分类号: U27;U46

10710-2011222002

长安大学

专 业 硕 士 学 位 论文

基于 16 位单片机 MC9S12XS128 的两轮自平衡智能车的系统研究与开发

丁 磊

导师姓名职称	韩 毅 副教授		
申请学位级别	硕士	学科专业名称	车辆工程
论文提交日期	2013.5.13	论文答辩日期	2013.6.8
学位授予单位	长安大学		

The Development and Research of Two-wheeledself-balancing intelligence vehicle System Based on 16-Bit MCU MC9S12XS128

A Dissertation Submitted for the Degree of Master

Candidate：Ding Lei

Supervisor：Associate Prof. Han Yi

Chang'an University, Xi'an, China

论文独创性声明

本人声明：本人所呈交的学位论文是在导师的指导下,独立进行研究工作所取得的成果。除论文中已经注明引用的内容外，对论文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本论文中不包含任何未加明确注明的其他个人或集体已经公开发表的成果。

本声明的法律责任由本人承担。

论文作者签名：丁磊　　2013年6月15日

论文知识产权权属声明

本人在导师指导下所完成的论文及相关的职务作品，知识产权归属学校。学校享有以任何方式发表、复制、公开阅览、借阅以及申请专利等权利。本人离校后发表或使用学位论文或与该论文直接相关的学术论文或成果时，署名单位仍然为长安大学。

（保密的论文在解密后应遵守此规定）

论文作者签名：丁磊　　2013 年 6月15日

导 师 签 名：　　2013 年 6月15日

摘　要

汽车的普及是大势所趋，汽车电控水平的高低及汽车电器智能化的程度被看作是衡量现代汽车水平的重要标志。随着汽车电子技术和机器人智能技术的快速发展，对智能车辆的研究已经成为自动控制领域内的一个研究热点。智能汽车是一种集环境感知，规划决策，自动行驶等功能于一体的综合高新科技系统，它的应用涉及到众多学科，诸如自动控制、模式识别、计算机视觉、传感器技术、车辆工程、电子与电气、单片机等众多学科。

教育部举办的“飞思卡尔”智能汽车大赛，是一项综合性很强的赛事，能够充分调动大学生对智能汽车的兴趣，培养大学生进行科学研究的能力，提高大学生的动手能力和科技创新能力。本文以第七届“飞思卡尔”智能汽车大赛为研究背景，运用MC9S12XS128 单片机，设计一种基于电磁导航的两轮自平衡车系统，并实现车模的直立行走，自主寻迹功能。系统采用飞思卡尔十六位微处理器 MC9S12XS128 作为核心控制单元，通过运用各种传感器，设计稳压模块、最小系统模块、双轮测速模块、倾角测量模块、电机驱动模块和人机交互模块并编写相应程序以完成平衡控制，速度控制，转向控制三大任务。文中通过对倒立摆进行动力学建模，类比得到车模的平衡条件。首先利用陀螺仪和加速度计获得车模的倾角和角速度，并进行卡尔曼滤波，并对倾角进行 PD 控制(比例-微分控制)实现车模的直立。通过光电编码器分别测得车模的线速度和转向角速度，对速度进行 PI 控制(比例-积分控制)。将转速控制信号与平衡控制信号叠加加载到后轮两电机上，实现车模的静止和直立行走。通过道路电磁中心线偏差检测与电机差动控制，保证车模在赛道上行驶，并对方向控制量和车体转动角速度进行 PD 控制可实现车模的方向控制。

最后对系统机械机构进行了安装调试，同时对三个控制模块的控制参数进行了静态及动态的调整，对整套系统进行实车赛道调试，实现智能车快捷、稳定、灵活的自主行驶。

关键词：自平衡，卡尔曼滤波，PID 控制，单片机

Abstract

The popularity of cars is the trend, and the level of automotive electronic control and automotive electronics intelligent degree is seen as the important symbol of the levels of modern car. As the rapid development of electronic information and Robot Intelligence Technology, the research on intelligent vehicle automatic control has become a hot research topic within the field. Intelligent vehicle is a set of environmental perception, planning decisions, automatic driving functions in an integrated high-tech systems. It covers multiple disciplines such as Automatic control, pattern recognition, Computer vision, sensor technology, Vehicle engineering , electronics and electrical , single-chip and so on.

The Freescale Intelligent Car Competition, which is held by the ministry of education. That event is strongly comprehensive. So, it can stimulate the college students to be in interested in intelligence automobile as well as cultivate college students' capability of doing research, practical ability and innovation. In this paper, the seventh "Freescale " Intelligent Car contest as research background , the use of MC9S12XS128 microcontroller, based on the electromagnetic design of two self-balancing vehicle navigation systems , and to achieve car models to walk upright , self- tracking capabilities. This system uses the Freescale 16 bit microprocessor MC9S12XS128 as the core control unit. By adding various sensors, designing the voltage regulator module, the minimum system module, double speed measure module, angle measurement module, motor drive module and man-machine interaction module, this system finally complete three complex tasks including balance control, speed control, steering control through designing corresponding control algorithms and other programs. This design first abtain the balance conditions of the two wheeled self-balanced car through establishing dynamic model of the inverted pendulum. Then, this system uses gyroscopes and accelerometers to measure the dip angle and angular velocity of the car, and utilizing Kalman filter to fuse dip angle and angular velocity. The design uses PD controller in the dip angle control to realize basic upright. Then the design uses photoelectric encoders to obtain the speed of two wheels and steering angular velocity. The speed is controlled by PI controller. The car could keep stationary and upright by combining the speed control signal and dip angle control signal together. The center line of the road electromagnetic deviation detection and motor differential control to achieve direction control, Cars driving on the racetrack . And directional control volume and body angular velocity of the PD control can be achieved in the control of the direction of the car models .

Eventually , after the installation and debugging of the mechanical system and the static

and dynamic adjustment of the three-control module control parameters .The entire system can be adjusted in the real car on the track to make the intelligent vehicle drive in fast, steady and flexible way

Key words: self-balanced; Kalman filter; PID control； single-chip

目　录

第一章 绪论

1.1 引言

自从上世纪 90 年代以来，伴随着科学技术的突飞猛进，经济的高速发展，汽车保有量持续增加，同时车辆的安全性，舒适性，经济性也越来越得到人们的重视。

在这种发展趋势和需求背景下，各种先进技术都运用到了汽车上，使得普通汽车越来越智能化。由于智能车涵盖多个学科，融入很多高新技术，因此很快就成为科学研究的热点[1]。

智能汽车是一个综合应用环境感知技术、规划决策技术、多等级辅助驾驶技术等的高科技产品。它主要侧重于计算机技术、图像处理识别技术、现代传感器技术、人工专家智能技术的应用。车载计算机是整个智能汽车系统的核心类似于人的大脑的功用，智能汽车对行驶道路环境的感知主要通过摄像头、雷达等传感器来完成，同时辅助设备对智能汽车也同样重要，该部分主要是一些电子设备。

无论在人们生活还是科研领域，智能车都有广泛的应用前景。在民用方面，它能自动识别出道路障碍物、制动的自动控制、安全距离的保持、自动泊车等；在军用方面，智能车可以代替人在危险的地带执行排雷及救援等任务。在科学研究领域，智能车可以代替人类完成外星球勘探，如美国的“机遇”号和“勇气”号火星车。

目前虽然智能车技术尚未完全成熟，它也没有真正进入到大众消费市场，但伴随着人类汽车生产制造技术的提高及半导体技术的逐步成熟，在一些高端汽车上较多的汽车电子技术和电子设备都成为这类车的标准配置。这些配置的总成本已经达到甚至超过总成本的一半以上。目前汽车的所有系统，都能看到电子产品的身影，例如 ABS（防抱死制动）、ASR（驱动防滑）、EPS（电动助力转向）、EBD（电子制动力分配系统）、ESP（车身电子稳定系统）等。这些装置的应用使汽车的智能化得到极好的体现。

智能汽车是未来汽车行业的发展方向，随着高新技术如信息技术和计算机技术的迅猛发展，以及模糊控制技术和人工神经网络技术等控制技术的出现和发展，智能汽车在将来肯定会得到突破性发展并能进入到大众消费市场中。

在这种发展趋势和行业背景下，教育部为给当代大学生提供动手实践的平台，发掘大学生的创新和创造能力，锻炼大学生的团队协作精神，为我国汽车行业培养后备人才，决定从 2006 年起，每年举办一届“飞思卡尔”大学生智能车竞赛。“飞思卡尔”智能车竞赛深受广大师生的欢迎，已经成为教育部主办的为数不多的全国性的竞赛之一[2]。2012

年第七届的全国大学生飞思卡尔智能车大赛，南京师范大学于 2012 年 8 月 22 日至 25 日成功承办该赛事的全国总决赛。参赛学校 118 所共有 195 支队伍参加比赛，分布如下：光电组 55 支，摄像头组 58 支，电磁组 57 支，创意组 25 支。在经过两天的激烈比拼后，电磁组共产生特等奖 3 名，一等奖 17 名，二等奖 37 名。长安大学自主设计开发的智能电磁车在该届西部赛区的比赛中获得二等奖的名次。

1.2 国内外的智能汽车的研究状况

智能汽车是一个综合应用环境感知技术、规划决策技术、多等级辅助驾驶技术等的高科技产品。计算机技术、图像处理识别技术、现代传感器技术、人工专家智能技术在智能车上得到充分的应用。所以它是高科技的产物也是未来汽车的发展方向，世界各个汽车公司均在该领域的研究投入巨大精力同时也为汽车工业的增长带来新动力[3]。

无人车红旗 HQ3 是由我国的国防科技大学独立研制完成同时也是我国自主研发的第一台无人车。在 2011 年 7 月 14 日首次完成从长沙到武汉 286km 的全部高速的无人驾驶试验，这表明我国在无人车自主研制开发和在复杂道路环境下自主驾驶方面取得了重大的突破。同时也展现了我国设计的无人驾驶车能识别出复杂道路环境、能进行精确的智能行为决策和控制。

随后中国第一汽车集团与国防科技大学共同研发的无人驾驶轿车，综合技术性能和指标均在世界无人驾驶研究领域得到认可。此无人驾驶轿车装备了激光雷达、摄像相机，它可自主实现导航并对识别到的道路路况、障碍物等进行判别，对速度可进行自行调整，这些操作的完成均不需人为的参与[7]。与 GPS 导航、电子巡航比较起来，无人驾驶轿车的导航定位系统具有定位精准，并具有在转弯和遇到复杂的道路交通路况下进行快速处理的优点。根据车内装备的环境识别系统可识别出道路状况，并可把与前方车辆之间的距离及无人驾驶轿车的相对速度和距离也能测量出来；同时由计算机视觉给出的道路信息、车前状况以及行驶状态，车载主控计算机和相应的路径规划软件来决定前进还是准备超车；为了能使汽车按规定的路径行驶，智能汽车控制系统需按照跟踪的时时道路路况信息并根据汽车动力学理论知识，来向轿车方向转动装置、刹车控制装置和油门做出相应的动作命令。

上世纪 50 年代，智能汽车的研究首先是从国外的一些经济比较发达国家开展起来的[4,5]。1954 年美国的贝瑞特电子公司研制出世界上第一台真正能实现自主行使的智能车辆，但该车也只能在 MercuryMotoFreight 公司设定的固定线路中行驶。到上世纪 70

年代末，美国、日本及欧洲的少数几个发达国家在智能车技术的探索和研究方面远远领先于其它国家[4]。

20 世纪 80 年代，科研实力雄厚的美国在强大经济的支持下，在智能车研究领域大胆的提出地面车辆计划（ALV）。为实现智能车能在校园环境中完成自主行驶的目的，该计划设计制作一辆 8 轮车，经试验该车出色的完成校园环境复杂道路路况下的自主驾驶，但行驶车速低的是该车的一个较大缺点。随后美国的卡耐基梅隆大学、麻省理工学院以及涉及汽车产业的公司等都先后加入到智能车辆的研究开发中。1995 年，美国著名的卡耐基梅隆大学自主设计研发一款无人驾驶智能车并将其命名为 Navlab_V，该车完美的实现了穿越美国东西部的驾驶任务。随后它又在美国 5000 公里的州际高速公路上的测试试验中，自主驾驶完成 96%以上的道路行驶，其中运行速度可以达到 50~60km/h[6]。

2005 年谷歌公司的工程师塞巴斯蒂安-特龙带领自己的团队设计制作出谷歌版的无人驾驶车。该智能车通过装载的激光雷达、车载相机和激光测距仪来观察周围行驶车辆，并依据详细的车载地图完成汽车导航。目前在无人驾驶和有人驾驶车辆混合在道路上有效避免交通事故是无人驾驶汽车研究人员面对的新挑战。

1.3 未来智能汽车的发展方向和前景

电子控制技术和智能技术在智能车上的应用，极大的提高智能汽车的各项性能指标，未来智能车正朝着电子化，智能化，信息化，网络化的方向发展，这也是当今信息社会发展的必然趋势。未来智能车的研究成果定能加快信息社会下的生活节奏方式。目前世界上各国主要在以下三个路况环境下开展无人驾驶智能车的研究与开发：特殊路况，城市路况和高速公路下的无人驾驶控制系统的研究与开发。这三个方面的研究内容具有相互重叠的特点，而在技术的研究方面具有不同程度的侧重[8]。

特殊路况下的无人驾驶控制系统：特殊路况是指军事上及一些较为危险的场合的路况环境，该路况下要求无人驾驶智能车对恶劣路况的适应性及对车辆的可靠性能力等方面有苛刻的要求，这些问题的解决才能为无人驾驶车在特殊路况下的使用提供极大的技术支持。

城市路况下的无人驾驶控制系统：考虑到城市路况的特殊性，这类无人驾驶车的开发必须具备速度慢，安全性指标高的特点，具备这些特点的无人驾驶车在未来的应用中具有良好的前景。例如在一些大型活动场所、休闲公园、学校、工业园区、机场等场所。

目前在这类车的开发过程中主要面临以下的技术难题如车辆的可靠性、多车辆间的协调以及人车相互交互等。所以，为适应复杂的城市路况，则应在无人驾驶车的感知和控制方面加大研究力度，以便车辆在该路况下能有较优异的性能发挥。

高速公路下的无人车驾驶控制系统：主要运用在具有良好标志的结构化高速公路上，进行道路标志线跟踪及车辆识别等方面工作。目前该研究已经实现在简单路况下能出色完成高速的自主行驶任务，而在现实高速公路上实现全自动高速驾驶是现如今的研究领域。

1.4 本论文的主要研究内容

（1）方案任务原理

为完成两轮自平衡智能车的直立行走任务，本方案采取模块化的设计理念。将此任务分为直立控制模块、速度控制模块、方向控制模块三个模块。通过控制电机的正反转，电机的转速及电机的转速差可实现上述模块的单独控制，但最终都是对车模的电机这一个控制对象进行控制，所以它们之间存在着耦合。因此，在分析其中任何一个模块时，均假设剩余的控制模块已达到稳定状态。在进行速度模块控制时，需保证车模的直立控制能够实现；在车模保证直立平衡和速度稳定的前提下，方向的控制才能得到好的实现；同理车模直立平衡的实现，则要求其它两个控制模块应处于稳定状态。经过上面的分析得知车模的直立平衡控制是实现以上三个任务的关键。实际运行中车模要同时受到三种控制的作用，所以从车模直立平衡控制出发，速度和方向控制模块就成为直立平衡控制的干扰项。因此对车模的这两种控制应该做到尽量保持平滑，以此来减少它们对于车模直立平衡控制产生的干扰。

（2）车模硬件电路的设计与制作

设计高品质的外围电路确保各个电子器件能够正常工作，同时应将系统的所有电路设计制作成 PCB 板，以便在出现问题时便于查找， 通过上述措施可有效提升系统的可靠性。

（3）控制系统设计

智能车的控制系统的设计是研究的重点内容，其主要功能是完成两轮自平衡小车的直立控制、速度控制、方向控制。PID 控制是一种非常成熟的工业控制方法，其在智能车辆控制领域得到了很好的应用。但 PID 控制在智能车领域的应用上需要克服控制器的

采样时间及各项系数对控制器调节能力的影响，所以通过何种手段确定上述参数将是控制器研究的重点内容。

（4）模型车的机械结构设计

模型车模的机械设计的好坏，对比赛成绩起着关键作用。所以各种传感器理想安装位置的选取及精确固定，同时在模型车上如何固定硬件电路是该部分研究的重点。安装传感器时主要考虑其定位精度及模型车在行驶时防止发生抖动，安装光电编码器应将同轴度和齿轮传动时产生的间隙问题考虑进来，最后通过准确的测量和计算确定其结构尺寸。

（5）基于目标控制器的软件系统开发

论文中按照既定的控制策略，解决方案和控制软件实现流程来实现目标控制。系统的开发环境采用了 Freescale 公司的集成开发环境 CodeWarrior，在该环境下利用 C 语言编写能实现目标控制的主程序和相关子模块的控制算法程序。

1.5 本章小结

本章介绍智能化汽车的发展及研究背景，同时对国内外智能车的研究发展现状进行介绍，最后列出本文的设计思路及论文要解决的内容。

第二章　　两轮自平衡智能车系统总体概述

2.1 两轮自平衡智能车的任务模块化设计

近年来随着汽车产业的发展壮大，汽车的研究领域也逐渐得到扩大，两轮自平衡电动车便是在此背景下得以进行并取得不错的研究成绩，该型号车最大的特点是行走灵活、便利、节能等。为了在大学生团体中开展对两轮自平衡车的研究，在 2012 年的全国大学生第七届飞思卡尔电磁组比赛中，即仿照两轮自平衡电动车的行进模式来完成电磁组两轮自平衡车直立行走，它以两后轮驱动车模完成直立行走任务。在实际的设计和研究中发现两轮自平衡车模的控制相比于四轮车情况更为复杂，为了更好的开展研究，本论文应用了模块化设计理念，将复杂的整体分解成较易实现的模块进行研究。本论文使用车模的两个后轮为车模保持直立及运行提供动力支持，并通过两个直流电机驱动车模的后轮转动。从控制角度分析，车模作为控制对象，两电机的转动速度是该控制对象的控制输入量。以下的三个基本控制模块构成了车模的运动控制任务：

（1） 车模直立控制模块：车模直立状态可通过对两电机正反向运动的控制来完成。

（2）车模速度控制模块：车模的速度控制可通过对车模倾角的调节来实现，最终车模的速度控制仍是通过对电机的转速控制来完成。

（3）车模的转向控制模块：车模转向控制是通过控制两个电机之间的转动差速来实现。

通过直接控制车模的两个后轮驱动电机可完成车模直立和方向控制任务。文中为了完成以上两个任务选择将车模电机虚拟地拆分为两个具有不同功用的驱动电机，并使它们同轴连接起来。论文在设计和实施时，将车模的直立及方向控制信号进行叠加然后加载到电机上，此时电机应保证处于线性状态。通过车模倾角的调节可完成车模速度的控制。车模的加减速由车模倾角的不同产生变化，从而达到对于速度的控制。三个分解后的模块分别进行独立控制，但最后还是演变成对车模电机进行控制，则三个模块之间存在耦合现象[9,10]。文中在进行分析其中任何一个模块时，均认为其它控制模块都处于稳定状态。若进行车模速度模块控制时，需保证车模的直立控制能够实现；在车模保证直立平衡和速度稳定的前提下，方向控制才能得到更好的实现；同理车模直立平衡的实现，则要求其它两个控制模块处于稳定状态。经过上面的分析得知车模的直立平衡控制是实现以上三个任务的关键。三个模块执行的优先级为：平衡控制 > 速度控制 > 转向控制。

实际运行中车模要同时受到三种控制的作用，所以从车模直立平衡控制出发，速度和方向控制模块就成为直立平衡控制的干扰项。因此对车模的这两种控制应该做到尽量保持平滑，以此来减少它们对于车模直立平衡控制产生的干扰。

