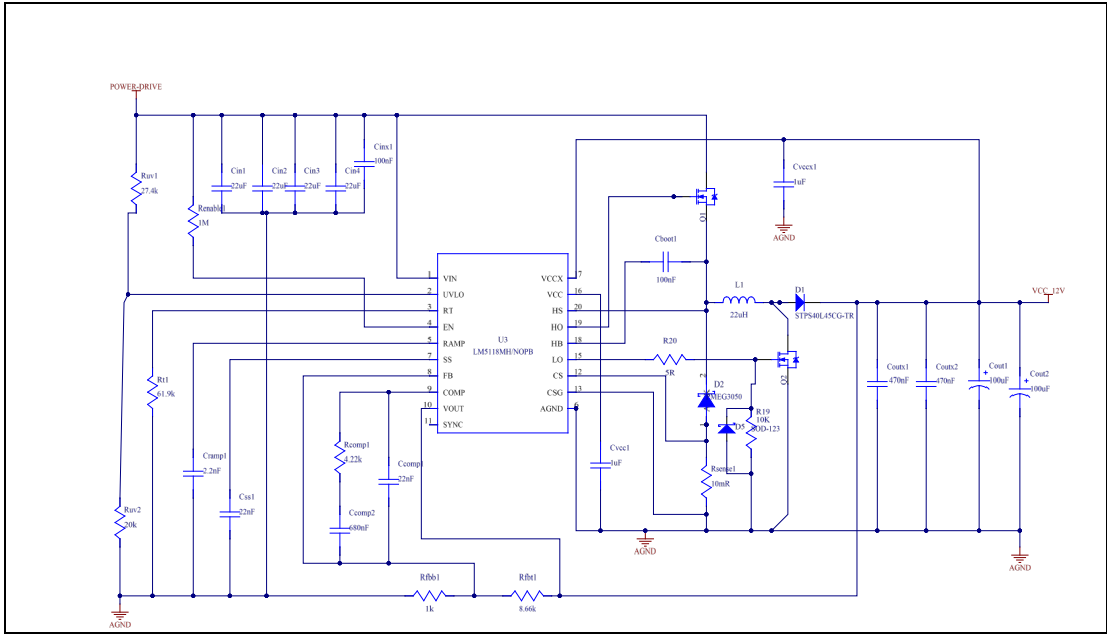
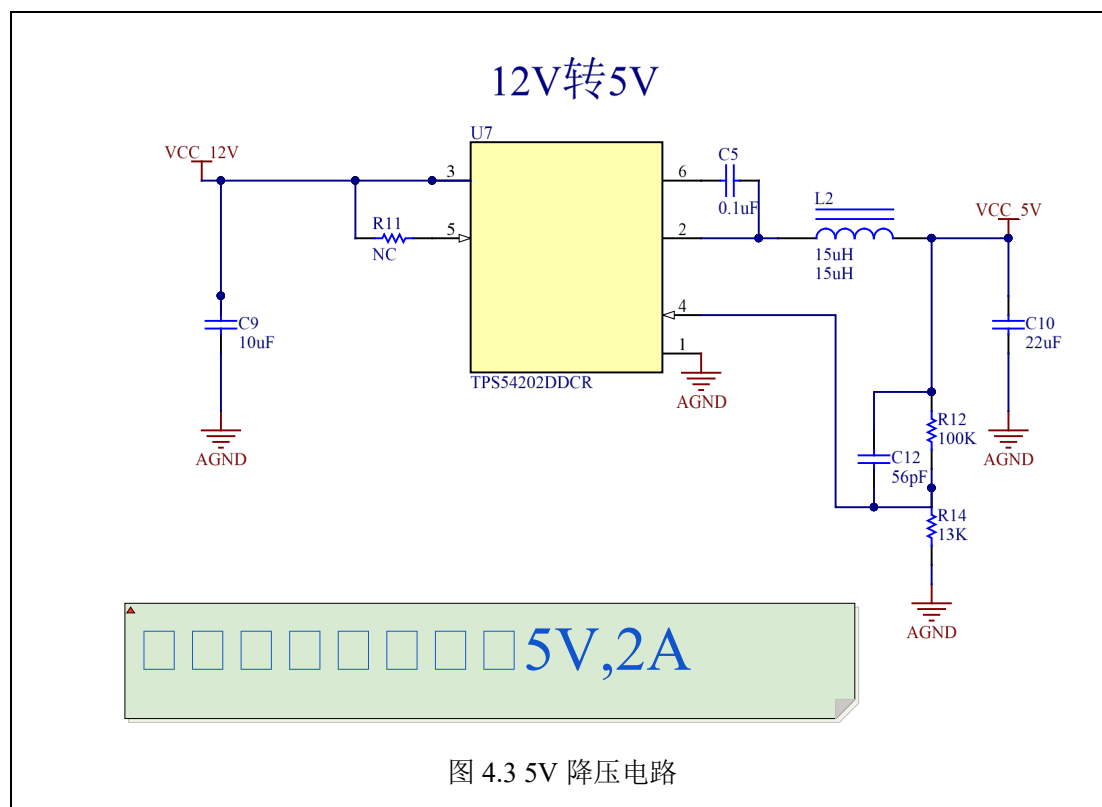


图 4.2 12V 升降压电路

使用 TPS54202 作为辅助电源为主控板和采样运放供电。

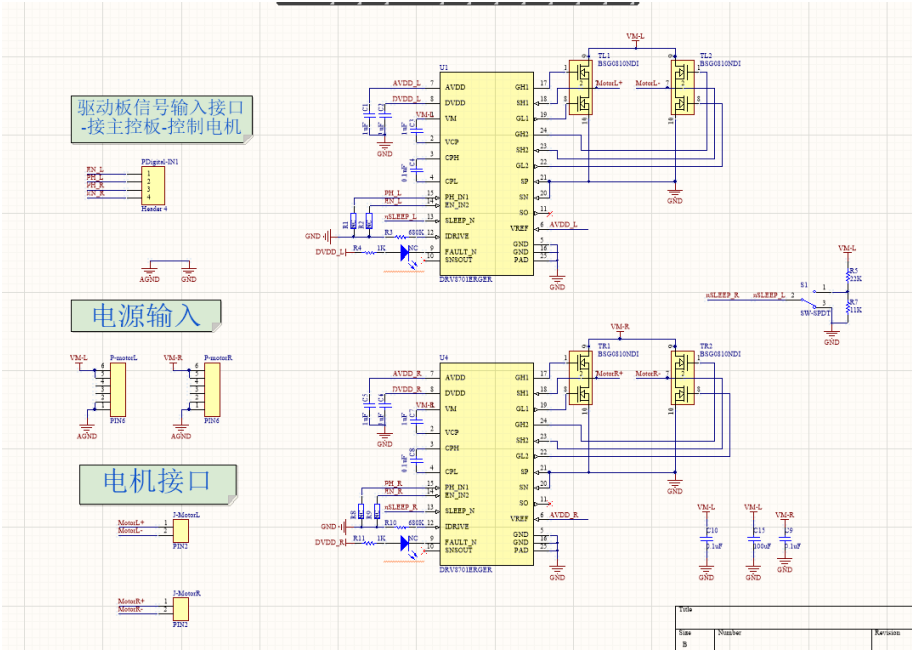


## 4.2 驱动板

比赛使用的电机内阻约为 33 欧，和 MOS 管导通内阻相比后者可以忽略不计。

驱动芯片使用 TI 公司型号为 DRV8701 驱动芯片，该芯片可以实现 H 桥中 4 个 MOS 管的控制，使用时最低工作电压可低至 5.5V，内部自带了自举升压电路，可提供驱动 MOS 管所需的电压。由于电机的功率很低，所以不做隔离措施。

