摘 要

随着交通的迅速发展，汽车已经成为现代社会的一种重要的交通工具，并且其数量仍在不断增加中。许多交通事故由于无任何行驶记录，而造成事故责任无法确定。汽车行驶记录仪是一种特殊的电子记录装置，它可以对车辆行驶速度、时间、里程以及有关车辆行驶的其他状态信息进行记录、存储并可通过接口实现数据输出。通过对汽车实时状态的监测，可以到达规范驾驶员行为和提供道路交通事故分析鉴定以及汽车运行状态分析的作用，同时可以集成电子仪表、故障诊断、报警、GPS、GIS 等功能。

根据汽车行驶记录仪的国家标准(GB/T19056-2003)规定，一个完整的汽车行驶记录仪系统包括主机和上位机管理分析软件两部。本文主要对汽车行驶记录仪的主机模块进行了研究与设计。首先对国内外汽车行驶记录仪的研究现状做了分析，然后阐述了本文的主要工作和结构。结合汽车行驶记录仪国家标准(GB/T19056-2003)，设计了一款基于车载网络的汽车行驶记录仪主机模块。该汽车行驶记录仪主机模块可分为中央处理模块、数据采集模块、数据存储模块、USB 主机模块、CAN 通信模块、串行通信模块、实时时钟模块、显示模块、IC 卡模块、键盘接口模块、数据打印模块及电源模块。针对各模块功能，给出各芯片电路实现方法。系统采用 DSP+CPLD 的电路结构，具有可扩展性好、升级空间大、电路简单的优点。对于汽车行使记录仪的软件设计，阐述了 DSP 系统开发的基本流程与概论，包括开发环境和语言、程序引导加载的方式、存储空间划分、命令文件编写。接着给出系统主循环的程序流程图，同时针对部分硬件模块，给出了各自的程序流程图，重点对 USB 主机程序设计相关概念和方法做了探讨和分析，最后对系统抗干扰措施和调试方法做了分析。在文章的结尾，进行了总结和展望，阐述了记录仪的有待改进和升级的几个部分。

关键字：汽车行驶记录仪，DSP，CPLD，USB 主机，Bootloader

Abstract

As the rapid development of transportation, car has become one of important transporting tools and the total amount of them increases continuously. Many traffic accidents happened because of absence of travelling data recording, and then the responsibility could not be clarified. Vehicle Travelling Data Recorder (VTDR) is an especial electronic equipment, which can record and store the data of velocity, date, distance and other information of vehicle then output them through the interfaces. Through the monitoring of real time state of car, the illogical behavior of dirvers can be restricted and the recorded data can be used as a proof for analyzing of traffic accident. The health of vehicle also can be checked. Meanwhile, some functions such as electronic watches, breakdown diagnosing, alarm, GPS, GIS, can also be integrated in it.

According to the national standard of VTDR (GB/T19056-2003), an Vehicle Travelling Data Recorder consists of the mainframe part and managment software part on PC. This thesis focus on the reseaching and designing of the mainframe of VTDR. At first, the domestic and foreign situation of VTDR is analysed and then the emphases and framework of thesis are expatiated. The basic framework of VTDR is introduced. It consists of the central processing module, data collecting module, data storing module, USB host module, CAN-BUS module, RS232 module, clock module, display module, IC card module, keyboard interface module, data printing module and power module. The chips of each module are introduced, so are the realized circuits. The DSP+CPLD framework is employed in this system, so it is easy to extend and upgrade and the complexity of circuit can be reduced. And then the sofeware designing of VTDR is expatiated. The development flow and environment of DSP system is introduced after that, which includes development enviorment and program language, memory file, command file. Then the flow of main loop is introduced. The software designing of USB host is an important part, and then software flows of hardware modules are given. At last, some useful measures to avoid the disturbance of PCB and hardware are introduced, so are the

test methods. At the end, some parts of VTDR which should be upgraded and improved are expatiated.

Key words: Vehicle Travelling Data Recorder (VTDR), DSP, CPLD, USB host, Bootloader

独 创 性 声 明

本人声明，所呈交的论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得武汉理工大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

签 名：钟明 日 期：2007.5.15

关于论文使用授权的说明

本人完全了解武汉理工大学有关保留、使用学位论文的规定，即学校有权保留、送交论文的复印件，允许论文被查阅和借阅；学校可以公布论文的全部或部分内容，可以采用影印、缩印或其他复制手段保存论文。

(保密的论文在解密后应遵守此规定)

签 名：钟明 导师签名：[illegible] 日 期：2007.5.15

第1章 引言

1.1 课题的提出

随着交通的迅速发展，汽车已经成为现代社会的一种重要的交通工具，并且其数量仍在不断增加中。但这直接导致了交通事故的增加，以及对事故责任的鉴定越来越困难。据统计，近 20 年来，由于交通事故造成的死亡人数已达 108 万人以上。并且，许多重大、特大交通事故均是由于发现晚、报案迟等原因造成事故责任无法认定，同时也给国家和人民生命财产造成了巨大的损失。据有关部门统计分析，在造成交通事故的诸多原因中，80%～90%都是非规范驾驶、超速行驶、疲劳驾驶造成的。同时许多交通事故由于无任何行驶记录、而造成事故责任无法确定。因此客观上，急需一种既能指导驾驶员安全驾驶又能记录和再现机动车行驶状态的智能装置，以提高驾驶人员的驾驶水平，降低车辆交通事故率。由此可见，要减少交通事故，特别是防止群死群伤的恶性交通事故的发生，必须对违章超速和疲劳驾驶行为进行有效的严格监控。其次，由于汽车的使用过程是在移动中的，用传统的管理手段，无法详细了解车辆在行驶过程中的动态情况，难以对汽车进行有效细致的动态管理，因此，各个汽车运输单位都面临着如何进行有效的安全管理和如何对司机进行有效监督的问题。

行驶状态记录仪，亦称“汽车黑匣子”，是安装在车辆上，能够记录、存储、显示、打印车辆运行速度、时间、里程以及有关车辆运行安全的其他状态信息的数字式电子记录装置。它对防止疲劳驾驶，车辆超速和违章、约束驾驶员的不良行为、分析鉴定事故、提高交通的管理执法水平和运输管理水平、保障车辆运行安全等有着重要的实际作用及意义。

1.2 国内外研究现状

(1) 国外汽车行驶记录仪研究现状

国外对汽车行驶记录仪的认识比较早，从 1970 年起，欧盟国家以立法的方

式规定，从事营运定的 6.5 吨以的货车和七座以上的客车必须强制安装机动车行驶记录仪，并规定记录仪中的资料要保留两个月，以备路面警察随时抽查。马来西亚、新加坡等国家已经通过正式立法，要求客运车辆在 2001 年 9 月底前必须安装汽车行驶记录仪，其它商用车也要限期安装。在美国，主管交通安全的国家安全委员会(National Transportation Safety Board, NTSB)一直在努力推广汽车行驶记录仪。NTSB 要求各汽车生产厂家安装记录仪，通用、福特等汽车公司已经开始行动。

20 世纪 70 年代后期，欧洲率先推出了机电模拟式汽车行驶记录仪。当时较为流行的为纸盘机械式汽车行驶记录仪。这种记录仪内通常放置着一张直径为 12 厘米的双面圆形纸盘，这种纸盘由国家统一印制，每天放一张在记录仪内。驾驶员在放入之前，要在纸盘上填写姓名、国籍、车号、日期、开车时间等。在行驶途中，记录仪自动在纸盘上一记录下各种数据(开车、收车时间，车速，休息时间，修车时间，最长运行时间等)。国外汽车行驶记录仪发展至今，一直以机械式记录仪为主。随着电子信息技术的迅速发展，欧盟开始制订数字式记录仪的技术标和法规，并规定从 2004 年 8 月起，在新注册的机动车上强制安装数字式记录仪，以逐渐替代现行传统的纸盘模拟记录仪。目前国外较常见的电子式记录仪包含记录器、显示器、数据采集卡、传感器以 PC 机处理软件等几个部分，其中数据采集卡为便携式的存储卡，可以插入记录仪进行数据采集，亦可将采集到的数据送计算机进行图像处理和事故分析。

2001 年 8 月，日本某汽车研究所开发小组研制出能记录交通事故发生时段车辆行驶数据的记录仪。它通过传感器记录速度、方向盘角度、刹车板及油门情况，当急刹车或急转弯使汽车达到一定的加速度以上时，便判断为“事故”，并将“事故”发生前五十五秒和后五秒间的各种数据自动记录入磁盘中。同时采用图象处理技术，利用相机自动收录“事故”发生前十秒和后五秒间从驾驶席上能看到的场面。

(2) 国内汽车行驶记录仪研究现状

国内对汽车行驶记录仪的认识晚于国外。80 年代末，几家国营、民营、合资企业始着手研制、推广记录仪。到目前为止，我国生产汽车行驶记录仪的企业多达 70 余家，产品上百种，并不断有新品推出[1]。经过国家有关权威部门检验合格后，开始在全国各省市推广使用。记录仪对提高车队的营运效率和降低

事故率等方面的显著效果，受到物流运输、汽车租赁、企事业车队、保险公司和交通管理部门的欢迎[2]。但由于它的安装配套成本相对较大，目前私家车安装使用的比较少。

虽然我国的研制工作起步较晚，但是无论在安装、操作、使用方面，还是在电性能、信息记录存储、数据的下传、上载方式等方面，我国的记录仪都比欧盟的起点高。欧盟推广使用的汽车行驶记录仪大多以机械式为主，以纸盘机械式汽车行驶记录仪为例，这种产品价格昂贵，维护费用高，还需要人工填写姓名、行驶里程，使用起来极为不便。而我国的产品均为数字式的电子设备，能够实时监测并一记录车辆行驶的各种状态信息、有效准确地鉴别驾驶员身份，同时还具备超速报警功能、具备串口通信接口以及打印输出功能。各类产品体积小巧、价格合理、无需专人维护且使用方便[3]。

国内记录仪的市场虽然已经初具规模，但是各类产品在数据结构和格式、数据传输方式等方面还是有较大差异，这对记录仪的规范管理和大面积推广极为不利。而且绝大多数产品采用的主处理器为 8 位或者 16 位的单片机，这也将极大地限制记录仪系统的接口扩展、功能的完善以及实时性能的提高[4]。

1.3 汽车行驶记录仪国家标准

在 2003 年 4 月 15 日，由公安部有关部门起草、国家标准化管理委员会、国家经贸委审定通过，国家质量监督检查检疫总局发布了汽车行驶记录仪的国家标准(GB/T19056-2003)，并于 2003 年 9 月 1 日起正式实施。本文所设计的汽车行驶记录仪完全符合国家标准，下面结合国标对记录仪的主机模块的构成和具体功能进行介绍[5]。

(1) 记录仪的组成

根据国家标准的规定，汽车行驶记录仪应由如下几部分组成：

① 主机：包括微处理器、数据存储器、实时时钟、显示器、操作键、打印机、数据通信接口等装置。如果主机本体上不包含显示器、打印机，则应留有相应的数据显示和打印输出接口；

② 车速传感器；

③ 数据分析软件；

(2) 记录仪的功能

根据国家标准的规定，汽车行驶记录仪应有如下功能：

① 自检功能

记录仪在通电开始工作时，应首先进行自检，自检正常后应以绿闪信号或显示屏显示方式指示工作正常，如有故障则应以红闪信号或显示屏显示方式指示故障信息。

② 实时时钟、日期及驾驶时间的采集、记录、存储

记录仪应能提供北京时间日期和时钟，该日期和时钟被用于为记录仪实现所有功能(记录、输出、显示、数据通信等)标注日期和时间。记录仪应能以年、月、日的方式记录实时日期；应能以时、分、秒的方式记录实时时钟。记录仪应能对连续驾驶时间进行记录。连续记录 24h 数据，记录时间允许误差在 5s 以内。

③ 车辆行驶速度的测量、记录、存储

事故疑点数据：记录仪应能以不大于 0.2s 的时间间隔持续记录并存储停车前 20s 实时时间对应的车辆行驶速度值及车辆制动状态信号、记录次数至少为 10 次。速度记录单位为 km/h 测量范围为 0km/h~220km/h 测量分辨率等于或优于 1 km/h。

行驶状态数据：无论车辆在行驶状态还是停驶状态，记录仪均应能提供实时时间对应的车辆行驶速度信息。当车速传感器输出的脉冲信号超过 1 脉冲/秒并且持续 5 秒以上时，可认为车辆是在行驶状态，否则认为车辆是在停驶状态。记录仪应能以不大于 1min 的时间间隔持续记录并存储车辆在最近 360h 内的行驶状态数据，该行驶状态数据为：车辆在行驶过程中与实时时间相对应的每分钟间隔内的平均行驶速度值。速度记录单位为 km/h 测量范围为 0km/h~220km/h，分辨率等于或优于 1km/h。

记录误差：分别输出相当于 20km/h，65km/h，100km/h，145km/h 的模拟速度信号对记录仪进行测试时，其速度记录允许误差为 1km/h。记录仪安装在测试用车上进行实车路试，在行驶速度恒定在 40km/h~1km/h 和行驶速度在 40km/h~60km/h 变化情况下分别进行测试时，其速度记录允许误差为 2km/h。记录仪在安装到车辆上使用后，在 40km/h 的行驶速度进行测试时其速度记录的最大允许误差为 6km/h。

④ 车辆行驶里程的测量、记录、存储

记录仪应能持续记录车辆从指定统计时间开始的累计行驶里程。车辆行驶里程记录单位为 km，行驶里程的测量范围为 0~999999.9km，分辨率应等于或优于 0.1km。行驶检验时，记录仪安装在测试用车上进行实车行驶里程误差测试，当测试距离为 5km 时，行驶里程允许误差为 0.1km 以内。

⑤ 驾驶员身份记录功能

记录仪应能实现驾驶人员身份记录功能，应能记录驾驶员代码和公安交通管理部门核发的机动车驾驶证证号。驾驶员代码为阿拉伯数字，其最大长度不超过 7 位，代码设置方法由使用者根据需要自定，在同一记录仪的数据记录中，某一驾驶员的代码应与其机动车驾驶证证号唯一相对应。在每次驾车前，驾驶人员首先应确认自己的代码，确认方式由制造商自定。

⑥ 显示及操作功能

显示器应符合如下要求：显示字符应笔划完整、清晰规范，在使用中不依靠环境光源也能正确读数；显示数据参数时字符高度不小于 4mm；在显示数据参数的同时，应以显示或面板标识的方式清楚表示数据参数的名称及单位，字符高度不小于 3mm；显示器在车辆点火开关通电后应处于工作状态；在任意恒定的速度下，车速显示值的变化范围不得超过 1 km/h。当无按键操作时，可默认显示车辆的实时行驶速度、实时时钟或驾驶员代码。通过操作按键应能实现如下显示：最近 15min 内每分钟的平均车速记录；最近 2 个日历天内同一驾驶员连续驾驶时间超过 3h 的所有数据记录；车辆特征系数。操作按键设置应能满足使用要求，并应在对应的位置标出各按键名称；仅使用面板按键应不能对速度、时间、里程等原始数据的进行修改、删除。

⑦ 数据打印输出功能

打印方式：数据打印只能在停车状态下进行；从打印开始到每分钟平均车速记录内容打印结束，时间不应超过 30s；打印字符字迹应清晰、规范；打印字符的高度应不小于 2.4mm，宽应不小于 1.5mm；打印纸上应留有足够的空白位置供驾驶员或其他人员签名及简单备注之用。

打印内容：记录仪至少应能打印输出车牌号码、车牌分类、驾驶员代码、驾驶证号码、打印实时时间、停车时刻前 15min 内每分钟的平均车速、疲劳驾驶记录(一次连续驾驶时间超过 3h 的所有记录)。

⑧ 数据通信功能

记录仪应同时配置以下两种标准接口：USB(通用串行总线)标准接口，建议；

标准 RS232CD 型 9 针接口。记录仪对每一次下传的日期及时间进行记录、存储。应能通过通信接口，向外部设备输出至少包含如下内容的信息：实时时钟；事故疑点数据；最近 360h 内车辆行驶速度数据（记录间隔为 1min，数据为每分钟内的平均速度）；对应实时时钟的车辆行驶里程数据；车辆识别代号、车牌号码、车牌分类；驾驶员代码、驾驶证证号；车辆特征系数；上载数据时，应不改变和删除记录仪内存中已存储的任何数据。记录仪应对每一次上载的日期和时间进行记录、存储。

(3) 数据安全性

记录仪应防止数据被更改或删除，应从下面几点来实现：

① 硬件上，应在记录仪主机上或其它适当的地方采取可靠安全措施(如铅封)防止数据储存器等重要器件被更换；

② 记录仪主机内车辆行驶速度、里程、驾驶时间等原始数据不能通过外部设备进行任何改写或删除操作。

1.4 主要工作与论文组织

汽车行驶记录仪包括主机模块以及上位机管理分析软件模块两个部分。本课题的目标是设计一个完整的主机模块。考虑到系统的扩展性、实时性以及大量数据处理能力等因素，主机模块的处理器采用 TI 公司的 TMS320VC5416 处理器，它通过与外围电路进行合理的交互来完成如下主要功能：自检功能，自检结束后能够对结果予以提示；实时时钟、日期以及驾驶时间的采集、记录和存储功能；驾驶员身份识别功能；对模拟输入、数字输入以及开关量输入信号的采集、处理和存储功能；车辆行驶速度、里程以及里程小计的测量、记录、存储功能；显示及操作功能；数据通信功能，具备 RS232 串口通信、USB 通信以及 CAN 总线通信接口；数据打印输出功能，能够通过微型打印机打印出车辆信息、驾驶员信息、疲劳驾驶时段、超速驾驶时段等；报警功能，对汽车状态和驾驶员违规驾驶等情况予以报警提示。

全文分为五章，结构安排如下：

第一章分析了汽车行驶记录仪的发展现状和相关背景知识，先阐述了国内外的研究现状，然后介绍了汽车行驶记录仪国家标准，最后给出了论文的主要

工作和论文组织结构。

第二章对汽车行驶记录仪主机模块的硬件结构进行了研究，主要侧重于各模块硬件实现与其他模块的连接。

第三章主要对汽车行驶记录仪主机模块的软件设计进行了研究，从开发环境到最后的每个模块的程序流程都作了详细探讨。

第四章主要分析汽车行驶记录仪的的抗干扰设计和调试方法。

第五章总结了本文所做的工作和不足之处，提出进一步研究需要解决的问题和以后的发展方向。

第 2 章　汽车行驶记录仪硬件设计

2.1 系统总体结构

系统总体结构主要由下列几个模块组成：中央处理模块、USB 主机模块、串口通信模块、CAN 通信模块、数据采集模块、数据存储模块、键盘接口模块、显示模块、IC 卡模块、键盘接口模块、数据打印模块、电源模块。系统硬件构成框图如图 2-1 所示。

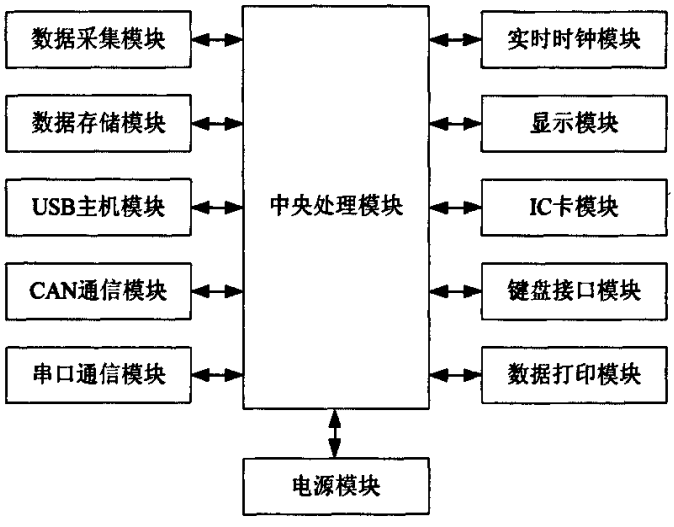


图 2-1 系统硬件框图

2.2 中央处理模块

中央处理模块由 TI 公司的 DSP 芯片 TMS320VC5416 和 ALTERA 公司的 CPLD 芯片 EPM7128ELC84 构成。其中 TMS320VC5416 负责程序的运行，而 EPM7128ELC84 负责外围器件的逻辑与译码。