2.2 两轮自平衡智能车的直立平衡控制模块

两轮自平衡车模的直立平衡控制是通过负反馈机制实现，由于车模只有两个轮子着地，车模的倾斜只能发生在车轮发生滚动的方向上。为了保持车模平衡，只要能控制车轮转动，并在一个维度上抵消车模发生倾斜的趋势[11]。如图2. 1所示，车模向左发生倾斜，车轮便产生向左的加速度；车体向右发生倾斜，车轮便产生向右的加速度。

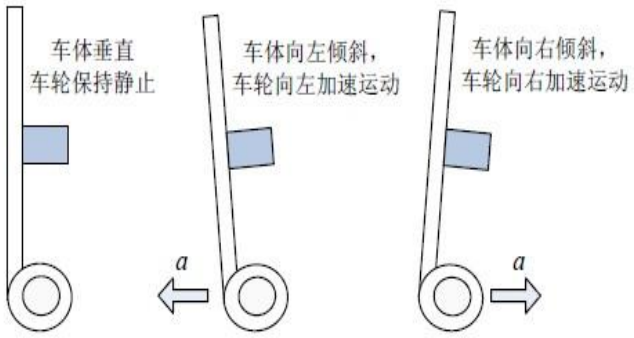

图 2.1　通过车轮运动保持车模平衡

文中通过建立车模的运动学和动力学数学模型，设计反馈控制来保证车模的平衡。下面通过对比单摆模型来说明保持车模平衡的控制规律。

重力场中使用细线悬挂着重物经过简化便形成理想化的单摆模型。直立着的车模可以看成放置在可以左右移动平台上的倒立着的单摆。如图 2.2,2.3 所示。

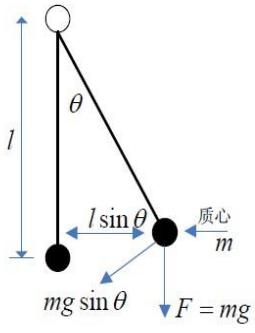

图 2. 2 普通单摆受力分析

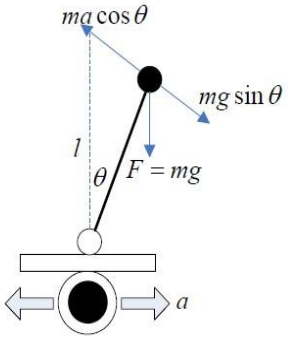

图 2.3 在车轮上倒立摆受力分析

由图 2.2 的受力分析知，当单摆离开垂直位置后，单摆受到重力与悬线的作用力，在此作用力的作用下单摆回到平衡位置。在图中θ角比较小的情况下，该作用力与θ大小成正比例关系，且方向相反。在此作用力作用下，单摆可进行往复的周期运动。由于单

摆在空气中运动时，会受到空气阻尼力的作用，则单摆最后会在垂直位置停止。单摆运动速度大小与空气阻尼力的大小成正比例关系，并具有方向相反的特点。而倒立摆在偏离垂直位置的时候，并不能像单摆在此位置稳定下来，是因为倒立摆受到的回复力与其运动位移方向相同，而此会加速倒立摆偏离垂直位置，最终倒立摆会倒下。通过给倒立摆增加额外的受力或者改变重力的方向，来使回复力与位移方向相反，此时倒立摆能够像单摆一样，在垂直位置稳定下来。根据经验可知只有第一项措施能达到目的。后轮的两驱动电机提供了车模运动所需的全部动力，所以通过对倒立摆的底部车轮进行控制，以便使它作加速运动。若从车模上的角度来对倒立摆的受力进行分析，额外的惯性力便加载到车模上，并且该惯性力与车轮的加速度大小成正比例关系，且方向相反。所以倒立摆此时受到的回复力为：

$$F = mg\sin\theta - m\alpha\cos\theta \approx mg\theta - mk_1\theta \tag{2-1}$$

式中由于θ角比较小，可对其进行线性化处理。在负反馈控制中，假设车轮的加速度a与偏角成θ正比例关系，且比值为k_1。如果保证比例值$k_1 > g$，那么回复力的方向与位移方向成相反关系。为使倒立摆在垂直位置尽快稳定下来，则只需增大阻尼力即可。虽然存在着空气和摩擦力等阻尼力，但数值相对较小。实际应用中额外增加了控制阻尼力。增加的阻尼力与偏角的速度成正比，且方向相反。因此式（2-1）可变为

$$F = mg\theta - mk_1\theta - mk_2\dot{\theta} \tag{2-2}$$

按照上面的控制方法，可把倒立摆模型变为单摆模型，能够稳定在垂直位置。因此，可得控制车轮加速度的控制算法为：

$$\alpha = k_1\theta + k_2\dot{\theta} \tag{2-3}$$

上式中，车模倾角为θ，车模角速度为$\dot{\theta}$，比例系数k_1，k_2，车轮加速度的控制量为a。为保证车模能像单摆一样维持直立平衡，须保证$k_1 > g$，$k_2 > 0$成立。其中式中的控制参数k_1决定着车模是否能在垂直位置稳定下来，此时必须使$k_1 > g$成立； 控制参数k_2决定着车模回到垂直平衡位置的阻尼系数的大小，为了使车模尽快能在垂直平衡位置稳定下来，合适阻尼系数的选取显得至关重要。在上述角度反馈控制中，比例控制和微分控制分别是与角度和角速度成比例的控制量。因此上面的控制系数k_1，k_2即是比例控制参数和微分控制参数。其中微分控制参数相当于车模运动时所受的阻尼力，在抑制车

模的震荡方面起到较好的效果。由以上分析可知只要能够精确测量车模倾角和角速度的大小及控制车模车轮的加速度即可保证车模的平衡控制。

2.3 两轮自平衡智能车的速度控制模块

始终保持两轮自平衡车模的直立是进行车模速度控制的前提保证，所以相对于普通车模的速度控制而言，两轮自平衡车模的速度控制则比较复杂，所以仅仅通过直接改变电机转速来完成车模的速度控制存在较大难度。欲进行车模速度控制，首先对引起车模速度变化的原因进行仔细的分析。

由于在实际情况下车模安装误差，车模角度与倾角传感器实测值之间存在偏差，所以此时车模实际上与地面并不保持垂直状态，而是与地面之间存在一个倾角偏差。所以即使车模在经过上面的直立控制调节处于平衡状态，但是由于上述倾角的存在，并在重力的作用下，车模仍会在倾斜的方向上产生加速度并使车模加速前进。所以控制车模的倾角就可以实现速度的控制。具体实现需要解决三个问题： 车模运行速度值的测定；车模在直立控制过程中如何实现改变车模的倾角；车模倾角如何由速度误差来进行控制。

第一个问题车模运行速度的测定可通过安装在电机上的编码器来完成测定。

假设车模在上面直立控制调节下已经能够保持平衡，则只要通过控制车模的倾角就可以实现速度的控制。车模直立控制时事先对倾角角度进行设定，并在角度控制的调节下，两轮自平衡车模将会在一个角度处维持原状。通过对车模直立平衡控制算法的研究得知，车模倾角与重力加速度 Z 轴上的角度成一定的关系。所以车模的倾角是由重力加速度 Z 轴上角度与车模的给定倾角值相减得到。

根据速度误差控制车模倾角的问题分析，由倾角控制引起的车模运行速度的变化部分，可忽略不计。倾角产生的加速度决定着车模的运行速度，速度的给定会影响到小车往前倾的角度，因此对车模倾角进行积分运算便得到车模运行速度。如图 2.4 为两轮自平衡车模速度控制框图。

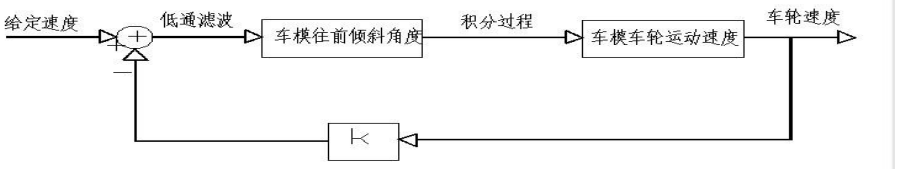

图 2.4 车模运动速度控制简化模型

结合小车角度的闭环控制，将角度控制闭环与速度控制闭环叠加在一起便可实现速度和角度的综合控制。但要注意，速度闭环的极性为“负”，则原来的负反馈相当于正反馈。因为原来在直立控制下的车模速度与车模倾角之间传递函数具有非最小相位特性，在反馈控制下容易造成系统的不稳定性。

2.4 两轮自平衡智能车的方向控制模块

对机器人行进原理进行分析可知通过给机器人的左右轮电机施加不同的控制电压，该控制电压驱动机器人左右轮前进和转向运动。车模的前进速度是两轮速度之和的一半，而其转向速度则与车模两轮速度之差成正比例关系，则类比得到机器人的前进和转向速度分别由两轮电压之和及电压之差决定。系统调试的环境是一铺有引导线的任意形状的跑道，包括直道，大弯，小弯，十字弯等。通过道路电磁中心线偏差检测与电机差动控制实现方向控制，保证车模在赛道上行驶。道路电磁中心线检测可通过安装在车模前方的两个电磁感应线圈实现。

根据两编码器测得的速度之差，可得到车体的转向角速度。对方向控制量和车体转动角速度进行 PD 控制可得到转向控制信号。根据相应的转向控制信号调节左右两个车轮的差速来实现转弯和寻迹[12,13]。

结合小车角度的闭环控制和速度的闭环控制，将角度闭环，速度闭环与差速闭环叠加在一起边可实现直立，速度和转向的综合控制。控制结构图如图 2.5 所示。

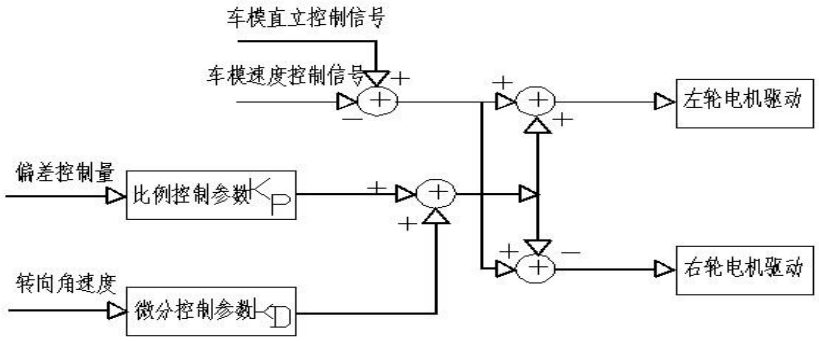

图 2.5 车模运动方向控制简化模型

为了实现两轮自平衡车模沿着赛道行走，车模的方向控制就显得至关重要。电磁组的赛道中心线上铺设有一根漆包线，在里面通有 100mA 的 20kHz 交变电流，所以道路中心线周围便产生了一个交变磁场。则车模方向的控制可通过检测道路电磁中心线偏差与电机差动控制一并进行来实现，这样能保证车模在赛道上良好的运行。

本文中道路电磁中心线的检测通过安装在车模前方的两个电磁感应线圈实现。此时左右驱动轮的差动控制电压，是由车模速度控制信号与电磁线偏差检测信号进行加减后形成的，该差动电压对车模方向的控制是通过车模左右驱动轮的运行角速度不一致来实现的。电机的差动控制量是由车模的方向控制算法根据车模检测到的电磁感应电压得到的，运行中车模距离赛道中心线的偏差消除是通过左右电机速度差驱动车模转向来完成的，而运行中车模距离赛道中心线的距离差可由车模方向的调整，以及车模的前行运动来消除。以上过程是个积分过程，所以车模的方向控制通过简单的比例控制就能完成。但在实际过程中车模安装一些较重的装置如电池等，所以车模具有较大的转动惯量，在赛道上运行时车模会出现转向过冲现象，如果此现象不加以抑制，车模将会存在冲出赛道的风险。论文在该部分的设计时增加微分控制过程，对该现象进行有效地抑制。微分控制的工作原理是根据车模方向的变化率来对控制驱动电机的差动量进行修正。论文中将左右两个线圈感应电动势之差除以两个线圈的感应电动势之和得一比值，使用该比值进行车模的方向控制可以将车模重心距离电磁线圈的距离差更好的反映出来，并避免对车模倾角的影响。

2.5 本章小结

本章主要运用模块化思想对两轮自平衡车的行走任务进行分解，对各模块进行分析并确定各模块的实现方案，为后续的硬件和软件设计打下良好的基础。

第三章　两轮自平衡智能车系统硬件电路设计

3.1 两轮自平衡智能车系统硬件总体设计

系统的硬件电路设计是系统控制的关键一步，在高质量的硬件电路下，智能车的控制策略及控制算法才能充分的发挥作用。对系统的输入、输出信号进行分析是车模控制系统电路设计的第一步，然后根据以上的分析结果选择合适型号的单片机本论文选用的16 位的 S12 系列的 MC9S12XS128 单片机。对系统的各个电路模块进行逐步设计，最终设计出完整的系统控制电路[14]。以下几个方面是该控制系统的输入输出：

（1）AD 转换接口（不少于 5 路）

电磁检测：两路，对左右两个感应线圈电压进行测量。

陀螺仪： 两路，分别用于车模转动角速度和倾斜角速度的检测。

加速度计：一路，测量加速度Z轴输出电压。

辅助调试：1到3路，主要是用来对车模的调试及设置。

（2）PWM接口（4路）

主要是对左右两个电机的双方向运行进行控制。根据PWM驱动采用的极性不同也决定着需要的PWM接口数目的不同，单极性驱动需四路PWM接口。双极性PWM驱动，需要两路PWM接口。

（3）定时器接口（2路）

主要是对两个电机的转速进行测量，它同时需要保证有两个定时器脉冲输入端口。

（4）通讯接口（备用）

SCI： 一路，主要应用于程序的下载及调试接口。

I2C： 如果选择飞思卡尔公司的数字加速度计，可以通过I2C接口直接读取加速度值。

（5）IO接口：4到8路输入输出，主要是对车模的运行状态进行显示及各个功能进行设置等功用。

Freescale公司生产的S12系列单片机主要使用了复杂指令集计算机架构，内部具有集成的中断控制器，同时具有极为丰富的寻址方式。中断有7个优先级，并且内核支持优先级的调度，最多可以有117个中断源，且内部集成有丰富的AD转换器和PWM发生器。本论文的核心控制器选用该系列的一款16位的MC9S12XS128单片机，经实际使用发现能

较好的完成比赛所需的各项要求[16]。

微处理器控制单元、稳压电源单元、电机驱动单元、倾角传感器单元、车速测定单元、电磁线检测单元、调试与设置单元等组成了系统的硬件部分。系统通过控制芯片来控制上述智能车的各个单元，并使得车模的各个单元之间能够协调并出色的完成工作。通过单片机内部AD采集各个传感器的信号，利用控制器进行滤波和信号处理，对车身姿态和当前的赛道环境进行识别，最后通过设计的控制策略及算法将控制量的输出通过PWM形式传送给后轮两个电机，以完成系统的所有任务。智能车整体系统的框架图如图3.1所示：

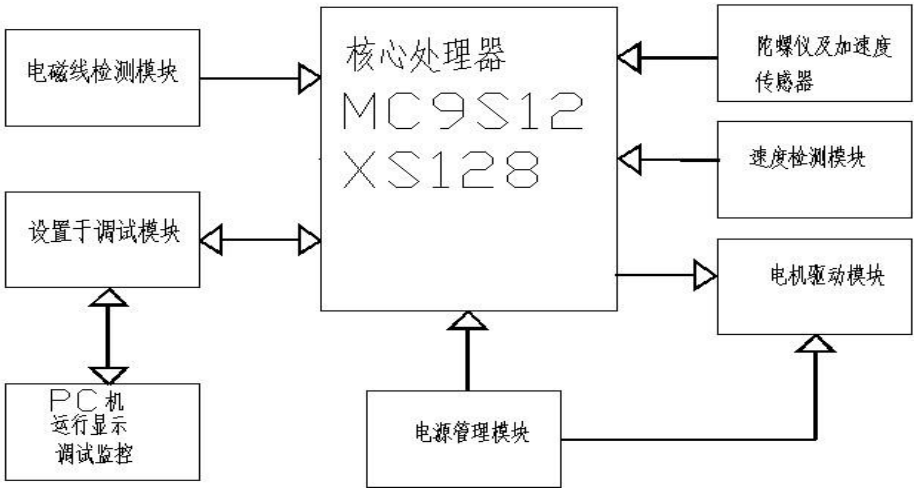

图 3.1 车模控制电路整体框图

3.2 单片机 MC9S12XS128 的内部资源

MC9S12XS128 是 16 位单片机，它的片内存储器有以下单元组成：16 位中央处理单元（CPU12X）、8KB 的 RAM、128KB 的程序 Flash、8KB 的数据 Flash。内部存储器，内部 PLL 锁相环模块，1 个同步串口外设接口 SPI，2 个异步串行通讯口 SCI，1 个 8 通道的比较定时器模块 TIM，周期中断定时器模块 PIT，16 通道的 A/D 转换模块 ADC，1 个 8 通道的 PWM 模块等构成了该芯片的主要功能模块[18]。

MC9S12XS128单片机具有在线编程的功能，实际应用中支持在线写入，擦除和程序的在线下载。在线编程的思路是：单片机片内部的CPU具有对片内 Flash 进行在线写入，擦除操作。本论文应用的是BDM调试方式，基本流程是单片机接受串行发送的数据和命令，单片机的编程接口可在实现 Flash 写入，擦除和调试程序方面发挥功用，且在运行应用程序时，CPU寄存器及存储器等的瞬间数值和信息可实现动态的获取。此外，另一种调试方式为RS—232调试，该方式是先通过BDM把监控程序写入到 Flash中，然后脱

离开BDM，再通过RS—232下载程序进行调试工作。

3.3 单片机 MC9S12XS128 的最小硬件系统

集成的电路芯片能够将单片机的CPU，RAM，ROM及I/O资源整合到一起，同时外部电路的支持能保证单片机性能的充分发挥。单片机系统的电源支持、时钟、I/O驱动、通信口等均需这些外部电路来支持，在具备上述条件后单片机能更好的发挥功用[19]。此最小系统，要能满足单片机与人的通信(即通过BDM或串口等来实现)，包括发送命令到单片机上，下载程序并进行程序调试等。拥有这些条件后，就可进行单片机硬件系统的调试，在硬件处于良好的状态下进行软件系统开发。 该最小硬件系统主要有以下几部分构成：

（1）供电电路。该系统的供电是靠外部的+5V电源提供，目前单片机应用的供电电压一般是2.5V、3.3V 有时会更低，以此来满足单片机CPU越来越高的运行速度。本论文的单片机片内电压由2.5V工作电压提供，使用5V工作电压给片外 I/O提供供电，若使CPU运转速度较快则应使片内电压较低，此情况下的功耗较低。本论文选用的单片机能保证在工作环境较差的控制系统中发挥出优异性能。 MC9S12XS128单片机内部具有电压自动调整器单元可以产生单片机内部其它单元所需的工作电压，实际应用中只需向MC9S12XS128单片机提供+5V 外部电源即可，同时为了保证不同电压的稳定运行可以接上外接电容。外接电容主要分为以下两类：一类是储能电容，它的电容值较大一般为1uF、10 uF,该类电容能消除三极管导通、截止时的电流变化；另一类是去祸电容，它的电容值比较小一般为0.0luF、0.luF，该类电容能将单片机在运行时刻产生出来的高频噪声消除掉。