图 4.4 驱动板电路



4.3 编码器板

由于使用了 AB 相输出的光电编码器，为了方便 MCU 处理速度信号，对 AB 相信号简单运算，输出信号 PUL，DIR。

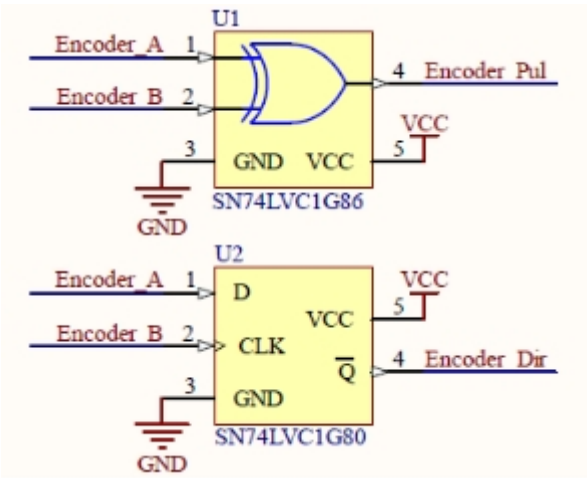


图 4.5 编码器板电路



## 4.4 控制板

控制板可分为电源管理、用户交互、传感器接口、主控，这几个部分。

### 4.4.1 电源管理

使用 1.1A 保险丝和肖特基二极管搭配，可以实现电流限制和反接保护。（实际上这种方案不如使用 TI 的集成芯片，功能更加强大，保护效果更好）使用 AMS1117 输出 3.3V 为摄像头，陀螺仪和编码器供电。（AMS1117 的功耗和压降都很大，功率密度很低，可以使用更好的芯片）