TMS320VC5416 是 16 位定点 DSP。它采用修正的哈佛结构，程序与数据分开存放，内部具有 8 条高度并行的总线[6]。集成有在片的存储器和在片的外

设以及专门用途的硬件逻辑，并配备有强大的指令系统，使得该芯片具有很高的处理速度和广泛的应用适应性[7]。再加上采用的模块化设计以及先进的集成电路技术，芯片功耗小，成本低，自推出以来已广泛应用于各种专门用途的实时嵌入式系统和仪器中。它有丰富的外设和大量的高速片上存储器。片上外设主要有主机接口（HPI）、DSP 的中断系统、片外扩展总线片上集成的存储器等[8]。

EPM7128ELC84 是 Altera 公司生产的 MAX7000AE 系列的 CPLD。CPLD 的设计主要是内部的时序和逻辑设计，采用的手段是使用 VHDL 语言进行“编程”。但是，这里所谓的“编程”与通常意义上对处理器的编程不一样。这里的“编程”只是对 CPLD 行为的描述，最终将转换成时序逻辑电路实现。

2.3 数据采集与存储模块

2.3.1 数据采集模块

数据采集系统包括速度信号采集电路、开关信号采集电路。速度采集电路的功能是采集行车时的实时速度信号。开关信号采集电路的功能是实时检测 8 路开关量信号状态并送主控模块处理。数据采集系统各模块组成框图如图 2-2 所示。

(1) 速度信号采集电路

速度信号采集电路的原理是：汽车行驶过程中，车轮旋转经过传感器，单位时间内输出一定数量的脉冲，传感器输出的脉冲通过放大电路放大整形，然后送至CPLD计数，得到的脉冲数经过DSP进行速度计算后得到汽车行驶速度。汽车变速器输出的速度信号是时间上连续的数字信号，为了便于处理、存储，将这些时间上连续的数字信号转换为时间上等间距的数字信号。系统采用霍尔传感器将速度信号转换为脉冲信号，霍尔式传感器[9]是利用霍尔元件制成的，霍尔元件是一种磁传感器。用它可以检测磁场及其变化，可在各种与磁场有关的场合中使用。霍尔元件以霍尔效应为其工作基础。霍尔元件具有许多优点，它具有结构牢固、体积小、重量轻、寿命长、安装方便、功耗小、频率高(可达1MHZ)，耐震动、不怕灰尘、油污、水汽及盐雾等的污染或腐蚀等特点[10]。

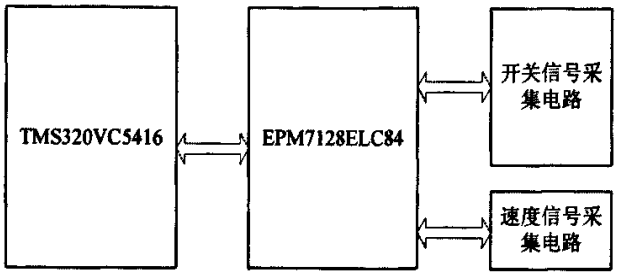


图 2-2 数据采集模块框图

(2) 霍尔转速传感器

霍尔转速传感器的结构原理如图 2-3 所示。它实际上是利用霍尔开关测转速。待测物上粘贴一对或多对小磁钢，小磁钢愈多，分辨率愈高。霍尔开关固定在小磁钢附近。待测物体以角速度ω旋转时，每当一个小磁钢转过霍尔开关集成电路，霍尔开关便产生一个相应的脉冲。检测出单位时间的脉冲数，即可确定待测物的转数。其主要技术指标如下：测量范围：0~10000r/min；输出波形：矩形脉冲波；供电电压：DC5~24V；工作距离：3~5mm；每转脉冲数：与磁钢数量一致；输出信号幅值：高电平接近供电电源，低电平≤0.5V；工作温度：-20℃~+80℃外型尺寸：M16×1 或 M12×1；

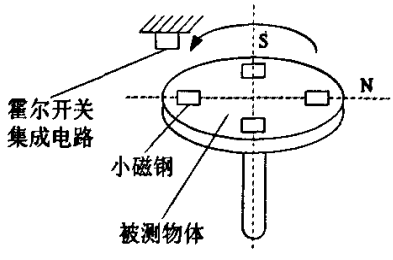


图 2-3 霍尔转速传感器的结构原理图

其主要特点如下：响应频率高：50~100kHz；作接近开关使用时，重复定位精确高(小于 0.02mm)；工作电压宽，负载能力强。抗干扰能力强，高可靠性，长寿命；适用于低转数测量。

(3) 速度测量原理

本系统选用霍尔传感器将速度信号转换为电信号，考虑到传感器的体积要小，便于安装，误差要尽量减小等要求，本文采用车轮旋转一周速度传感器要输出若干个脉冲的方法，具体采用在变速器上安装 3 个小磁钢相应的输出 3 个脉冲的方法[11]。

车轮旋转一周速度传感器输出的脉冲个数为 n，则脉冲个数如式 2-1:

$$n=3 \tag{2-1}$$

设 0.2 秒内计数器测得的脉冲数为 f，则车轮旋转圈数 N 如式 2-2：

$$N=\frac{f}{n} \tag{2-2}$$

设车轮半径为 R，则在 0.2 秒钟内汽车行驶的距离 L 如式 2-3：

$$L=2\pi RN \tag{2-3}$$

在 1 秒钟内汽车行驶的距离 L1 如式 2-4：

$$L1=5L \tag{2-4}$$

这样就可以得到最终的速度计算公式 V 如式 2-5：

$$V=5\times 2\pi R\times\frac{f}{n} \tag{2-5}$$

(4) 由 THS118 型霍尔元件组成的速度信号采集电路

在系统中，采用了 THS118 型霍尔元件[12]作为速度信号采集的速度传感器，THS118 的特性如下：超小 4 引脚封装；良好的温度特性；温度工作范围大(-55℃~125℃)；输出电压线性良好。

图 2-4 是采用 THS118 的速度信号采集电路，磁转子 M 旋转的同时，使霍尔元件 H 的磁极(N，S)产生变化，从而检测转子的转速[13]。图中 a，b 为输入端，c，d 为输出端。从霍尔元件结构上看，输出端包含共模电压 Vc，电压 Vc 对霍尔电压没有关系，使用时此电压必须除去，本系统中采用差动输入的运算放大器来忽略此电压。霍尔元件的输出端接到差动放大器的输入端，因此，c 点电压等于 d 点电压时，运放无输出，c 点电压大于 d 点电压或小于 d 点电压时，有差动信号输入，这时，运放输出端有较大的输出电压。THS118 输出的波形为矩形波。

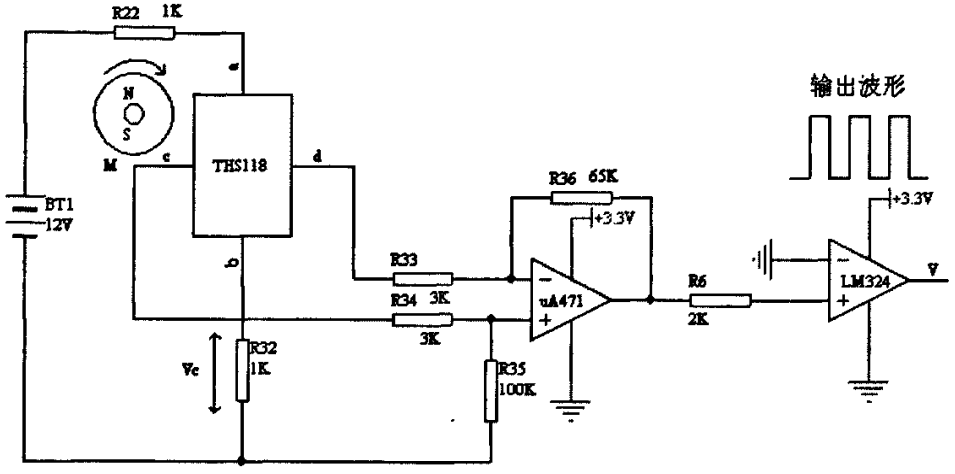


图 2-4 速度信号采集电路

Vo 的计算公式下所示：

$$V_o = \frac{R_{36}}{R_{33}} V_h$$

式中，Vh 为 THS118 的输出电压，由于 THS118 的输出电压是当 c 点电压等于 d 点电压时，输出电压 0V，c 点电压大于 d 点电压或小于 d 点电压时，输出电压为 55~140mV，由以上公式可知放大电路的增益为 65/3=21.67，这样可以得到 Vo 电压为 0~3.03V。将所得信号 LM324 进行整形，输出幅值为 3.3V 的方波。

(5) 开关信号的采集电路

开关信号主要包括刹车、喇叭、左右灯、大灯、气压、1 号和 2 号门、报警器等[14]。从各部件采集到的开关信号必须进行电压的变换，使其能符合 CPLD 的电平要求，这里采用 LM324 进行整形。如图 2-5 开关信号采集电路所示，D0~D7 为经过整形后的波形。

(6) 数据处理的 CPLD 实现

对采集到的开关信号和速度信号，必须经过 CPLD 的处理，才能被 DSP 接收并进行处理。这里对 CPLD 的编程采用原理图输入的形式。图 2-6 为数据处理的 CPLD 实现。

该模块的输入端意义分别为：

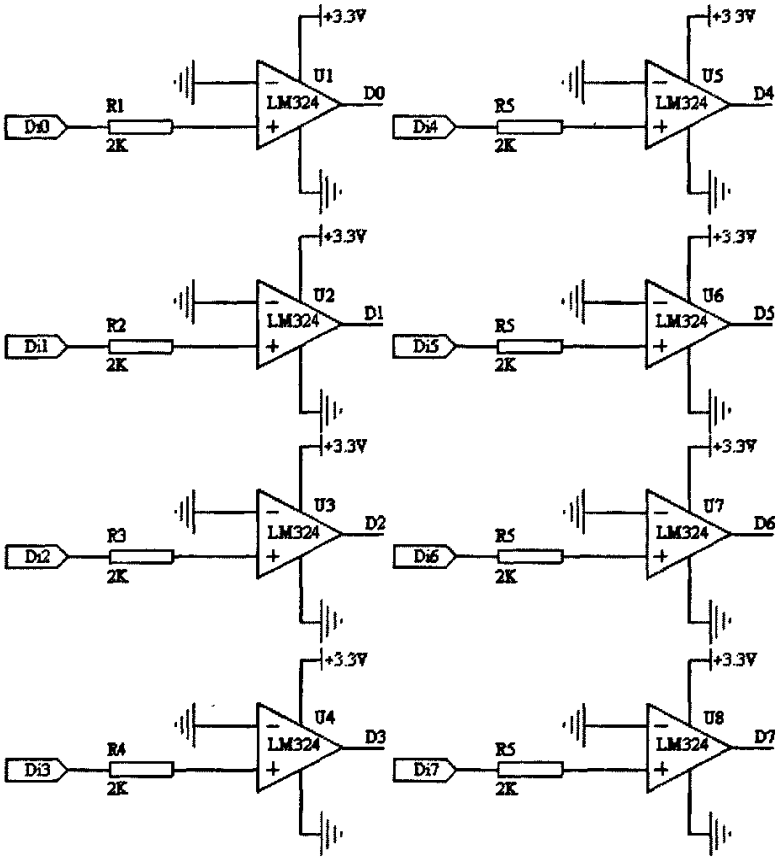


图 2-5 开关信号采集电路

CLK：工作时钟输入端；

V：速度信号输入端，该信号即图 2-4 的电路输出；

V_en：输出使能端；

D[7..0]：开关信号输入端；

该模块的输出端意义分别为：

V_int：数据输出等待状态信号；

Vo[7..0]：对采集到的速度信号脉冲进行记数，以 8 位数据的方式输出；

Do[7..0]：经处理的开关量的输出。

inst12 为一个 10 位计数器，这里对输入的工作时钟进行分频，inst7 为一个

8 位的计数器，对分频后的时钟信号 CLKDIV 进行记数，当脉冲计数到 0.2S 中时，输出信号 Vo_int 来表示 0.2S 的计数完成。inst8 为一个 8 位计数器，对输出的速度脉冲来进行计数。inst10 为一个 D 触发器，输入端为 Vo_int，输出端为 inst8 的清零信号。inst11 用于对输出数据的保持，方便 DSP 对数据的读取，inst13 为 74373b 锁存器，可以通过使能端口来完成对输出端口的状态控制，当不使能的时候，使起输出端口保持为高阻态，防止对数据线的干扰。inst5 的功能同 inst13。

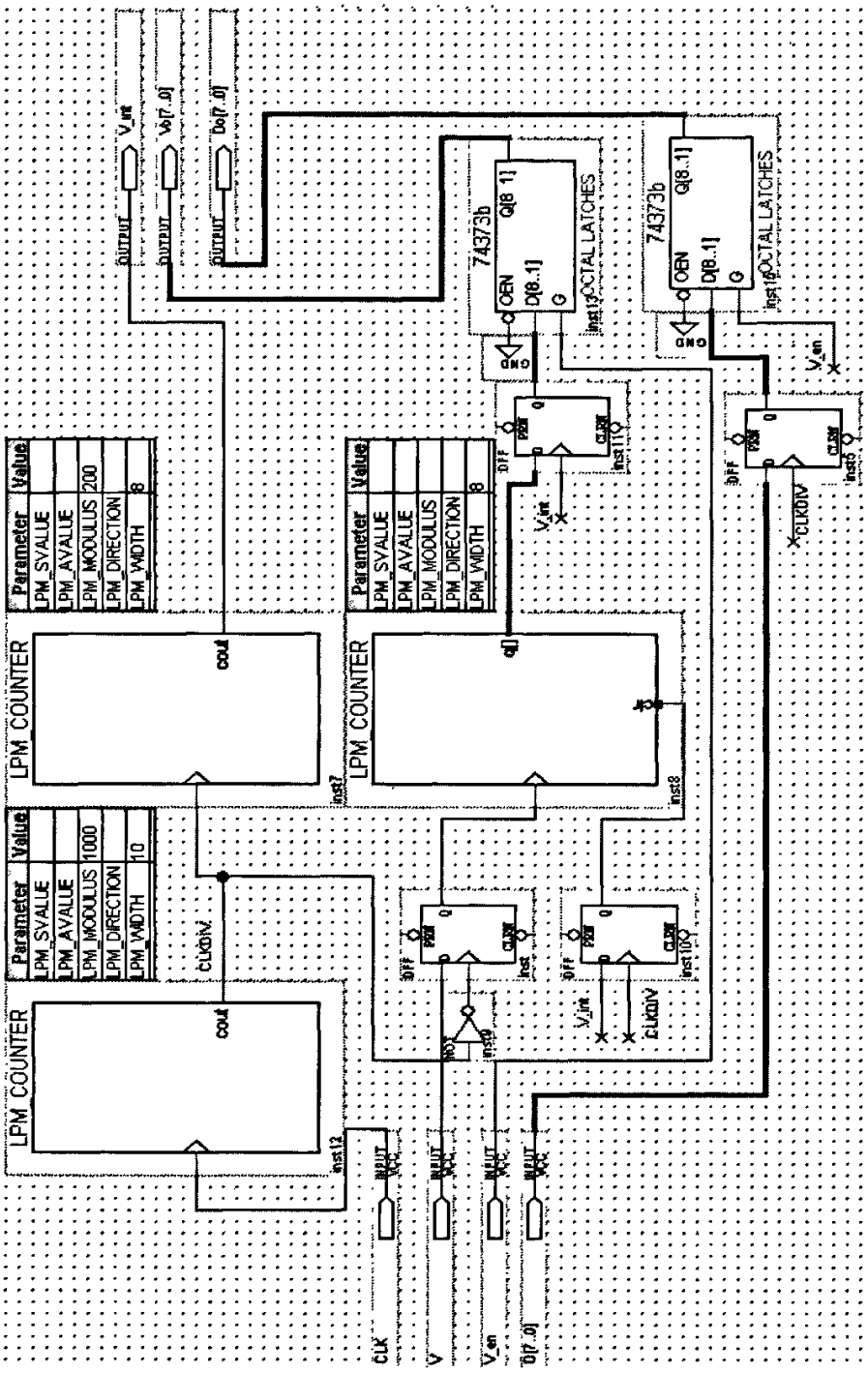


图 2-6 数据处理的 CPLD 实现

2.3.2 数据存储模块

由于汽车行驶状态数据需要保存到汽车出现交通事故后，因此数据存储芯片应具有掉电长时间保存数据的功能，并自身具有很高的可靠性。对本系统而言，数据存储分为两部分，一部分为汽车行驶行驶记录仪的程序的存储，另一部分是采集到的汽车状态数据的存储。这里采用 SST39VF010 作为汽车行驶记录仪程序的存储，而 SST39VF1601 作为汽车状态数据的存储。

SST39VF010 芯片是大容量存储器，容量为 32K*16 位，储存汽车行驶记录仪程序。SST39VF010 与 TMS320VC5416 的连接图如图 2-7 所示。

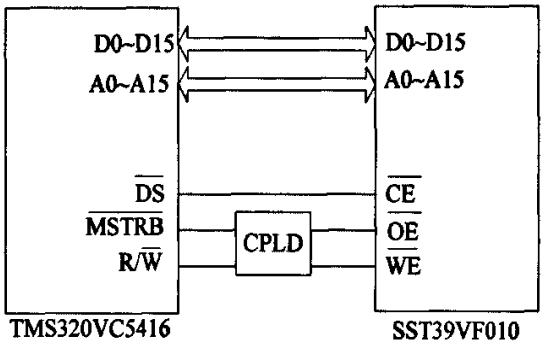


图 2-7 SST39VF010 与 DSP 连接图

逻辑门部分功能由 CPLD 来完成，其逻辑关系如下：

$$\overline{OE} = \overline{MSTRB} + \overline{R/W}$$

$$\overline{WE} = \overline{MSTRB} + R/\overline{W}$$

在对 CPLD 编程时采用原理图输入方式，如图 2-8 所示。

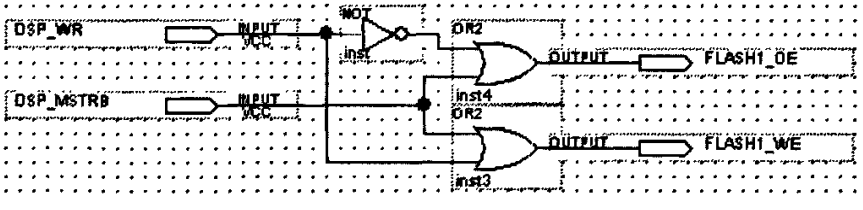


图 2-8 SST39VF010 与 DSP 连接关系的逻辑实现

SST39VF1601 芯片是大容量存储器，容量为 4M*16 位，采集到的汽车状态信息，可以把汽车在相当长的时期内运行状态记录下来。

SST39VF1601 与 TMS320VC5416 的连接图如图 2-9 所示。

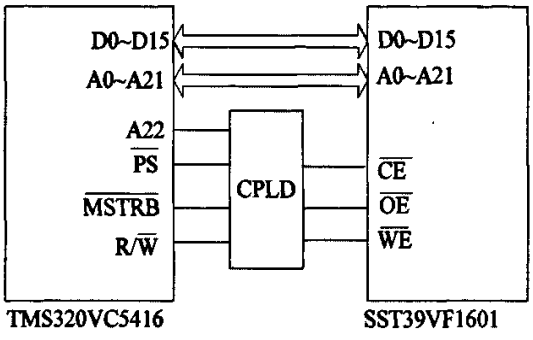


图 2-9 SST39VF1601 与 DSP 连接图

逻辑门部分功能由 CPLD 来完成，其逻辑关系如下：

$$\overline{CE} = A22 + \overline{PS}$$

$$\overline{OE} = A22 + \overline{MSTRB} + \overline{R/W}$$

$$\overline{WE} = A22 + \overline{MSTRB} + R/\overline{W}$$

在对 CPLD 编程时采用原理图输入方式，如图 2-10 所示。

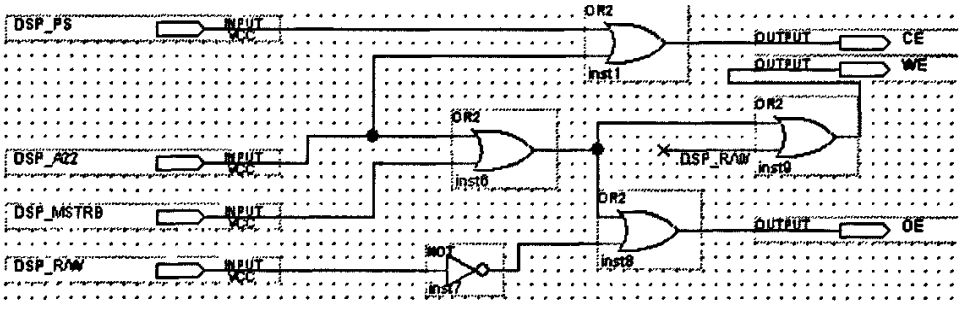


图 2-10 SST39VF1601 与 DSP 连接的 CPLD 逻辑图

2.4 通信模块

2.4.1 USB 主机模块

在记录仪的设计中，基于 CYPRESS 公司的 USB Host/Slave SL811HS 芯片构造 USB 主机系统(USB HOST)遵从 USB 1.1 通信协议，海量存储设备类协议和 FAT16 文件系统规范，实现对 U 盘的识别、读写文件等操作[15]。