(2)时钟电路。电阻、电容和石英晶体振荡器是该时钟电路的组成部分。单片机运行时所需的时钟虽然可由单片机内部的振荡器产生出来，但稳定性较差是此时钟电路频率的最大缺点，实际情况下添加外部晶振来提高时钟电路频率的稳定性。单片机的运行速度越快，越需使用单片机片内集成的压控振荡器产生出来的高频振荡作为系统时钟，但此系统时钟需使用外部晶振为压控振荡器提供稳定的时钟电路频率。外部晶振电路分为并联振荡电路和串联振荡电路两种。本论文应用主板选用了并联振荡电路。具体的电路如图 3.2所示：

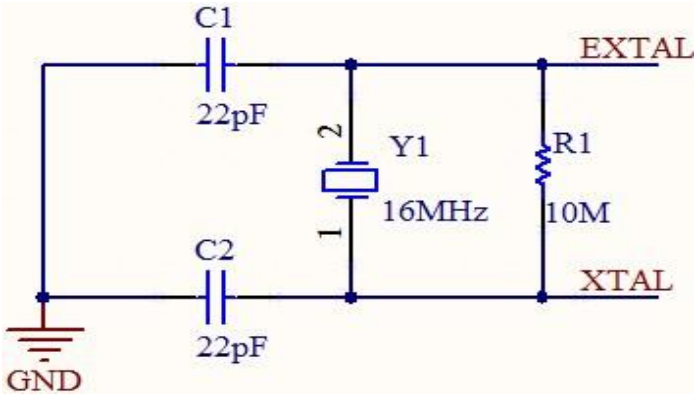

图 3.2 时钟电路图

(3)复位电路。复位表示单片机重新启动。若复位引脚RESET处于低电平状态时，RESET会产生一个低电平脉冲，单片机复位。复位原理：单片机正常工作时RESET通过$10k\Omega$的上拉电阻接到电源正极，当复位按钮按下时，RESET引脚接地，为低电平，单片机复位。具体的电路如图 3.3所示：

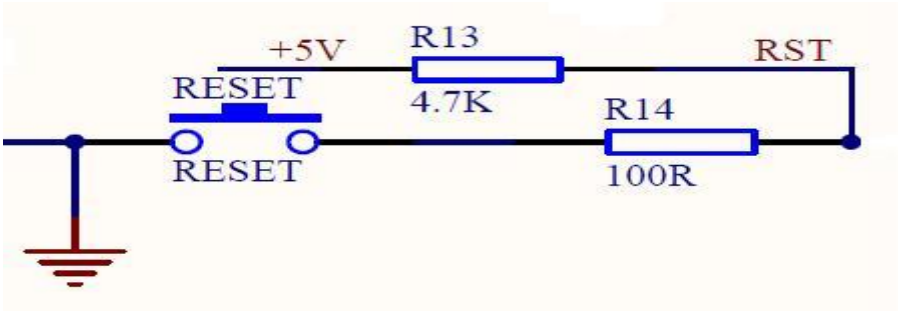

图 3.3 复位电路图

（4）BDM接口电路。背景调试模式BDM（Background Debug Mode）是由Freescale半导体公司自定义的片上调试规范，是一种底层的调试手段。开发人员利用BDM向目标板下载程序，也可以通过BDM进行在线调试和编程。本文采用常用的BDM调试插头如图3-4所示：

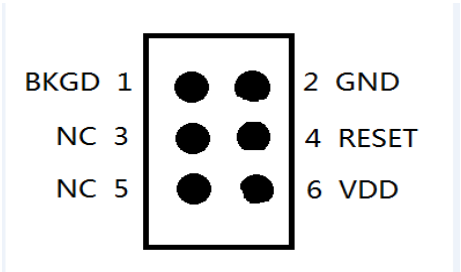

图 3.4 BDM调试接头示意图

引脚定义如下：

BKGD：单线背景调试模式引脚

GND：接地

VDD：电源

RESET：目标机复位引脚

(6)调试显示。为了方便系统的调试，实际应用中在单片机的某I/0接口上接上一些发光二极管，例如将调试时用的发光二极管接在PORTK口上，虽然这些发光二极管不一定是系统所必须的。同时为了便于单片机与外界电路的联接，可把单片机的所有I/O均引出。

3.4 电源管理模块

电源为智能车系统提供动力支持，目前市面上较为常见的充电电池主要有以下几种：镍氢电池、镍镉电池、碱性电池等。考虑到镍镉电池具备价格的优势、电路比较简单、技术比较成熟、能提供瞬间大电流等优点。实际应用中能很好的达到智能模型车的各模块功能要求，所以确定由镍镉电池提供智能车的电源支持[17]。

进行电池电压的调节是电源管理模块的主要功能，它为智能车各模块的正常工作提供稳定的工作电压。在智能车系统中，核心控制模块、电磁线检测模块、人机交互模块、车速传感器模块需要 5V 电压；倾角传感器模块需要 3.3V；电机驱动模块可使用 7.2V 2000mAh 镍镉蓄电池直接供电。考虑到由于驱动电机引起的电压瞬间下降的现象，因此采用低压降的三端稳压器成为必然。本文采用 LM1117 和 LM2940 作为稳压芯片，经测试电压纹波小，低压降，完全可以满足要求。智能模型车系统电源管理框图如图 3.5

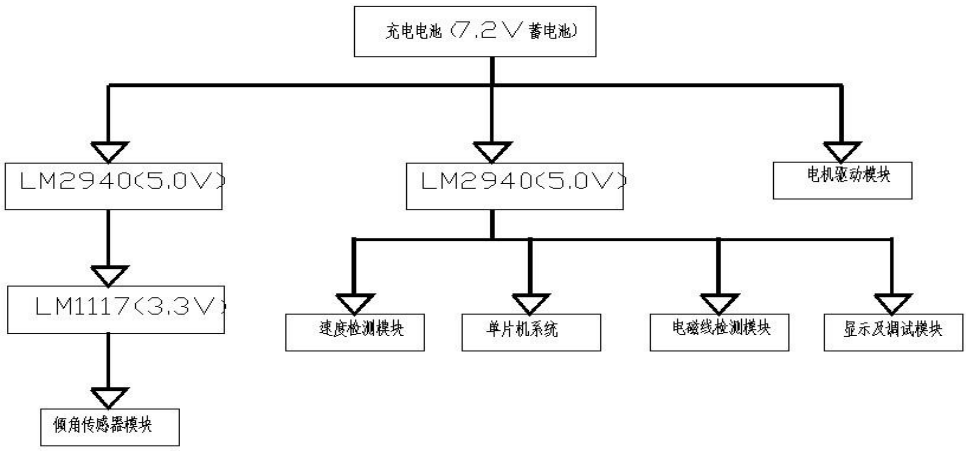

图 3.5　智能模型车系统电源管理框图

3.4.1　3.3V 稳压模块设计

LM1117 是一个低压差电压调节器系列，采用 LM1117 进行降压为倾角传感器模块提供 3.3V 工作电压。电路图如 3.6：

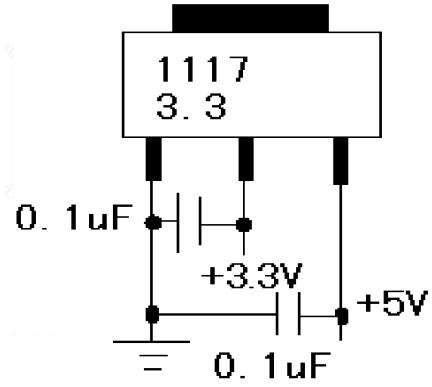

图 3.6　3.3V 稳压电路原理图

3.4.2　5V 稳压模块设计

智能车系统使用的是 7.2V 镍镉可充电电池，由于电机在使用的过程中，会出现电池电压的波动现象。从提高电池电能利用率的角度出发，电池的最低使用阀值及电池有限的充放电次数在实际使用过程中必须被考虑进来。所以本论文中选择了高效率、低压差的电源管理芯片，有效的提高蓄电池可利用率并同时延长蓄电池的寿命长度。实际应用中纹波小，电路结构简单是 LM2940 串联线性稳压模块所具有的优点，但其缺点是效率较低，功耗较大。但稳定的 5V 工作电源对于单片机性能发挥起着关键的作用，所以考虑到 LM2940 具有较优异的稳压线性度，所以论文最终将 LM2940 选定为 5V 电源稳压芯片。电路图如 3.7：

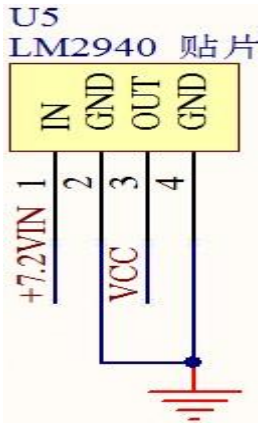

图 3.7 5V 稳压电路原理图

3.5 电机驱动模块

近年来随着直流电机的机械结构和控制方式的变化，以及计算机在控制领域的深入应用和新型电子元器件的出现，则脉宽调制(pulse width modulation 简称 PWM)控制方式在电机控制领域得到极大的应用，且使用单片机能较容易的实现该方式的控制。单片机

控制的小功率直流伺服系统可以采用专用的集成电路芯片搭建组成。本文选用的驱动芯片为一片 74ALS244 和两片 BTS7970，共同搭建成 H 桥电路来驱动电机[15]。

74ALS244 内部有两个四位三态缓冲器，其三态输出受到使能输出端的控制，当使能输出有效时，器件实现正常逻辑状态输出，当使能输出无效时，输出为高阻状态，即相当于和所连的电路断开。BTS7970 是一种大功率驱动芯片，输入电压为 6V-24V，输出电流最大可达 40A。由 BTS7970 搭建的驱动桥与 MC33886 相比，它的加减速性能更加卓越，稳定性更好，效率更高。应用电路如图 3.8 所示。

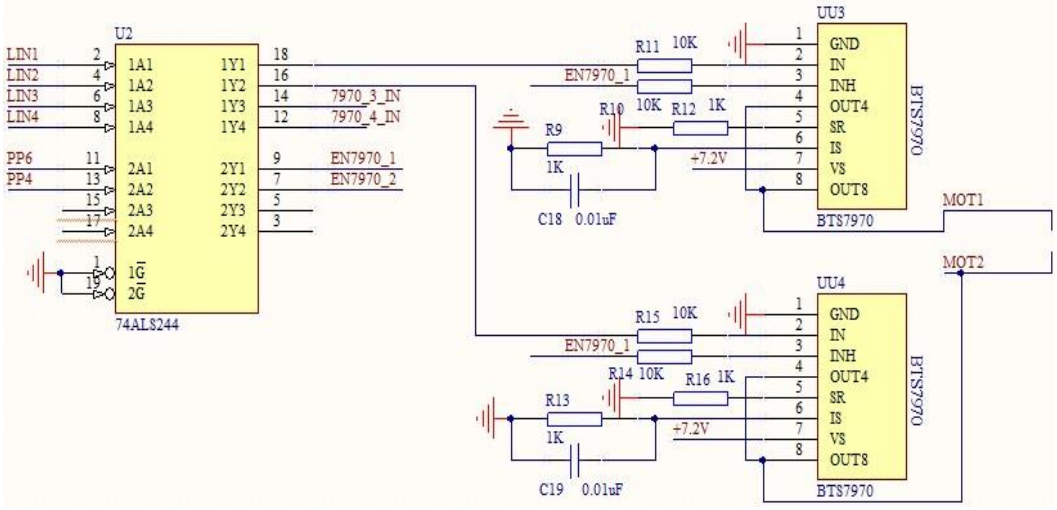

图 3.8 电机驱动电路图

3.6　车模倾角测定模块

3.6.1　车模倾角测定方案的选择

由第二章车模平衡原理可知，通过测量车模的倾角和倾角加速度控制车模车轮的加速度以此消除车模发生倾斜产生的倾角。所以控制车模平衡的关键即是测量车模倾角以及倾角加速度。

车模倾角的测定首先选用了一款加速度传感器，它可以测量物体运动所产生和地球引力作用下的加速度。论文设计的方案中，通过对车模 Z 轴方向上的加速度信号的测量来得到此方向上的加速度值，并最终计算出车模的倾角[9]。车模直立时，在 Z 轴的水平方向上固定加速度传感器，此时输出的信号为零偏电压信号。当车模发生倾斜时，重力加速度 g 值便会在 Z 轴方向上形成加速度分量，并造成该轴输出电压的变化。变化的规律为

$$\Delta u = kg\sin\theta \tag{3-1}$$

式中，g 为重力加速度；k 为比例系数；θ 为车模倾角。当车模倾角较小时输出电压的变化可近似与倾角成正比。

上述可测得车模的倾角，而倾角速度可通过对此信号进行微分运算得到。但实际情况是车模本身的运动会产生一定的加速度，此干扰信号叠加在上述测量信号上，会使得此输出信号无法准确的把车模倾角反映出来。

所以该方案具有一定的局限性，采用第二套方案利用角速度传感器陀螺仪来获得车模倾角和倾角加速度。将陀螺仪安装在车模上，可以测量车模倾斜的角速度，将角速度信号进行积分便可以得到车模的倾角。

陀螺仪输出的车模角速度受车模振动影响比较小，且此信号噪声较小。车模倾角可通过对上述测得的角速度进行积分运算得到，通过对该信号进行进一步的平滑，可增强角度信号的稳定性能。所以可通过陀螺仪来获得车模直立平衡控制所需的角速度和角度信号。但此方案的唯一缺点是获得陀螺仪的角速度信号后必须经过积分运算再能获得角度信号。此种情况下就是角速度信号存在微小的偏差但经过积分运算之后，都会形成积累误差，同时随着时间的推移该积累误差会逐渐增大，最终会使电路达到饱和状态，则无法产生正确的角度信号。

最终，本论文测定车模倾角通过加速度传感计和陀螺仪共同完成，此种情况下测量结果的精确性更高。

3.6.2 加速度传感器的硬件设计

论文选用一款三轴低 g 半导体模拟加速度计 MMA7361，可同时输出三个方向上的加速度模拟信号，各轴信号最大输出灵敏度为 800mv/g，文中采取测量 Z 轴方向上的加速度值，来计算出车模倾角。当车模直立时，将加速度器固定在 Z 轴水平方向，此时输出信号为零偏电压信号。当车模发生倾斜时，重力加速度 g 便会在 Z 轴方向产生加速度分量，从而会使该轴的输出电压发生变化。而当车模倾角较小时，该倾角与输出电压的变化近似成正比例的关系。MMA7361 是本论文选用的加速度传感器，它是一款三轴低 g 值的半导体加速度器，它可将车模三个方向上的加速度模拟信号同时输出。如图 3.9 所示：

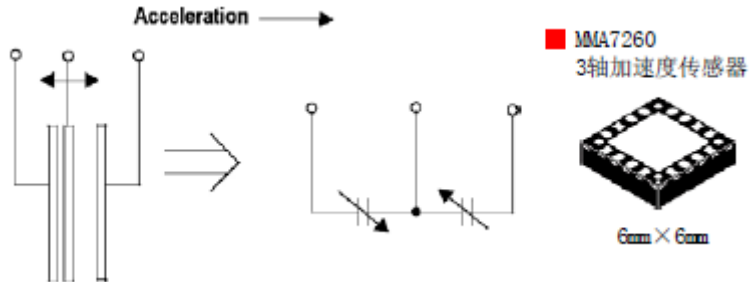

图 3.9　MMA7261 三轴加速度传感器

加速度器 MMA7361 是一款低 g 值的传感器，它的输出信号很大，不再需要额外放大电路，此信号可直接输入到单片机进行 A/D 转换。加速度器的电路图如 3.10 所示：

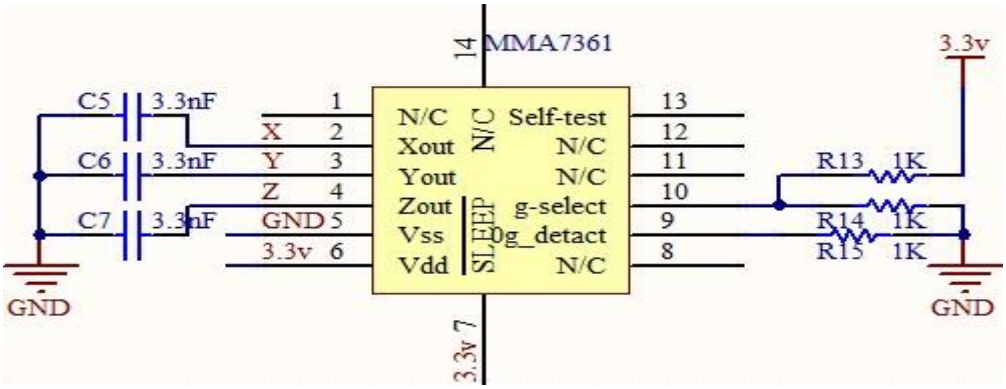

图 3.10　加速度计电路图

3.6.3　陀螺仪的硬件设计

陀螺仪论文选取村田公司出品的 ENC-03 系列，它利用了物体在旋转坐标系中会受到科里奥利力的原理，在器件中利用压电陶瓷做成振动单元，当旋转器件时会改变振动频率从而反映出物体旋转的角速度。 ENC-03 是一款可以用来测量物体旋转角速度的模拟陀螺仪，它可将车模前后方向上的角速度信号输出，且不会受到车体振动影响，将测得的车模倾斜角速度信号进行积分运算后，便可得到车模倾角。 论文选用的此款陀螺仪灵敏度为 0.67mv/deg/s，经测试，当小车的摆动幅度很大时，输出的电压范围很小，造成测量的精确性不够，而且输出信号受温漂的影响较大。所以文中将陀螺仪的输出信号进行 10 倍的放大处理，并将零点偏置电压相应的调整到工作电压的一半（1.65V）左右。然后将此信号输入到单片机中，利用软件部分来完成角度和角加速度的计算[27]。陀螺仪的放大电路图如 3.11 所示：

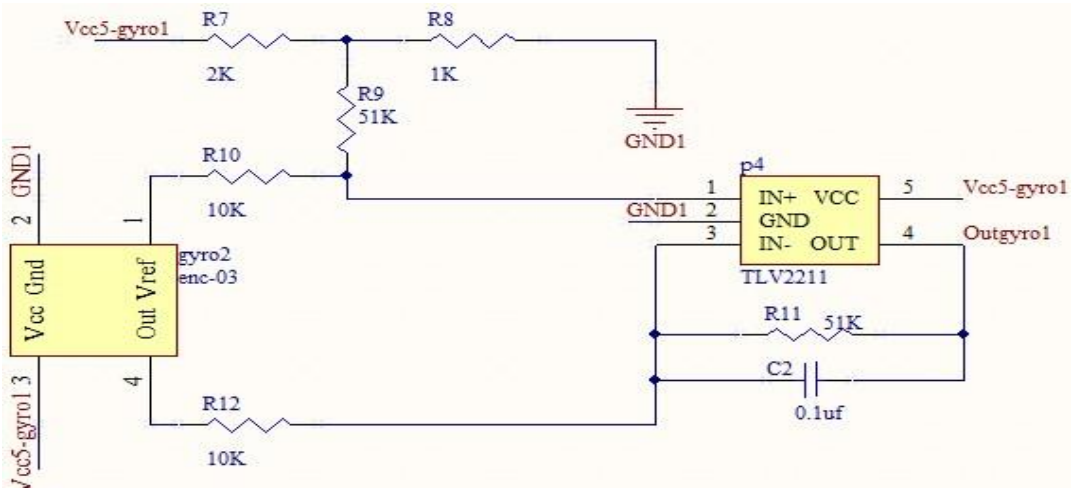

图 3.11　陀螺仪的放大电路图

3.7　速度检测模块

车模转速控制系统采用闭环控制，该闭环的反馈量是电机的转速，控制系统性能的良好发挥主要取决于转速测量的实时性及准确性，转速测量的精度也在很大程度上决定着系统的测量精度。在电机的实际转速测量中，其测量精度的精确性主要受以下两个因素影响：一是采样频率，为了提高采样数据的准确性，一般采样频率设定较高的值；二是采样点的多少，为了提高速度测量结果的精确度，通常采集较多的采样点，此方法在低转速的测量中优势更为明显。电机转速的数字检测是利用与电机同轴相连的光电脉冲发生器的输出脉冲频率与其转速成正比例的关系，同时在由脉冲发生器发出的脉冲速度和序列，来测量电机转速并判别电机的转动方向。常用的电机转速的测量主要有以下三种：测频法，测周期法和测频率/周期法。测频法的工作原理：是在一定时间间隔内对光电码盘的输出脉冲数目进行计数，然后计算出电机转速，此方法适合在电机转速较高的场合下使用；测周期法的工作原理：是对光电码盘产生的脉冲周期进行测量然后计算出电机转速；频率/周期法的工作原理：是同时测量出检测时间和在此时间周期内光电脉冲发生器发出的转速脉冲信号的数目然后来计算出电机转速。