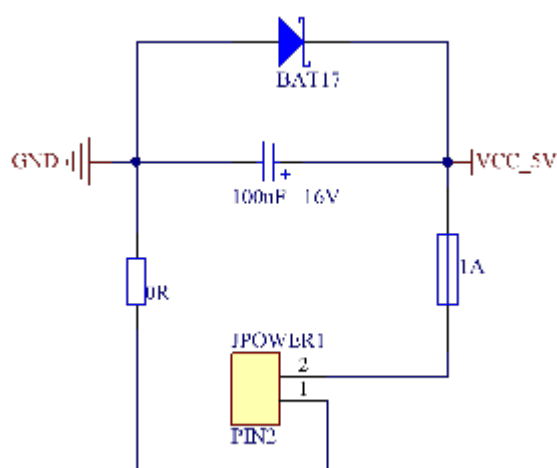


图 4.6 电源保护电路

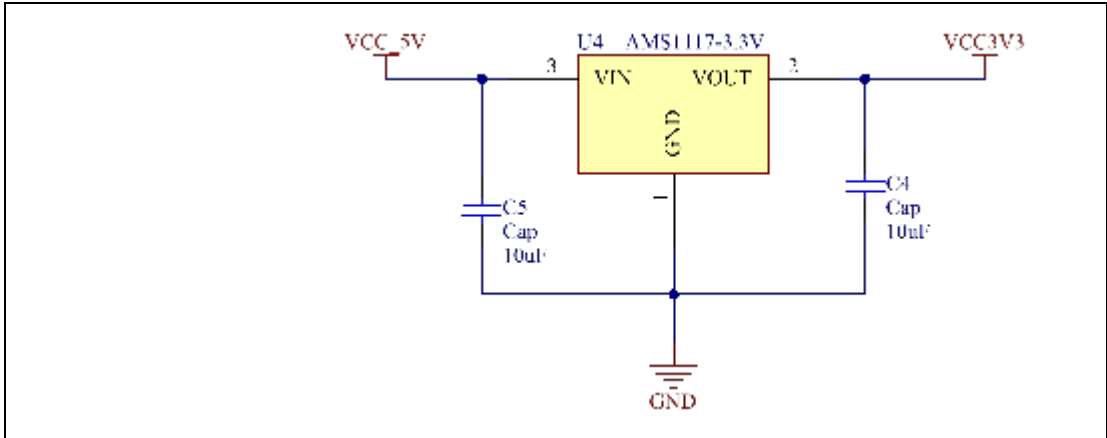


图 4.7 3.3V 电源电路

4.4.2 用户交互

使用拨码开关、TFT 与用户交互。

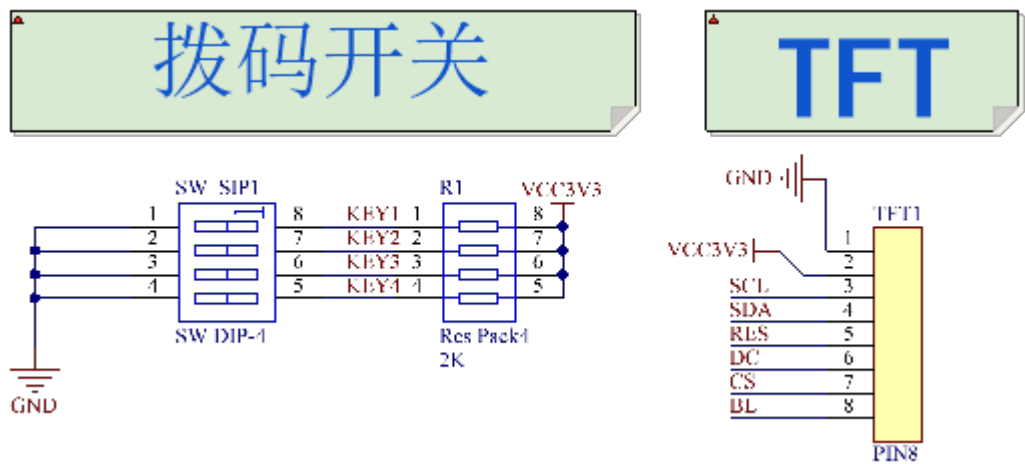


图 4.8 用户接口电路

#### 4.4.3 模拟量采样信号处理

使用单电源供电运放对充电电压和电流进行采样。由于小车在强磁场中，受到干扰严重，所以采用 RC 滤波。

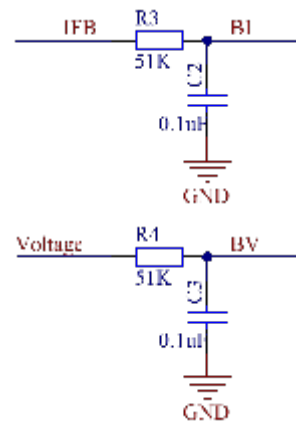


图 4.9 采样信号处理电路

#### 4.4.4 接口

小车运行中需要传递来自传感器的数据并输出控制信号，这些都由接口完成。传感器包括摄像头，获得和信标灯的偏差角，以及 ICM 陀螺仪控制获得小车角加速度。

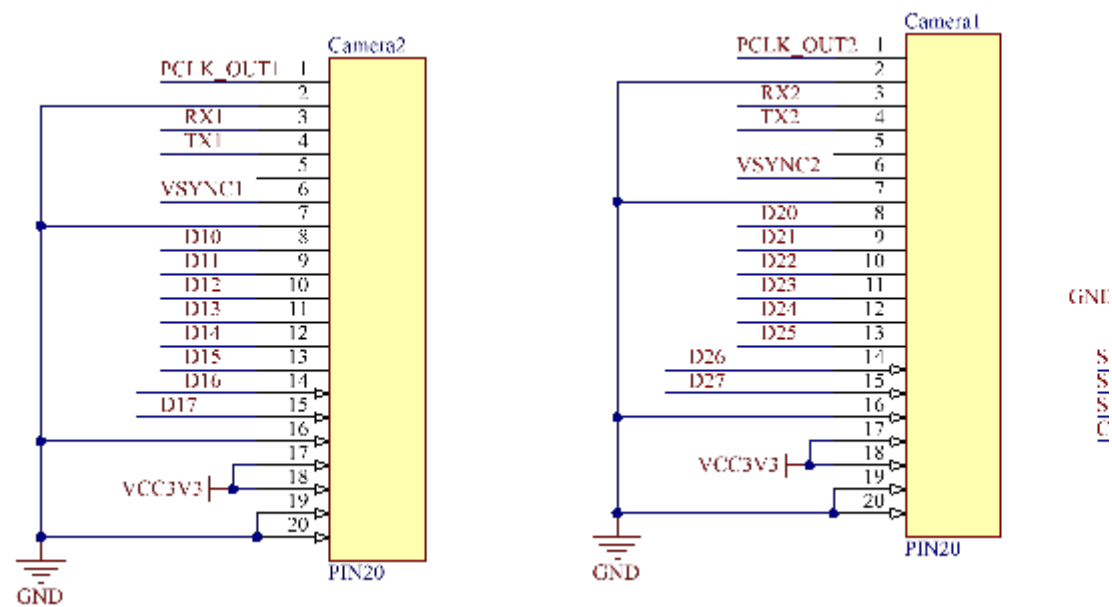


图 4.10 接口电路

#### 4.4.5 单片机主控

使用 TC264 核心板控制车体运行，工作在 200M 主频下。

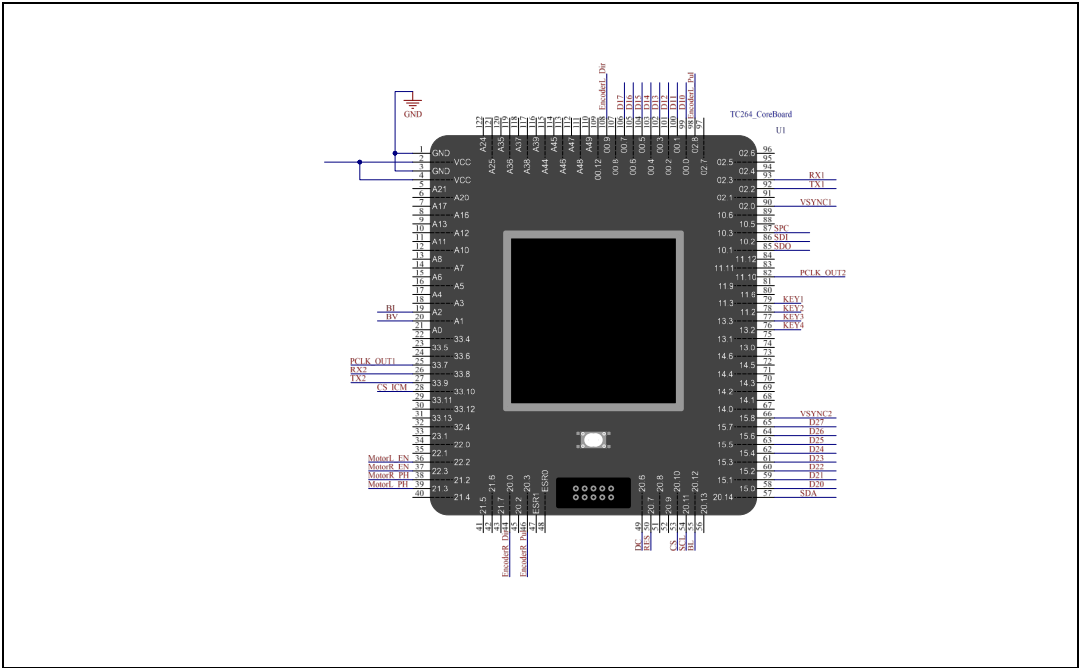


图 4.11 单片机主控电路

## 第五章 软件设计

### 5.1 运动感知

#### 5.1.1 主要思路

单片机使用的是英飞凌 TriCore Family 系列的 TC264，为双核单片机，运算速度快的 CPU1 用于计算量大的摄像头外设。

信标灯导航信号有红外、闪烁的红色以及 150kHz 电磁导引信号。红外信号适用于灰度摄像头，由于常亮，便于检测，故导航信号选择红外信号，传感器为灰度摄像头。

运动感知的任务为发现信标灯并且计算偏差角度，便于后续运动控制。

寻找信标灯大致分为两步，将灰度图二值化、对二值化后的图像提取信标灯。

#### 5.1.2 灰度图二值化

方法一：平均滤波

思路：所有像素点累加平均

优点：算法简单，计算速度快，适用于高斯噪声

缺点：降低噪声同时使图像产生模糊，特别是物体边缘和细节部分

方法二：动态阈值

思路：遍历每一个灰度值，找出灰度直方图峰值所对应的灰度值

优点：比赛场地变化带来的影响小，鲁棒性高

缺点：更适合于赛道组别而非信标组，计算量大

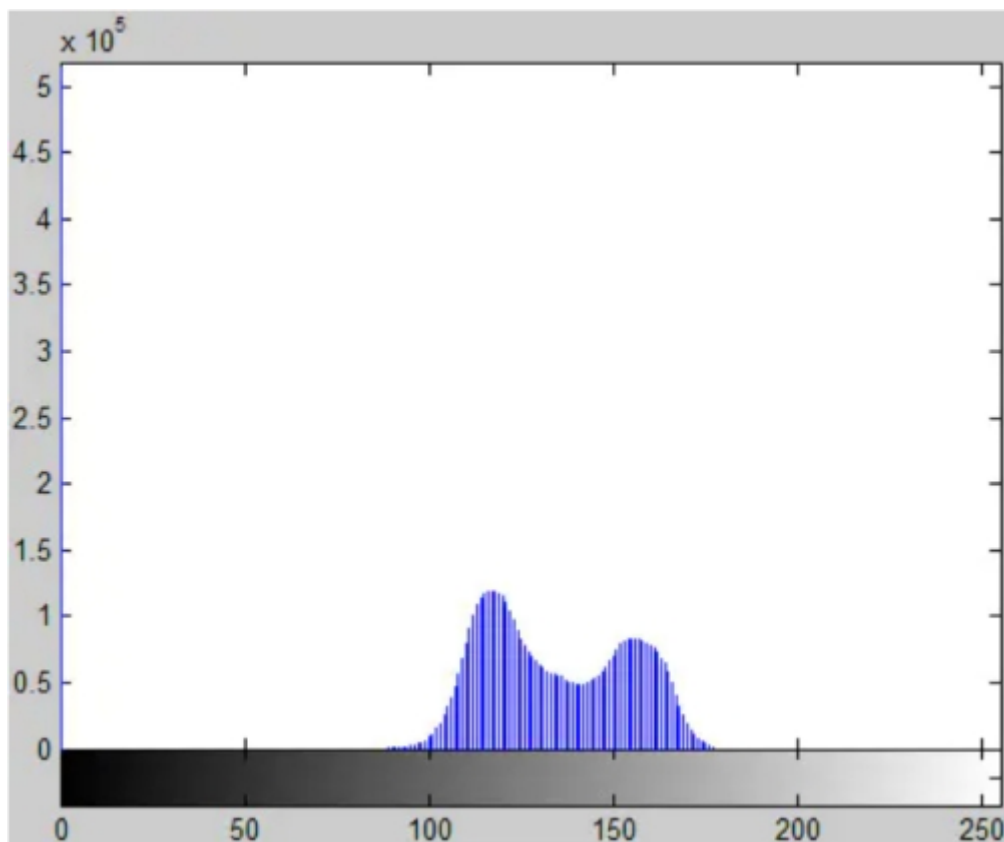


图 5.1 动态阈值

### 5.1.3 提取信标灯

方法一：遍历法

思路：遍历整个二值化后的数组，找到最左边白色像素点，即为信标灯左边界，找到最右边白色像素点，即为信标灯右边界。

优点：计算简单

缺点：比较依赖二值化的效果

方法二：闪烁识别算法

思路：异或前后两组图像，找出信标灯

优点：简单高效

缺点：适用于红色引导信号，内存要求高