鉴于 USB 系统的复杂性，整个 USB 系统的核心由 TMS320VC5416 和 CYPRESS 公司的 USB Host/Slave 接口芯片 SL811HS 组成。SL811HS 支持 USB1.1 全速和低速设备，提供 USB 主机的硬件接口及总线管理的物理机制，带有 USB 发送器及根集线器，满足了嵌入式 USB 主机的所需要的功能。由于 TMS320VC5416 外扩 FLASH，足够 USB 主机协议栈软件的运行要求。

SL811HS 主机模式下的功能框图如图 2-11 所示。在片选信号 nCS、读闸门信号 nRD、写闸门信号 nWR 等控制信号、地址线 A0 和数据总线 D0~D7 的作用下，它可以映射到处理器的 I/O 空间或存储器空间。SL811HS 片内有 256 字节的 RAM，其中低 16 字节是控制寄存器和状态寄存器，剩下的 240 字节用作数据缓存。地址线 A0 的使用比较特殊，访问芯片时，首先将 A0 置 0，通过 D0~D7 写入目标 RAM 地址，然后在下一次读写周期中，将 A0 置 1，这样 D0~D7 上就变成访问的数据。在每次读写操作后，RAM 地址指针会自动指向下一个数据单元。SL811HS 片内寄存器分为二部分，第一部分负责 USB 的传输，包括 USB 主机控制寄存器(USB Host Control Register)，USB 主机基址寄存器(USB Host Base Address)，USB 主机长度寄存器(USB Host Base Length)，USB 主机令牌，端点寄存器(USB Host PID， Device Endpoint)，USB 状态寄存器(USB Status)，USB 主机设备地址寄存器(USB Host Device Address)，USB 传输计数器(USB Transfer Count)；第二部分负责 SL811HS 的工作，包括控制寄存器 1(Control Register 1)、中断使能寄存器(Interrupt Enable Register)、中断状态寄存器(Interrupt Status Register)、SOF 低位计数器(SOF Counter LOW)、硬件版本寄存器(HW Revision Register)、SOF 高位计数器(SOF Counter HIGH)、控制寄存器 2(Control Register 2)。

处理器通过访问上述寄存器来进行数据传输控制和获取传输状态。另外，SL811HS 提供了 USB-A 和 USB-B 二组 USB 主机控制寄存器。因此可以采用乒

乓方式进行 USB 传输。而数据的 CRC 校验则由芯片自动完成[16]。

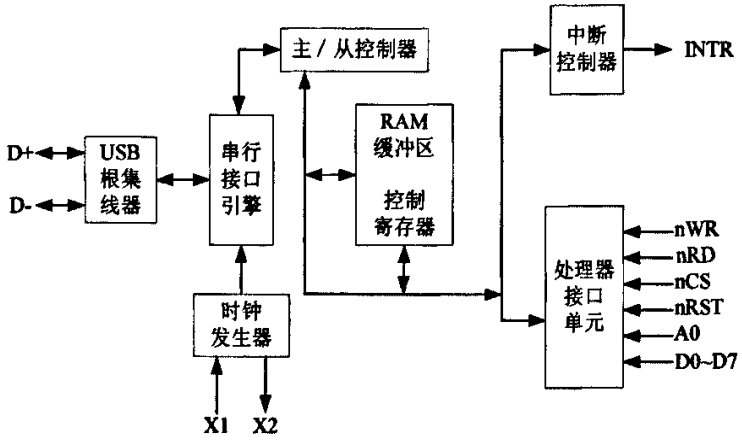


图 2-11 SL811HS 主机模式下的功能框图

TMS320VC5416 与 SL811HS 的硬件连接图如图 2-12 所示。其中 SL811HS 的 M/S 接低电平，工作在主机模式下，其中对 SL811HS 的逻辑译码由 CPLD 完成，其逻辑关系可用下列式表示：

$$\overline{CE} = \overline{A13} + A14 + A15 + \overline{IS}$$

$$\overline{OE} = \overline{A13} + A14 + A15 + \overline{IOSTRB} + R/\overline{W}$$

$$\overline{WE} = \overline{A13} + A14 + A15 + \overline{IOSTRB} + \overline{R/\overline{W}}$$

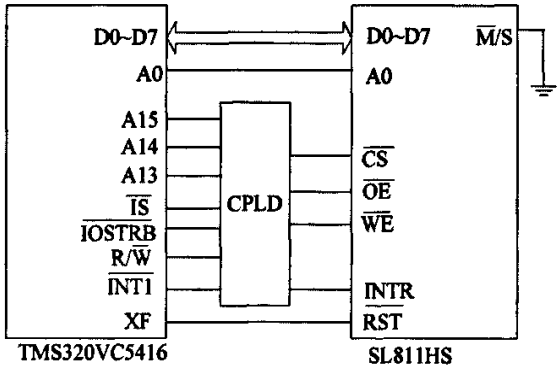


图 2-12 TMS320VC5416 与 SL811HS 的硬件连接图

在这里对 CPLD 的编程采用逻辑方框图来实现，如图 2-13 所示。

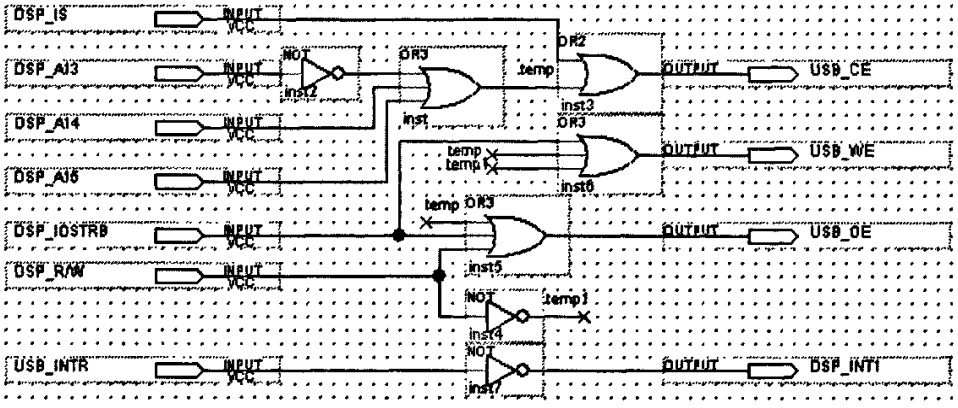


图 2-13 TMS320VC5416 与 SL811HS 的硬件连接的 CPLD 实现

两者的地址线 A0 和数据线 D0~D7 对应连接；DSP 的输出 XF 提供硬件复位脉冲。此外，由于 SL811HS 的中断请求 INTR 高电平有效，而 DSP 的中断是下降沿触发，所以 INTR 要经过非运算后才能接到 $\overline{INT1}$ 上。经过如上的映射，DSP 向奇地址 I/O 空间写数据对选择 SL811HS 的目标 RAM，访问偶地址 I/O 空间时传输相应的数据输出和时钟信号。

2.4.2 CAN 通信模块

CAN 总线最早是由德国 BOSCH 公司为解决现代汽车中的控制与测试仪器之间的数据交换而开发的一种数据通信协议。按照 ISO 有关标准，CAN 的拓扑结构为总线式，因此也称为 CAN 总线。相比于其他总线，CAN 总线由于其着重考虑并较好地解决了实时环境下数据传输的高速、可靠、抗干扰等诸多问题。如今 CAN 总线在汽车控制领域应用十分广泛。

在本系统中，CAN 总线接口使用 Philips 公司的独立 CAN 线控制器 SJA1000，并由光耦 6N137 进行总线隔离，CAN 总线收发器采用 MCP2551。SJA1000 兼 CAN2.0A 和 CAN2.0B 两种技术规范，最高位速率可达 1Mbits，支持 Intel 和 Motorola 微控制器类型。SJA1000 与 82C200 比较而言各方面的性能都有了很大的，提高尤其是在错误处理超载能力以及接收滤波等方面。SJA1000 有两种应用模式：BasicCAN 模式和 PeliCAN 模式。BasicCAN 模式符合 CAN

协议的 2.0A 标准，接收缓冲器有 64 个字节；PeliCAN 模式符合 2.0B 标准，能实现扩展数据格式，具有仲裁丢失捕获、错误代码读取等功能，设计灵活方便，其核心模块集成了位流处理、位定时、数据收发及错误管理等功能。本系统中采用的是 PeliCAN 模式[17]。

SJA1000 的地址和数据线是时分复用的，是通过 ALE 来锁存地址的；而 TMS320VC5416 则是地址线和数据线分开的，也就是说 TMS320VC5416 没有提供与 SJA1000 直接接口信号线。DSP 与 SJA1000 的连接框如图 2-14 所示。为使 TMS320VC5416 满足 SJA1000 的接口信号要求，从以下几点进行设计：

(1) 地址数据复用线的设计：将 VC5416 的数据线 D0～D7 通过电平转换芯片 SN74LVTH16245(目的是更好地带动及电平匹配)转换后，再作为 CAN 的地址/数据复用线，用 DSP 的数据线去选择 CAN 的内部端口和传送数据。

(2) 地址有效信号 ALE 及读写信号的产生：用 TMS320VC5416 的 A0、IOSTRB 和 R /W 的逻辑组合产生 SJA1000 的读和写选通信号。逻辑关系如图 2-15 所示。

(3) 片选信号 CS 的产生：用 TMS320VC5416 的 I/O 空间选通信号 IS 和高位地址的译码信号的逻辑组合产生 CAN 的片选 CS。逻辑关系如图 2-15 所示。以上的逻辑关系都是通过 CPLD 芯片实现。

这种方法是将 DSP 的数据线改为适应 CAN 控制器的数据地址线，为此将 DSP 的 A0 作为地址数据选择线。A0 =0 时，地址有效；A0 =1 时，数据有效，即用奇数地址选择端口，用偶数地址传送数据。同时，通过信号的逻辑组合，在地址有效期间不产生读写信号，而是产生满足 CAN 的地址有效信号 ALE；在数据有效期间产生满足 CAN 的读和写逻辑信号时序。

其中对 SJA1000 的逻辑译码由 CPLD 完成，其逻辑关系可用下列式表示：

$$\overline{CS} = A13 + \overline{A14} + A15 + \overline{IS}$$

$$ALE = A0 \cdot R/\overline{W} \cdot \overline{IOSTRB}$$

$$\overline{WR} = A0 + \overline{IOSTRB} + R/\overline{W}$$

$$\overline{RD} = A0 + \overline{IOSTRB} + \overline{R/\overline{W}}$$

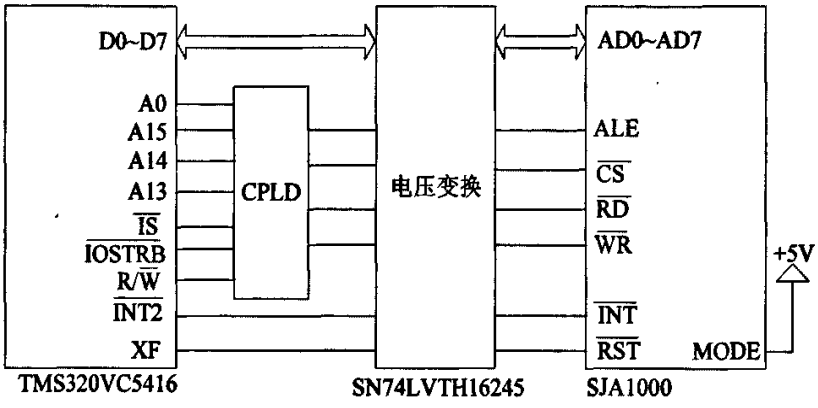


图 2-14 DSP 与 SJA1000 的连接框

在这里对 CPLD 的编程采用逻辑方框图来实现，如下图 2-15 所示。

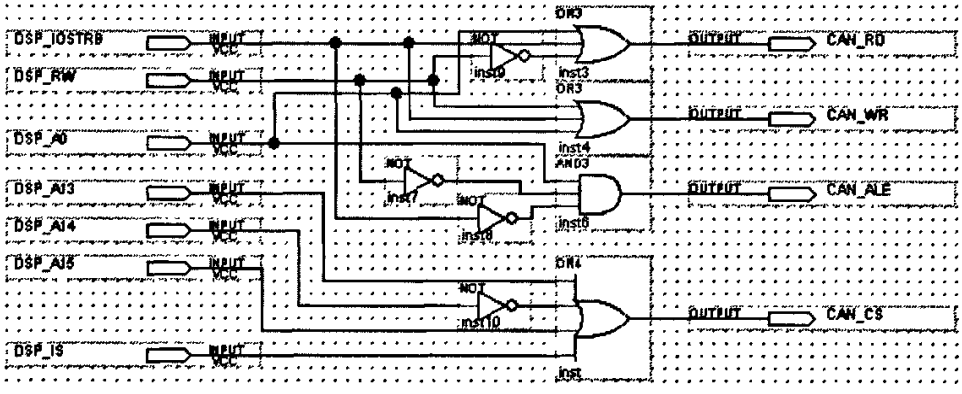


图 2-15 DSP 与 SJA1000 的连接逻辑的 CPLD 实现

2.4.3 串行通信模块

MAX3111E 是 Maxim 公司推出的全功能收发器，内部包括 UART、内置电泵电容和±15kVESD 保护的双 RS232 收发器其中 UART 部分采用了兼容 SPI/QSPI/MICROWIRE 串行接口，所以可节省线路板空间和微处理器的 I/O 引脚。UART 和 RS232 两部分电路采用公共电源和地，因此它们可以联合使用也可独立使用由于 RS232 部分使用低压差输出级，从而使双接收/发送接口能够在高速通信、正常电源下提供真正的 RS232 特性，而功耗仅为 600uA。

MAX3111E 可实现同步串行数据接口和异步串行数据接口的转换，并支持

1.8432MHz 或 3.6864MHz 两种晶振，可产生 300~230kB/s 范围内的通用波特率、内置一个 8 字接收 FIFO 缓冲；又由于其体积小，功耗低、通信速率高等特点，使其在控制和通信领域中有很好的应用前景。

MAX3111E 内部结构包括 UART 和 RS232 两个独立的部分其中，UART 部分包括兼容于 SPI 的串行接口、可编程波特率发生器、发送缓冲器和发送 1 位寄存器、接收缓冲器及接收 1 位寄存器、8 字接收 FIFO 以及 4 种可屏蔽中断发生器。RS232 部分包括自带电容的电泵，信号/SHDN 可实现硬件关断同时具有 ESD 保护结构，可以对±5kV 静电起保护作用。

McBSP 主要包括数据通路和控制通路两部分，通过 7 个引脚与外部器件相连。接口信号包括与接收数据总线相连的接收引脚 DR、与 McBSP 相连发送数据的发送引脚 DX、发送时钟引脚 CLKX、接收时钟引脚 CLKR、发送帧同步 FSX、接收帧同步 FSR。当 McBSP 处于时钟停止模式时，发送器和接收器是内部同步的，此时 McBSP 可以设置为 SPI 通信的主设备。发送端输出信号(BDX)作为 SPI 协议从设备的 MISO 信号，接收端输人信号(BDR)作为 SPI 协议从设备的 MISO 信号；发送帧同步脉冲信号(BFSX)作为从设备片选信号，而发送时钟信号(BCLKX)与 SPI 协议的串行时钟信号（SCLK)相对应。TMS320VC5416 与 MAX3111E 的接口电路如图 2-16 所示[18]。

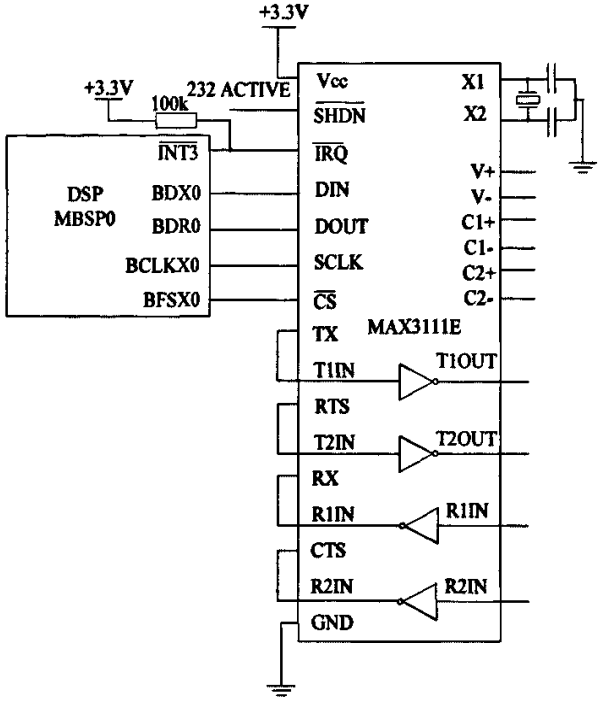


图 2-16 MAX3111E 与 DSP 连接图

2.5 人机接口模块

2.5.1 液晶显示模块

汽车行驶记录仪显示模块的功能是实时反映汽车行驶状态，同时提供人机操作接口。由于 LCD 具有低功耗、体积小、质量轻、超薄等诸多其他显示器无法比拟的优点，它广泛用于各种智能型仪器和低功耗电子产品中。本系统采用信利(TRULY)公司的 M240128-1A1 液晶显示模块，它集成 T6963C 控制器，典型的工作电压为 5V，同时具有宽电压输入的特性，能与 DSP 无缝结合，XD0~XD7 是 3 态数据总线；/RD，/WE 是读写选通信号，低电平有效；/CS 为片选信号，低电平有效；C/D 为通道选择信号，1 为指令通道，0 为数据通道；RST 为复位信号，低电平有效。“VO”引脚提供 M2401281A1 液晶显示的对比度电压，可以将液晶的“VOUT”引脚经一个 5kB 电位器分压后接至 VO 脚，将液晶调到适合人眼的亮度。“FS”引脚为液晶字体选择信号，高电平为 6×8 的点阵，低电平为 8×8 的点阵[19]。

本文以比较常见的 T6963C 液晶显示控制器为例做介绍。T6963C 液晶显示控制器多用于中小规模的液晶显示器件，常被装配在中小规模图形液晶显示模块上， 以内藏控制器型图形液晶显示模块的形式出现。其结构框图如图 2-17 所示。

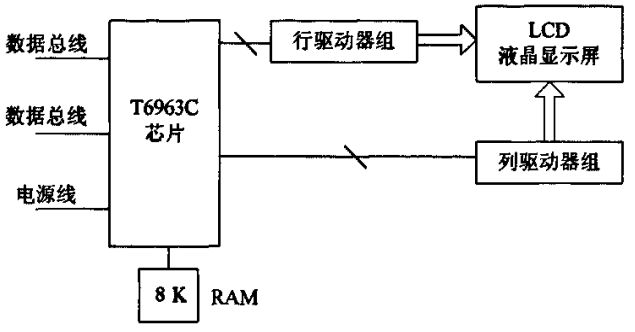


图 2-17 T6963C 结构图

TMS320VC5416 与 T6963C 的硬件连接图如图 2-18 所示。由于 DSP 和液晶控制器 T6963C 的工作电压不一样，需要电压变换。由于 DSP 相对与液晶响应速度而言比较快，这里需要对其数据端口进行锁存一下，这里采用 CPLD 完成。同时 T6963C 的逻辑译码也由 CPLD 完成。在这里对 CPLD 的编程采用逻辑方框图来实现，如图 2-19 所示。