测频法测量转速在极端情况下会产生±1 个转速脉冲的误差，所以只有被测转速比较大时，再能保证测量的精度。测周期法在被测电机转速较低时，再能保证其具有较高的测量精度，因为在被测电机转速较高的情况下，时间的测量将会产生±1 个高频脉冲周期的误差。频率/周期法在高速和低速时都能保证较高的测速精度，主要是此方法是同时对两种脉冲信号进行计数，所以只要保证“同时性”的实现既能完成测速任务。

本文选取的光电编码器测速时为了同时兼顾高低转速下的精确度，论文采用将测频

法和测周期法相结合的测速方案，即测频法应用在电机高转速状态下，测周期法在电机转速较低的情况下使用。特别是在车轮不发生打滑的情况下，电机的转速与车速成正比例关系。

论文设计时，将测速部分使用的光电编码器通过齿轮与主齿轮连接起来，通过查阅选型手册，选定了 157 线的日本内密控 NEMIC 。线数是编码器的分辨率，即一转所产生的脉冲数目，由于编码器没有倍频技术，是接收器处理脉冲时通过编码器的输出脉冲（A 与 B 相）的相位差关系实现倍频技术的。在单位时间内累计编码器输出的脉冲数，计算车模的行驶速度只需知道编码器单位时间内输出的脉冲数目即可。此外，车模当前行驶过的距离计算也可通过累计编码器脉冲值得到，同时为赛道的记忆提供信息支持。本文采用 157 线的光电编码器，将编码器安装在驱动大齿轮的后方并与之啮合良好。这种方法的优点是测速精度高，安装相对较容易，缺点是质量较大。在车体中具体的安装位置如图 3.12 所示：

图 3.12　编码器的安装位置图

3.8　LCD、键盘、存储芯片系统设计

论文设计时加入液晶显示模块，它可在一定程度上完善智能车控制系统的功能，它能使其具有更加人性化，并方便调试。目前电子信息产品的显示器件主要应用的是液晶显示器（Liquid Crystal Display, LCD），本论文的电子显示器件选用的是诺基亚公司生产的 NOKIA5110。该显示器为 48*84 的点阵 LCD，可以显示 4 行汉字，采用串行接口与 ECU 进行通信，接口信号线有 8 条，支持 SPI 串行外设接口[25]。本文采用普通 I/O 口软件模拟 SPI 通信。各引脚及功能如表 3.1 所示：

表 3.1 LCD 引脚名称

引脚名	功能	备注
GND	接地	
VCC	电源输入脚	3.3V
SCLK	同步时钟输入	最高可达 4Mbps
SDIN	数据输入	时钟上升沿采样
D/C	数据/命令切换	0：命令；1：数据
SEC	片选型号	低电平有效
RST	LCD 复位信号	低电平有效
LIGHT	背光	
OSC	外部时钟输入	

NOKIA5110 与单片机的电路连接图如 3.13 所示：

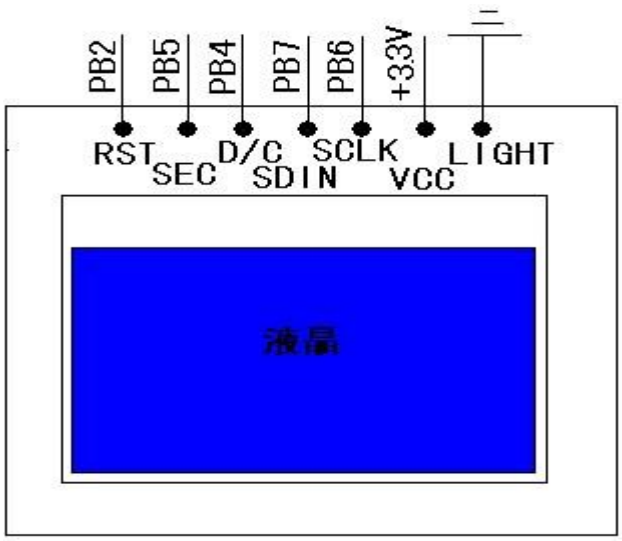

图 3.13 NOKIA5110 与单片机的电路连接示意图

键盘是最简单的 MCU 数字量输入设备，本文共设 8 个键组成一个键盘，键盘的排布方式为独立式，MCU 通过检测与键盘相连的 I/O 口来确定键盘状态。键盘的排布方式为独立式，对应的 I/O 口为 A0，A1，A2，A3，A4，A5，A6，A7。电路原理图如 3.14 所示：

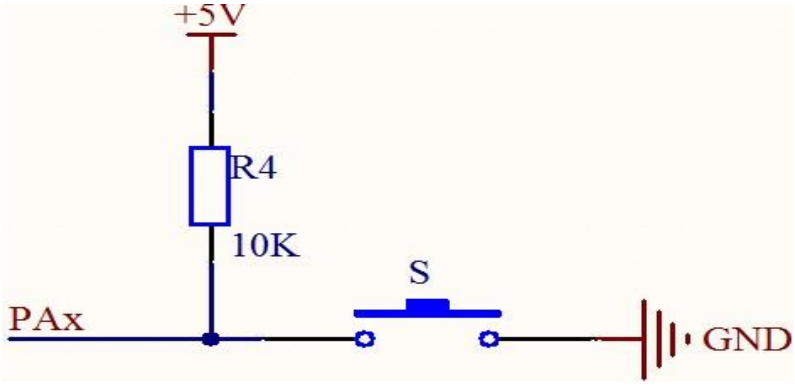

图 3.14　按键连接电路图

当开关断开时，PAx 上拉，处在高电平；开关闭合时，PAx 接地，产生一个低电平脉冲，程序检测到该脉冲信号后，开始执行信号对应的程序。按键的作用分别为换行、保存、+1、−1、+10、−10、+100、−100。

本论文的存储芯片选用 Atmel 公司制造的 AT24C02，它具有低功耗 CMOS 型 E2PROM，内含 8×256 位的存储空间，具有的工作电压宽为(2.5～5.5 V)、具有擦写次数比较多、写入速度比较快、抗干扰能力比较强、数据不易丢失等特点[26]。而且它是采用 I2C 总线式来进行数据读写的串行器件，占用较少的资源和 I/O 线，同时支持在线编程，可十分方便的进行数据实时存取。AT24CO2 与单片机的电路连接图如 3.15 所示。

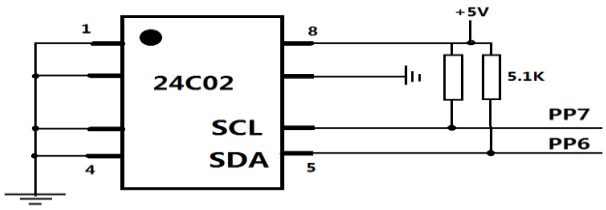

图 3.15　AT24C02 与单片机的电路连接图

芯片的 1，2，3 引脚是三根地址线，用来确定芯片的硬件地址。第 5 引脚 SDA 为串行数据线，数据可通过这根双向 I2C 总线串行传送。第 6 引脚 SCL 为串行时钟，SDA 和 SCI 为漏极开路端，在实际的应用当中都需要和正电源间各接一个 5.1k 的上拉电阻。第 7 脚为 WP 写保护端，接电源正极时只允许器件的读操作执行，接地时允许芯片一般读写操作的执行。

I2C 总线简述：I2C 总线的数据传输是使用两根信号线来进行的，这两根信号线分别是：串行时钟线(SCL)和串行数据线(SDA)，总线上所有器件均可依靠 SDA 发送的地址信号来进行寻址。

本文通过软件来模拟 I2C 总线时序实现数据的传输。从 24C02 中读取数据的结构图如 3.16 所示：

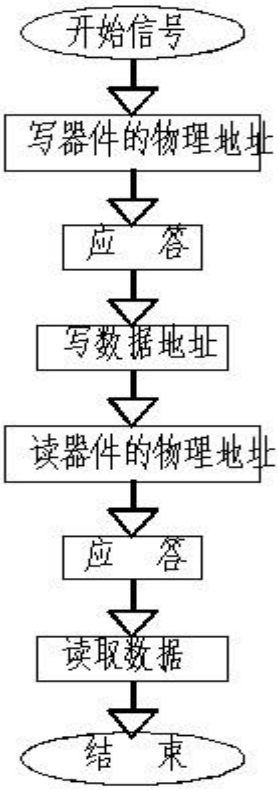

图 3.16　从 24C02 中读取数据的结构图

3.9　本章小结

本章主要对 16 位单片机 MC9S12XS128 进行介绍，包括其内部资源及单片机最小硬件系统，该部分的介绍能加深对单片机的理解更利于后续软件的编写。然后重点介绍该智能车的各个硬件模块并对各个模块进行电路的设计，主要包括电源管理模块，倾角测定模块，电机驱动模块，测速模块以及人机交互和调试模块，这些硬件模块性能的优异直接决定着后续编写的软件能否真正发挥功用。

第四章　两轮自平衡智能车的控制算法

4.1 系统的开发环境及总体结构

系统软件的开发环境选用 Metrowerk 公司专门为 freescale 公司开发的集成 CodeWarrior 开发环境。它是 Freescale 公司开发的专门面向 Freescale MCU 及 DSP 嵌入式应用开发的一款商业软件工具，XS128MCU 的汇编语言、C 语言和 C++语言程序，在此环境下均可进行编制和调试[20]。由于 C 语言具有易读性强、移植性好以及容易维护等特点，因此，本文软件设计主要采用 C 语言来进行编程。CodeWarrior 开发环境主要包括以下功能模块：工程管理器、编辑器、调试器、源码浏览器、构造系统、搜索引擎。其中软件开发工程的四个主要阶段对应的分别是编辑器、编译器、连接器和调试器。而代码浏览和构造控制则需要其它模块的支持再能完成工作，工程管理器是对整个过程进行控制，CW 环境能跟踪源码的变化，进行自动编译和连接。

整个系统的程序总体结构如图 4-1 所示。主要有以下几部分组成：主程序、初始化程序、中断程序及功能函数，其中功能函数是整个系统的核心部分，它主要包括车模直立控制算法、车模速度控制算法、车模方向控制算法、陀螺仪和加速度计的卡尔曼滤波算法等[21,22]。单片机系统通过接收倾角传感器信号、电磁传感器信号、车速传感器的信号，采用 PID 控制算法进行控制，进而实现车模的直立控制，速度控制、方向控制。

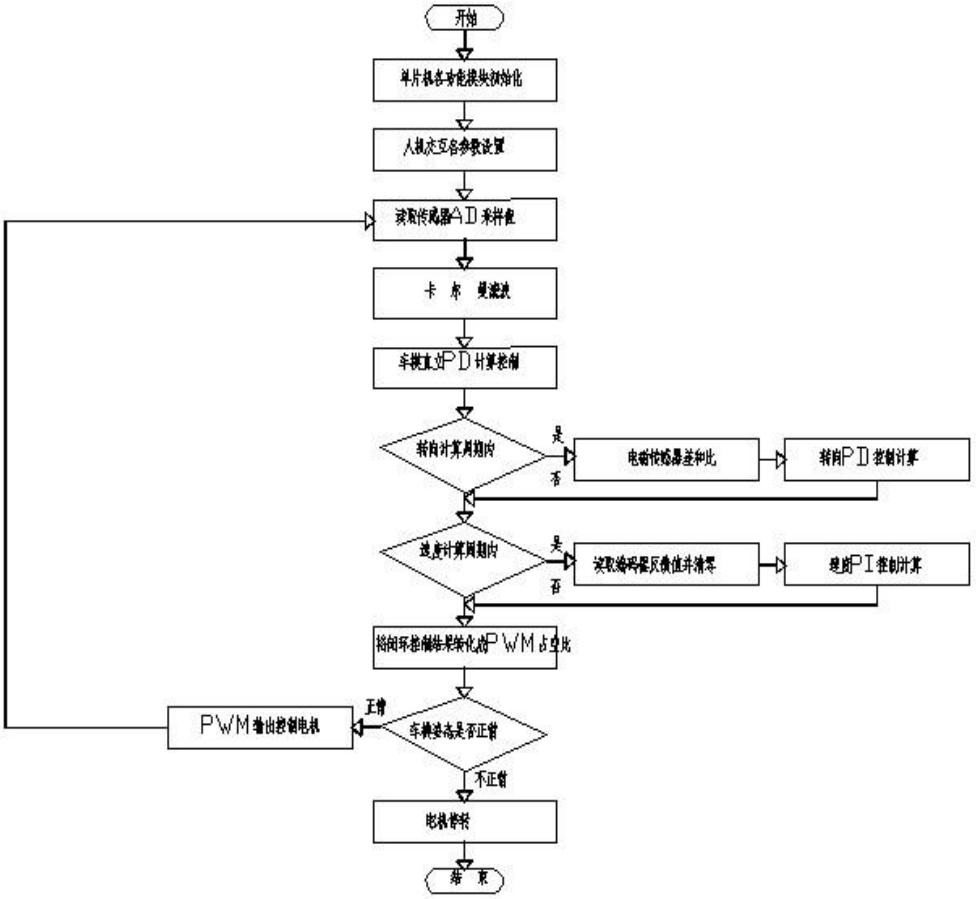

图 4.1 系统的程序总体结构图

4.2 系统的初始化设置

本论文系统设计中，用到单片机以下几个基本功能模块：PWM 模块、PIT 模块、ECT 模块、ATD 模块、串口通信模块及普通 I/O 口模块。系统根据实际需求，对各个模块进行初始化配置并通过对相应状态寄存器或数据寄存器的读写，实现相应的功能[24]。

4.2.1 PWM 模块

电机的控制信号为脉宽调制信号，本论文使用 16 位单片机 MC9S12XS128 的核心板拥有 8 路独立的可设置占空比和周期的 8 位 PWM 通道，并且每个通道都配备了专门的计数器装置。该模块共有 4 个时钟源，能分别控制 8 路信号，并通过配置寄存器来设置 PWM 的使能与否、每个通道的工作脉冲极性、每个通道的输出对齐方式、时钟源以及使用方式(四个 16 位通道还是八个 8 位通道)。本论文中通过级联提高占空比精度，共用到四路 PWM 输出分别控制两个电机的正反转。

具体设置如下：

```
void PWM_Init(void)
```

```
{
   PWME=0x00;
   PWMPRCLK=0x33;
   PWMSCLA=0x02;
   PWMSCLB=0x02;
   PWMCTL=0xf0;
   PWMCLK=0x00;
   PWMCAE=0x00;
   PWMPOL=0xff;
   PWMPER67= 300;
   PWMPER45 =300;
   PWMDTY45 =0;
   PWME =0xaa;
 }
```

4.2.2　ECT 模块

增强型捕捉定时器模块，它可通过一系列可供设置的可供读写的数据寄存器和控制寄存器来进行扩展端口功能，ECT 模块的两大功能分别是实现输入捕捉和输出波形。在本论文中，主要应用 ECT 模块检测车速。论文应用的 16 位 MC9S12XS128 单片机中内置 8 路事件捕捉通道，文中主要使用两路事件捕捉通道来对编码器进行计数，最终并将其转化为车速。

具体设置如下：

```
void ECT_Init(void)
{
    TIOS_IOS0=0;
    TIOS_IOS1=0;
    TIOS_IOS2=0;
    TSCR1_TEN=1;
    TCTL4=0X15;
    TIE=0X05;
    TSCR2_TOI=0;
}
```

4.2.3 ATD 模块

MC9S12XS128 单片机内置 2 组精度可设置为 8 位或 10 位的 A/D 模块：ATD0 和 ATD1。该模块具有 16 路通道，可自行设置单次转换时间、转换结果类型、转换完成是否产生中断、转换序列长度等。在论文中，共用到了 6 路 AD 通道，2 路用于加速度计 MMA7361；2 路应用于陀螺仪 ENC-03；电磁传感器 2 路。采用 8 位的采样精度和连续转换的方式进行工作。

具体设置如下：

```
void AD_Init(void)
{
   ATD0CTL1=0x40;
   ATD0CTL2=0x40;
   ATD0CTL3=0x80;
   ATD0CTL4=0x01;
   ATD0CTL5=0x30;
   ATD0DIEN=0x00;
}
   int ReadATD(byte ch)
{
   int ad=0;
   DisableInterrupts;
   while(!ATD0STAT0_SCF);
   switch(ch)
   {
      default:
      case 0:
         ad= ATD0DR0;
      break;
      case 1:
         ad= ATD0DR1;
      break;
      case 2:
         ad= ATD0DR2;
```

```
        break;
        case 3:
           ad= ATD0DR3;
        break;
        case 4:
           ad= ATD0DR4;
        break;
        case 5:
           ad= ATD0DR5;
        break;
        case 6:
           ad= ATD0DR6;
        break;
        case 7:
           ad= ATD0DR7;
        break;
        }
      EnableInterrupts;
    return ad;
    }
```

4.2.4　PIT 模块

论文使用的 MC9S12XS128 单片机提供实施中断模块，它是一个模数递减计数器。工作原理是首先给计数寄存器设定初值，则当每经过一个总线周期时，计数器便进行一次减 1 操作，而在计数器自减溢出时触发中断。由于总线周期是设定的，即可以通过计数器自减实现定时，本论文利用该模块来实时终端模块计数并提供时钟信号[23]。

具体设置如下：

```
void PIT_init(void)
{
   PITCFLMT_PITE=0;
   PITMUX_PMUX0=0;
   PITMUX_PMUX3=1;
   PITMTLD0=160-1;
```

```
    PITLD0=500-1;
    PITCE_PCE0=1;
    PITINTE_PINTE0=1;
    PITMUX_PMUX1=1;
    PITMTLD1=160-1;
    PITLD1=5000-1;
    PITCE_PCE1=1;
    PITCFLMT_PITE=1;
}
```

4.2.5 IO 模块

论文选用的 MC9S12XS128 单片机内置丰富 I/O 资源，其中大部分 I/O 引脚可由相应的寄存器位来配置选择数据方向、驱动能力，使能上拉或下拉式装置。当用作通用 IO 口时，所有的端口都有数据寄存器和数据方向寄存器。

文中用到 A 口和 B 口，其中 A 口与 MCU 数字量输入设备键盘相连接；采用 B 口软件模拟 SPI 通信，来使液晶显示器工作；端口 P 与数据存储元件 AT24CO2 的各引脚相连。