### 方法三：寻找最大连通域

思路：二值化后，由于信标灯为面积最大的一块白色区域，噪声干扰则是面积较小的白色区域，故选用数据结构——并查集、算法——TwoPass 算法。

优点：快速、准确

缺点：算法实现有一点难度

### 方法四：形状匹配（好用）

思路：借鉴霍夫变换的思想，由于信标灯是椭圆形，场地灯管长条形，地面反光在上一步已滤除，故找出图像中每个白色连通域的水平距离和竖直距离，分别作为椭圆长轴短轴，通过椭圆面积公式计算出测量面积  $S_1$ ，通过遍历该连通域像素点个数得到实际面积  $S_2$ ，两者越接近则认为是椭圆形信标灯的概率越高。

优点：简单好用

缺点：如果二值化后出现单像素点连通域则形成误判；需要浮点运算，对于无 FPU 模块的单片机运算较慢。

## 5.2 运动控制

### 5.2.1 主要思路

由于 CPU0 计算速度较慢但内存大，故使用 CPU0 做控制。为了节能，不需要舵机，使用两轮控制。

使用编码器反馈速度，定时中断 PID 来调速。



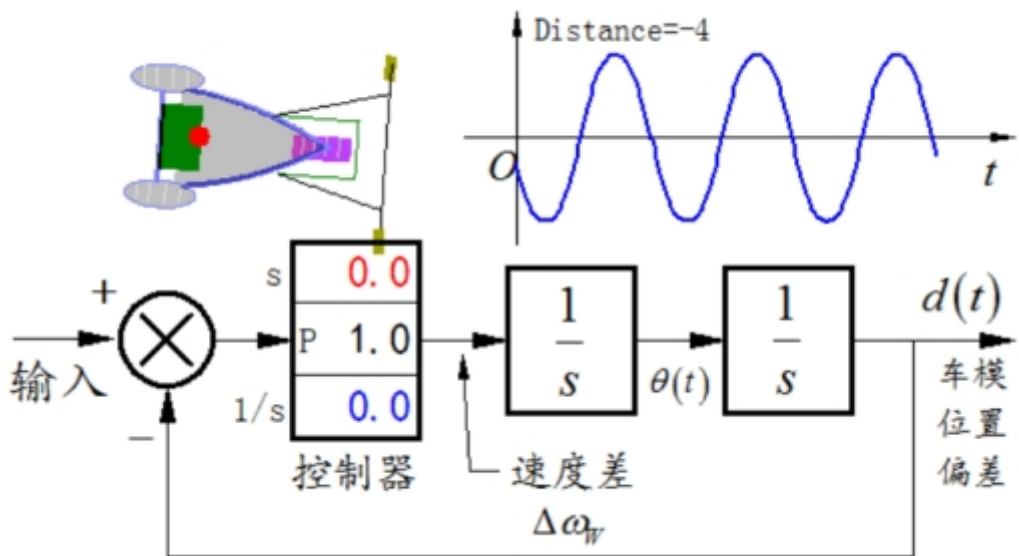


图 5.2 运动控制

### 5.2.2 PID 控制

PID (Proportional Integral Derivative) 控制是最早发展起来的控制策略之一，由于其算法简单，鲁棒性好和可靠性高，被广泛应用于过程控制。任何闭环系统首要任务就是要稳定、快速、准确的响应命令。PID 调整主要工作就是如何实现这一任务。

增大比例系数  $P$  将加快系统响应，它的作用于输出值较快，但不能很好的稳定在一个理想的数值。过大的比例环节会使得系统有比较大的超调，甚至产生震荡。

积分环节可以在比例环节基础上消除稳态误差。它可以对于稳定后的系统进行稳态误差积累，但过大的比例会导致超调过大。

微分环节具有超前作用，有利于提高系统动态指标，使超调减小。但过大的微分环节会导致调节时间过长。

## 第六章 开发工具及调试说明

### 6.1 单片机型号介绍

节能信标车总共使用了两个 TC 的单片机，型号分别为 TC264 和 TC211。分别为双核和单核单片机。该款单片机提供了丰富的数字外设（串口、定时器、PCA、PWM 以及 I2C、SPI）接口与模拟外设（超高速 ADC、比较器），可满足不同的应用需求。

### 6.2 单片机开发工具

单片机程序编译软件使用 ADS（AURIX Development Studio）。该软件提供 TC 系类兼容单片机 C 语言开发系统。ADS 提供了包括 C 编译器、宏汇编、连接器、库管理和一个功能强大的仿真调试器等在内的完整开发方案，通过一个集成开发环境将这些部分组合在一起。

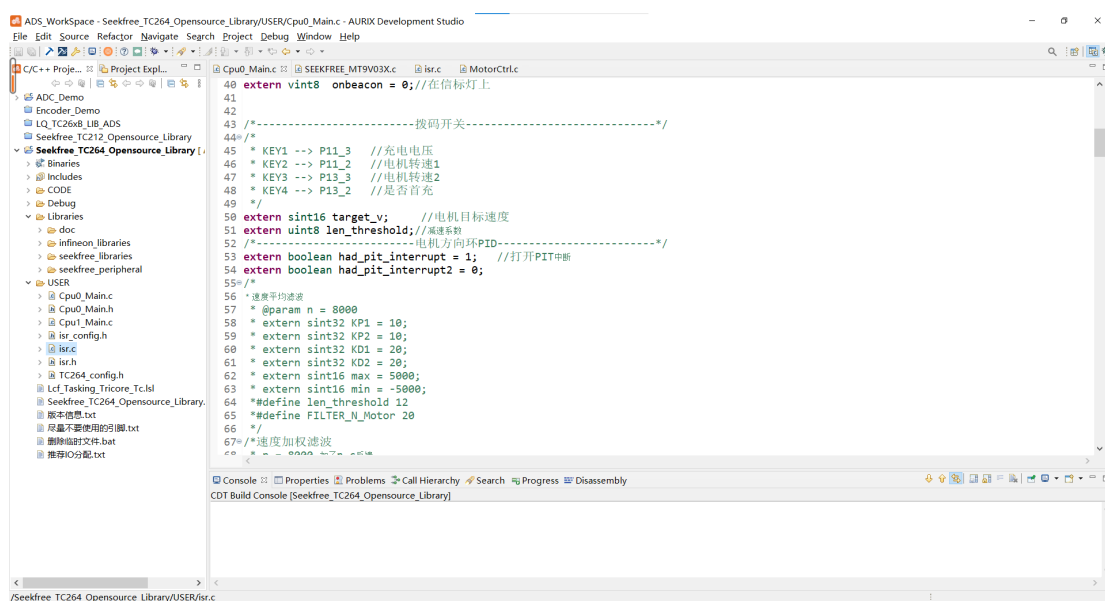


图 6.1 ADS 集成开发环境界面

由于 ADS 有许多的开源库，开发者可以避开最底层的单片机开发，使用 C 语言的知识进行开发，通过调用库函数，查阅单片机用户手册以及相应的例程进行模仿学习，然后再尝试新的功能，达到创新的目的。