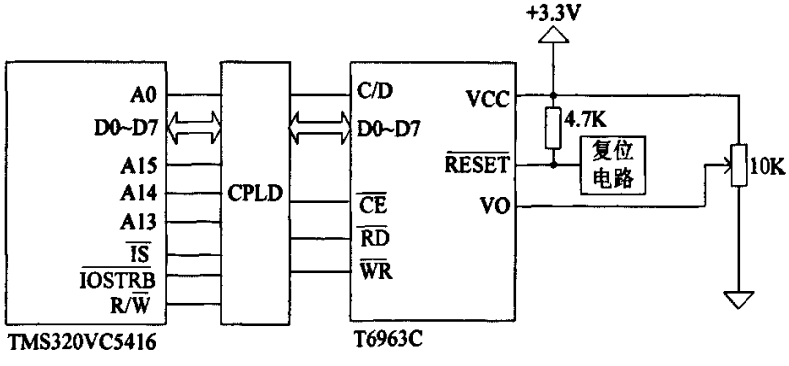


图 2-18 DSP 与 T6963C 连接图

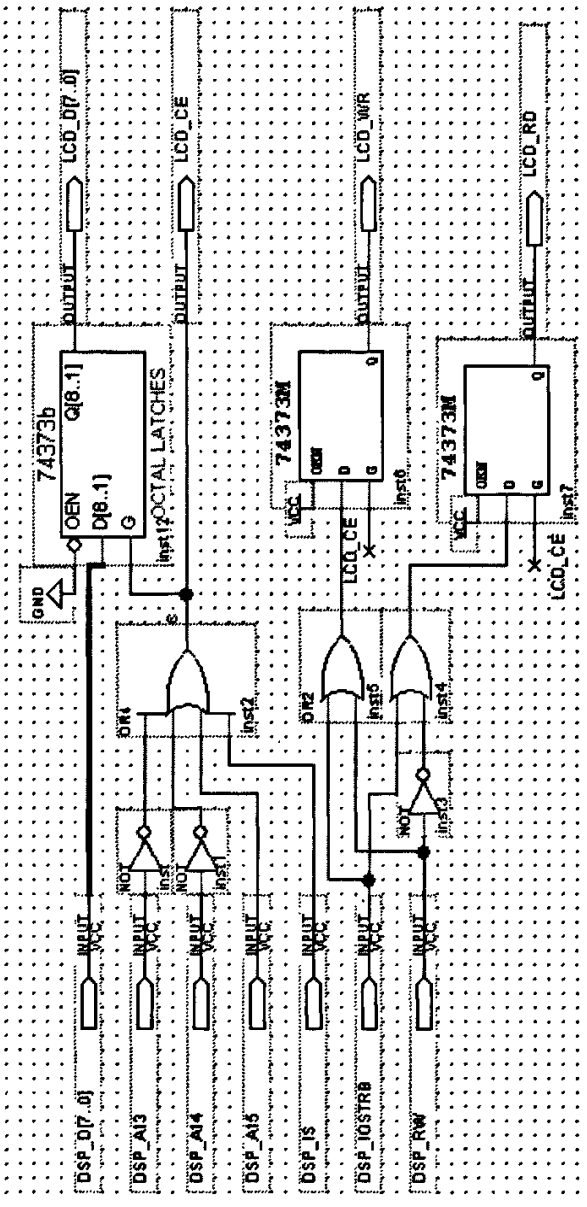


图 2-19 DSP 与 T6963C 逻辑连接的 CPLD 实现

2.5.2 IC 卡模块

目前市场上主要存在的身份识别的技术包括接触式 IC 卡，非接触式 IC 卡（RF 卡）及密钥等，而接触式 IC 卡因为成本较低，安全可靠，目前在市场上

的应用范围最广，所以在本系统中最终选用接触式 IC 卡作为司机身份识别的方式。接触式 IC 卡通常分为三类：(1) 存储器卡：含有 EEPROM 及其控制电路，但无加密逻辑，缺乏安全保护，适用于不需要保密要求的地方。(2) 逻辑加密卡：由加密逻辑电路和 EEPROM 组成。适用于需要保密的地方。(3) CPU 卡：称为智能卡(SmartCard)，这种卡内不仅有 EEPROM 等存储器，还带有 CPU 及其操作系统和加密算法(DEA 或 RSA)。它具有处理和存储两大功能，保密性能好，不过价格相对较贵。由于逻辑加密卡具有一定的保密逻辑功能，不像存储器卡那样可以被自由地擦写，也不需要 CPU 卡那样进行复杂的密码计算同时逻辑加密卡也没有 CPU 卡的成本高。本系统基于保密性和成本两方面考虑采用逻辑加密卡。目前市场上的逻辑加密卡种类较多，比较有代表性的有 ATMEL 公司的 AT 系列和 SIEMENS 公司的 SLE 系列。根据市场的供给情况逻辑加密卡选用了 SIEMENS 公司设计的 SLE4442 卡[20]。图 2-20 给出了 SLE4442 卡的存储示意图。主存储器的容量为 256 个字节，每个字节为 8 位。主存储器可分为保护区和应用区，地址单元为 00H~1FH 的 32 个字节是保护区，带位保护功能，一旦实行保护后，被保护的单元不可擦除和改写。保护区中没有设置为保护状态的字节，其使用与应用区完全相同。SLE4442 还提供一个 4 字节的密码存储器，其中 0 单元是错误计数器(EC)，只用了该单元的后三位，在 SLE4442 卡个人化后， (EC) = 111。其余 3 个字节是密码存放单元(PSC)。上电后，除了密码以外，整个存储器都是可读的。如要擦除或改写卡中内容，必须校验密码，相同才可进行。这时才可读出密码内容，如果需要的话，还可以改写新的密码。如果输入的数据与原有密码比较为不正确，错一次，(EC) 为 011，再一次不正确， (EC)为 001，三次不正确的话，(EC)则为 000，这时卡就会自锁，不能再改写卡中内容卡就失去作用了。如三次比较里面有一次正确，则(EC)恢复为 111。

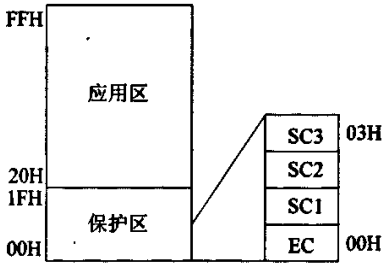


图 2-20 SLE4442 卡的存储示意图

2.5.3 键盘接口模块

汽车行驶记录仪须提供人机接口，这里采用 4×4 键盘，采用 CPLD 对键值进行扫描，所采集到的数据再送 DSP 进行处理。键盘模块接口图如图 2-21 所示。

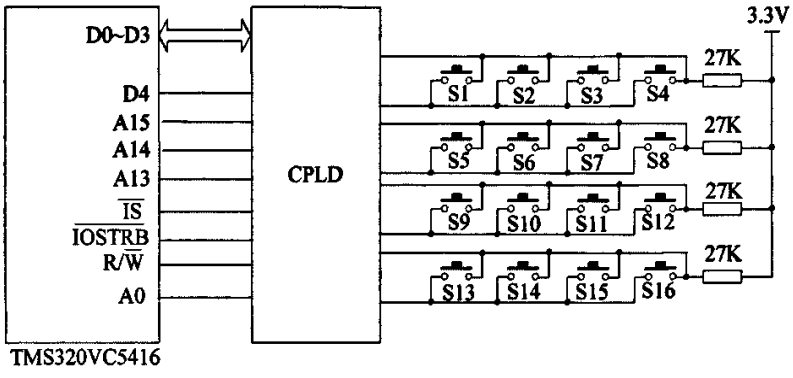


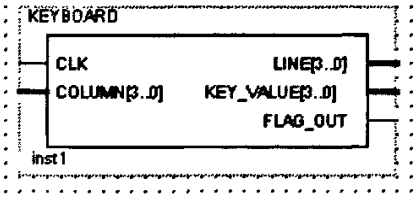
图 2-21 键盘模块接口框图

这里对键盘模块的处理主要用 CPLD 完成，这里对 CPLD 编程采用了两种方法，原理图法和程序输入法。首先完成对键盘扫描的模块的 VHDL 程序编写，程序清单如下所示：

```
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
ENTITY KEYBOARD IS
 PORT(
  CLK   :IN       STD_LOGIC;
  COLUMN      :IN       STD_LOGIC_VECTOR(3 DOWNTO 0);
  LINE        :BUFFER STD_LOGIC_VECTOR(3 DOWNTO 0);
  KEY_VALUE :OUT      STD_LOGIC_VECTOR(3 DOWNTO 0);
  FLAG_OUT   :OUT      STD_LOGIC
  );
END ENTITY;
```

```
ARCHITECTURE A OF KEYBOARD IS
 BEGIN
----------输出扫描行线的值----------
 PROCESS
  VARIABLE X_VALUE :STD_LOGIC_VECTOR(3 DOWNTO 0):="1110";
 BEGIN
  WAIT UNTIL RISING_EDGE(CLK);
  IF X_VALUE="1111"THEN
   X_VALUE:="1110";
  ELSE
   X_VALUE:=X_VALUE(2 DOWNTO 0)&'1';
  END IF;
  LINE<=X_VALUE;
 END PROCESS;
----------检测按键操作----------
 PROCESS(CLK)
  VARIABLE HIT_FLAG   :STD_LOGIC; --按键标志
  VARIABLE KEYVAL     :STD_LOGIC_VECTOR(3 DOWNTO 0);
  VARIABLE XY_VALUE   :STD_LOGIC_VECTOR(7 DOWNTO 0);
 BEGIN
 WAIT UNTIL RISING_EDGE(CLK);
 IF COLUMN="1111"THEN                --松开按键，传递键值
  IF HIT_FLAG ='1'THEN
   HIT_FLAG:='0';
   FLAG_OUT<='1';                   --输出键值标志
   KEY_VALUE<=KEYVAL;                --输出键值
  ELSE
   FLAG_OUT<='0';
  END IF;
 ELSE
  FLAG_OUT<='0';
```

```
  HIT_FLAG:='1';                    --有键按下
  XY_VALUE:=COLUMN & LINE;
  CASE XY_VALUE IS
   WHEN "11101110"=>KEYVAL:="0000";  --0
   WHEN "11101101"=>KEYVAL:="0001";  --1
   WHEN "11101011"=>KEYVAL:="0010";  --2
   WHEN "11100111"=>KEYVAL:="0011";  --3
   WHEN "11011110"=>KEYVAL:="0100";  --4
   WHEN "11011101"=>KEYVAL:="0101";  --5
   WHEN "11011011"=>KEYVAL:="0110";  --6
   WHEN "11010111"=>KEYVAL:="0111";  --7
   WHEN "10111110"=>KEYVAL:="1000";  --8
   WHEN "10111101"=>KEYVAL:="1001";  --9
   WHEN "10111011"=>KEYVAL:="1010";
   WHEN "10110111"=>KEYVAL:="1011";
   WHEN "01111110"=>KEYVAL:="1100";
   WHEN "01111101"=>KEYVAL:="1101";
   WHEN "01111011"=>KEYVAL:="1110";
   WHEN "01110111"=>KEYVAL:="1111";  --15
   WHEN OTHERS=>NULL;
  END CASE;
 END IF;
 END PROCESS;
END a;
```



2-22 键盘扫描模块图

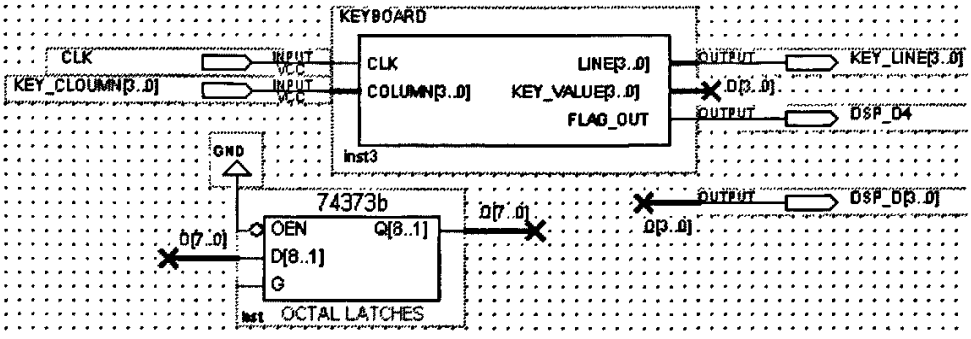
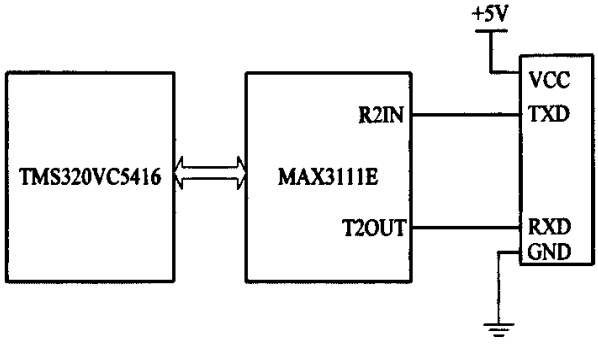


图 2-23 键盘完整模块

对上述清单所列程序将其编译成模块，可得键盘扫描模块图 2-22。其中 CLK 为时钟输入端，COLUMN[3..0]接在图 2-21 所示的 4*4 键盘的行上，而 LINE[3..0]接在 4*4 键盘的列上，KEY_VALUE[3..0]为输出的键值，FLAG_OUT 为有键值输入的信号。键盘扫描模块必须配合外围模块才能构成一个完成的功能模块，这里总称为键盘模块，如图 2-23 所示。

2.5.4 数据打印模块

本设计中选用的是讯普公司的 SP-T 台式打印机，通过串口与记录仪连接。该台式微型打印机具有体积小、重量轻、功能强、易操作、使用方便等特点。该机的内藏式供纸方式使打印机更加紧凑，轻触按键和指示灯集中于上盖表面，操作直接，状态醒目；具有在线选择功能，可以暂停打印，随时上纸。采用 25 线 D 型插座作为并口或 RS-232C 串行接口的连接器。符合 ESC/P 标准的控制命令与标准的 IBM 和 EPSON 打印机兼容[21][22]。作为各种智能化仪器、仪表和各种微型计算机、单片机的打印输出设备。它不但可以打印标准的国标二级字库、ASCII 字符及块图符，还可打印点阵图形和曲线，打考虑到记录仪所需打印的数据量不大，设计中使用串行接口与记录仪相连。由于打印机的串行接口可以为 RS-232C 或 TTL 电平，因此可以直接和 MAX3111E 的 R2IN，T2OUT 端口连接。打印机部分电路示意如图 2-24 所示。



2-24 打印机接口图

2.6 其他模块

2.6.1 时钟模块

汽车行驶记录仪在对数据进行采集过程中，需要实时记录和存储各种信息，这就要求此系统必须要有实时时钟发生器。它对行车记录仪的实时性、准确性和可靠性都起着至关重要的作用。为此，我们选用 DALLAS 半导体公司的 DS12CR887 实时时钟芯片作为该系统的时钟发生器。它是一种高性能、低功耗并且内部带有静态 RAM 的实时时钟芯片，时钟校准也较为容易，若采用专用晶振器，几乎无需调整即可达到国家要求的时钟误差标准[23]。此外，考虑到当汽车行驶记录仪停电时，需具备对时钟电路单独供电的系统，而该芯片正好具有备用电池充电和切换管理功能。DS12CR887 片内地址空间为 00H~7FH，其中 00H 为秒单元，01H 为闹秒单元，02H 为分钟单元，03H 为闹分单元， 04H 为时单元，05H 为闹时单元，06H 为星期单元，07H 为日单元，08H 为月单元，09H 为年单元，0AH~0DH 单元分别为控制寄存器 A、B、C、D。0EH~7FH 为用户 RAM 区，可用来在系统掉电时保存数据。通过访问 A、B、C、D 四个寄存器，可随时设置和了解 DS12CR887 的工作方式。

DSP 与 DS12CR887 电路连接图如图 2-25 所示，DS12CR887 有两种接口总线时序工作方式，此系统中 DS12CR887 工作在 Intel 总线时序方式，其写命令时序如图 2-26 所示，读命令时序如图 2-27 所示。从 DS12CR887 的时序图可以看出，在一次读或写操作中，地址/数据复用总线上先出现地址，后出现数据。

写操作时，当片选信号 CS 有效时，地址锁存信号 AS 的下降沿将 AD0～AD7 上的数据锁存作为地址(AS 高电平的宽度 PWASH 不小于 45ns 时，锁存地址有效)；随后读写信号 R/W 为低电平(低电平宽度 PWEH 不小于 90 ns)，在 R/W 的上升沿将 AD0～AD7 上的数据写入 DS12CR887，在 R/W 的上升沿要求 AD0～AD7 的数据稳定时间不为小于 70ns(即 tdsw>70ns)，通过上述时序，才完成一次写操作。读操作同样首先将数据线(AD0～AD7)上的信号锁存为 DS12CR887 需要的地址，然后 DS12CR887 才能在 AD0～AD7 上输出有效数据。DSPTMS320VC5416 在一次操作中，数据线输出数据，地址线输出地址。从这个特点出发，设想用 TMS320VC5416 的两次操作产生的时序来完成 DS12CR887 的一次操作。具体思路如下：首先在 TMS320VC5416 的数据线 D0～D7 上输出 DS12CR887 需要的地址；如果是写操作，经过一定延迟后在数据线图 D0～D7 上输出需要写入到 DS12CR887 的数据；如果是读操作，则经过一定延迟后通过数据线 D0～D7 读入数据。具体的逻辑可参考文献[24]。

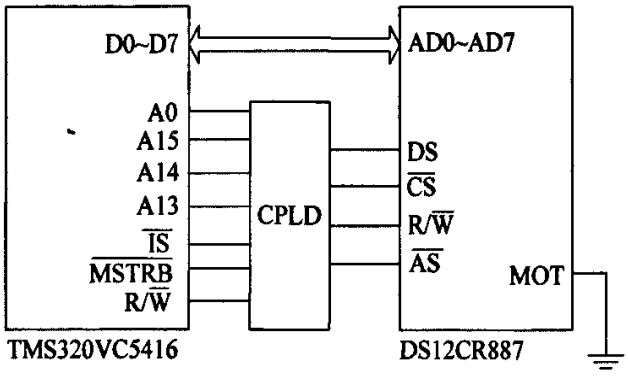


图 2-25 DS12CR887 电路实现

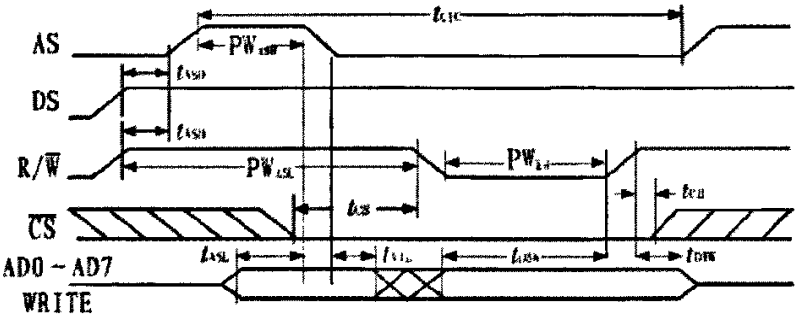


图 2-26 DS12CR887 写命令时序图

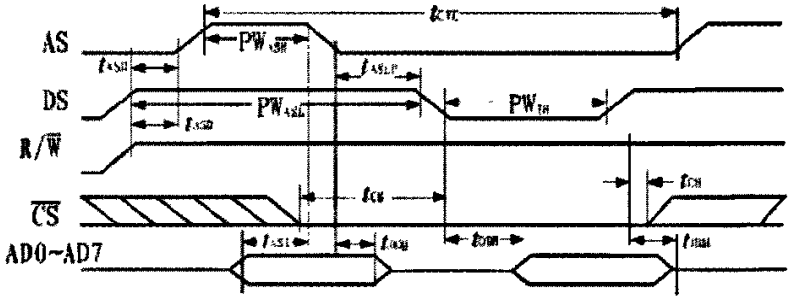


图 2-27 DS12CR887 读命令时序图

2.6.2 电源模块

在系统设计中，本文考虑的是正常的车载电压 DC12V，考虑电源电压到波动情况。采用输入范围较宽的 DC-DC 将输入的车载电压转换成 5 伏直流电源，提供给内部电路。在本设计中采用了 DC-DC 转换芯片在系统设计中，本文考虑的是正常的车载电压 DC12V，考虑电源电压到波动情况，采用输入范围较宽的 DC-DC 将输入的车载电压转换成 5 伏直流电源，提供给内部电路。在本设计中采用了 DC-DC 转换芯片 MC34063[25][26][27]。MC34063 是 MOTOROLA 推出的一款高性能、外围电路简单的 DC-DC 转换芯片。可组成降压式、升压式或电压反转式开关电源。其输入电压范围为 3~40V，工作电流典型值为 2.5mA，内部开关管电流 1.5A，输出电压可设定或连续可调，工作频率由外部电容确定，最高 100kHz，内部有精度为 2%的 1.25V 基准电压源，并有输出电流限制电路。