4.3 卡尔曼算法设计

4.3.1 倾角传感器误差的分析

加速度器：在车模上安装加速度器，加速度器能将 Z 轴方向上的重力加速度分量值输出，并根据分量值和反三角函数关系计算得到倾角值。

陀螺仪：在车模上安装陀螺仪，将输出车模在其旋转方向上的角速度信号。通过对角速度进行积分即得到车模倾斜的角度。

原理上车模的姿态信息可通过加速度器和陀螺仪计算得到。但在对陀螺仪和加速度器的实际性能测试中，发现在实际车模运行过程中，由于车模本身的运动所产生的加速度会产生很大的干扰信号叠加在上述测量信号上，使得输出信号无法准确反映车模的倾角。而两轮自平衡车模对倾角的测量有较高要求，所以该噪声已经影响了输出的信号的精确性。陀螺仪自身的输出精度较高，但是对陀螺仪的长时间测试中，发现当输入固定时，陀螺仪输出存在漂移现象。如果该漂移值不加处理会严重影响信号的可靠性，而且角速度经过积分运算后，变化形成积累误差。该误差会随着时间延长逐步增加，最终导

致电路饱和，无法形成正确的角度信号。所以，加速度器和陀螺仪都不能单独的用于检测两轮自平衡车模的姿态信息。本论文采用 kalman 滤波对加速度器和陀螺仪的数据进行有效融合，以得到准确的倾角信息[28,29]。

4.3.2 卡尔曼滤波算法设计

1960 年，R.E.Kalman 在论文（A New Approach to Linear Filtering and Prediction Problems）中介绍一种应用于离散线性滤波的迭代算法，这既是后来被广泛应用的著名卡尔曼滤波方法[30]。 由于测量值中混杂有噪声，而且系统的行为也会受到随机干扰，卡尔曼滤波的目的就是将噪声的影响减少到最低，并且从含有噪声的测量值中得到系统状态最优估计。同时卡尔曼滤波的解是递归计算的，其状态的每一次更新估计都需要前一次估计和新的输入数据计算得到，所以只需存储前一次估计，这样提高计算机的计算效率。论文在考虑卡尔曼滤波的优点后，决定应用该滤波对倾角及角速度状态进行估计。

卡尔曼滤波器解决线性离散时间控制系统的状态估计问题，一般而言这个系统可用离散线性差分方程描述。

（1）线性离散系统状态方程。

$$x(k) = Ax(k-1) + B\mu(K-1) + w(k-1) \tag{4-1}$$

式中$x(k) \in R^n$...k 时刻系统的状态变量

$\mu(k)$.....k 时刻系统的控制变量

$w(k) \in R^n$.....过程噪声，可建模为零均值的白噪声过程，且其相关的矩阵定义：

$$E[w_1(n)w_2(k)^H] = \begin{cases} Q_1(n)\cdots\cdots n = k \\ 0\cdots\cdots\cdots\cdots n \neq k \end{cases} \tag{4-2}$$

$A \in R^{n,n}$.....状态转移矩阵；

B.....输入控制矩阵；

(2) 传感器测量方程

$$z(k) = Hx(k) + v(k) \tag{4-3}$$

式中$z(k) \in R^n$.....k 时刻传感器的测量向量

$v(k) \in R^n$.....观测噪声，可将其建模成零均值的白噪声过程，且相关矩阵定义是：

$$E[\mathrm{v}_1(n)v_2(\mathrm{k})^H]=\begin{cases}R_1(\mathrm{n})\cdots\cdots n=k\\0\cdots\cdots\cdots\cdots n\neq k\end{cases} \tag{4-4}$$

H-------观测矩阵

（1）初始状态的描述

状态方程从初始状态$x(t_0)$开始传播，对于全部的真实系统的特定时刻来说，该初值是一个固定的具体向量。但是，由于该具体值事先并不能得知，所以实际建模时应该把此初始状态当作能满足高斯分布的一个随机向量。同时均值$\mathbf{x_0}$和方差$\mathbf{p_0}$可表示$x(t_0)$

$$\begin{cases}E\{\mathrm{x}(t_0)\}=x_0\\E\{[\mathrm{x}(t_0)-x_0][x(t_0)-x_0]^T\}=P_0\end{cases} \tag{4-5}$$

P_0是所有元素都是分布在对角线的正数的对称矩阵，它能把估计状态和真实状态之间的方差给出。每个状态和真值的方差可由对角线上的元素来表示。

（2）卡尔曼算法

卡尔曼滤波算法是利用反馈控制机制来实现对状态的估计，它根据前一时刻的系统状态来进行当前系统状态的估计，然后根据当前时刻的实际观测值作为反馈，来修正估计的状态。因此，卡尔曼滤波算法具有以下两个步骤：状态预测和测量修正。根据当前系统状态和噪声方差，状态更新方程能及时地把下一步的系统状态估计出来。而测量更新方程则负责反馈，它是将新测到的信号加入到先前在状态更新方程中得到的先验估计状态，并最终得到系统状态的后验估计。状态更新方程可被视为状态预计方程，而测量更新方程则可当成状态修正方程[31]。卡尔曼滤波具体程序如下：

```
void Kalman_Filter(float angle_m,float gyro_m)
{
  NowData = RealData + gyro_m*0.01;
  NowData_P = sqrtf(Q*Q+RealData_P*RealData_P);
  Kg = sqrtf(NowData_P*NowData_P/(NowData_P*NowData_P+R*R));
  RealData = NowData+Kg*(angle_m-NowData);
  RealData_P = sqrtf((1-Kg)*NowData_P*NowData_P);
  angle= RealData;
  angle_dot=gyro_m;
}
```

4.4 智能车控制算法研究

智能控制起源于 20 世纪 70 年代，智能控制是人工智能与控制论相结合的产物，它是自动控制的一种高级控制技术。虽然智能控制的目标仍是实现自动控制，但其主要是通过“知识”和“推理思考”来实现智能控制的，其中最重要的是模仿人的智能，主要是利用人类现有的控制知识和推理方法来实现智能控制。目前在智能控制方面比较成熟的技术主要包括：PID 控制、模糊控制、神经网络控制、专家系统控制等控制类型[20,32]。

本论文主要应用比较成熟且应用广泛的 PID 算法。PID 控制器是一种线性控制器，它根据事先的设定值与实际输出值来构成控制偏差。PID 的控制量是将偏差的比例(Proportional)、积分（Integral）和微分(Derivative)通过线性组合来构成，对被控对象进行控制故称 PID 控制器，原理框图如 4.2 所示。

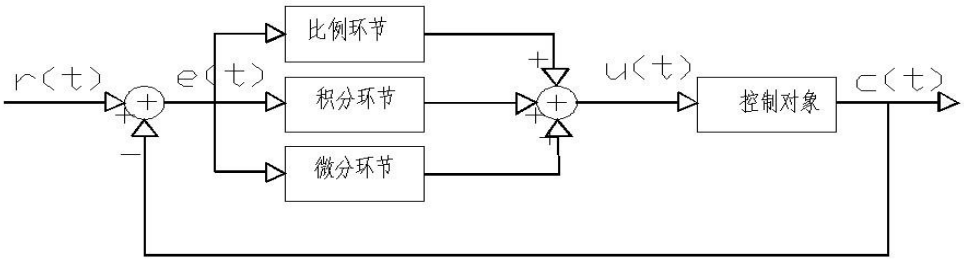

图 4.2　PID 控制原理图

式中 $r(t)$ 是希望得到的控制对象的取值；$c(t)$ 是对象变量；$e(t)$ 是偏差值；$\mu(t)$ 是控制变量。在计算机控制系统中，使用的是数字 PID 控制器，其控制规律如下：

$$u(t) = K_p e(t) + K_I \int_0^t e(t)dt + K_D \frac{de(t)}{dt} \tag{4-6}$$

$$e(t) = r(t) - c(t) \tag{4-7}$$

相应的传递函数为

$$G(s) = \frac{U(s)}{E(s)} = K_P + \frac{K_I}{s} + K_D s \tag{4-8}$$

其中，最简单的控制方法是比例控制，其特点是控制器的输出与输入误差信号成一定比例关系。当偏差一旦产生，控制器就会立刻发生作用来进行控制输出的调节，其目的是使被控量朝着减小偏差的方向变化，比例因子 k_p 的大小决定着偏差减小的速度快

慢。k_p 值越小偏差减小的越慢，不易发生振荡，但是达到稳态需要的时间较长；k_p 值越大偏差减小的越快，发生振荡的可能性越大。单纯的比例控制的缺点是其存在一个不能消除的稳态误差[33,34]。

积分控制，其特点是输入误差信号的积分值与控制器的输出成正比例关系。当一个系统在进入稳态之后存在稳态误差，在控制器中加入“积分项”后便能消除此稳态误差。而积分项是误差对于时间的积分，所以随着时间的增加，积分项也将会变大。误差“积少成多”使得控制器输出信号增大，进一步减小误差，直到误差为 0。

微分控制，其特点是输入误差信号的微分值与控制器的输出成正比例关系，也即是误差的变化率与控制器的输出成正比例关系。较大的惯性环节能使自动控制系统在消除误差的调节过程中可能会出现振荡甚至失稳的现象，所以其变化总是落后于误差的变化。微分控制能提前预测误差变化的趋势，提前抑制误差，从而避免被控量的严重超调。

4.4.1 智能车直立的控制算法

直立控制首先需要根据加速度器和陀螺仪的值得到车体的倾角和角速度值。可知加速度器的量程为 1.5g，800mv，陀螺仪的灵敏度为 0.67mv/deg/s。由于陀螺仪和加速度器均为模拟器件，故先将 AD 读到的电压值减去零偏置电压，再通过一定的关系分别转换成角度和角速度。将该角度和角速度的值送入卡尔曼滤波器则得到准确的角度和角速度。对经过卡尔曼滤波滞后的角度和角速度进行 PD 控制可使车模基本实现直立[39]。车模直立控制流程图如 4.3 所示：

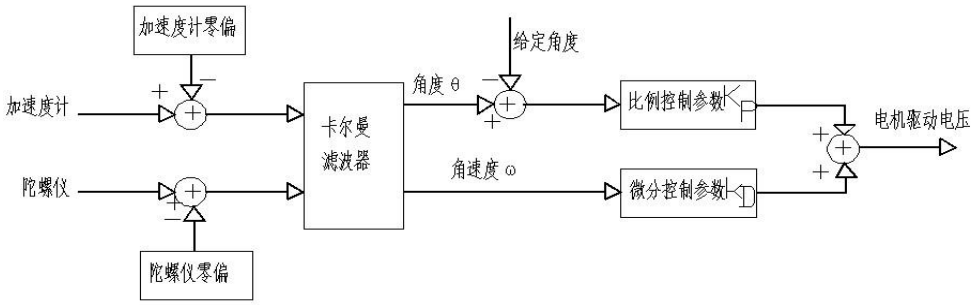

图 4.3 车模直立控制流程图

PD 控制器是比例-微分控制的简称。在闭环控制系统中，PD 调节器的控制作用是在保证系统稳定的前提下，来使偏差达到最小。在连续系统中，PD 控制器的输入输出均是时间的函数，将其离散化后的运算关系为

$$M_n = k_p \times (SP_n - PV_n) + k_D \times (PV_{n-1} - PV_n) \quad (4\text{-}9)$$

式中：M_n为对控制量的调节量；　k_p为比例控制系数；k_D为微分控制系数；SP_n为控制量的目标状态；PV_n为调节量在第 N 个采样周期的实际状态。

比例控制参数k_p相当于回复力，能克服重力的作用使车模保持直立，k_p参数一般应大于重力加速度的等效数值。k_p的值越大越能增加车模维持直立的倾向，但过大会引起车模的震荡。所以单靠比例控制难以使系统维持稳定。

微分控制参数k_D的作用相当于阻尼力，在抑制车模震荡方面能产生较好的效果，并能克服震荡的自由能量的消失，但该值过大则会使车模产生高频的抖动。一般相对k_p参数来说都较小。

论文中车模直立平衡控制的程序：

```
void CarAngleAdjust(void)
{
float fValue;
fValue=(0-gyroscope_rate)*ANGLE_CONTROL_D
+(0-g_nCarGyroVal+SubValue)*ANGLE_CONTROL_P;
if(fValue>ANGLE_CONTROL_OUT_MAX)
    fValue=ANGLE_CONTROL_OUT_MAX;
if(fValue<ANGLE_CONTROL_OUT_MIN)
    fValue=ANGLE_CONTROL_OUT_MIN;
    g_fAngleControlOut=fValue; }
```

4.4.2 智能车速度的控制算法

本论文速度控制采用 PI 即是比例-积分控制。而具有比例-积分控制规律的控制器称为 PI 控制器，同时其输出信号$\mu(t)$能成比例的反应出输入信号$e(t)$及其积分，即：

$$\mu(t) = k_p\left[e(t) + \frac{1}{T}\int_0^t e(t)dt\right] \qquad (4\text{-}10)$$

式中k_p为比例控制参数，T_i为可调积分时间常数；PI 控制器的结构图如4.4所示 ：

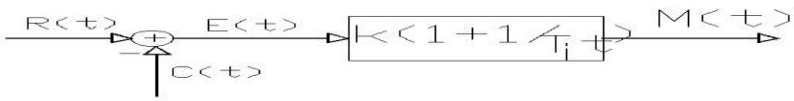

图 4.4　PI 控制器结构图

文中为使车模在速度为零时保持直立，必须使控制移动的距离最终达到零。所以车

模速度控制在比例控制的基础上还添加了积分控制，这样能保证车模保持平衡状态运动[35]。维持车模空速的前提是要能够实时知道车模的运动速度，文中通过安装在车模上的光电编码器，捕获其脉冲数从而算出速度。由第二章速度控制模块的介绍可知车模的速度最终由倾角所产生的加速度决定，速度的给定会影响到车模往前倾的角度。因此将倾角的进行积分便可以得到车模的运动速度，结合车模角度的闭环控制，将角度控制闭环与速度控制闭环叠加一起加载到电机上便可实现对车模运动速度的控制。车模速度控制框图如图4.5所示。

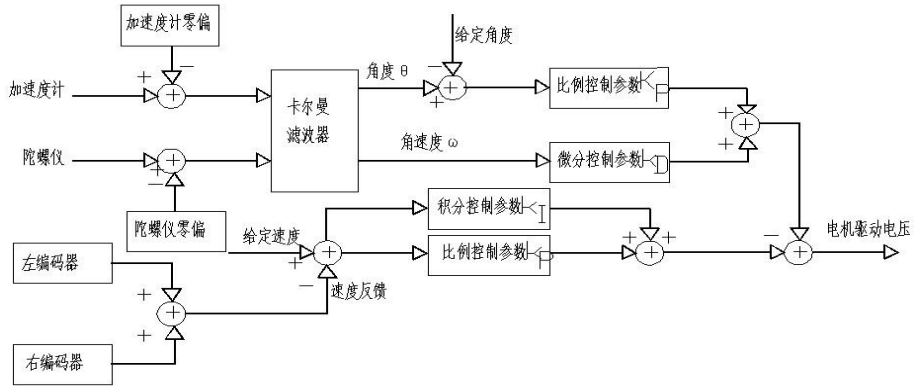

图 4.5　车模速度控制框图

图中速度比例控制参数k_p可以消除车模来回摆动，使其静止在某一点，提高k_p参数可以提高系统抗外部干扰的能力。但k_p过大，会削弱比例控制参数的作用，从而使角度产生震荡。

速度积分控制参数k_I的作用，对速度进行积分，通过积分的误差来抵消车模的运动倾向，阻碍其向一个方向发生倾斜。k_I参数越大，停止在某一点的速度越快，但k_I过大会在某一点来回摆动。

文中将速度控制周期设定为 100 毫秒。将读取的两个电机的光码盘脉冲进行累加，利用 100 毫秒的脉冲数量可以反映车模的电机的转速。车模速度采用两个电机速度的平均值。具体程序如下：

```
void SpeedControl(void)
{
float    fDelta;
float fP, fI;
speed_left=g_nLeftMotorPulseSigma;
speed_right=g_nRightMotorPulseSigma;
```

```
g_fCarSpeed = (g_nLeftMotorPulseSigma + g_nRightMotorPulseSigma) / 2;
g_nLeftMotorPulseSigma=0;
g_nRightMotorPulseSigma=0;
g_fCarSpeed =g_fCarSpeed * CAR_SPEED_CONSTANT;
fDelta = CAR_SPEED_SETfDelta - g_fCarSpeed;
fP = fDelta * SPEED_CONTROL_P;
Speed_P=fP;
fI = fDelta * SPEED_CONTROL_I;
if((g_fSpeedControlIntegral+fI)>-60000&&(g_fSpeedControlIntegral+fI)<60000)
g_fSpeedControlIntegral += fI;
Speed_I=g_fSpeedControlIntegral;
g_fSpeedControlOutOld = g_fSpeedControlOutNew;
g_fSpeedControlOutNew = fP + g_fSpeedControlIntegral;
}
void SpeedControlOutput(void)
 {
float fValue;
fValue = g_fSpeedControlOutNew - g_fSpeedControlOutOld;
g_fSpeedControlOut = fValue * (count1)
if(g_fSpeedControlOut>MOTOR_SPEED_SET_MAX)
      g_fSpeedControlOut=MOTOR_SPEED_SET_MAX;
 if(g_fSpeedControlOut<MOTOR_SPEED_SET_MIN)
g_fSpeedControlOut=MOTOR_SPEED_SET_MIN;
}
```

4.4.3 智能车转向的控制算法

根据第二章车模方向控制原理可知，方向控制是根据车模检测到的电磁感应电压来生成电机差动控制量。通过左右电机速度差驱动车模转向消除车模距离道路中心的偏差。为了逐步消除车模距离道路中心线的距离差可通过车模方向的调整以及加上车的前行运动来完成。该过程是一个积分过程，因此简单的比例控制就可以进行车模差动控制并从而完成车模的方向控制。但考虑到车模自身安装的电池等比较重的物体，产生较大的转动惯量，所以车模在调整过程中容易出现转向过冲现象，此现象如果不加以抑制，车模在很大程度上会冲出赛道。为了消除车模方向控制中的过冲，并结合前面车模直立

和速度控制的经验，论文中增加了微分控制来抑制此现象的发生[36,37]。

微分控制就是根据车模方向的变化率对电机差动控制量进行修正的控制方式。同时为了更能准确反映车模重心距离电磁线的距离差别，避免角度的影响，在进行方向控制时，使用左右两个线圈感应电动势之差除以左右两个线圈感应电动势之和得一比值，使用该比值进行方向控制，可以消除检测线圈角度的影响。论文采用 PD 控制来对方向进行控制结构图如 4.6 所示：

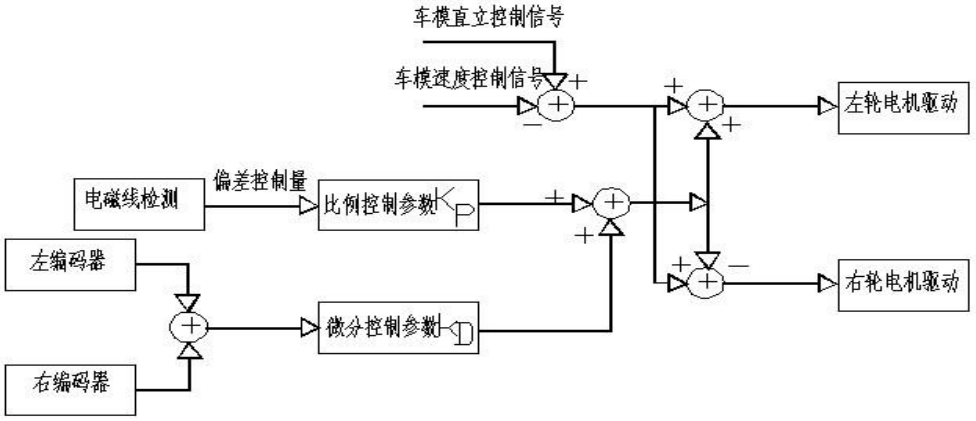

图 4.6　方向 PD 控制结构图

方向控制参数 k_P 能迅速减小偏差，使系统快速跟踪给定，决定于系统的响应速度和快速性。k_P 参数太小，系统的控制作用太弱不利于克服外界扰动，系统反应迟钝，延长调节时间长，转弯不及时。增加 k_P 参数可以尽快消除车模与中心线的偏差，但 k_P 过大系统容易产生超调。