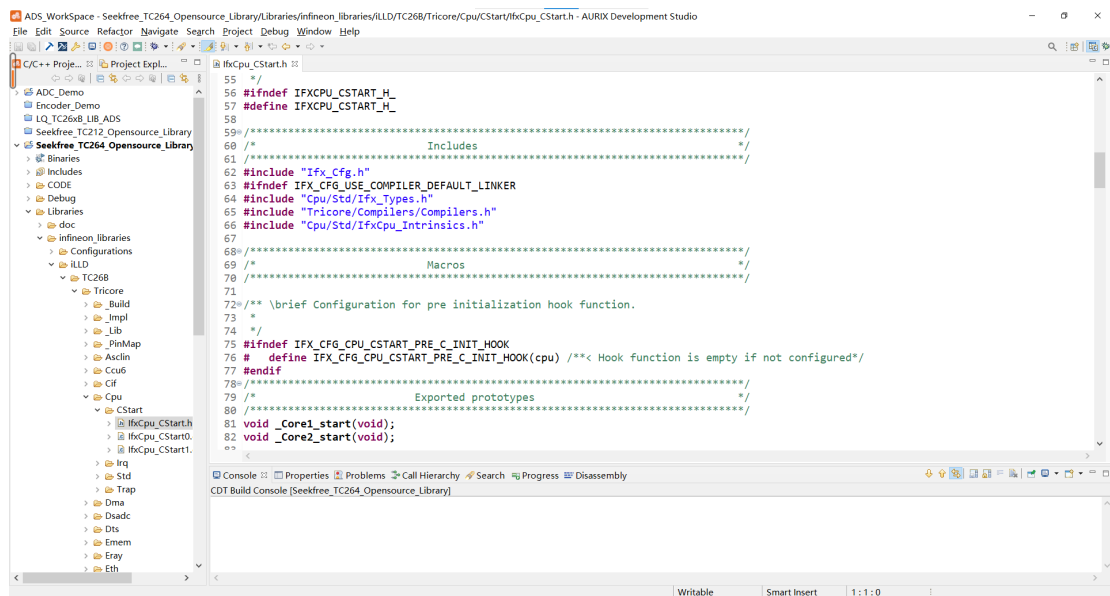


图 6.2 ADS 底层库示例

程序烧录软件使用英飞凌公司提供的专门的烧录软件——Infineon Memtool。该软件通过串口对单片机进行程序烧录，同时提供单片机的配置、调试和加密功能。

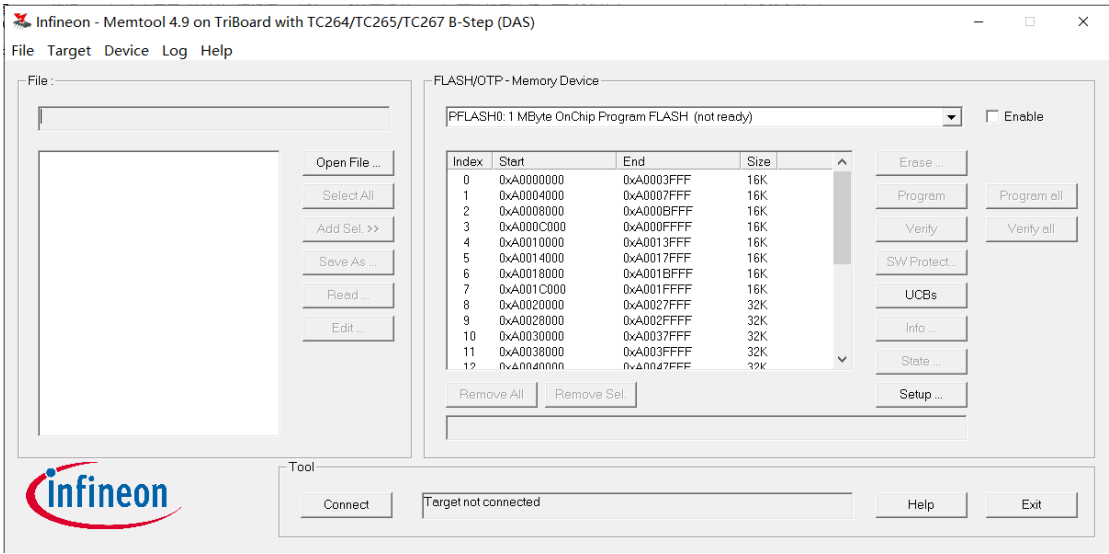


图 6.3 Memtool 程序烧录软件界面

在节能信标车的代码调试和调参过程中，我们使用了串口调试助手和串口绘图软件来监控代码的调试信息和各个变量的值变换。串口调试助手使用的软件为 SSCOM，串口绘图软件为 Serial Plot。



图 6.4 SSCOM 串口调试助手界面

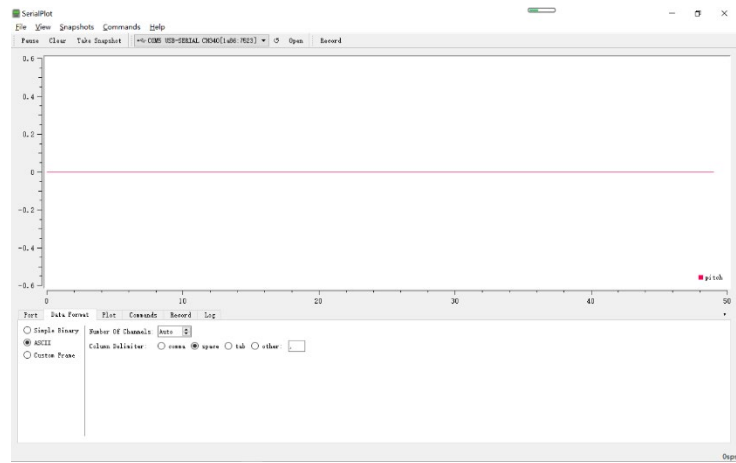


图 6.5 Serial Plot 串口绘图软件界面

## 第七章 总结与展望

### 7.1 车模的具体参数

车模的主要参数如下表所示：

表 7.1

车模类型	自制	
车模尺寸（长、宽、高）	长（厘米）	29
	宽（厘米）	20
	高（厘米）	28
功耗	静止	4w
	正常运行	平均 18w
	上下灯罩	24w
电容容量	10F	
微控制器型号和数量	TC264, TC212	各一个
传感器种类和数量	光电编码器	2
	总钻风摄像头	2
	陀螺仪	1
车模驱动电机型号	海泰机电	
赛道信息检测精度及频率	5ms	

### 7.2 比赛总结

自备赛以来，我们尝试了多种控制算法，期间遇到了很多困难，经过不断的摸索和尝试最终确定了一套比较满意的控制方案，通过了实践和比赛的检验。对于两轮车来说，小车的重心不均匀会影响最终的控制效果，我们大胆尝试了加入陀螺仪使得小车方向环更加可靠，同时小车在方向环性能上也获得了比较好的效果。在控制算法上，速度环和循迹闭环都使用了串级 PID 结构，提高了控制算法的鲁棒性。

对于无线充电来说，由于往届留下的资料少，效果不好。因此我们在备赛过程中，从零摸索无线充电。在此期间尝试了多种方案，走了不少弯路，但是我们也在不断地改进，总结。最后探索出了一套比较好的方案。正因为如此，在技术报告中我们重新梳理了一遍目前无线充电的方案及要点，希望之后的同学可以少走一些弯路。