相应的电路图如图 2-28、图 2-29 和图 2-30 所示。

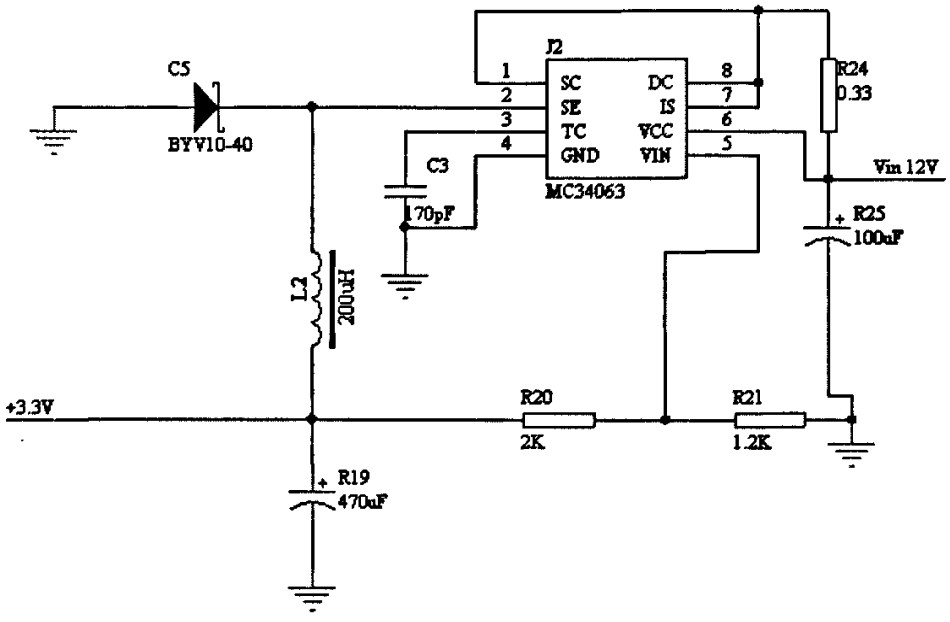


图 2-28 +3.3V 电源

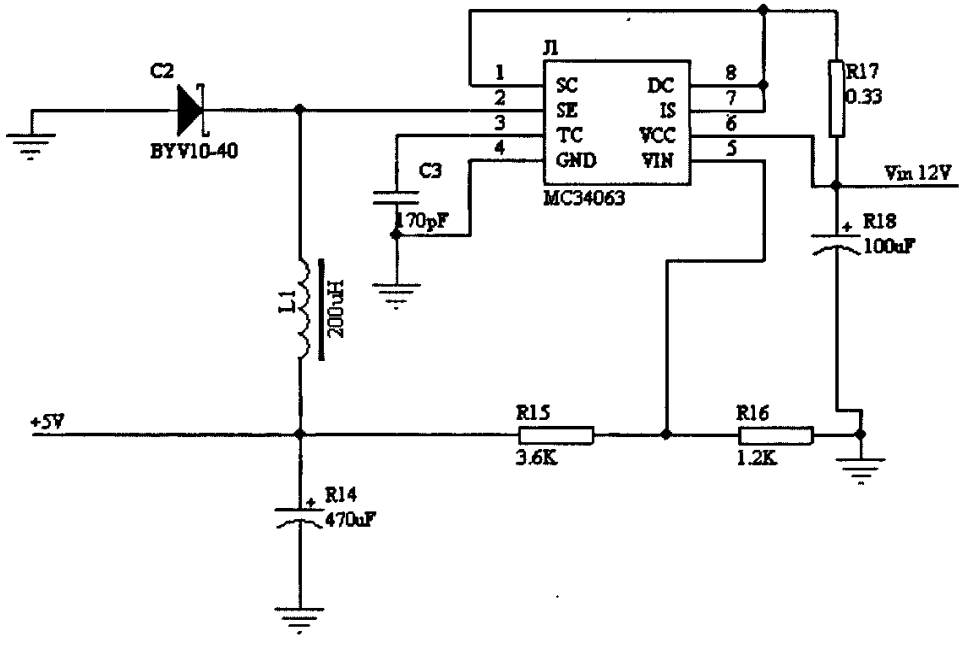


图 2-29 +5V 电源

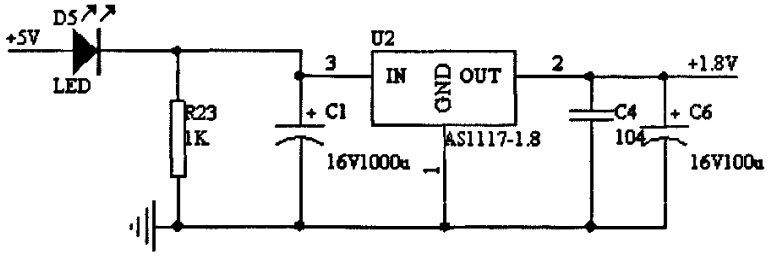

D5
+5V
LED
R23
1K
U2
3
IN
GND
OUT
2
+1.8V
+ C1
16V1000u
AS1117-1.8
C4
104
+ C6
16V100u

图 2-30 +1.8V 电源

第 3 章　汽车行驶记录仪软件设计

系统的硬件电路设计完成后，必须掌握应用程序开发环境，这样才能对所设计的硬件平台中各模块进行测试以及编写相应的底层驱动进行软硬件联调。

3.1 DSP 软件开发环境及相关文件

3.1.1 DSP 开发环境与语言

TI (Texas Instruments)公司 CCS(Code Composer Studio)是一个完整的 DSP 集成开发软件环境，也是目前最优秀、最流行的 DSP 开发软件之一。它具有实量、多任务、可视化的软件开发特点，已经成为程序设计、制作、调试、优化的有效工具[28]。

TI 公司的 DSP 芯片既可以使用汇编语言，也可以使用 C 语言进行软件开发。多数情况下，考虑到软件的可移植性问题，应尽量采用 C 语言进行开发。只有对代码效率要求很高的软件模块，使用汇编语言来编写。在 CCS 下进行软件开发，用户首先要创建一个工程，该工程包含所有的源程序文件以及要调用的程序库文件。然后用户用 C 语言或汇编语言编写程序并添加至工程。源程序编写完毕后，就可以进行编译、连接。C 编译器 (C Compiler) 将 C 源程序编译成汇编语言代码。编译时，可以选择是否让 CCS 进行编译优化。汇编器 (Assembler) 将汇编语言代码汇编为机器语言代码，经过汇编后的机器语言代码使用 COFF 文件格式。链接器 (Linker)将汇编器输出的各个目标文件链接成为一个可执行的 COFF 文件。在链接的过程中，链接器会根据.CMD 文件提供的信息将存储空间分段，并将相应的程序代码分配到各个存储空间段。

在用 C 语言进行程序开发时，除了源程序文件外，还有两个重要的文件需要编写，它们是.CMD 文件和中断向量表文件。.CMD 文件后面将做详细介绍，这里首先简单介绍一下 DSP 的中断向量文件[29]。

中断向量文件中存放的是中断向量表，当 DSP 产生中断时，PC 指针会根据中断的类型指向中断向量表中该中断向量所在的位置，然后执行中断程序。

由于 DSP 的中断向量空间有限而可能导致不能存放下完整的中断服务程序，因此在编写程序时一般会在每个中断向量所在的位置处放置一条跳转指令以使程序能跳向真正的中断服务程序处在的位置处执行。在编写本系统的中断向量表时，需要注意五个问题：一是每个中断向量都是 4 个字对齐并且没有用到的中断要用“RETE”指令返回；二是第一个复位向量必须跳转到_c_int00 地址处；三是必须使能外部中断 0 和 DMA 中断 0，并在相应的中断处加上跳转指令以跳转到中断服务程序处；四是中断向量表的位置必须由处理器状态寄存 PMST 中的 IPTR 位（高 9 位）和左移两位后的中断向量序号（低 7 位）决定，因此中断向量表必须加载到相应的位置处；五是中断向量表必须和源程序在一个程序页面空间内，因为当发生中断跳转时，并不会加载扩展程序页面指针 XPC 的值，也就是说程序跳转只会发生在现在程序运行的页面内。

对于源程序的编写，由于 C 语言具有修改和移植方便、可读性好等特点而汇编语言代码效率高，对硬件操作更方便，因此在设计时使用 C 和汇编混合编程。在 DSP 的开发中，C 和汇编混合编程一般有四种方法：

1. 单独汇编汇编语言模块并将它们与编译后的 C 模块链接。
2. 在 C 源程序中使用汇编语言变量和常数。
3. 使用在线汇编语言直接嵌入到 C 源程序中。
4. 在 C 源程序中直接使用内部函数调用汇编汇编语句。

3.1.2 DSP 程序的引导加载

DSP 应用系统开发中的应用日益广泛，为了满足不同的应用系统要求，DSP 提供了五种上电引导方式[30]，即 HPI 引导模式、8 位/16 位的并行引导模式、8 位/16 位的标准串口引导模式、8 位串行 EEPROM 引导模式和 8 位/16 位的 I/O 引导模式[31][32]。引脚 MP/MC 用来配置 DSP 上电时的处理器模式，当 MP/MC=1 时将处理器配置为微处理器模式；在该模式下，DSP 上电时，内部集成 ROM 在程序空间透明不可见，程序指针 PC 自动指向外部程序空间第 0 页的 0FF80H 地址，从该地址处直接取指执行程序[33]。

当 MP/MC=0 时将处理器配置为微计算机模式，内部集成 4K 字的 ROM 被映射在程序空间，包括引导装载程序律和律扩充表、正弦对照表、工厂测试码和中断向量表。其中，引导装载程序位于地址空间 0F800H~0FBFFH。程序从片内 ROM 的 0FF80H 地址开始执行程序。0FF80H 地址存放的是中断向量表，它

实为一条分支转移指令(BD 0F800H)，使程序跳转至 0F800H 执行自举引导程序(BootLoader)。BootLoader 是固化在 DSP 芯片内 ROM 中的一段程序代码，其功能是将用户程序从外部加载至片内 RAM 成扩展的 RAM 中，引导程序首先对 CPU 进行初始化，包括关闭所有可屏蔽中断，设置数据页指针为 0、清中断标志寄存器；置 HM=0 将外部地址、数据线挂起，处理器执行内部操作；设置 OVLY=1 将片上 DARAM0-3(0080H-7FFFH)既映射到程序空间，又映射到数据空间，DROM=0 将片上 DARAM4-7(08000H-0FFFFh)映射在外部数据空间；SWWSR=07FFFH 在外部扩展程序、数据空间插入 7 个等待状态使其高速运行。在搬运程序之前，BootLoader 首先完成初始化工作。初始化设置内容包括：设置使中断无效(INTM=1)，内部 RAM 映射到程序数据区 (OVLY=1)，对访问程序区和数据区均设置 7 个等待状态等。详细的引导过程如下图 3-1 所示[34][35]。由于本系统采用并行引导方式，故 MP/MC=0，其中程序数据存放 SST39VF010 中，具体的硬件连接可见前介绍。DSP 工作在微控制器模式下，并行引导方式从数据空间 0FFFFH 开始读取引导表的第一个字。

加电后，如果 MP/MC=0，DSP 工作于微计算机方式，执行片内 ROM 中的引导装载程序，引导的过程为：引导程序从外部存储器 FLASH 中特定地址获得引导装载表的存放地址，从该地址读取引导装载表，将该表的可执行代码搬移到 DSP 相应的 RAM 中执行。

引导装载表需按 TI 公司规定的格式来创建，该表中存放着 DSP 特殊寄存器如 SWWSR，BSCR 等的值，程序入口地址，各段目标首地址，各段长度及程序代码。引导装载表的内容和顺序如图 3-2 所示。

在线编程实现并行自举的步骤可归纳如下：

(1)编写烧写程序和 C5416 脱机运行的用户程序。烧写程序用于把用户程序及中断向量表按照(引导装载表)格式写入 FLASH。

(2)将编译链接后的用户程序下载到 C5416 芯片的 RAM。

(3)把编译链接后的烧写程序下载到 C5416 芯片的 RAM。

(4)设置 MP/MC=1，DSP 工作于微处理器方式，运行烧写程序，对 FLASH 在线编程。

(5)设置 MP/MC=0，DSP 工作于微计算机方式，C5416 上电后开始并行自举过程。

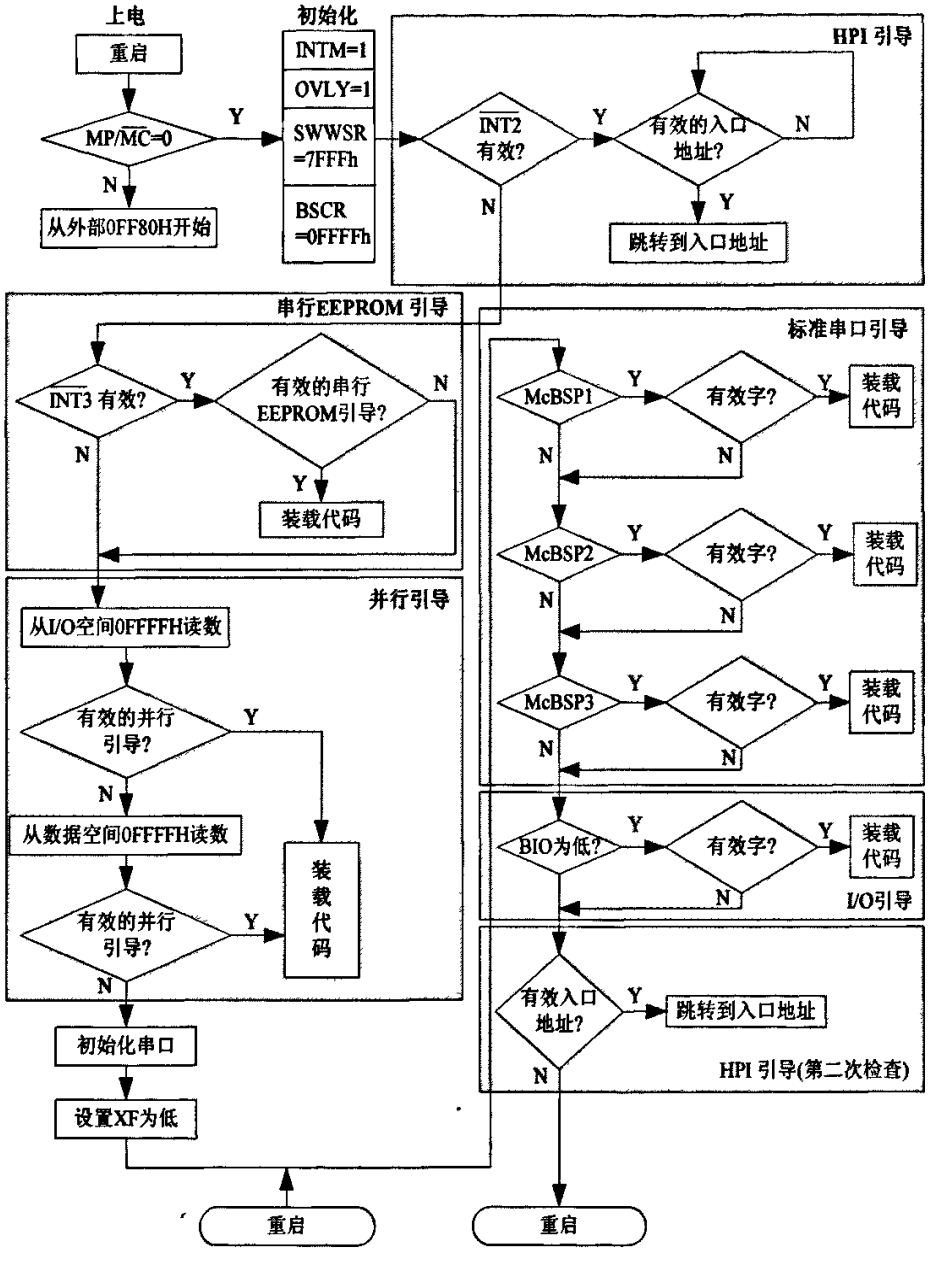


图 3-1 DSP 的引导加载过程

08AAH 或 10AAH
SWWSR 寄存器初始化值
BSCR 寄存器初始化值
程序入口地址 XPC
程序入口地址 PC
第一个程序块大小
第一个程序块目标地址 XPC
第一个程序块目标地址指针
程序代码 1
……
程序代码 N
……
最后程序块大小
最后程序块目标地址 XPC
最后程序块目标地址指针
程序代码 1
……
程序代码 N
……
代码结束标志 0000H

图 3-2 引导装载表的内容和顺序

3.1.3 DSP 存储空间的划分

汽车行驶记录仪要完成对大量的汽车状态数据采集，最后存放到相应存储器中。所以 DSP 的存储空间可以分为几个部分，一个部分是用来存放汽车行驶记录仪的程序代码，另一部分存放采集到的汽车状态数据，还有一部分用来保存程序运行时所需的数据和变量。TMS320VC5416 的存储空间分为三个部分：分别是程序空间，数据空间和 I/O 空间，其中程序空间可以扩展到 8192K（8M）字，因此 TMS320VC5416 的地址线有 23 根，并且 DSP 内部增加了一个额外的

寄存器—程序计数器扩展寄存器（XPC），可以计数 0-127，表明了程序空间的 128 页，每页有 64K 字；数据空间和 I/O 空间大小都为 64K 字，其中 I/O 空间全部映射到了片外，可以使 DSP 很方便的访问外部的器件[36]。

TMS320VC5416片上包含了随机存取存储器(RAM)和只读存储器(ROM)，其中 RAM 又可分为双访问 RAM（DARAM）和单访问 RAM（SARAM），其大小都是 64K 字，ROM 大小为 16K 字。DARAM 可以在一个机器周期内对其同时读或写，SARAM 在一个机器周期内只能读或写，显然对 DARAM 的访问要快于对 SARAM 的访问。RAM 始终映射到数据空间，但也可以映射到程序空间，ROM 只能映射到程序空间。TMS320VC5416 的 PMST 寄存器的三个位影响着存储器的配置：MP/MC，OVLY 和 DROM。具体影响如下：

MP/MC 位

若 MP/MC=0，则片内 ROM 安排到程序空间

若 MP/MC=1， 则片内 ROM 不安排到程序空间

OVLY 位

若 OVLY=1， 则片内 RAM 安排到程序和数据空间

若 OVLY=0， 则片内 RAM 只安排到数据存储空间

DROM 位

当 DROM=1，则部分片内 ROM 安排到数据空间

当 DROM=0，则片内 ROM 不安排到数据空间

DROM 如何用法与 MP/MC 的用法无关，程序和数据空间映射和扩展程序空间映射可见其芯片手册。

根据以上介绍，由于本系统采用外部引导方式，故 MP/MC=0，DSP 工作在微控制器模式下。在本系统中，因为系统汽车行驶记录仪器程序存放在外部 FLASH 中，由 3.1.2 节中，上电后，进入片内 ROM 的 BootLoader 后，DROM 被初始化为 0，此时片内 BootLoader 从外部数据空间加载程序，加载完成以后将 OVLY 置 0。

程序数据空间、数据空间、I/O 空间各地址分布分别如表 3-1、表 3-2、表 3-3 所示。

表 3-1 程序空间地址范围

页数	地址范围	片内/外	属性与功能
第 0 页	000000H~00BFFFH	片外	存放汽车行驶记录仪采集到的数据
	00C000H~00FEFFH	片内	固化 ROM，详细情况可见芯片手册
	00FF80H~00FFFFH	片内	中断向量表
第 1 页	010000H~017FFFH	片外	存放汽车行驶记录仪采集到的数据
	018000H~001FFFH	片内	DARAM4~7，运行记录仪程序
第 2 页	020000H~027FFFH	片外	存放汽车行驶记录仪采集到的数据
	028000H~002FFFH	片内	DARAM0~3，运行记录仪程序
第 3 页	030000H~037FFFH	片外	存放汽车行驶记录仪采集到的数据
	038000H~03FFFFH	片内	片内 DARAM4~7，运行记录仪程序
第 4 页	040000H~04FFFFH	片外	存放汽车行驶记录仪采集到的数据
.....			
第 64 页	400000H~40FFFFH	片外	存放汽车行驶记录仪采集到的数据

表 3-2 数据空间地址范围

地址范围	片内/外	属性与功能
0000H~005FH	片内	寄存器空间
0060H~007FH	片内	暂存 RAM
0080H~7FFFH	片内	DARAM0-3
8000H~0FFFFH	片外	外部 FLASH，用于存放汽车行驶记录仪程序

表 3-3 I/O 空间地址范围

地址范围	属性与功能
0000H~1FFFH	数据采集模块访问地址，实际上只有 A15 A14 A13 用来片选
2000H~3FFFH	USB 主机模块访问地址，实际上只有 A15 A14 A13 A0 有意义，A0=1 时表示访问数据端口，A0=0 表示访问地址端口，A15 A14 A13 用来片选
4000H~5FFFH	CAN 通信模块访问地址，实际上只有 A15 A14 A13 A0 有意义，A0=1 时表示访问数据端口，A0=0 表示访问地址端口，A15 A14 A13 用来片选