方向控制参数 k_D 的作用：微分控制能够按照偏差的趋势进行控制，提高系统的动态性能。在偏差未引起太大变化之前，首先在系统中添加一个有效的早期修正信号，该修正信号起到提前纠正的作用，且它的控制量大小与偏差的变化快慢成正比例关系。所以它既能加快系统的响应速度同时能克服震荡的产生，最终使系统的稳定性更高。它能起预测前瞻和迅速调整车身方向的作用，加快响应速度，消除振荡，提高系统的稳定性，但是 k_D 过大会导致系统频繁抖动，且对噪声很敏感。

具体程序如下：

```
void DirectionControl(void)
 {
      float sumxy=0;
      float sumy=0;
```

```
float sumx;
float fValue,fDValue;
for(i=0;i<4;i++)
{
 sumxy+=dif_x[i]*(AD_wData[i]-Min_AD[i]);
 sumy+=(AD_wData[i]-Min_AD[i]);
}
if(sumy>0)
 sumx=sumxy/sumy;
else
 sumx=0;
 g_fDirectionControlOutOld= g_fDirectionControlOutNew;
 minx=0;
for(i=0;i<4;i++)
 {
    if(AD_wData[i]>minx)
    {
          maxi=i;
          minx=AD_wData[i];
     }
 }
 dif_old=dif;
if(minx>=setnum[5]*0.1)
{
   dif=sumx*DIR_CONTROL_P+pow(sumx,2)/setnum[7];
 if(dif>500)    dif=500;
 if(dif<-500)    dif=-500;
 fValue=dif;
}
else
{
        if(dif_old>1)
             fValue=500;
        else if(dif_old<-1)
```

```
                fValue=-500;

            dif=fValue;
        }
        Gyro_Angle=fValue;
        fDValue=Dir_AD_gyro-Dir_Gyro_Zero;
        OutData[3]=fDValue;
        fDValue*=DIR_CONTROL_D;
        Gyro_Dir=fDValue;
        fValue+=fDValue;
    g_fDirectionControlOutNew=fValue;
    g_fDirectionControlOutNew=fValue*setnum[5]*0.1+abs(fValue*fValue)/setnum[7]+
    (fValue-g_fDirectionControlOutOld)*setnum[5]*0.01;
    if(g_fDirectionControlOutNew>DIRECTION_SET_MAX)
  g_fDirectionControlOutNew=DIRECTION_SET_MAX;
if(g_fDirectionControlOutNew<DIRECTION_SET_MIN)
g_fDirectionControlOutNew=DIRECTION_SET_MIN;
}
void DirectionControlOutput(void)
{
    float fValue;
    fValue=g_fDirectionControlOutNew-g_fDirectionControlOutOld;
    g_fDirectionControlOut=fValue*(countdir+1)/2+g_fDirectionControlOutOld;
}
```

4.5 本章小结

本章首先介绍系统的软件开发环境并确定软件系统的总体结构。对单片机各个模块进行初始化配置并通过对相应状态寄存器或数据寄存器的读写，实现相应的功能。重点介绍软件算法中的卡尔曼滤波，车模直立 PD 控制算法，车模速度 PI 控制算法，车模方向 PD 控制算法并给出相应的程序。

第五章　系统的安装与调试

5.1 系统机械的安装

5.1.1 车模倾角传感器的安装

本论文使用的倾角传感器包括：陀螺仪和加速度器，它们是表贴元件。实际使用时为了在最大程度上减少车模运行时的前后振动对于倾角测量的干扰，论文将这块带有陀螺仪和加速度器的电路板固定在车模的底部。安装角度传感器电路板时应该尽量保证陀螺仪传感器水平安装。如图 5.1 所示：

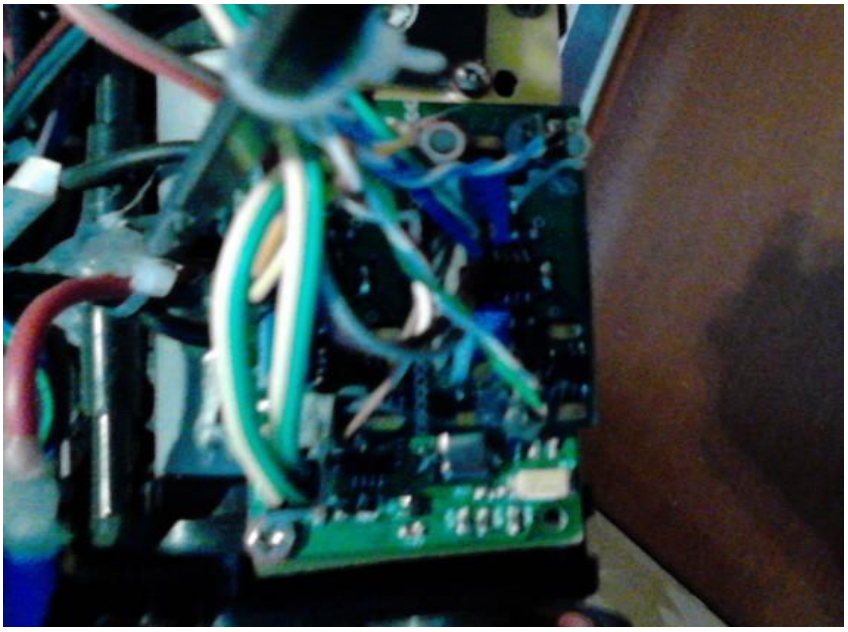

图 5.1 倾角传感器电路图

经过分析可知车模在过弯道时同时具有平动和转动两种运动。而车模的转动同时将会带动陀螺仪的转动，如果陀螺仪此时在电路板上的安装不是处于绝对的水平位置，那么陀螺仪在 Z 轴方向上会由于该转动产生一个分量，此分量会在车模过弯道时造成速度变慢或变快。其示意图如图 5.2 所示：

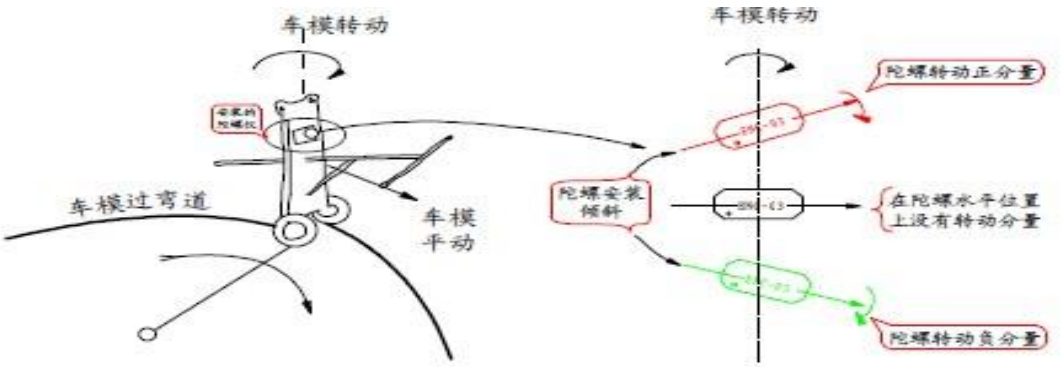

图 5.2　陀螺仪安装倾角对车模过弯道的影响

5.1.2 车模速度传感器的安装

本论文的测速编码器选用是 157 线的日本内密控 NEMIC，它以不影响智能车的重心和支点作为安装准则。本论文采用单个编码器来测得车模后轮的转速，它通过齿轮与主齿轮连接，如图 5.3 所示。

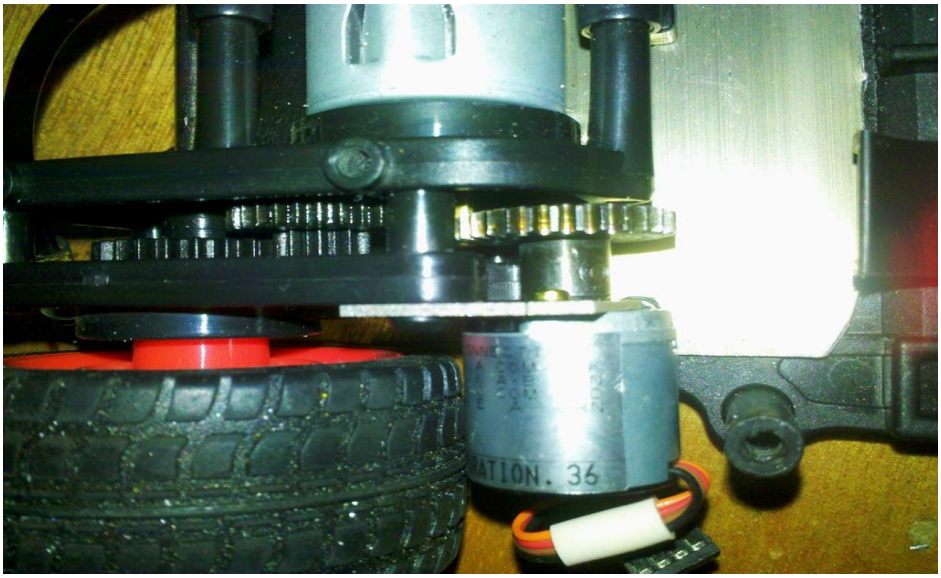

图 5.3 编码器的安装位置图

5.1.3 电磁传感器的安装

本论文选用两个工字型的 10 毫亨电感作为电磁传感器。文中将电感尽可能的安装在车模运行前方较远的地方，这样能够更好的检测前面的道路。考虑到车模直立运行的特殊性，设计了如下的电磁传感器安装方案，如图 5.4,5.5 所示。

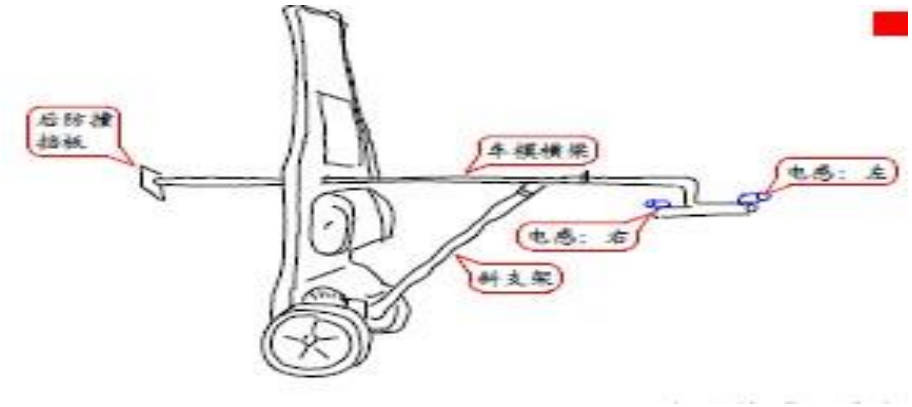

图 5.4 电磁传感器支架

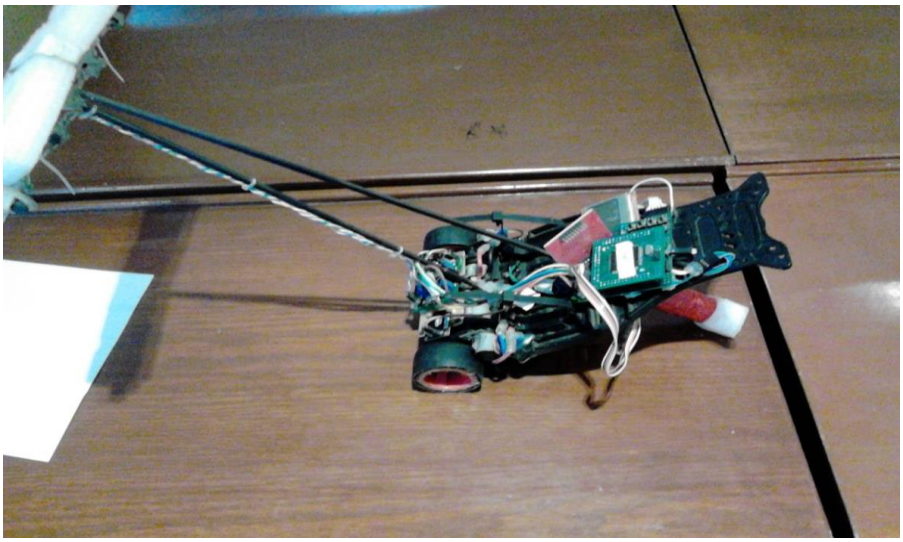

图 5.5 电磁传感器安装实物图

为了保证车模运行时的方向控制的稳定性，文中将两电磁线圈安装在同一条直线上，并呈现出水平位置。因为若两电磁线圈不是安装在一条水平线上，则在车模过赛道的十字交叉口的时候，水平方向的电线中的磁场就会在这两线圈上产生额外的感应电动势。此时造成车模方向控制的不稳定性，从而影响比赛成绩。

5.2 系统的调试

5.2.1 系统的开发平台

MC9S12XS128 单片机的开发主要使用 Netrowerks 公司 Codewarrior 软件进行了编写和调试。我们使用的是 Codewarrior5.1 版本，该新版本能完美支持如 Windows 7 等系统，使用开发更加人性化。CodeWarrior 是专门面向 Freescale 所有 MCU 与 DSP 嵌入式应用开发的软件工具，该开发环境主要包括以下功能模块：IDE、编辑器、调试器、源码浏览器、构造系统、搜索引擎。其中软件开发工程的四个主要阶段对应的分别是编辑

器、编译器、连接器和调试器。而代码浏览和构造控制则需要其它模块的支持再能完成工作，工程管理器是对整个过程进行控制。CodeWarrior 开发环境能跟踪源码的变化，进行自动编译和连接。CodeWarrior 开发环境的优点主要是以下两点：第一它具有友好的界面，可以对目标单片机型号和开发调试场景进行实时改变，开发人员可以进行程序在线实时调试。第二是 CodeWarrior IDE 能够自动地检查代码中的明显错误，该功能是通过一个集成的调试器和编辑器来完成的，这在很大程度上减少明显的错误的发生，然后通过编译并链接程序以便计算机来执行程序[38]。CodeWarrior 的界面如图 5.6 所示。

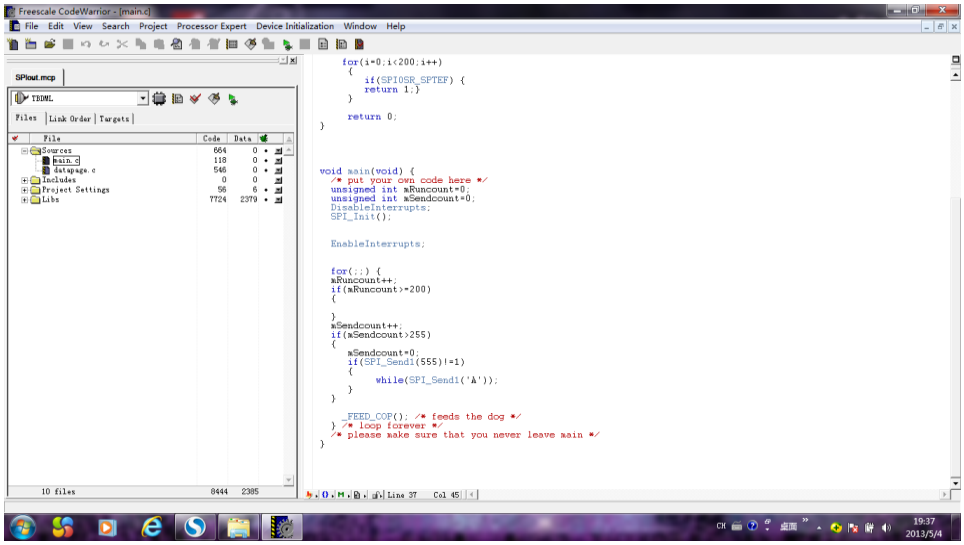

图 5.6 系统调试软件界面

5.2.2 系统的调试及参数整定

车模各模块设计安装好之后，车模进入了最重要的调试阶段。整车调试分为 3 个部分分别是：硬件调试，静态调试和动态运动调试。 硬件调试过程中，为了测量各个稳压电路的输出值和电路的工作电流是否在安全范围之内，故先将电路加上 7.2V 电池电压再进行观察。该过程主要是测试软件信号是否正确，系统的供电电路、PWM 输出、编码器的采样、各传感器的值等是否正常。 静态参数调试：测量车模运动状态下各传感器在车模静止时的零点偏移量以及单位换算的比例值[41,42]。

（1）车模直立平衡控制参数的整定。直立平衡控制参数包括比例控制和微分控制两个参数。比例控制参数 k_p 相当于倒立摆的回复力，欲使车模保持直立，需保证该参数的值要大于重力加速度即 $k_p > g$ 成立。然后逐步增大该参数，车模开始能够保持直立，随着该参数的进一步增大，车模便会出现来回摆动的现象。微分控制参数 k_D 相当于倒立摆中的阻尼力，在抑制车模摆动方面具有良好的效果，这两个参数可以先进行比例控制

参数的调整然后再对微分控制参数进行调整。调试先给比例控制参数赋各初值，观察在哪个初值下车模能够直立并能来回的摆动，则此情景下该初值能保证克服重力的影响。在比例控制参数值不变的情况下，可进行微分控制参数的调整，将其值逐渐增大直到车模能处于直立稳定状态。为了得到微分控制参数的最大值可以将该参数值一直增大，直到出现车模共振现象，该值即是微分控制参数的最大值。然后适当减小该值，同时将比例控制参数值逐渐增大，当车模再次发生震荡时，此时即得到比例控制参数的最大值。然后对两参数值进行微调，直至能得到车模直立控制参数的最优值。

（2）车模速度控制参数整定。车模速度控制参数的调整是建立在车模能很好的保证直立的情况下进行的，速度控制参数包括比例参数k_p和积分参数k_I。在进行静态调整时，速度闭环控制的速度初值应先置为零，然后就可对上述两参数值进行调整。实际调试过程中先将比例参数k_p逐渐增大，当车模能够在一定平衡点处进行往复运动时即可停止该参数的调整。为加快车模停在平衡点处可将积分参数逐渐增大，若车模受到外力冲击时，车模将会快速的处于静止状态。为了增强车模抵抗冲击的能力可将比例和积分控制参数值逐步加大。考虑到速度的调节实质上是通过调节车模的倾角来完成的，从车模倾角控制的角度来说，车模的速度控制就会对其产生干扰。假若速度控制参数调整的范围过大，就会削弱车模角度的控制作用。所以应经过多次试验来得到一组最优值来同时保证两种控制都能发挥良好的作用。

（3）车模方向控制参数的整定。车模方向控制包括比例控制参数k_p和微分控制参数k_D。在调整方向控制参数时，比例控制参数k_p可以保证车模方向能恢复到正确位置上，为了加快车模的方向回复速度，可将该参数逐渐增大。然而当该值过大时会造成车模方向回复过快，且车模方向易出现过冲。为了抑制方向过冲的现象的出现，可通过增加微分控制参数k_D来实现。微分控制参数k_D可保证车模在直道入弯道时能迅速转弯，弯道进直道时能迅速摆正车身，并且在蛇形弯处能迅速调整车身的方向。为了避免过冲现象的发生，应经过多次组合测试，最终选择一组合适的比例控制和微分控制参数[43,44]。

以上即是车模的三个控制模块控制参数在静态状态下的调整过程，三者模块之间调试具有相互影响的特点。在后面车模运动过程中应根据前面的调整步骤和三者之间相互影响的关系，逐步优化这些参数，最终使车模在运动过程中能保持稳定、运行速度较快，通过综合调整参数，保证车模能够兼顾稳定性和速度性。