### 7.3 未来展望

由于备赛时间紧张，在小车的性能发挥上还存在很大的提升空间。我们在比赛过程中小车的速度为固定值，这样并没有发挥小车全部的性能，因此需要做进一步的改进，可以对小车进行识别并记录，然后对不同的距离以不同的速度进行循迹，最大限度的发挥小车的性能。小车目前使用的控制算法都是基于传统的PID 控制算法，控制效果并没有达到最优，未来可以尝试一些其他的控制算法，使小车的性能得到进一步的提升。





## 参考文献

- [1] Texas Instruments.Exploring the evolution and optimization of wireless power transfer[J/OL].www.ti.com,July, 2018.
- [2] 黄程. 磁谐振无线电能传输系统的特性分析及失谐控制策略研究[D].广西大学,2018.
- [3] 卓晴.竹篮打水[J/OL]. <https://blog.csdn.net/zhuoqingjoking97298/article/details/104120711>, 2017-12-31.
- [4] 卓晴.简易无线电能接收方法[J/OL].  
<https://blog.csdn.net/zhuoqingjoking97298/article/details/104120716>, 2020-1-31.
- [5] 卓晴.如何把大象装进冰箱[J/OL].  
<https://blog.csdn.net/zhuoqingjoking97298/article/details/104120669>, 2020-1-31.
- [6] Texas Instrument.Understanding Boost Power Stages in Switchmode Power Supplies[J/OL].www.ti.com,March 1999.



## 附录 A 主要代码

### 灰度图二值化代码

```
uint16 mean_filter(uint16 width,uint16 height)
{
    uint16 Threshold = 0;
    uint32 tv = 0;
    uint8 i,j;

    //累加
    for (i = 0; i < height; i++)
    {
        for (j = 0; j < width; j++)
        {
            tv += mt9v03x_image[i][j];    //累加
        }
    }
    Threshold=(unsigned short)(tv / height / width);    //求平均值,光线越暗越小,全黑约 35,
    对着屏幕约 160, 一般情况下大约 100
    Threshold = Threshold + 30;    //此处阈值设置, 根据环境的光线来设定

    for (i = 0; i < height; i++)
    {
        for (j = 0; j < width; j++)
        {
            if(mt9v03x_image[i][j] > Threshold)//数值越大, 显示的内容越多, 较浅的图像
            也能显示出来
                mt9v03x_bin_image[i][j] = 255;    //255 而不是 1,TFT 显示彩屏
            else
                mt9v03x_bin_image[i][j] = 0;
        }
    }

    return Threshold;    //返回最佳阈值;
}

/*****大津法
*
* 返回值: 二值化阈值: 0~255
*
*/
uint16 GetOSTU (uint16 width,uint16 height)    //二维数组作为参数是否可行??
{
    sint16 i,j;
```

```

uint32 Amount = 0;
uint32 PixelBack = 0;
uint32 PixelshortegralBack = 0;
uint32 Pixelshortegral = 0;
sint32 PixelshortegralFore = 0;
sint32 PixelFore = 0;
float32 OmegaBack, OmegaFore, MicroBack, MicroFore, SigmaB, Sigma; // 类间方差;
sint16 MinValue, MaxValue; //最大灰度和最小灰度
uint16 Threshold = 0;
uint8 HistoGram[256]; //灰度直方图

for (j = 0; j < 256; j++)
    HistoGram[j] = 0; //初始化灰度直方图

for (j = 0; j < height; j++)
{
    for (i = 0; i < width; i++)
    {
        HistoGram[mt9v03x_image[j][i]]++; //统计灰度级中每个像素在整幅图像中的
        个数
    }
}

for (MinValue = 0; MinValue < 256 && HistoGram[MinValue] == 0; MinValue++);
//获取最小灰度的值
for (MaxValue = 255; MaxValue > MinValue && HistoGram[MinValue] == 0; MaxValue--);
//获取最大灰度的值

if (MaxValue == MinValue)
    return MaxValue; // 图像中只有一个颜色
if (MinValue + 1 == MaxValue)
    return MinValue; // 图像中只有二个颜色

for (j = MinValue; j <= MaxValue; j++)
    Amount += HistoGram[j]; // 像素总数

Pixelshortegral = 0;
for (j = MinValue; j <= MaxValue; j++)
{
    Pixelshortegral += HistoGram[j] * j; //灰度值总数
}
SigmaB = -1;
for (j = MinValue; j < MaxValue; j++)
{
    PixelBack = PixelBack + HistoGram[j]; //前景像素点数
    PixelFore = Amount - PixelBack; //背景像素点数
    OmegaBack = (float) PixelBack / Amount; //前景像素百分比
    OmegaFore = (float) PixelFore / Amount; //背景像素百分比
    PixelshortegralBack += HistoGram[j] * j; //前景灰度值
    PixelshortegralFore = Pixelshortegral - PixelshortegralBack; //背景灰度值
    MicroBack = (float) PixelshortegralBack / PixelBack; //前景灰度百分比
    MicroFore = (float) PixelshortegralFore / PixelFore; //背景灰度百分比
}

```

```

        Sigma = OmegaBack * OmegaFore * (MicroBack - MicroFore) * (MicroBack -
MicroFore); //计算类间方差
        if (Sigma > SigmaB) //遍历最大的类间方差 g //找出最大类间
方差以及对应的阈值
        {
            SigmaB = Sigma;
            Threshold = j;
        }
    }

    for (i = 0; i < height; i++)
    {
        for (j = 0; j < width; j++)
        {
            if(mt9v03x_image[i][j] > Threshold) //数值越大，显示的内容越多，较浅的图像
也能显示出来
                mt9v03x_bin_image[i][j] = 255; //255 而不是 1,TFT 显示彩屏
            else
                mt9v03x_bin_image[i][j] = 0;
        }
    }

    return Threshold; //返回最佳阈值;
}

```

## 提取信标

```

/*****二值后图像提取信标面积和中心****中心扫描法
*
*
*地址传参： 信标面积      信标中心
*
*
*
*/
void extract_beacon2(uint16 *area,uint16 *mid)
{
    int16 left = MT9V03X_W-1;
    int16 right = 0;
    uint16 S = 0;

    int16 i,j,k;
    uint8 flag1=0;
    uint8 flag2=0;

    for(j=0;j<MT9V03X_H/2;j++)

```

```

{
    if(mt9v03x_bin_image[j][MT9V03X_W/2] == 255)
    {
        flag1=1;break;
    }
}

if(flag1 == 0) //图像不在中间
{
    for(i=MT9V03X_W/2 - 1;i>=0;i--) //逐列扫描，从中间往左
        for(j=0;j<MT9V03X_H/2;j++)
        {
            if(mt9v03x_bin_image[j][i] == 255)
            {
                flag2=1;
                if(mt9v03x_bin_image[j][i+1]==0 && mt9v03x_bin_image[j][i+2]==0
&& mt9v03x_bin_image[j][i+3]==0)
                {
                    uint8 flag3 = 0;
                    uint8 L = (0>(i-3))?0:(i-3);
                    for(k=i-1;k>=L;k--)
                        if(mt9v03x_bin_image[j][k] == 0){flag3=1;break;};
                    if(!flag3)
                    {
                        right = (i>right?i:right); //更新右界
                        break; //下一列
                    }
                }
            }
            else if(mt9v03x_bin_image[j][i+1]==255 &&
mt9v03x_bin_image[j][i+2]==255 && mt9v03x_bin_image[j][i+3]==255)
            {
                uint8 flag3 = 0;
                uint8 L = (0>(i-3))?0:(i-3);
                for(k=i-1;k>=L;k--)
                    if(mt9v03x_bin_image[j][k] == 255){flag3=1;break;};
                if(!flag3)
                {
                    left = (i<left)?i:left; //更新左界
                    break; //下一列
                }
            }
        }
    }
}

if(!flag2)
{
    for(i=MT9V03X_W/2 - 1;i<MT9V03X_W;i++) //逐列扫描，从中间往右
        for(j=0;j<MT9V03X_H/2;j++)
        {