6000H~7FFFH	实时时钟模块访问地址，实际上只有 A15 A14 A13 A0 有意义，A0=1 时表示访问数据端口，A0=0 表示访问地址端口，A15 A14 A13 用来片选
8000H~9FFFH	显示模块访问地址，实际上只有 A15 A14 A13 A0 有意义，A0=0 时表示访问数据端口，A0=1 表示访问指令端口，A15 A14 A13 用来片选
0A000H~0BFFFH	键盘接口模块访问地址，实际上只有 A15 A14 A13 用来片选

3.1.4 DSP 的命令文件编写

为了使用户在编写 C 程序时，无需考虑目标硬件的具体情况，语言采用了在链接时配置存储空间的方法。用户在 CCS 集成开发环境中编写 C 程序时，C 语言程序设计无需考虑硬件的存储空间配置状况，只需单独编写一个.CMD 命令文件[37]，指明目标硬件的存储空间配置情况，链接器就可以自动的将数据和程序装载到指定的位置空间。.CMD 文件的总体框架如下：

```
MEMORY
{
PAGE 0: Mname 0 [(attrib)]:origin=constant, length=constant;
PAGE 1: Mname 1 [(attrib)]:origin=constant, length=constant;
}
SECTIONS
{
Sname: [property, property, property,...]
Sname: [property, property, property,...]
}
```

其中："MEMORY" 是关键字，它的内容是对目标硬件分配可用的储存空间段。"MEMORY" 伪指令在命令文件中的书写方式为：以大写"MEMORY"关键字开始，后面由大括号括起来的一系列对存储器区间说明，每个存储器区间具有名称、起始地址和分配的存储器长度。一般来说包括两页，PAGE 0 和 PAGE 1。PAGE 0 指示分配程序空间，PAGE 1 指示分配数据空间。"Mname" 是一个存储空间段的名称，每个名称可由 1-8 个字符组成，合法的字符包括 "A~Z"，"a~z"， "$"和"_"。每个名称对链接器来说没有特殊的意义，仅仅代表一个存储器的可用段。不同页的可用存储器段可以有相同的名称，但同一页中各段的名

称必须互不相同。"Attrib"指定 1-4 个属性，合法的属性包括："R"表明存储器可读；"W" 表明存储器可写；"X"表明存储器包含可执行代码；"L"表明存储器可以被初始化。属性是可选的，所有没有指明属性的存储器同时包括"R，W，X，L"四种属性。

"Origin" 指示可用存储器段的起始地址，可用 "origin"，"org"或"o"表示；起始地址是 16 位的常数。"Length"指示可用存储器段的长度，用"length"，"len"或"l" 表示，其值是 16 位常数。

"SECTIONS" 是关键字，它的内容是指示代码和数据将如何装载到在 "MEMORY" 中定义的各个存储器可用段。"SECTIONS" 伪指令在命令文件中的书写方式为：以大写 "SECTIONS" 开始，后面为大括号，括号内为一系列输出段的说明语句。其功能有：说明如何将输入的段结合成输出段；在可执行程序中定义输出段；指定输出段放置在存储器中的位置（包括各段之间的相互关系及在整个存储器中的位置）；允许输出段重新命名。

"Sname" 表示各段代码/数据的名称。"property" 用来定义本段代码/数据的内容和分配情况。C 编译器对 C 程序编译后会自动生成若干个可以进行重定位的代码和数据段[38]。这些段分为两种类型，一种是已初始化段，另一种是未初始化段。C 编译器共生成 4 个已初始化段如下：

- .text 段：包含可执行代码和字符串；
- .cinit 段：包含 C 环境初始化变量和常数表；
- .const 段：包含字符串，在大存储模式下，常数表也包含在本段中；
- .swith 段：包含为 swith 语句建立的表格。

未初始化段用于保留存储空间，程序利用这些空间在运行时创建和存储变量。C 编译器创建 3 个未初始化段如下：

- .bss 段：保留全局和静态变量空间。在程序开始运行时，C 语言初始化程序将数据从 cinit 段拷贝到.bss 段。
- .stack 段：系统堆栈段。
- .system 段：系统段，为动态存储函数 malloc， calloc 和 realloc 分配存储空间。若程序中没有用到这些函数，编译器就不创建.system 段。

下面给出本系统程序设计.CMD 文件的具体例子：

```
MEMORY
{
```

```
PAGE 0 :        /*程序空间块的划分*/
  VECTORS:    origin = 0x028000, length = 0x80   /* interrupt vector table
*/
  PRO:        origin = 0x028080, length = 0x80
  PROG:       origin = 0x028100, length = 0x7E00    /* code */

PAGE 1 :        /*数据空间块的划分*/
  DARAM:      origin = 0x0080, length = 0x7700    /* data buffer */
  STACK:      origin = 0x7780, length = 0x0800    /* stack */
}

SECTIONS
{
  .text          : load = PROG       page 0
  .cinit         : load = PROG       page 0
  .switch        : load = PRO        page 0
  .const         : load = DARAM      page 1
  .bss           : load = DARAM      page 1
  .stack         : load = STACK      page 1
  .coeff         : load = DARAM      page 1
  .vectors       : >VECTORS          page 0   /*interrupt vector table */
}
```

3.2 汽车行驶记录仪部分模块程序设计

3.2.1 系统主循环程序设计

汽车行驶记录仪主程序流程图如图 3-2 所示，在系统上电后，首先通过 IC 卡模块确认驾驶员信息，然后对 DSP 内部各寄存器进行设置，完成自检功能，自检不通过则输出错误信息。通过则查询键盘模块是否有按键，根据不同的键值运行相应的程序。无键盘按下则依次轮循环相应模块，虽然 CAN 模块和 USB 模块都是基于中断的，但是由于在同一时刻只有一个模块处于片选状态，只有处于片选状态下，其中断信号引脚才能输出有效，所以要依次轮询，而数据采集模块其中断不受片选信号的影响所以这里没有参与轮询过程。

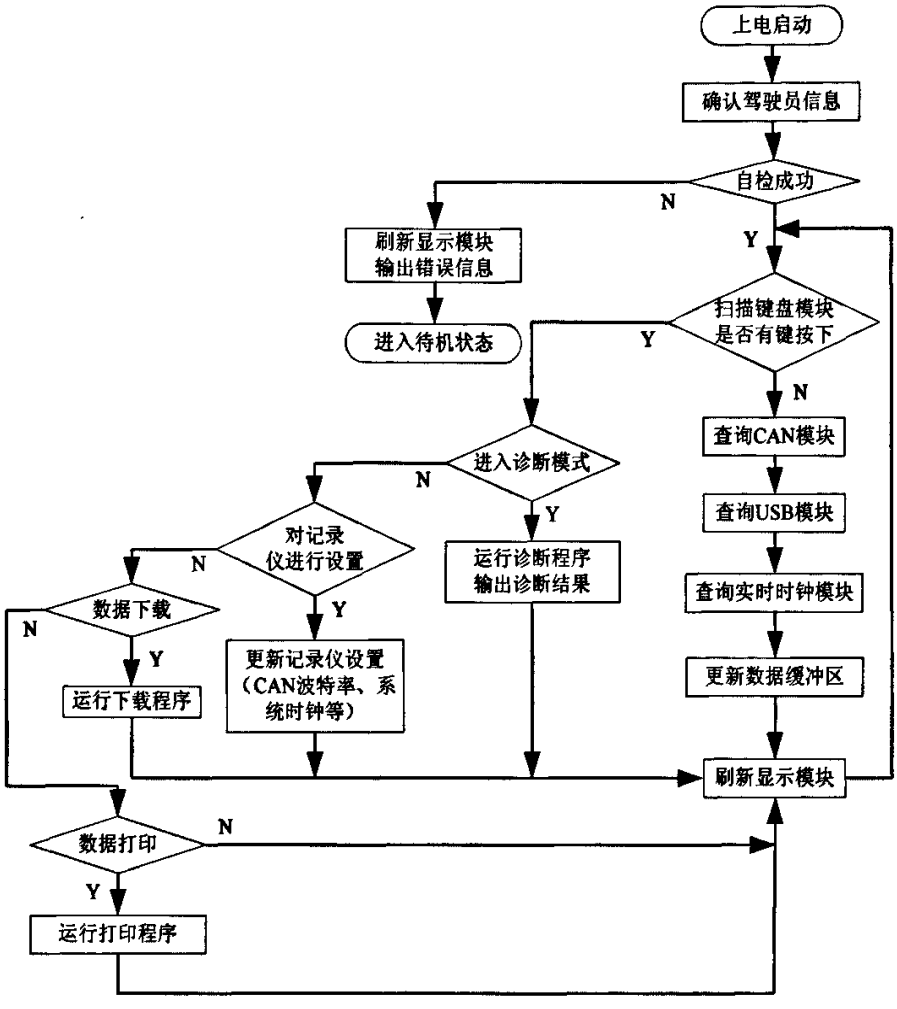


图 3-2 系统主程序流程图

3.2.2 数据采集模块程序设计

数据采集模块程序流程图如图 3-3 所示。数据采集模块硬件连接图如 2.3.1 节中所述，采集模块通过中断引脚通知 DSP 新采集的数据到来。DSP 访问数据采集模块采用端口访问方式即可完成。其中数据采集模块端口地址为 0000H，它映射在 I/O 空间，故可使用下列语句完成对端口状态数据的读取：

```
ioport unsigned port0000;
unsigned int velocity;
```

velocity=port0000;

将所得数据高 8 位为开关量，低 8 位为 0.2s 里所测得脉冲数 f，代入式(2-5)即：

$$V = 5 \times 2\pi R \times \frac{f}{n}$$

其中 R 为汽车车轮半径，n 为车轮旋转一周输出脉冲数，这里取 3，所的结果 V 即为所测速度。将所得结果存入 FLASH 中，同时更新数据缓冲区并刷新显示模块。这里的数据缓冲区即为图 3-2 中的所列数据缓冲区，它为各种模块的信息交流提供了一个公共的渠道。

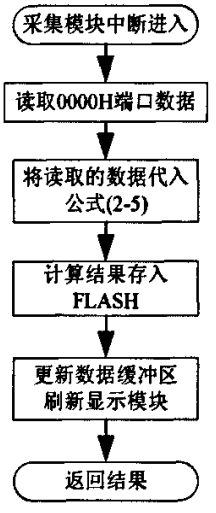


图 3-3 数据采集模块程序流程图

3.2.3 数据存储模块程序设计

在使用 FLASH 存储器时，用户通过向特定地址中写入特定的指令序列，通过这些命令用户即可启动内部写状态机，从而实现 FLASH 自动完成指令序列要求的内部操作，包括：复位、整片擦除、块擦除、扇区擦除、操作字写入等。其整个编程过程由用户完成，所以必须了解 FLASH 的状态来确定其操作是否完成。这里我们通过检测 DQ6（数据线的第 6 位）来检测内部操作。SST39VF010 或 SST39VF1601 在进行内部编程或擦除时，对任何地址进行连续

读取都会引起 DQ6 跳变，当操作停止时跳变结束。因此可通过连续两次读取检测 DQ6 的变换情况来判断编程或擦除操作是否完成。对与每一个数据的写入操作都要经过如图 3-4 所示的程序流程。另外在写 FLASH 时应先对其进行擦除，表 3-4 列出了擦除指令序列。

TMS320VC5416 在加电后是以外部数据空间来寻址 SST39VF010 的，只能寻址 8000~0FFFFH 外部地址范围的数据。在 OVLY＝1 的情况下，地址 5555H 和 2AAAH 是位于 DSP 内部，在外部是不可见的，因此对于写 0AAH 到 5555H 和写 55H 到 2AAAH 的操作，其实根本未对 SST39VF010 进行操作。但通过分析我们发现 5555H 和 2AAAH 中真正起作用的是 A0~A14 位，A15 位可以是任意值，于是考虑加一个偏移量 8000H。则此时 5555H 变为 D555H，而 2AAAH 变为 0AAAAH 在外部可见，这样就可以通过 DSP 对外部数据空间的 SST39VF010 进行操作。

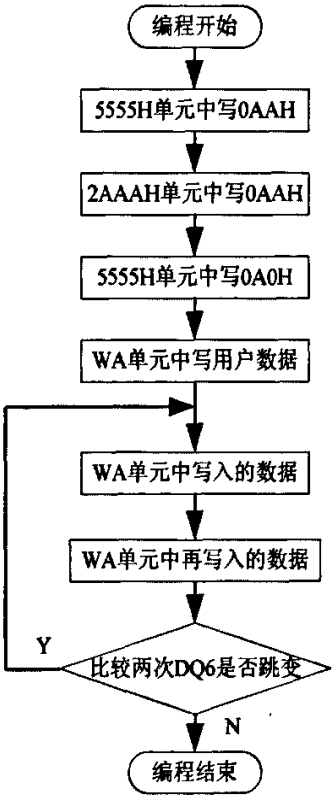


图 3-4 数据的写入操作

表 3-4 擦除指令序列

指令字	第 1 个总线写周期		第 2 个总线写周期		第 3 个总线写周期		第 4 个总线写周期	
	地址	数据	地址	数据	地址	数据	地址	数据
编程	5555H	AAH	2AAAH	55H	5555H	A0H	WAH	DATA
扇区擦除	5555H	AAH	2AAAH	55H	5555H	80H	5555H	AAH
整块擦除	5555H	AAH	2AAAH	55H	5555H	80H	5555H	AAH
整片擦除	5555H	AAH	2AAAH	55H	5555H	80H	5555H	AAH
指令字	第 5 个总线写周期		第 6 个总线写周期					
	地址	数据	地址	数据				
编程								
扇区擦除	2AAAH	55H	SAX	30H				
整块擦除	2AAAH	55H	BAX	50H				
整片擦除	2AAAH	55H	5555H	10H				

其中：

(1)WA 表示可编程字的地址。

(2)SAx 表示要擦除第 x 个扇区。

(3)BAx 表示要擦除第 x 块。

系统的另一块 FLASH SST39VF1601 用来存放所采集到的汽车状态数据，它映射在外部程序空间，而外部程序空间的读写只能使用 READA 和 WRITA 指令。指令格式如下：

READA Smem

WRITA Smem

READA 指令把累加器 A 所确定的程序存储器单元中的一个字，传送到数据存储器单元 Smem 中。

WRITA 指令把数据单元 Smem 中的一个字，传送到累加器 A 确定的程序存储器单元。

采集到的汽车数据存储到 SST39VF1601 中，必须首先对 SST39VF1601 的

内部存储结构进行规划，同时要规定所存储数据格式。SST39VF1601 存储区中主要分为两个存储模块。

第一个为驾驶员信息存储区，用于用户身份认证，其结构如图 3-7 所示，其起始地址为 PAGE 0 中的 0000H 单元，结束地址为 PAGE 0 中的 0FFFH，所以用于存储驾驶员信息的空间长度为 1000H 个字，这里身份 ID 和密码长度皆为 16 位，所以系统可以存储的用户信息总数为 2048 。

另一个为汽车状态信息存储区，用户存储汽车状态信息，其结构如图 3-8 所示。记录仪数据存储区的开始地址为 1000H，这样只需知道记录数即可从起始地址遍历所有记录。根据表 3-1 所列地址列表，可知对于数据存储的区域，地址处于分块不连续状态，但是所有数据的存储都是按照顺序进行，所以并不影响数据的存储和读取。

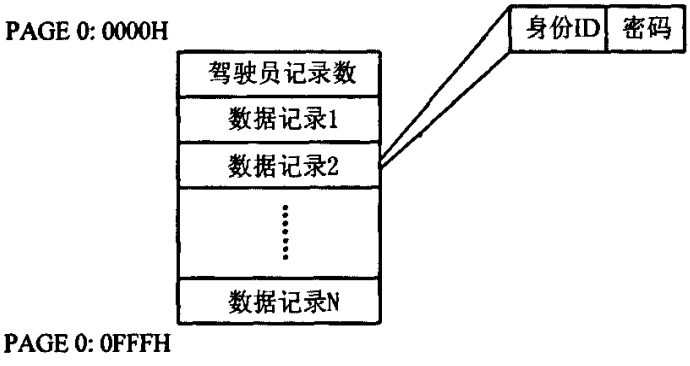


图 3-7 驾驶员信息存储区

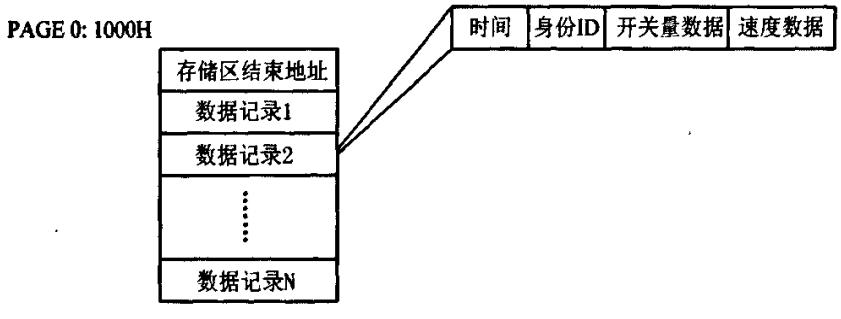
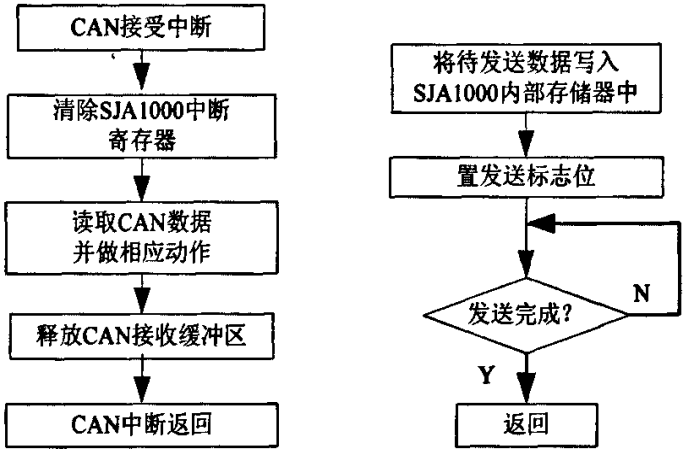


图 3-8 记录仪信息存储区

3.2.4 CAN 通信模块程序设计

CAN 通信模块的程序设计是基于中断的，根据第 2.4.2 节中 CAN 连接图，CAN 模块地址映射在 I/O 空间，其数据端口地址为 4000H，地址端口地址为 4001H。系统使用 CAN 总线接入汽车后，可以方便记录仪与汽车内部控制器件的通信，完成基本的发送和接受功能，同时由于可以实时监测 CAN 总线数据可以完成对 CAN 总线故障诊断的功能。

CAN 模块流程图如图 3-9 所示，进入 CAN 中断后，首先清除 SJA1000 中断寄存器，读取 CAN 数据并做相应动作。接着释放 CAN 接受缓冲区，并中断返回，这是一个最基本的 CAN 接受流程图。规定了 CAN 物理层的基本动作，对于高层协议可以在 CAN 数据包中进行规定。对于 SJA1000 内部寄存器的状况可参见相关芯片手册。



(a) CAN接收流程　　(b) CAN发送流程

图 3-9 CAN 模块流程图

3.2.5 实时时钟模块程序设计

实时时钟模块映射在外部 I/O 空间，其地址为 6000H~7FFFH。对于地址线而言，实际上只有 A15 A14 A13 A0 有意义，A0=1 时表示访问数据端口，A0=0 表示访问地址端口，A15 A14 A13 用来片选。

由于 DS12CR887 片内地址空间为 00H~7FH，其中 00H 为秒单元，01H 为闹秒单元，02H 为分钟单元，03H 为闹分单元，04H 为时单元，05H 为闹时单元，06H 为星期单元，07H 为日单元，08H 为月单元,09H 为年单元，0AH~0DH 单元分别为控制寄存器 A、B、C、D。0EH~7FH 为用户 RAM 区，可用来在系统掉电时保存数据。通过访问 A、B、C、D 四个寄存器，可随时设置和了解 DS12CR887 的工作方式。

```
ioport unsigned char port6001;
ioport unsigned char port6000;
unsigned int temp
```

读取一个字节程序如下：

```
port6000=0x00; //读取单元地址
asm("nop");
asm("nop");
asm("nop");
temp=port6001;
```

写入一个字节程序如下:

```
port6000=0x00;  //写入单元地址
asm("nop");
asm("nop");
port6001=0x20;//写入的数据
```