5.3 实际赛道的测试

调试赛道如图 5.7 所示，赛道总长 26.53m，其中弯道 19.73m，直道 6.8m，试验测试结果

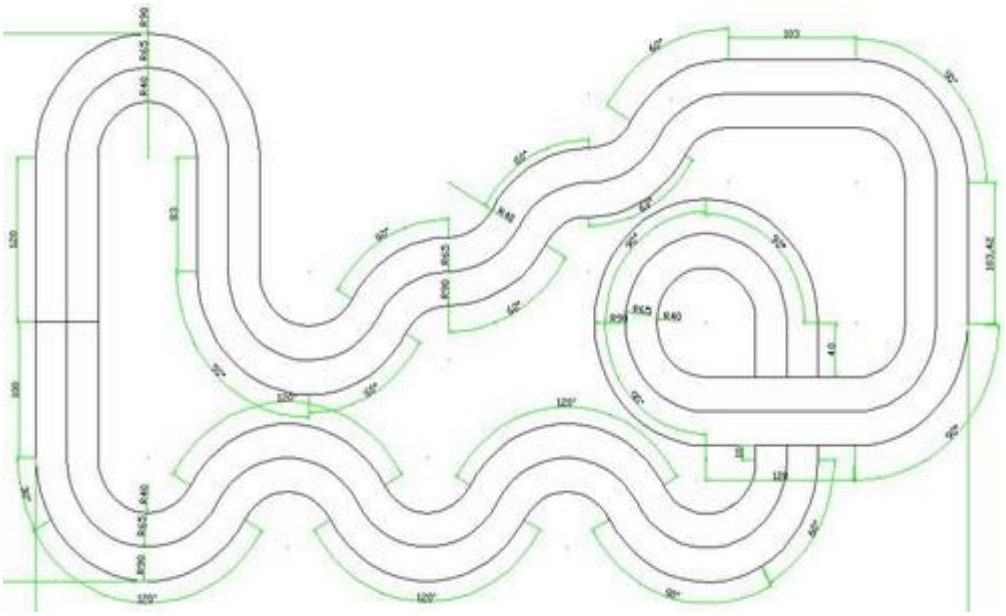

图 5.7 试车赛道图

表 5.1 试验结果

序号	最快单圈时间（s）	最快速度(m/s)	全程平均速度(m/s)	冲出跑道次数
1	16.69	1.7	1.59	0
2	17.22	1.7	1.54	0
3	16.80	1.8	1.65	0
4	15.61	1.9	1.70	0
5	15.80	1.8	1.68	0
6	14.90	1.9	1.78	0
7	14.74	1.9	1.80	0
8	15.24	1.8	1.74	0
9	14.66	1.9	1.81	0
10	15.16	1.8	1.75	0

经过车模硬件的制作并进行软件系统的编写，在对三个控制模块的控制参数进行静态及动态调整后便进行实际的赛道调试，以上数据便是经过多次进行参数调整及相应

程序修改后得到的，上面的数据说明本论文在硬件及软件设计方面达到设计要求。

5.4 本章小结

本章介绍系统的机械结构安装，包括倾角传感器，速度传感器及电磁传感器。并把它们若安装不当会造成不良影响考虑在内，优化系统机械结构的安装。然后介绍软件系统的调试环境，及车模在直立、速度和方向控制参数的静态，动态调整所遵循的原则。随后在实际赛道中对其进行调试，事实证明该系统在硬件和软件方面的设计达到前期的任务要求，证明该论文的设计方案的可行性。

第六章　结论及展望

6.1：结论

2012 年我国举办了第七届全国大学生“飞思卡尔”智能汽车竞赛，该竞赛是以快速发展的汽车电子为背景，并涵盖了控制技术、模式识别技术、传感技术、电子、计算机、机械等多个学科交叉的科技创新比赛。本课题“基于 16 位单片机 MC9S12XS128 的两轮自平衡智能车的系统研究与开发 ”即是在此背景下设计完成的。

两轮自平衡车是基于倒立摆模型的复杂非线性系统，是传统四轮车的进一步提升，具有重大的理论和实际意义。随着社会及科学技术的发展，两轮自平衡机器人的研究与应用得到较大的重视。 本论文以两轮自平衡车为研究对象，通过系统研究实现两轮小车的自平衡控制，速度控制，转向控制及自主寻迹。对倒立摆动力学模型、传感器数据融合、PID 控制器等展开了研究。本论文的关键技术为：车模倾角和角速度的滤波处理，硬件系统的搭建，自平衡控制的 PID 控制器设计，车模转向的控制，角度闭环和速度闭环的相互协调。本文主要做了以下几个方面的工作：

（1）论文完成系统的总体设计，包括系统任务要求的分析、系统原理概述、核心控制器及系统平台的选择，并给出了系统的总体结构框图。

（2）论文对系统相关模块的硬件电路进行设计，主要包括电源管理模块、电机驱动模块、车模倾角测定模块、车模速度检测模块、人机交互及调试模块。在经过查阅资料、科学论证、方案验证、电路设计和调试及最后的实践完成各个模块的设计，并采取措施，使每一个模块的性能得以优化和完善。

（3）论文利用卡尔曼滤波对采集到的加速度器和陀螺仪信号进行处理，经过处理后在车模倾角的滤波效果和收敛速度方面取得良好的平衡效果。倾角的滤波处理也是论文的重点和难点所在，因为其处理的好坏直接影响到后续的车模平衡及速度控制。

（4）文中的软件系统设计是基于相应的硬件系统来进行开发设计的。软件的开发环境是基于 Freescale 公司开发的 CodeWarrior 集成开发环境。对于倾角测定模块分别设计卡尔曼滤波算法；对于车模平衡控制采用 PD 闭环控制算法；对于速度控制模块采用 PI 控制算法； 对于车模方向控制模块采用 PD 闭环控制算法。

（5）车模调试方面，论文设计液晶显示屏按键模块，在程序中将相关的参数定义为全局变量，实际使用中该模块可以直接调整和更改程序中的相关参数，可省去程序的重

新下载，调试过程方便快捷。

6.2：展望

基于对 16 位 MC9S12XS128 单片机的两轮自平衡智能车模型系统的设计、开发及研究，并取得了一定的科研成果，但是由于时间的仓促，加之个人能力的局限性，所以本论文还有以下地方需要改进：

（1）机械结构方面。智能车在相对高速运行时有时会出现抖动的现象，从而影响到智能车的整体性能，因此需要从汽车理论的角度来分析车辆重心分布等各个机械结构方面对模型车驱动及转向性能的影响。

（2）控制算法方面。论文中采用经典 PID 控制算法，有些参数的整定需要在调试过程中不断的试验，才能得出理想的数据，增加了调试的工作量，需要开发出更先进的调试工具，以便能够简化调试过程。

（3）设计的两轮自平衡车在比赛时当其倒下时，自身是无法站立起来的，只能借助外力的辅助才能重新站起来。所以在两轮自平衡车模的自主性功能方面，以后还需要加大研究力度。

参考文献

[1] 朱政合. 飞思卡尔智能赛车及调试平台开发研究[D]. 大连: 大连理工大学, 2010

[2] 任娜. 基于16位单片机MC9S12DG128智能车系统的设计[D]. 西安: 长安大学, 2008

[3] Bishop R. Survey of Intelligent Vehicle Applieations Woridwid[J]. Proeeedings of the IEEE Intelligent Vehicles Symposium. 2000, 108(16): 509-516

[4] 付梦印, 邓志红, 刘丹. 智能车辆导航技术[M]. 北京: 科学出版社, 2009: 1-5

[5] 乔维高, 徐学进. 无人驾驶汽车的发展现状及方向[J]. 上海汽车, 2007(7): 40-43

[6] 张文杰. 基于CAN总线的智能小车控制系统研制[D]. 成都: 西南交通大学, 2009

[7] 万忠涛. 基于激光雷达的道路与障碍检测研究[D]. 长沙: 国防科学技术大学, 2010

[8] 刘欣. 无人车智能行为验证平台的虚拟交通场景研究[D]. 合肥: 中国科学技术大学, 2009

[9] 徐曙. 基于摄像头寻迹的两轮自平衡下车的设计与开发[D]. 武汉: 武汉科技大学, 2012

[10] 刘明，王洪军，李永科. 直立行走的智能车设计方案[J]. 科技信息, 2012（7）23-25

[11] 李虹瑶，冯开凯，李庆源. 自平衡小车原理研究[J]. 硅谷, 20012 (9) 31-33

[12] 孙绍杰，齐晓慧. 一种基于模糊控制的智能车转向控制算法研究[J]. 微计算机应用 , 2011（01）27-29

[13] 袁泽睿. 两轮自平衡机器人控制算法的研究[D]. 哈尔滨: 哈尔滨工业大学, 2006

[14] 李波, 杨卫，张文栋，黄伟. 一种智能小车自主寻迹系统设计[J]. 计算机测量与控制, 2012（10）15-18

[15] Rosenblum M. Neuons that How to Drive proeeeding[J]. Processings of IEEE Intelligent Vehicles Symposium. 2000, 118(12): 537-550

[16] 于虹. 基于16位单片机MC9S12DG128智能模型车系统开发研究[D]. 包头: 内蒙古科技大学, 2009

[17] 卓晴，黄开胜，邵贝贝. 学做智能车——挑战“飞思卡尔”杯[M]. 北京: 北京航空航天大学出版社, 2007: 267-275

[18] Freescale S. S12CPUV2 Reference Manual S12CPUV2/D Rev[Z]. 2003

[19] 胡杰. 基于16位单片机MC9S12DG128智能模型车系统开发研究[D]. 武汉: 武汉理工大学, 2008

[20] 张晖. 智能车导航控制硬件设计与控制算法研究[D]. 呼和浩特: 内蒙古大学, 2009

[21] 邱广萍. 移动机器视觉定位导航和自主避障系统的研究[D]. 广州: 华南理工大学, 2011

[22] 鲁云, 赵亮，陈晓东，武丽. 基于磁道航的两轮智能车系统设计[J]. 湖南师范大学自然科学学报, 2013（02）30-32

[23] Agrawal, Yang A K. Compensation of time-delay for control engineering strueture[J]. Earthquake Engineering & Structural Dynamics. 2000, 29(1): 37-62

[24] 林忠海. 基于单片机的两轮自平衡小车设计[J]. 科技创新与生产力, 2013（02）15-17

[25] 王广祥，张生元，亢俊健. 单片机及外围接口芯片的复位问题[J]. 石家庄经济学院学报. 2000(6): 643-646

[26] 卢旭锦. 基于Keil C的AT24CO2串行E2PROM的编程[J]. 现代电子技术, 2007（04）20-22

[27] 梁丽娟. 基于机器人及军用导航系统的MEMS陀螺仪性能研究[D]. 北京: 北京交通大学, 2011

[28] 杨承凯，曾军，黄华. 多传感器融合中的卡尔曼滤波探讨[J]. 现代电子技术, 2009（07）40-42

[29] 彭丁聪. 卡尔曼滤波的基本原理及应用[J]. 软件导刊, 2009（11）31-34

[30] An Introduction to the Kalman Filter. Greg Welch and Gary Bishop.TR 95-041.Department of Computer Science. University of North Carolina at Chapel Hill. Chapel Hill, NC 27599-3175.