```

```

        if(mt9v03x_bin_image[j][i] == 255)
        {
            if(mt9v03x_bin_image[j][i-1]==0 && mt9v03x_bin_image[j][i-2]==0
&& mt9v03x_bin_image[j][i-3]==0)
            {
                uint8 flag3 = 0;
                uint8 R = ((MT9V03X_W-1)>(i+3))?(i+3):(MT9V03X_W-1);
                for(k=i+1;k<=R;k++)
                    if(mt9v03x_bin_image[j][k]==0){flag3=1;break;};
                if(!flag3)
                {
                    left = (i<left)?i:left;    //更新左界
                    break;    //下一列
                }
            }
            else if(mt9v03x_bin_image[j][i-1]==255 && mt9v03x_bin_image[j][i-2]==255 && mt9v03x_bin_image[j][i-3]==255)
            {
                uint8 flag3 = 0;
                uint8 R = ((MT9V03X_W-1)>(i+3))?(i+3):(MT9V03X_W-1);
                for(k=i+1;k<=R;k++)
                    if(mt9v03x_bin_image[j][k] == 255){flag3=1;break;};
                if(!flag3)
                {
                    right = (i>right)?i:right;    //更新右界
                    break;    //下一列
                }
            }
        }
    }
}
if(flag2)lcd_showchar(0,0,'L');
else lcd_showchar(0,0,'R');
}
// 中间有图像
else
{
    for(i=MT9V03X_W/2;i>=0;i--)
    {
        uint8 flag3=0;
        for(j=0;j<MT9V03X_H/2;j++)
            if(mt9v03x_bin_image[j][i]==255){left=i;flag3=1;break;}
        if(!flag3)break;    //超出左边界
    }
    for(i=MT9V03X_W/2;i<MT9V03X_W;i++)
    {
        uint8 flag3=0;
        for(j=0;j<MT9V03X_H/2;j++)
            if(mt9v03x_bin_image[j][i]==255){right=i;flag3=1;break;}
        if(!flag3)break;    //超出右边界
    }
    lcd_showchar(0,0,'M');
}

```

```

    }

    //计算中心和面积
    if(left<right)*mid = ((uint16)left + (uint16)right)/2;
    else *mid = 0; //出错
    for(i=left;i<=right;i++)
        for(j=0;j<MT9V03X_H/2;j++)
            if(mt9v03x_bin_image[j][i]==255)S++;

    *area = S;
}

```

## 运动方向外环

```

uint8 len_threshold = 0;
sint16 ramp_v = 0;
extern boolean slowdown = 0;
extern uint8 mode_dis = 0;
extern uint8 judgecounter = 0;
void Guiding(sint16 target_v)
{
    sint32 dis_v = 40*(beacon_dis+len_threshold);//速度是距离的函数

    if(camera_change==1)
    {
        target_v = -target_v;
        dis_v = -dis_v;
    }

    if(mode_dis == 1)
    {
        ramp_v = MovingAverageFilter(0);//这是为了更新滤波器的数组方便后续减速
        ramp_v = (sint16)dis_v;
        if(beacon_dis < 50 )
            judgecounter++;
        if (judgecounter > 5)
            mode_dis = 2;
    }

    else if(mode_dis == 2)
        ramp_v = target_v/8;
    else
    {
        ramp_v = MovingAverageFilter(target_v);
        judgecounter=0;
    }
}

```



```

//距离速度模式转换条件判断
if(mode_dis == 0 &&
    ((dis_v - ramp_v < 0 && camera_change==0) ||
    (dis_v - ramp_v > 0 && camera_change==1)))
    mode_dis = 1;

AngularVelocityLoop(ramp_v, beacon_bias); //用角速度做导引
}

//////////滑动平均滤波//////////
#define FILTER_N_Motor 20//30
sint16 MovingAverageFilter(sint16 target_v)
{
    uint8 i;
    static sint32 maFilter_sum = 0;
    static sint16 maFilter_buff[FILTER_N_Motor] = {0};
    static sint16 last_target_v;

    if(last_target_v*target_v<0 || mode_dis==1)
    {
        for(i = 0; i < FILTER_N_Motor ; i++)
            maFilter_buff[i] = 0; // 所有数据清空，防止车后退
        maFilter_sum = 0;
    }

    maFilter_sum -= maFilter_buff[0];
    for(i = 0; i < FILTER_N_Motor-1 ; i++)
        maFilter_buff[i] = maFilter_buff[i + 1]; // 所有数据左移，低位扔掉
    maFilter_buff[FILTER_N_Motor-1] = target_v;
    maFilter_sum += maFilter_buff[FILTER_N_Motor-1];

    last_target_v = target_v;
    return (sint16)(maFilter_sum/(sint16)FILTER_N_Motor);
}

//-----
// @brief      角速度环，电机转速 PD 调节
// @param      n_c          //给定小车质心的速度（未知量纲转速，取值-
//                          1,000,000~+1,000,000，测试建议取值 300,000），即左右轮平均值，正值表示 headflag = 0，
//                          负值 headflag = 1
// @param      bias          //摄像头图像得到的信标灯与中心位置的偏差，左侧为负
//                          数，此值最好归一化
// @return      void
// @note      得到的 n_l,n_r 正值表示向前，负值表示向后转，注意两轮对称放置若输入
//            同一控制字则旋转方向相反

```

```
//          n_l 表示 01 号电机转速， n_r 表示 02 号电机转速
```

```
//-----
```

```
extern sint16 omiga_c = 0;
```

```
void AngularVelocityLoop(sint16 n_c, sint16 bias)
{
    static sint16 last_bias;//角速度环控制里储存上一次偏差
    sint32 KP,KD;
    sint16 max = 39999;
    sint16 min = 0;
    if(n_c > 8100)
    {
        KP = 30;
        KD = 30;
    }
    else if(n_c > 3000)
    {
        KP = 20;
        KD = 20;
    }
    else
    {
        KP = 10;
        KD = 10;
    }
    omiga_c = ((sint16)KP*bias + (sint16)KD*(bias - last_bias));
    //限幅          应该对 omiga_c 限幅          !!!!!
    if(omiga_c > max)omiga_c = max;
    else if(omiga_c < min)omiga_c = min;    //参数可改

    last_bias = bias;
}
```

## 运动方向环内环

```
//变量 omiga_c 为内环输入，由外环赋值
```

```
//陀螺仪速度反馈阻尼回路
```

```
void GyroscopeDampingLoop(sint16 omiga_c)
```

```
{
    get_icm20602_gyro_spi();

    sint16 bias = omiga_c + (sint16)icm_gyro_z;    //负负得正
    static sint16 last_bias = 0;

    sint16 omiga_loop = omiga_c;
    omiga_loop = omiga_c + bias;
```

```
//限幅  
  
//执行  
n_l = ramp_v + omiga_loop;  
n_r = ramp_v - omiga_loop;  
  
last_bias = bias;  
}
```