对 DS12CR887 的写入与读取与外部其他 I/O 设备的操作一样，在主循环里，要不停的采集时钟数据来刷新当前状态，同时在储存所采集数据时也必须读取当前时钟，由于该时钟芯片具有电池切换功能，可以很好的满足系统要求。

3.2.6 液晶显示模块程序设计

通过初始化液晶显示屏以及调用自建的汉字库显示连续的汉字，来具体说明 DSP 控制液晶显示屏的程序流程。如表 3-3 中所列，控制器指令的写入端口地址是 8001H，写参数及显示数据端口地址为 8000H，定义形式如下：

```
ioport unsigned char port8001;  /*指令通道地址*/
ioport unsigned char port8000;  /*数据通道地址*/
#define LCD_CODE_ADDR   port8001   /*指令通道地址*/
```

#define LCD_DATA_ADDR port8000 /*数据通道地址*/

对液晶操作时，首先要根据用户系统的需要对控制器的各项指令代码及其参数进行设置，以完成液晶模块的参数(如液晶的行数、列数、扫描频率及光标的位置等)以及显示方式等一系列过程的初始化。在 DSP 操作 T6963C 控制的液晶显示模块时，必须首先写入 SYSTEM SET 40H 指令。如果该指令设置出现错误，则显示必定不正常。

文本显示区域首地址和宽度、图形显示区域首地址和宽度、光标形状、显示方式和显示开关等液晶的初始化设置完成后，通过写指令和写数据子程序完成液晶的汉字图形显示，中间需要调用自建的汉字库和图形库。对液晶模块的写指令、写数据和读数据的流程如图 3-10 所示。

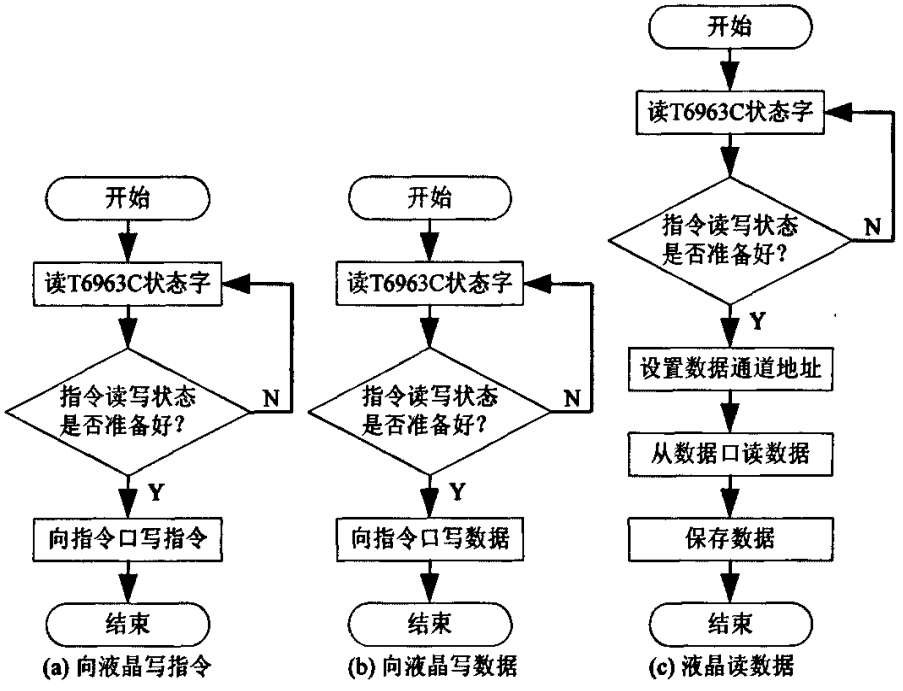


图 3-10 液晶模块读写程序

3.2.7 数据打印模块程序设计

SP 系列打印机提供了 36 条打印命令。这些命令规定了打印机的功能，如选择字符类别和字符集、定义格式、放大或缩小字符、打印汉字、打印点阵图形和定义用户可定义字符等。

打印命令是由一字节控制码或 ESC 控制码序列组成。字节控制码用十进制或十六进制数字序列表示，ESC 控制码是以“ESC”码开头，后跟其它字符码。

SP 系列打印机打印控制码是在参考 IBM 和 EPSON 打印机的基础上设计的。因此，它能够和大多数打印机兼容。下面介绍几个在打印程序设计中常用的命令。

(1) 汉字打印命令

SP 系列打印机自带国标一、二级硬汉字库(12×12 点阵或 16×16 点阵)，可打印汉字库中全部汉字。16×16 点阵字库还可选择打印 16×16, 8×16, 16×8, 8×8 点阵汉字，不同点阵汉字可同行打印，汉字与 ASCII 字符可以同行混合打印。汉字打印使用双字节标准机内码（或国标码）调用。汉字打印命令包括进入和退出汉字打印、执行和取消汉字倍宽打印和设置汉字点阵打印规格等功能，详细命令字可见产品手册。

(2) 选择字符集命令

SP 系列打印机除上述汉字打印外，还提供了字符集 1 及 2 的字符进行打印。字符集 1 中有 6X8 点阵字符 224 个，包括 ASCII 字符及一些图形及字符。字符集 2 中也有 6X8 点阵字符 224 个，包括希腊、德、法、俄、日文片假名及部分图形。详细命令字可见产品手册。

(3) 设置走纸命令

设置走纸命令包括换行、n 点行走纸、换页等内容。详细命令字可见产品手册。

根据以上的命令，在停车的状态下通过串口对打印机发送相应数据，即可实现相应的功能。设计数据打印的相关函数如下;

```
void printer_init(void);          //打印机初始化
void print_chi(void);             //打印中文字符
void exit_chi(void);              //退出打印中文字符
void print_space(void);           //打印空格
```

```
void print_space6(void);          //打印六个空格
void change_line(void);           //换行
void print_driv_num(void);        //打印驾驶证号
void print_fatigue_data(void);    //打印疲劳驾驶数据
void seri_send(unsigned int);     //通过串口向打印机发送数据
```

3.2.8 CAN 故障诊断程序设计

利用现有平台，可以完成对汽车 CAN 总线故障的诊断，使记录仪工作在诊断模式，当 CAN 总线上的数据被采集到记录仪后，就可以进行故障诊断了，故障诊断代码是依照 KWP2000 应用层规定的故障代码设计的，是目前国际上通用的，现在将其应用于 CAN 的应用层，将来可以用全新的 CAN 上层协议取代。故障诊断代码定义了在 SSF 14230。SAE J1979 中被车辆制造商或系统供应者定义的服务标志符数值的不同范围如表 3-5 所示[39]。

表 3-5 SAE J1979 中的服务标志符数值的不同范围

服务标志符 16 进制数	服务种类（bit 6）	定义来源
00-0F	请求	SAE J1979
10-1F	请求（bit 6=0）	SSF 14230-3
20-2F		
30-3E		
3F	不做应用	文档保留
40-4F	响应	SAE J1979
50-5F	对服务（$10-$3E）的肯定响应（bit 6=1）	SSF 14230-3
60-6F		
70-7E		
7F	否定响应	SSF 14230-3
80	请求"ESC"-字码	ISO 14230-3
81-8F	请求（bit 6=0）	SSF 14230-2
90-9F	请求（bit 6=0）	为将来扩展预留
A0-B9	请求（bit 6=0）	由车辆制造商定义

BA-BF	请求（bit 6=0）	由系统供应者定义
C0	肯定响应"ESC"-字码	ISO 14230-3
C1-CF	肯定响应（bit 6=1）	SSF 14230-2
D0-DF	肯定响应（bit 6=1）	为将来扩展预留
E0-F9	肯定响应（bit 6=1）	由车辆制造商重定义
FA-FF	肯定响应（bit 6=1）	由系统供应者重定义

此表中以 16 进制数表示的服务标志符，同数据链路层中数据字节内的 SID 服务识别字节对应。不同的 SID 值代表不同的服务请求，故障诊断程序必须要符合此应用层标准，才能识别不同的 16 进制代码所代表的不同的故障信息。将故障信息输出到显示模块，即可以实现汽车内部 CAN 总线的诊断功能[40]。

3.3 USB 主机程序设计

关于 USB1.1 通信协议这里不作详细介绍，可以从 www.usb.org 上下载到相应的手册和资料。USB 分为主机和设备两方面的开发。在本系统中，SL811HS 这里作为主机控制器来使用。主机是 USB 的核心，每一次数据通信都必须是由 USB 主机发起的，主机管理着每个 USB 设备[40]。USB 主机结构包括以下三个层次：

(1) 总线接口层

总线接口层主要是指以主机控制器为核心的硬件部分，包括 USB 主机控制器芯片、USB Hub 控制器芯片、串行接口引擎、USB 端口连接件及控制器外围电路等。总线接口层负责处理电气和协议层的互连，提供了 USB 数据收发的物理层。

(2) USB 主机系统软件层

USB 主机系统软件层是 USB 主机用来管理 USB 系统资源的核心软件，与主机控制器一起，实现了客户数据和 USB 总线事务的转换传输。它包括了两部分软件，即 USB 核心驱动程序(USBD)和 USB 主机控制器驱动程序(HCD)。USBD 是 USB 是主机软件体系的核心部分，解释 USB 设备类驱动程序发来的命令并将其划分为一系列的 USB 事务，然后发送给 USB 主机控制器驱动程序，起到一个中间桥梁的作用。HCD 是主机控制器硬件的抽象，负责最底层的驱动

任务，控制和管理着底层主机控制器硬件，负责将 USB 事务发送给 USB 主控制器芯片，并最终将串行数据发送给 USB 设备。

(3) USB 客户层

客户层描述了直接与 USB 设备进行交互时所需的软件包，包括用户软件和 USB 设备类驱动程序。用户软件是指用户与 USB 系统之间的一种接口，针对特定的应用场合完成用户对 USB 设备的控制，以及实时的进行一些数据交互。USB 设备类驱动程序是实现特定 USB 设备功能的软件。每一种 USB 类设备都需要设计相应的设备类驱动程序。比如：移动硬盘、U 盘属于 Mass Storage 类，键盘、鼠标属于人机接口设备类[41]。

总的来说，USB 主机为 USB 系统提供了以下 5 种功能：

(1) 检测 USB 设备的安装和拆卸；

(2) 管理主机和 USB 设备之间的控制流；

(3) 管理主机和 USB 设备之间的数据流；

(4) 收集状态和动作信息；

(5) 控制主机控制器与 USB 设备的电气接口，包括提供能源。

记录仪写 U 盘的功能可分为 5 个层次。最底层是直接与硬件相关，用来实现对 SL811HS 芯片的读写，为主机控制器驱动层。在主机控制器驱动层的上层是用来实现 USB 协议的，包含多个子程序，称为 USB 驱动层。在 USB 协议之上是海量存储设备类协议，该层提供了直接访问 U 盘的函数。再上层为 FAT 文件系统层，创建文件、写文件等函数都是属于该层。顶层是用户应用层，用户根据 FAT 协议层提供的函数来实现写 U 盘功能。

3.3.1 主机控制器驱动(HCD)设计

HCD 是 USB 协议栈的最底层代码，是 TMS320VC5416 与 SL811HS 的通信接口，通过数据线、读写控制线、片选线完成对 SL811HS 主机芯片的 I/O 读写等操作，可为上层驱动提供对 SL811HS 的访问接口。其实质上是对 USB 主机控制器硬件和数据传输的抽象，受核心驱动程序(USBD)的调用和管理。主要实现了对主机控制器芯片寄存器的读写功能[42]。

根据对 USB 主控制器 SL811HS 的介绍，SL811HS 片内 RAM 的访问可以用两个函数实现：Wr811hsBuf(u16 address,u16 * buffer,u16 length)和 Rd811hsBuf(u16 address,u16 * buffer,u16 length)，其中的参数分别为 SL811HS 的

RAM 地址、DSP 数据单元指针和访问字节数，以下给出它们的源代码。

```
void Wr811hsBuf(u16 address,u16 * buffer,u16 length)
{
Port2000=address;//选择 SL811HS 要写的第 1 个 RAM 单元
While(length--) //将 DSP 缓冲区的数据逐一写入 SL811HS
Port2001=*buffer++;//每读 1 次，SL811HS 自动指向下一个 RAM 单元
}

void Rd811hsBuf(u16 address,u16 * buffer,u16 length)
{
Port2000=address;//选择 SL811HS 要读的第 1 个 RAM 单元
While(length--) //将 SL811HS 数据的数据逐一读出放到 DSP 缓冲区
*buffer++=port2001;//每读 1 次，SL811HS 自动指向下一个 RAM 单元
}
```

3.3.2 USB 总线驱动(USBD)设计

USBD 是 USB 主机系统的核心，负责检测和管理 USB 系统的所有活动，向上接收 USB 设备驱动程序和用户程序的各种请求命令和数据，向下把处理好的数据发送给 HCD，并最终完成与设备的通信。针对批量传输(Bulk Only)子类，一般具有 3 个端点，涉及两种传输类型：一个是端点 0，用于处理控制传输；一个是批量输出(Bulk Out)端点，用于向设备发送批量数据，另一个是批量输入(Bulk In)端点，用于接收设备发送来的批量数据。USB 的各种传输类型都是由事务组成的，所以要实现这两种传输，就必须先完成相应事务的程序设计[43]。

(1) USB 事务的实现

在数据事务处理过程中，程序将要发送的数据写入 SL811HS 缓冲存储区，然后开始发送，接着等待 USB 设备的响应，程序根据响应的结果采取相应的动作。该过程实现了 USB 通信协议的主要部分，令牌包，数据包和握手包在该过程中都得到了体现，只有在全面而又准确地了解 USB 通信协议的基础上才能用程序实现该过程。设备枚举、批量传输、控制传输都是基于这个过程的。该过程的函数名是 usbXfer()，通过一个 usbXfer()函数来实现 USB 的 IN、OUT、SETUP

三种事务。函数形式为 usbXfer(uchar usbaddr, uchar endpoint, uchar pid, uchar iso, uchar wPayload, ucharwLen, uchar * buffer)；此函数包含 7 个输入参数:usbaddr 设备地址；endpoint 端点号；pid 事务传输类型标识；iso 同步数据传输标识；wPayloadUSB 数据包的最大尺寸；wLen 事务输入的数据长度；buffer 事务的收发数据缓冲区。传输成功返回 TRUE，传输失败返回 FALSE。

USB 主机与设备的事务交换过程完全是由 SL811HS 的硬件完成的，我们所需要实现的是对各种事务的控制。实现流程如下：

① 根据事务标识符 pid 和端点 endpoint 确定状态寄存器，以控制主机发起的事务类型和传输通道。

② 根据事务标识符 pid、传输方向、同步标识确定控制寄存器的命令字，以便控制 SL811HS 启动不同的事务。

③ 需要根据传输数据长度 wLen 确定是否分批发送数据包，若 OUT，IN 事务的总数据长度超过数据包的最大长度 wPayload，需要分批传送。分批传输时，必须不断切换数据包的标识，以实现乒乓机制；而 SETUP 事务的数据包是一个固定的 8 字节的命令信息，所以只需要一个数据包就够了。

④ 启动数据交换事务过程，进入查询等待状态，这时候，由 SL811HS 自动与设备进行事务交换。

⑤ 当事务交换完成，SL811HS 产生中断，通过读取状态寄存器判断事务返回握手包类型，如果握手信息为确认 ACK，则说明事务交换成功，对 IN 事务来说，就可以把设备传送上来的数据 SL811HS 的接收缓冲区中读出来；对于 OUT， SETUP 来说，主机所发送的数据已经被设备接收，如果握手信息为 STALL，说明设备端点被禁止，如果握手信息为 NAK，说明设备忙，无法进行事务交换。流程图如图 3-11 所示。

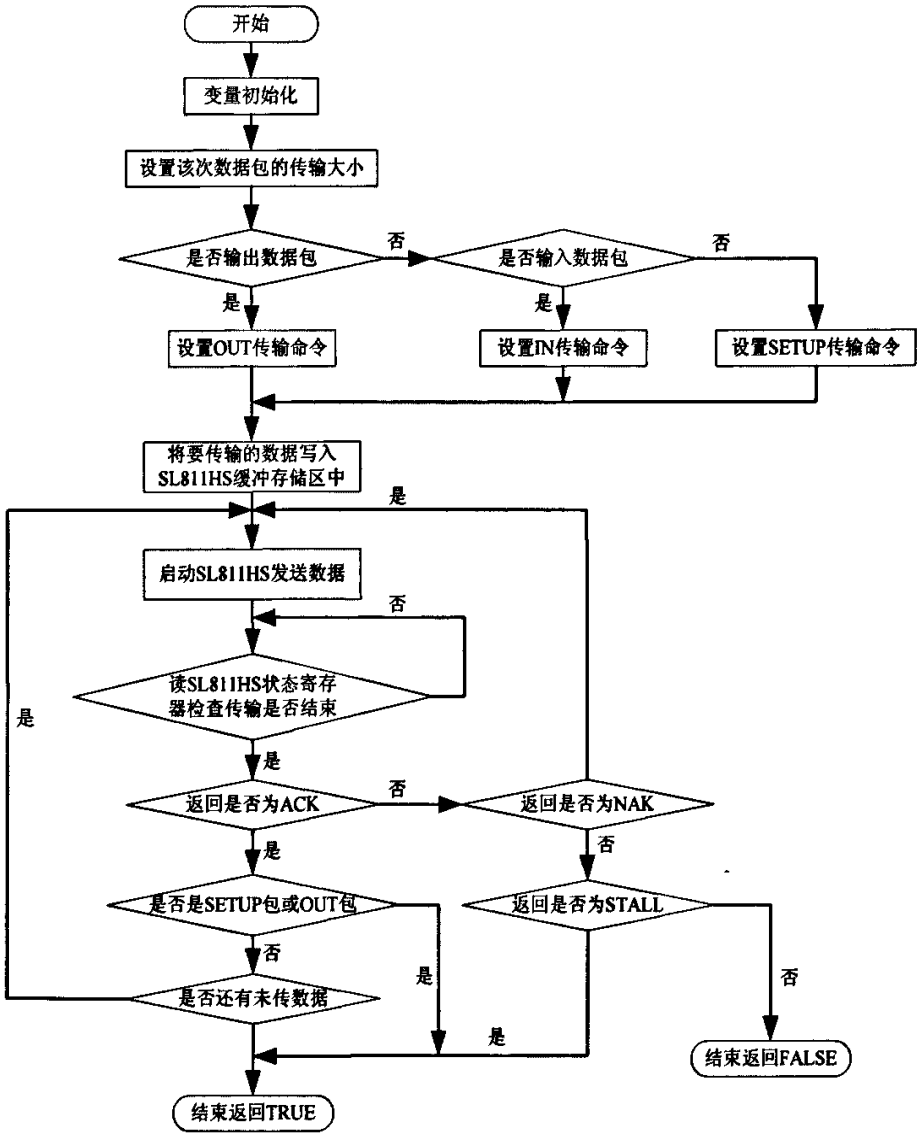


图 3-11 USB 事务交换流程图

(2) 控制传输的实现

控制传输是最为复杂的传输类型，也是最为重要的传输类型，是 USB 枚举阶段最主要的数据交换方式。当 USB 设备初次连接到主机之后，就通过控制传输来交换信息、设置地址和读取设备的描述符。控制传输是通过访问专用的控

制端点 0 来实现的，只要 USB 设备连接到主机上，控制端点就可以被访问。控制传输一般分为初始设置、可选数据传输、状态返回三个步骤：初始设置步骤把标准请求命令通过 setup 事务发送给设备，数据传输步骤利用 IN 或 OUT 事务传送命令请求中要发送的数据，状态返回步骤将设备执行命令的结果返回给主机。

(3) U 盘枚举的实现

U 盘枚举过程实际上为主机通过控制传输执行的一系列标准 USB 设备请求命令，首先通过端点 0 获取描述符命令(GET DESCRIPTOR)获得设备的基本配置信息；然后通过设置地址(SET ADDRESS)、设置配置(SET CONFIGURATION)等命令来达到对设备的动态识别、配置和访问。标准 USB 设备请求命令共有 11 个，大小都是 8 个字节，具有相同的数据格式。USB 枚举函数名是 EnumUsbDev()，枚举成功返回 TRUE，枚举失败返回 FALSE[44]。流程图如图 3-12。