[31] 吴至秋，常韫恒，王佳蔚. 东北大学电磁一队技术报告[R]. 南京: 第七届飞思卡尔大赛组委会, 2012

[32] 郭巍. 模块化智能机器人平台研究[D]. 南京: 东南大学, 2009

[33] 李海涛，张晓波，何胜江. 成都信息工程学院疯狂飞车队技术报告[R]. 南京: 第七届飞思卡尔大赛组委会, 2012

[34] 李陈荣，常子敬，马志凯. 浙工大银江电磁三队技术报告[R]. 南京: 第七届飞思卡尔大赛组委会, 2012

[35] 周海旭. 两轮自平衡机器人系统设计[D]. 哈尔滨: 哈尔滨工业大学, 2009

[36] 袁景明. 汽车电动助力转向系统助力与回正控制研究[D]. 重庆: 重庆大学, 2011

[37] 徐济安. 微小型移动机器人组合导航定位系统的研究[D]. 哈尔滨: 哈尔滨工业大学, 2008

[38] 张川，刘帅，陆明伟. 浙江师范大学尖峰电磁一队技术报告[R]. 南京: 第七届飞思卡尔大赛组委会, 2012

[39] 谢英强，江晨露，王科祥. 杭州电子科技大学钱江一号队技术报告[R]. 西安: 第六届飞思卡尔大赛组委会, 2011

[40] 郭伟，刘玉章，刘辉. 哈尔滨工程大学极品飞车3号技术报告[R]. 西安: 第六届飞思卡尔大赛组委会, 2011

[41] 李剑. PID参数整定方法进展[J],电力情报, 2001（09）20-22

[42] 段力学. PID参数整定方法分类与概述[J],现代计算机, 2012（03）15-17

[43] 黄宜庆. PID控制器参数整定及其应用研究[D]. 淮南: 安徽理工大学, 2009

[44] 刘玲玲. PID参数整定技术的研究及应用[D]. 郑州: 郑州大学, 2010

攻读学位期间发表的论文和专利

[1] 韩毅，丁磊，王金洋. 一种基于单片机的汽车报警装置[P].中国专利 ZL201120492365.2，2012-07-25

[2] 韩毅，王金洋，丁磊. 一种基于 MC9S12XS 单片机的节气门控制装置[P].中国专利：ZL201120430064.7，2012-07-01

程序附录

```
#include    "PIT.h"
#include    "AD.h"
#include    "PORT.h"
#include "PWM.h"
#include "ECT.h"
#include "main.h"
#include"lcd5110.h"
#include"24c02.h"
#define CONTROL_SPEED_COUNT 19
#define    MOTOR_SPEED_SET_MAX          300
#define    MOTOR_SPEED_SET_MIN        -300
#define    MOTOR_OUT_MAX                 300
#define    MOTOR_OUT_MIN                -300
#define    ANGLE_CONTROL_OUT_MAX    300
#define    ANGLE_CONTROL_OUT_MIN    -300
#define    DIRECTION_SET_MAX           300
#define    DIRECTION_SET_MIN         -300
//#define      Gyro_Zero    1293      //1.35v    1167
#define    MOTOR_OUT_DEAD_VAL      0 //32 28// 60
#define ENCONDE_CONSTANT               157
#define DRIVE_RATIO                        30/9
#define SPEED_CONTRO_PERIOD          100
#define CAR_SPEED_CONSTANT
100./471//(1000.0*DRIVE_RATIO/SPEED_CONTRO_PERIOD/ENCONDE_CONSTANT)
//-----------------SD 输出参数--------------------
float Gyro_Angle=0,Gyro_Dir=0;
float Speed_P=0,Speed_I=0;
float AngleValue=0,GyroValue=0;
//-----------------SD 输出参数--------------------
  char    i_anglefirst=0;
    unsigned int setnum[25];
```

```
// int      Gyro_Zero=1261;      //1.35v    1293
//--------------------角度相关-----------------------
float g_fLeftMotorOut,g_fRightMotorOut;
float SubValue=-0.3;
int count0=0,count1=0,count11=0;
float g_nCarGyroVal=0,anglez=0;
float gyroscope_rate=0;
 int x,y,z;
 float    Ax,Ay,Az,z_1;
 float    Axyz,Ayxz,Azxy,angle_YZ;
 float    Gyro_b =0,angle_kalman,angle_kalman_last, angle_dot_kalman;
 int GRAVITY_ANGLE_RATIO,Acc_Z_Zero,Acc_Z_Max,Acc_Z_Min;
 float ANGLE_CONTROL_P=150,ANGLE_CONTROL_D=6; //300 6.8
 float g_fAngleControlOut;
 //---------------串口发送----------------
 int    gyro, s_a,s_b,flag=0;
 int j;
int AD_x[20]=0,AD_y[20]=0,AD_z[20]=0,AD_gy[20]=0,Dir_AD_gy[20]=0;
long AD_xmax,AD_xmin,AD_xsum;
//----------------------速度相关--------------------------------
float SPEED_CONTROL_P=48;//82;
float SPEED_CONTROL_I=3.8;//8.28;
int CAR_SPEED_SETfDelta=0;          //25
int i_t0_test=0,i_t1_test=0,i_t2_test=0,cur_speed_l=0,cur_speed_r=0;
int      g_nLeftMotorPulseSigma=0,g_nRightMotorPulseSigma=0;
float g_fCarSpeed;
float
g_fSpeedControlIntegral=0,g_fSpeedControlOutOld=0,g_fSpeedControlOutNew=0;
float    g_fSpeedControlOut=0;
int SetMaxSpeed=0,SetMinSpeed=0;
//-----------------------方向相关-------------------------------
float sumxy1=0;
float sumy1=0;
```

```
float sumx1;
float minx=0;
int maxi;
float dif_old=0,dif=0,dif_old2,dif_old1;
float AD_wData[4]=0;
const float dif_x[4]={-10.6,-4.3,3.3,8.8};
float AD_0[20]=0,AD_1[20]=0,AD_2[20]=0,AD_3[20]=0;
float Max_AD[4]={5,5,5,5},Min_AD[4]={0.524,0.561,0.561,0.508};;
float g_fDirectionControlValue=0;
int i,countdir=0;
float g_fDirectionControlOutOld=0,g_fDirectionControlOutNew=0;
float DIR_CONTROL_P=115,DIR_CONTROL_D=1.1;       //125
float g_fDirectionControlOut=0;
//-----------------------显示屏相关-----------------------------
unsigned int P=0,Q=0,R=0;
unsigned int setnum[25];
unsigned char cur_sel;
const char ad[4][4]={"A0\0","A1\0","A2\0","A3\0"};
const char lcd_show[12][4]={"AP\0","DP\0","AD\0","SP\0","SI\0","TY\0","LS
  \0","CK\0","DD\0","SV\0","V\0","DIR\0"};
unsigned int HS,ST,ZQ,XS, DZ,TY,LS,CK,XQ,XZ,DJ,TJ;
void Delay(int c)
{
      int cnt;
      for(cnt=0;cnt<2*c;cnt++)
      {
            _asm NOP; _asm NOP;_asm NOP; _asm NOP; _asm NOP;_asm NOP;
            _asm NOP;_asm NOP;
      }
      }
   void BusCLK_ini(void)
{
         CLKSEL=0X00;
```

```
        PLLCTL_PLLON=1;
        SYNR = 0xc9;    f(VCO)=2*osc*(1+SYNR)/(1+REFDV)=160MHz;
        REFDV= 0x81;
        pllclock=f(VCO)=160M
        _asm(nop);
        _asm(nop);
        while(!(CRGFLG_LOCK==1));
        CLKSEL_PLLSEL =1;
}
void acc_z(void)
  {
for(j=0;j<19;j++)
   {
          AD_z[j]=AD_z[j+1];
      AD_gy[j]=AD_gy[j+1];
      Dir_AD_gy[j]= Dir_AD_gy[j+1];
   }
   AD_wData[0]=ReadATD(0)/4096.*5;
   AD_wData[1]=ReadATD(1)/4096.*5;
   AD_wData[2]=ReadATD(2)/4096.*5;
   AD_wData[3]=ReadATD(3)/4096.*5;
   AD_z[19]=ReadATD(4);
   AD_gy[19]=ReadATD(5);
   Dir_AD_gy[19]=ReadATD(6);
//----------去掉最大值和最小值并求平均值----------
// ----------z----------
   AD_xmax=0;
   AD_xmin=4096;
   AD_xsum=0;
   for(j=0;j<20;j++)
   {
      AD_xsum=AD_xsum+AD_z[j];
      if(AD_z[j]>AD_xmax) AD_xmax=AD_z[j];
```

```
    if(AD_z[j]<AD_xmin) AD_xmin=AD_z[j];
    }
    AD_xsum=AD_xsum-AD_xmax-AD_xmin;
    Acc_Z_7260=AD_xsum/18.0;
// --------------Dir gyro-----------------
  AD_xmax=0;
  AD_xmin=4096;
  AD_xsum=0;
  for(j=0;j<20;j++)
  {
    AD_xsum=AD_xsum+Dir_AD_gy[j];
    if(Dir_AD_gy[j]>AD_xmax) AD_xmax=Dir_AD_gy[j];
    if(Dir_AD_gy[j]<AD_xmin) AD_xmin=Dir_AD_gy[j];
  }
    AD_xsum=AD_xsum-AD_xmax-AD_xmin;
    Dir_AD_gyro=AD_xsum/18.0;
   // --------------gyro-----------------
  AD_xmax=0;
  AD_xmin=4096;
  AD_xsum=0;
  for(j=0;j<20;j++)
  {
    AD_xsum=AD_xsum+AD_gy[j];
    if(AD_gy[j]>AD_xmax) AD_xmax=AD_gy[j];
    if(AD_gy[j]<AD_xmin) AD_xmin=AD_gy[j];
  }
    AD_xsum=AD_xsum-AD_xmax-AD_xmin;
    AD_gyro=AD_xsum/18.0;
    Gyro_b=(AD_gyro-Gyro_Zero);          //
    gyroscope_rate=Gyro_b*0.3382;//0.5002,0.4382

        OutData[2]=gyroscope_rate;
       Az=(Acc_Z_7260-1350)/600.0;
```

```
        if(Az>1.0) Az=1.0;
        if(Az<-1.0)    Az=-1.0;
        anglez=asinf(Az)/_M_PI*180;
        OutData[0]=anglez*10;
  }
//////////////////控制电机正反转//////////////////////////
void    PWMoutVecRight(float Velocity)
{
int pwm_motor;
PTM_PTM0=0;
PTM_PTM6=0;
PTM_PTM2=1;
PTM_PTM4=1;
pwm_motor=Velocity;
if(pwm_motor>0)
{
   if(pwm_motor>PWMPER45)
   {
     pwm_motor=PWMPER45;
   }
}
else pwm_motor=0;
PWMDTY45=pwm_motor;
}
void PWMoutVecLeft(float Velocity)
{
int pwm_motor;
PTM_PTM1=0;
PTM_PTM7=0;
PTM_PTM3=1;
PTM_PTM5=1;

pwm_motor=Velocity;
```

```
if(pwm_motor>0)
{
   if(pwm_motor>PWMPER67)
   {
      pwm_motor=PWMPER67;
   }
    }
else pwm_motor=0;
PWMDTY67=pwm_motor;
}
void PWMoutStopVecRight(float Velocity)
 {
int pwm_motor;
 PTM_PTM6=0;
PTM_PTM4=0;
PTM_PTM0=1;
PTM_PTM2=1;
pwm_motor=Velocity;
if(pwm_motor>0)
{
   if(pwm_motor>PWMPER45)
      pwm_motor=PWMPER45;
}
else pwm_motor=0;
PWMDTY45=pwm_motor;
}
void PWMoutStopVecLeft(float Velocity) {
int pwm_motor;
PTM_PTM7=0;
PTM_PTM5=0;
PTM_PTM1=1;
PTM_PTM3=1;
```

```
pwm_motor=Velocity;
if(pwm_motor>0)
{
   if(pwm_motor>PWMPER67)
      pwm_motor=PWMPER67;
}
else pwm_motor=0;
PWMDTY67=pwm_motor;
}
void PWMoutReverseVecRight(float Velocity)
{
int pwm_motor;
PTM_PTM2=0;
PTM_PTM4=0;
PTM_PTM0=1;
PTM_PTM6=1;
pwm_motor=Velocity;
if(pwm_motor>0)
{
   if(pwm_motor>PWMPER45)
   pwm_motor=PWMPER45;
}
else pwm_motor=0;
PWMDTY45=pwm_motor;
}
void PWMoutReverseVecLeft(float Velocity)
{
int pwm_motor;
PTM_PTM3=0;
PTM_PTM5=0;
PTM_PTM1=1;
PTM_PTM7=1;
pwm_motor=Velocity;
```

```
if(pwm_motor>0)
{
   if(pwm_motor>PWMPER67)
      pwm_motor=PWMPER67;
}

else pwm_motor=0;
PWMDTY67=pwm_motor;
}
//////////////////控制电机正反转//////////////////////////
void SC_black_Init()
{ int i;
   for(i=0;i<400;i++)      //4000
   {
      for(j=0;j<8;j++)
      {
           acc_z();
      }
     Delay(8000);
         }
         Gyro_Zero=AD_gyro;
         Dir_Gyro_Zero= Dir_AD_gyro;
          }
//----------------方向控制函数--------------------
 void DirectionControl(void)
 {
        float sumxy=0;
        float sumy=0;
        float sumx;
        float fValue,fDValue;
        for(i=0;i<4;i++)
        {
```

```
  sumxy+=dif_x[i]*(AD_wData[i]-Min_AD[i]);
  sumy+=(AD_wData[i]-Min_AD[i]);
}
if(sumy>0)
  sumx=sumxy/sumy;
else
  sumx=0;
  g_fDirectionControlOutOld= g_fDirectionControlOutNew;
minx=0;
for(i=0;i<4;i++)
  {
     if(AD_wData[i]>minx)
     {
           maxi=i;
           minx=AD_wData[i];
      }
  }
dif_old=dif;
if(minx>=setnum[5]*0.1)
{
  dif=sumx*DIR_CONTROL_P+pow(sumx,2)/setnum[7];
  if(dif>500)    dif=500;
  if(dif<-500)    dif=-500;
  fValue=dif;
   }
else
{
        if(dif_old>1)
            fValue=500;
        else if(dif_old<-1)
            fValue=-500;
            dif=fValue;
}
```

```
        Gyro_Angle=fValue;

         fDValue=Dir_AD_gyro-Dir_Gyro_Zero;
OutData[3]=fDValue;
fDValue*=DIR_CONTROL_D;
Gyro_Dir=fDValue;

fValue+=fDValue;//转弯添加陀螺仪后的代码

g_fDirectionControlOutNew=fValue;
if(g_fDirectionControlOutNew>DIRECTION_SET_MAX)
g_fDirectionControlOutNew=DIRECTION_SET_MAX;
if(g_fDirectionControlOutNew<DIRECTION_SET_MIN)
g_fDirectionControlOutNew=DIRECTION_SET_MIN;
}
voidDirectionControlOutput(void)
{
floatfValue;
fValue=g_fDirectionControlOutNew-g_fDirectionControlOutOld;
g_fDirectionControlOut=fValue*(countdir+1)/2+g_fDirectionControlOutOld;
}
//----------------电机控制----------------
voidMOTOR_SET(floatnLeftPWM,floatnRightPWM)
{
if(fabs(g_nCarGyroVal-SubValue)<setnum[11])
{
if(nLeftPWM>0)
PWMoutVecLeft(nLeftPWM);
elseif(nLeftPWM<0)
PWMoutReverseVecLeft(fabs(nLeftPWM));
else
PWMoutVecLeft(0);
if(nRightPWM>0)
```

```
PWMoutVecRight(nRightPWM);
elseif(nRightPWM<0)
PWMoutReverseVecRight(fabs(nRightPWM));
else
PWMoutVecRight(0);
}
else
{
PWMoutStopVecLeft(0);
PWMoutStopVecRight(0);
}
}
voidMotorSpeedOut(void)
{
floatfLeftVal,fRightVal;
fLeftVal=g_fLeftMotorOut;
fRightVal=g_fRightMotorOut;
//-------------死区--------------------------
if(fLeftVal>0)fLeftVal+=MOTOR_OUT_DEAD_VAL;
elseif(fLeftVal<0)fLeftVal-=MOTOR_OUT_DEAD_VAL;
if(fRightVal>0)fRightVal+=MOTOR_OUT_DEAD_VAL;
elseif(fRightVal<0)fRightVal-=MOTOR_OUT_DEAD_VAL;
if(fLeftVal>MOTOR_OUT_MAX)fLeftVal=MOTOR_OUT_MAX;
if(fLeftVal<MOTOR_OUT_MIN)fLeftVal=MOTOR_OUT_MIN;
if(fRightVal>MOTOR_OUT_MAX)fRightVal=MOTOR_OUT_MAX;
if(fRightVal<MOTOR_OUT_MIN)fRightVal=MOTOR_OUT_MIN;
//nLeftVal=nLeftVal<<4;
//nRightVal=nRightVal<<4;
MOTOR_SET(fLeftVal,fRightVal);
}
voidMotorOutput(void)
{
floatfLeft,fRight;
```

```
fLeft=g_fAngleControlOut-g_fSpeedControlOut+g_fDirectionControlOut;
fRight=g_fAngleControlOut-g_fSpeedControlOut-g_fDirectionControlOut;
g_fLeftMotorOut=fLeft;
g_fRightMotorOut=fRight;
if(g_fLeftMotorOut>MOTOR_OUT_MAX)g_fLeftMotorOut=MOTOR_OUT_MAX;
if(g_fLeftMotorOut<MOTOR_OUT_MIN)g_fLeftMotorOut=MOTOR_OUT_MIN;
if(g_fRightMotorOut>MOTOR_OUT_MAX)g_fRightMotorOut=MOTOR_OUT_MAX;
if(g_fRightMotorOut<MOTOR_OUT_MIN)g_fRightMotorOut=MOTOR_OUT_MIN;
MotorSpeedOut();

}
voidCarAngleAdjust(void){
intnLeft,nRight;
floatfValue,nSpeed;
DisableInterrupts;
GyroValue=(0-gyroscope_rate)*ANGLE_CONTROL_D;
fValue=(0-gyroscope_rate)*ANGLE_CONTROL_D+(0-g_nCarGyroVal+SubValue)*A
        NGLE_CONTROL_P;//6.28195
if(fValue>ANGLE_CONTROL_OUT_MAX)
fValue=ANGLE_CONTROL_OUT_MAX;
if(fValue<ANGLE_CONTROL_OUT_MIN)
fValue=ANGLE_CONTROL_OUT_MIN;
g_fAngleControlOut=fValue;
EnableInterrupts;
MotorOutput();
}
voidSpeedControl(void)
{
floatfDelta;
floatfP,fI;
speed_left=g_nLeftMotorPulseSigma;
speed_right=g_nRightMotorPulseSigma;
g_fCarSpeed=(g_nLeftMotorPulseSigma+g_nRightMotorPulseSigma)/2;
```

```
g_nLeftMotorPulseSigma=0;
g_nRightMotorPulseSigma=0;
g_fCarSpeed=g_fCarSpeed*CAR_SPEED_CONSTANT;
fDelta=CAR_SPEED_SETfDelta-g_fCarSpeed;
fP=fDelta*SPEED_CONTROL_P;
Speed_P=fP;
fI=fDelta*SPEED_CONTROL_I;
if((g_fSpeedControlIntegral+fI)>-60000&&(g_fSpeedControlIntegral+fI)<60000)
g_fSpeedControlIntegral+=fI;
Speed_I=g_fSpeedControlIntegral;

g_fSpeedControlOutOld=g_fSpeedControlOutNew;
g_fSpeedControlOutNew=fP+g_fSpeedControlIntegral;
}
voidSpeedControlOutput(void)
{
floatfValue;
fValue=g_fSpeedControlOutNew-g_fSpeedControlOutOld;
g_fSpeedControlOut=fValue*(count1)/CONTROL_SPEED_COUNT+g_fSpeedControl
                    OutOld;
if(g_fSpeedControlOut>MOTOR_SPEED_SET_MAX)
g_fSpeedControlOut=MOTOR_SPEED_SET_MAX;
if(g_fSpeedControlOut<MOTOR_SPEED_SET_MIN)
g_fSpeedControlOut=MOTOR_SPEED_SET_MIN;
}
/**************中断函数**********************/
#pragmaCODE_SEG__NEAR_SEGNON_BANKED
voidinterrupt66PIT0(void)//定时器 0 中断服务函数
{
count0++;
acc_z();
if(count0>=5)
{count0=0;
```

```
g_nCarGyroVal+=((anglez-g_nCarGyroVal)/1.3+gyroscope_rate)*0.005;//4
OutData[1]=(g_nCarGyroVal-SubValue)*10;
SpeedControlOutput();
DirectionControlOutput();
CarAngleAdjust();
if(g_fLeftMotorOut<-1200&&i_t2_test<10)
cur_speed_l=0-i_t2_test;
elsecur_speed_l=i_t2_test;
if(g_fRightMotorOut<-1200&&i_t0_test<10)
cur_speed_r=0-i_t0_test;
else
cur_speed_r=i_t0_test;
if(g_fLeftMotorOut>-500||i_t2_test>20)
cur_speed_l=i_t2_test;
elsecur_speed_l=0-i_t2_test;
if(g_fRightMotorOut>-500||i_t0_test>20)
cur_speed_r=i_t0_test;
else
cur_speed_r=0-i_t0_test;
i_t0_test=0;//对右电机的速度清零。
i_t2_test=0;
g_nLeftMotorPulseSigma+=cur_speed_l;
g_nRightMotorPulseSigma+=cur_speed_r;
count1++;
if(count1>=CONTROL_SPEED_COUNT+1)//每 100ms 清零一次
{
count1=0;
SpeedControl();
}
countdir++;
if(countdir>=2)//每 10ms 清零一次
{
countdir=0;
```

```
DirectionControl();
}
if(count11>=0)
{
count11++;
if(count11>50)//200
{
count11=0;
CAR_SPEED_SETfDelta+=3;
}
if(CAR_SPEED_SETfDelta>SetMaxSpeed)
{
count11=-1;
CAR_SPEED_SETfDelta=SetMaxSpeed;
}
}
}
//#endif
}
PITTF_PTF0=1;//清中断标志
}
voidinterrupt67PIT1(void)//定时器 110ms
{
if(count11==50)
{
PORTB_PB0=~PORTB_PB0;
}
PITTF_PTF1=1;//清中断标志位
}
//--------------测速度的------------------------------
voidinterrupt8Irupt_t(void)//////T0 口中断
{
i_t0_test+=2;
```

```
TFLG1_C0F=1;//清除中断标志寄存器中被置位的位标志位
TIE_C0I=0;
TIE_C2I=1;
}
voidinterrupt9Irupt_tt1(void)//T1 口中断//右电机
{
i_t1_test++;
TFLG1_C1F=1;//清除中断标志寄存器中被置位的位标志位
}
voidinterrupt10Irupt_tt2(void)//////T2 口中断//左电机
{
i_t2_test+=2;
TFLG1_C2F=1;//清除中断标志寄存器中被置位的位标志位
TIE_C2I=0;
TIE_C0I=1;
}
/***************主函数*****************************/
voidmain(void)
{
longbaa[4]={1,2,3,4};
unsignedcharcur_sel;
unsignedintP,Q,R;
unsignedinti;
unsignedintj;
constcharad[4][4]={"A0\0","A1\0","A2\0","A3\0"};
floatKp,Kd,Identify;
floatsetted_speed_n;
unsignedcharminy,minx,miny2,minx2;
floatK1,K2,K3,v;
//------------初始化函数段--------------------
BusCLK_ini();
PORT_ini();
```

```
AD_Init();
DisableInterrupts;
PWM_Init();
UART_Init();
ECT_ini();
lcd_init();
init_24c02();
for(i=2;i<22;i++)setnum[i]=RdEEPROM(i);//读取 24c02
setnum[0]=setnum[12]*256+setnum[13];
setnum[1]=setnum[14]*256+setnum[15];
setnum[22]=setnum[16]*256+setnum[17];
setnum[23]=setnum[18]*256+setnum[19];
setnum[24]=setnum[20]*256+setnum[21];
Acc_Z_Max=setnum[22];
Acc_Z_Min=setnum[23];
Acc_Z_Zero=(Acc_Z_Max+Acc_Z_Min)/2;
GRAVITY_ANGLE_RATIO=(Acc_Z_Max-Acc_Z_Min)/2.;
HS=setnum[0];
ST=setnum[1];
ZQ=setnum[2];
XS=setnum[3];
DZ=setnum[4];
TY=setnum[5];
LS=setnum[6];
CK=setnum[7];
XQ=setnum[8];
XZ=setnum[9];
DJ=setnum[10];
TJ=setnum[11];
if(PORTA_PA4==0&&PORTA_PA4==0&&PORTA_PA5==0&&PORTA_PA5==0)
{
//////////陀螺仪水平零点值检测/////////
disply_listchar(0,0,"LevelGyroZero",1);
```

```
for(i=0;i<2000;i++)
{
for(j=0;j<4;j++)
{
}
delayms(1);
}
disply_listchar(0,0,"",1);
disply_listchar(0,0,"LevelEnd",1);
disply_listchar(0,0,"",1);
////////////////保存数值///////////////
disply_listchar(0,0,"Save",1);
setnum[2]=ZQ;
setnum[3]=XS;
setnum[4]=DZ;
setnum[5]=TY;
setnum[6]=LS;
setnum[7]=CK;
setnum[8]=XQ;
setnum[9]=XZ;
setnum[10]=DJ;
setnum[11]=TJ;
setnum[12]=HS/256;
setnum[13]=HS%256;
setnum[14]=ST/256;
setnum[15]=ST%256;
setnum[16]=Acc_Z_Max/256;
setnum[17]=Acc_Z_Max%256;
setnum[18]=Acc_Z_Min/256;
setnum[19]=Acc_Z_Min%256;
setnum[20]=Gyro_Zero/256;
setnum[21]=Gyro_Zero%256;
for(i=2;i<22;i++)
```

```
{
WrEEPROM(i,setnum[i]);
}
disply_listchar(0,0,"",1);
disply_listchar(0,0,"SMARTCAR",1);
disply_listchar(0,1,"SAVECOMPLETE",1);
disply_listchar(0,2,"================",1);
disply_listchar(0,3,"TURNPOWEROFF",1);
disply_listchar(0,4,"================",1);
disply_listchar(0,5,"WWW.CHD.EDU.CN",1);
while(1);
}
if(setnum[6]==1)
{
ANGLE_CONTROL_P=1.0*setnum[0]/4;
DIR_CONTROL_P=1.0*setnum[1]/4;
ANGLE_CONTROL_D=1.0*setnum[2]/4;//motor_pwm_frequ;
SPEED_CONTROL_P=1.0*setnum[3]/4;//motor_pwm_frequ;
SPEED_CONTROL_I=1.0*setnum[4]*0.1/4;//motor_pwm_frequ;
SubValue=setnum[9];
}
PIT_init();
EnableInterrupts;
for(i=0;i<200;i++)
Delay(10000);
for(;;)
{
}
}
```

致　谢

在论文完成之际，首先要感谢我的导师韩毅副教授，这篇论文是在他的悉心指导和热情关怀下完成的。在这两年的硕士学习和生活中，导师的待人态度、渊博的学识、严谨的科研态度使我受益终生。在论文的选题和立意方面，韩老师给予我大量的支持和精心指导。同时，导师也为我创造良好的学习、科研环境和宽松的研究氛围，在导师的精心栽培下，自己的理论水平和实践能力得到极大的提高。在此，谨对导师的辛勤培养和关心致之我最真挚的谢意。

同时还要感谢同门师兄弟和读研的同班同学们。在论文选题及方案论证中、在软硬件开发及调试过程中、在论文的撰写阶段以及在平时的生活中，得到他们的无私帮助和热情关怀，在此向他们表示最诚挚的谢意。

最后衷心感谢我的父母和其他亲朋好友对我在学业和生活上的关心、支持和理解，在他们的关心、鼓励和支持下，使我顺利完成两年硕士学业的学习。

谨以此文献给所有关心和帮助过我的人们。