附录 B 电路原理图

控制板原理图

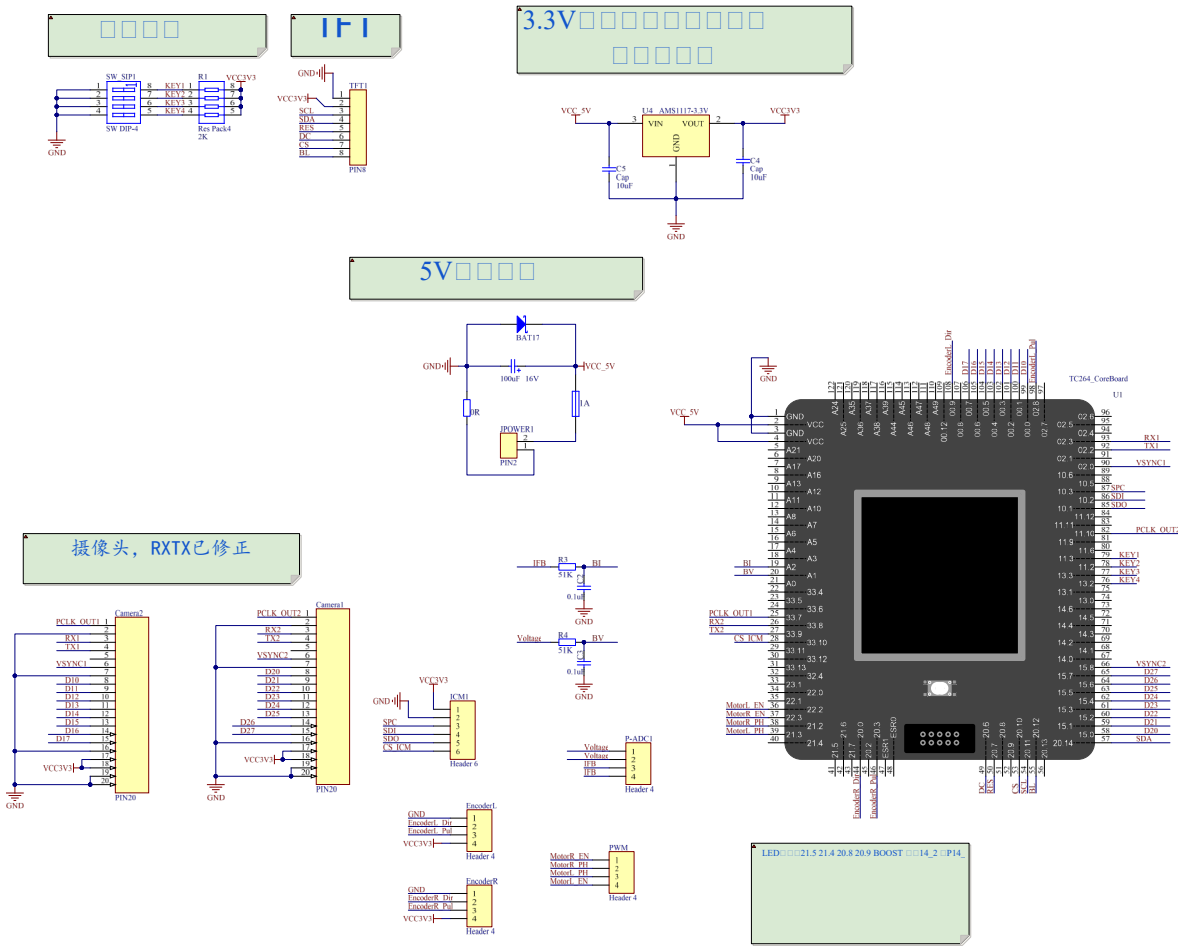


图 B-1 控制板原理图

## 电源板原理图

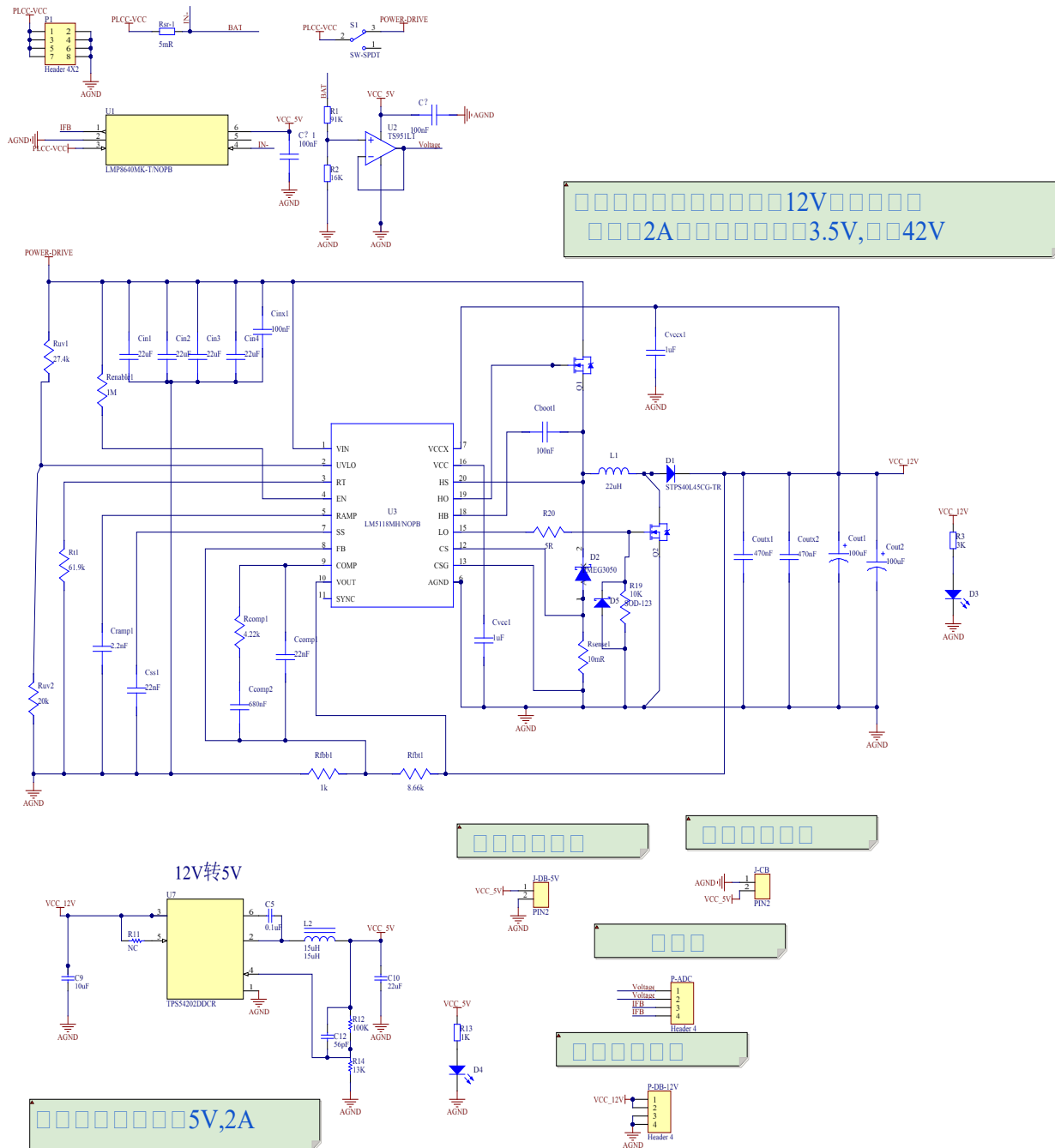


图 B-2 电源板原理图