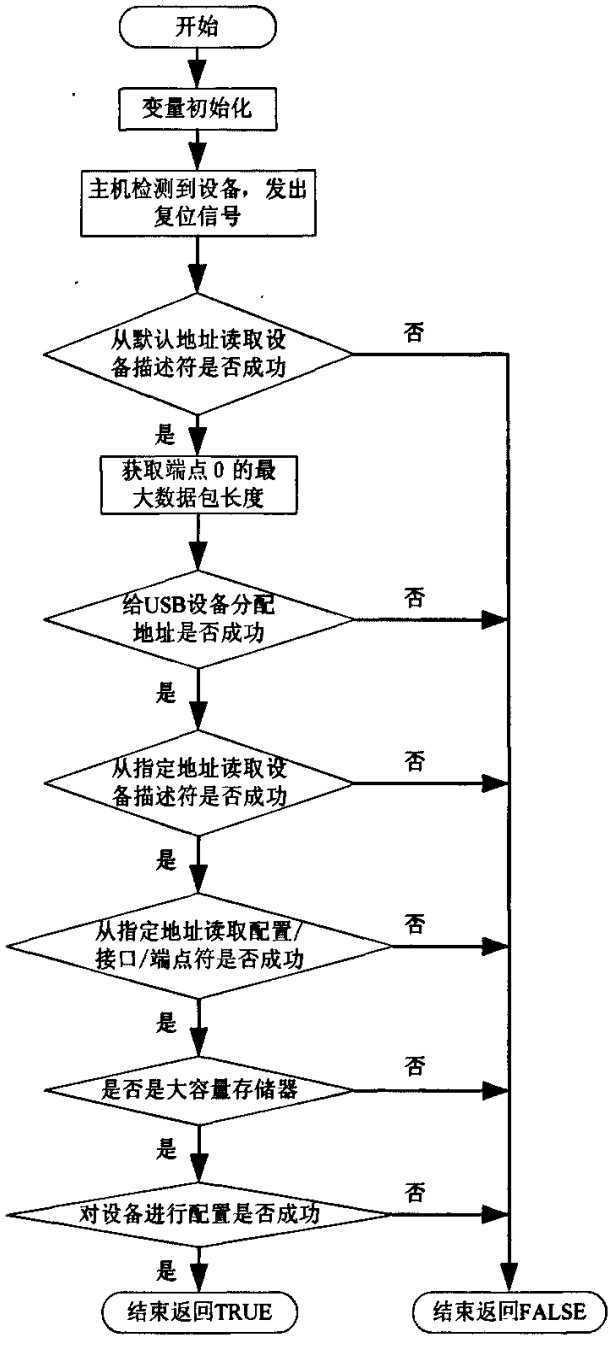

开始
变量初始化
主机检测到设备，发出复位信号
从默认地址读取设备描述符是否成功
否
是
获取端点0的最大数据包长度
给USB设备分配地址是否成功
否
是
从指定地址读取设备描述符是否成功
否
是
从指定地址读取配置/接口/端点符是否成功
否
是
是否是大容量存储器
否
是
对设备进行配置是否成功
否
是
结束返回TRUE
结束返回FALSE

图 3-12 U 盘枚举过程

(4) 批量传输的实现

对于基于 Bulk Only 传输模式的 U 盘，其默认配置只含一个 Bulk Only 数据接口，此接口使用三个端点：端点 0，Bulk-In 和 Bulk-Out。枚举结束后，主机识别出为 Bulk-Only 的 Mass Storage 设备，然后即进入 Bulk-Only 的传输方式。在此方式下，USB 与设备间的所有数据均通过 Bulk-In 和 Bulk-Out 端点来进行批量传输，不再通过控制端点 0 传输任何数据。

批量传输主要由 IN 和 OUT 事务组成，通过调用事务函数 usbXfer()实现，批量发送和批量接收的实现函数如下：

① unsigned char epBulkSend(unsigned char * pBuffer,unsigned int len)

往设备进行批量发送。发送长度由 len 指出，该长度就是设备批量传输端点的最大数据包的长度，pBuffer 指向要发送的字符串的首地址单元。但是究竟发送到 U 盘的哪个端点上，则是由 U 盘决定。在 U 盘枚举的过程中，U 盘告诉记录仪哪个端点是它的批量接收端点。

```
unsigned char epBulkSend(unsigned char *pBuffer,unsigned int len)
{
usbstack.usbaddr=USB_ADDR;//设备的地址
usbstack.endpoint=usbstack.epbulkout;//设备批量接收端点号
usbstack.pid=PID_OUT;//令牌包类型
usbstack.wPayload=64;//数据包的最大长度
usbstack.wLen=len;//一共要传输的数据
usbstack.buffer=pBuffer;指向要发送数据在 RAM 中首地址
while (len>0)
    {
    //如果有数据需要发送
    if (len>usbstack.wPayload)//确定每次事务传输的数据量
        usbstack.wLen=usbstack.wPayload:
    else
        usbstack.wLen=len:
    if (!usbXfer () )//进行数据传输
        return FALSE:
    len-=usbstack.wLen;还剩多少数据没有发送
```

```
        usbstack.buffer=usbstack.buffer+usbstack.wLen;//指向要发送的数据
        }
        return TRUE;
```

② unsigned char epBulkRcv(unsigned char *pBuffer,unsigned int len)

从设备中进行批量接收。接收长度由 len 指出，该长度就是设备批量传输端点的最大数据包的长度，pBuffer 指向存放接收数据的首地址单元。但是究竟从 U 盘的哪个端点来接收数据，则是由 U 盘本身决定的，该端点号是在 U 盘的枚举过程中 U 盘告诉记录仪的[45]。

```
    unsigned char epBulkRcv(unsigned char *pBuffer,unsigned int len)
    {
        usbstack.usbaddr= USB_ADDR;//设备的地址
        usbstack.endpoint=usbstack.epbulkin;//设备批量发送端点号
        usbstack. pid=PID_IN;//令牌包类型
        usbstack.wPayload=64;//数据包的最大长度
        usbstack.wLen=len;//一共要传输的数据
        usbstack.buffer=pBuffer;//指向在 RAM 存放接收数据的首地址
        if(usbstack.wLen)//是否有数据需要接收
        {
          if(!usbXfer())//接收数据
          return FALSE;//接收失败返回 FALSE
        }
          return TRUE;//接收成功返回 TRUE
        }
```

③ unsigned char SetAddress(unsigned char addr)

分配一个地址给设备(在记录仪中就是指 U 盘)，地址号由 addr 决定，在序中调用该函数的实参是个定值。命令完全遵照 USB 标准命令格式。

```
    unsigned char SetAddress(unsigned char addr)
    {
        .//完全根据 USB 设置地址命令格式对参数进行赋值
         usbstack.usbaddr=0;//设备初始设备号是 0
          usbstack.setup.bmRequest=0;
```

```
    usbstack.setup.bRequest=SET_ADDRESS:
    usbstack.setup.wValue=addr;//给设备分配地址
    usbstack.setup.wIndex=0;
    usbstack.setup.wLength=0;
    return ep0Xfer ();//通过端点 0 发送数据
}
```

3.3.3 海量存储设备类驱动设计

海量存储设备类(USB Mass Storage)是 USB 协议定义的存储设备类规范，主要用于软磁盘接口、ATA 接口、IDE 硬盘接口及 FLASH 存储器等设备的磁盘管理及大容量数据传输。这个类规范包括四个独立的子类规范：控制/批量/中断传输协议(CBI)，单批量传输协议(Bulk-Only)，ATA 命令集和 UFI 命令集。前两个子规范定义了数据/命令/状态在 USB 上的传输方法。Bulk-Only 传输规范仅使用 Bulk 端点传送数据/命令/状态，CBI 传输规范则使用控制/批量/中断(Control/Bulk/Interrupt)三种类型的端点进行数据/命令/状态传送。后两个子规范则定义了存储介质的操作命令。ATA 命令规范一般用于硬盘，而 UFI 命令基于 SCSI-2'和 SFF-80701 命令规范发展而来，针对 USB 移动存储设备[46]。

在 Bulk-Only 传输方式中，有三种类型的数据在 USB 和设备之间传送:CBW，CSW 和普通数据。CBW(Command Block Wrapper，即命令块包)是从 Host(主机)发送到设备的命令；命令格式遵从接口中的 bInterfaceSubClass 所指定的命令块，这里为 SCSI 传输命令(UFI)集。USB 设备需要将 SCSI 命令从 CBW 中提取出来，执行相应的命令，完成以后，向 Host 发出反映当前命令执行状态的 CSW (Command Status Wrapper)；Host 根据 CSW 来决定是否继续发送下一个 CBW 或是数据。Bulk-Only 传输过程如图 3-13 所示。

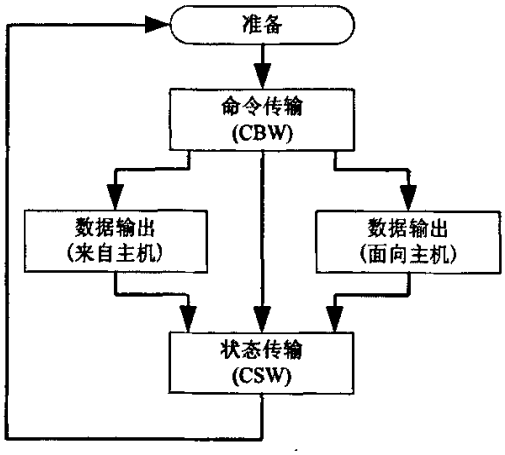


图 3-13 Bulk-Only 传输过程

(1) CBW 命令格式：

USB HOST 向设备发送 CBW 的格式如表 3-6 所示。

表 3-6 CBW 的格式

bit / byte	7	6	5	4	3	2	1	0
0-3	dCBWSignature							
4-7	dCBWTag							
8-11	dCBWDataTransferLength							
12	bmCBWFlags							
13	保留					bCBWLUN		
14	保留					bCBWCBLength		
15-30	CBWCB							

其中 dCBWSignature 的值为 43425355h(LSB)，表示当前发送的是一个 CBWdCBWTag 的内容需要原样作为 dCBWTag 再发送给 HOST；dCBWDataTransferLength 为本次 CBW 需要传输的数据；bmCBWFlags 反映数据传输的方向，0 表示来自主机，1 表示发送至主机；bCBWLUN 一般为 0，但是当设备有多个逻辑单元时，用此位指定数据发送的目标单元；bCBWCBLength

为本次命令字的度；CBWCB 才是真正的传输命令集(UFI)的命令。

(2) CSW 命令格式：

设备得到一个 CBW 命令之后，解析出 CBWCD 中所代表的命令，然后按照 UFI 命令集中的定义来执行相应的操作:接收下一个从 Bulk-Out 端点发来的数据，还是要向 Host 传送速据。完成以后需要向 USB Host 发送 CSW，反映命令执行的状态。USB Host 也是通过这个来了解设备的工作情况。CSW 的格式如表 3-7 所示。

表 3-7 CSW 的格式

bit / byte	7	6	5	4	3	2	1	0
0-3	dCSWSignature							
4-7	dCSWTag							
8-11	dCSWDataResidue							
12	bCSWStatus							

dCSWSignature 的内容为 53425355h；dCSWTag 的内容为 dCBWTag 的内容；dCSWDataResidue 为还需要传送的数据，此数据根据 dCBWDataTransferLength 本次已经传送的数据得到。HOST 端根据此值决定下一次 CBW 的内容，如果没有完成则继续;如果命令正确执行，bCSWStatus 返回 0。按照这个规则组装好 CSW 后，通过 Bulk-In 端点将其发出。

UFI 命令集定义了一系列的磁盘操作命令和命令返回内容，例如 READIO，WRITEIO，READCAPACITY 等。每一个 UFI 命令的实现都是通过单批量传输方式来实现。其中几个主要的命令函数如下。

① READIO

unsigned char ReadIO(unsigned long lba, unsigned char len, unsigned char *pBufer);

从 U 盘的逻辑块地址 lba 上，读出 len 字节长度的数据，存放在 pBuffer 指向的缓冲区中。返回值 1 表示读取成功，且等于读取的数据长度，返回值为零表示读取失败。

② WRITEIO

unsigned char WriteIO(unsigned long lba, unsigned char len, nsigned char *pBufer);

把 pBuffer 指向的 len 字节的数据写到 U 盘的逻辑块地址 lba 上。返回值 1 表示写成功，且等于写入的数据长度；返回值为零表示写入失败。

③ READCAPACITY

unsigned char ReadCapacity();读取 U 盘的容量。

3.3.4 FAT 文件操作程序设计

U 盘使用 FAT16 文件系统，因此我们需要在 USB 主机上实现对 FAT16 文件系统的操作，主要是利用 USB 海量存储设备类驱动层提供的读、写、查询等命令，对 FAT 表进行定位、搜索和读、写操作，实现查找、新建、读、写文件的功能[47]。

一个 FAT 文件系统卷是由四个部分组成:

(1) 保留区(Reserved Region)

(2) FAT 区(FAT Region)

(3) 根目录区(Root Directory Region)

(4) 文件和目录数据区(File and Directory Data Region)

主机端的文件系统程序利用设备类驱动层提供的各种文件操作命令来读写以上 FAT16 文件信息，从而实现文件的读写和查询。当主机发出 READ 命令后，开始读取操作，首先读取 BPB，得到存储介质的有关信息，如扇区长度、扇区数以及总扇区数等内容。实现该过程的函数名是 EnumMassDev()，流程图如图 3-14 图[48]。

文件操作协议的实现还包括下面的函数，下面分别说明每个函数的用途：

(1) unsigned long FirstSectorofCluster(unsigned int clusterNum)

计算指定簇的第一个扇区号。一个簇包含多个扇区，每个扇区在整个 U 盘中都有一个 LBA(逻辑块地址)号，即扇区号。该函数就是计算指定的簇中的第一个扇区的扇区号。clusterNum 是簇号，返回值为扇区号(LBA 号)。

(2) unsigned int ThisFatSecNum(unsigned int clusterNum)

根据文件簇号计算 FAT 表中描述该簇的表项的扇区的扇区号。由于一个 FAT 表将占用 1 个以上的扇区，而 FAT 表中描述一个簇的使用情况的表项(两个字节长度)究竟在哪个扇区里，其扇区号由该函数得出。clusterNum 是簇号，返

回值是 FAT 表中描述该簇的表项的扇区的扇区号。

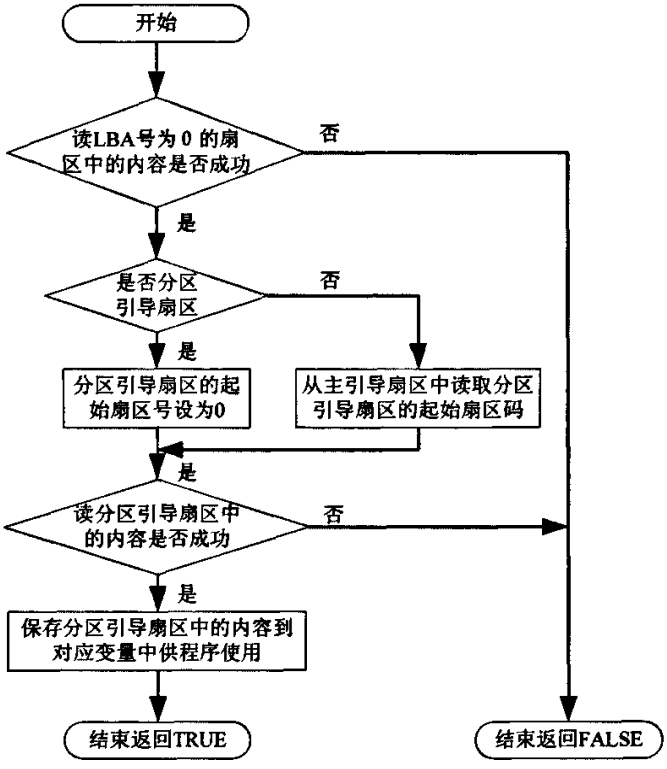


图 3-14 EnumMassDev()流程图

(3) unsigned int ThisFatEntOffset(unsigned int clusterNum)

根据文件簇号计算描述该簇的表项在 FAT 表中的偏移。FAT 表用来描述簇的使用情况，为了在 FAT 表中找到描述某簇的使用情况的表项(两个字节)，除了要知道该表项所在的扇区号，还要知道该表项在其所在扇区中的偏移值。该函数指出描述某簇。使用情况的表项在其所在扇区中的偏移值。clusterNum 是簇号，返回值是描述该簇的表项在其所在扇区的偏移值。

(4) unsigned int GetNextClusterNum(unsigned int clusterNum)

在 FAT 表中搜索当前文件的下一个簇。一个文件占用的簇的数目可能不止一个，当大于一个的时候，必须知道紧跟该文件的某个簇的下一个簇的簇号。该函数指出文件的某个簇的下一个簇的簇号。clusterNum 是簇号，返回值是紧

跟该文件的 clusterNum 簇的下一个簇的簇号。

(5) unsigned char DeleteClusterLink(unsigned int clusterNum)

删除一个文件在 FAT 表中的所有标志。删除某个文件，其实就是将 RDR(根目录区)描述该文件属性的 16 个字节的第一个字节变成 ORES，同时在 FAT，表中找出描述该文件占用的所有的簇的表项，将其清 0。而该函数只是在 FAT 表中找出描述该文件的所有簇的表项，并将其清 0，并没有删除该文件。因为一个分区的 FAT 表的份数有两份，其结构和数据完全一样，因此清 0 时是两个表中的表项一起清 0。clusterNum 是簇号，成功则返回 TRUE，失败返回 FALSE。

(6) unsigned int GetClusterNumFromSectorNum(unsigned long sectorNum)

根据扇区号计算簇号。文件的存储是以簇为单位的，一个簇包含至少一个扇区，已知一个扇区号(LBA 号)，必须知道其所属的簇的簇号。该函数就是求扇区号所对应的簇号。sectorNum 为扇区号，返回其对应的簇号。

(7) unsigned char GoToPointer(unsigned long pointer)

将文件的指针位置转换成簇号，扇区号以及在扇区内的地址偏移。一个文件包含的字节数至少大于等于 0 个，如果要在文件的某个字节后面增加一个或多个字节，则必须找到该字节所在的位置。这个位置包括该字节所在的簇号、该字节所在的扇区位于该扇区所在的簇中的第几个扇区、该字节所在的扇区的扇区号(LBA 号)、该字节在其所在簇的中的偏移值。该函数就是找到指定字节的字节号的位置，并将这些有关位置的值赋给全局变量。pointer 就是字节号。成功则返回 TRUE，失败则返回 FALSE。

(8) unsigned int GetFreeCusterNum(void)

在 FAT 表中搜索下一个空簇的簇号。由于一个文件占用的簇可能不止一个，当需要新的一个簇的时候，就必须到 FAT 表中寻找表项的值为 0 的簇所对应的簇号，并将该表项值变为 0xFFFF。该函数就是寻找一个空簇，返回值为该簇号。

(9) unsigned int CreateClusterLink(unsigned int currentCluster)

当文件长度超过一个簇的时候，新建一个新的簇，并在 FAT 一表中建立相关信息。通过 GetFreeClusterNum()函数找到一个新的簇，并将该簇的簇号放入 currentCluster 簇号所对应的 FAT 表项中去，这样就将新的簇连接到一个文件上了。currentCluster 为本文件的最后一个簇号，要增加的簇的簇号就放在该簇号对应的表项里面。成功就返回 TRUE，失败就返回 FALSE。

(10) unsigned char RBC_Read(unsigned long lba,unsigned char len,unsigned

char * pBuffer)

读取从指定扇区中开始的数据。文件在U盘中存放的最小单位是簇，簇是由扇区组成的，读取文件中数据的时候，最好是以扇区为单位读取。该函数就是读取从指定扇区开始的某几个扇区的数据，并将这些数据存放到指定的单元中去。lba指定了起始扇区号，len表明要读取几个扇区，pBuffer指向存放读出的数据的起始单元地址。读取成功返回TRUE，读取失败返回FALSE[49]。

(11) unsigned char RBC_Write(unsigned long lba,unsigned char len,unsigned char * pBuffer)将数据写入到从指定扇区开始的扇区中去。指定的数据必须是每个扇区中字节数的整数倍(一般情况下一个扇区中包含512个字节，所以数据长度必须是512的整数倍)。lba表明要写入的起始扇区，len表明写入的长度，其一个单位代表512个字节，pBuffer指向存放要写入的数据的起始单元。写入成功返回TRUE，写入失败返回FALSE。

用以上基本函数可以构成在记录仪写U盘中的常用函数：

(1) unsigned char CreateFile(unsigned char *pBuffer)

创建一个文件。描述该文件的属性共32个字节，这32个字节必须完全符合文件目录项的格式。pBuffer指向该32个字节的首地址。在创建完一个文件后，该文件就处于打开状态，不需要再次打开，当然如果再次调用打开文件函数也不会出错。如果创建成功则返回TRUE，失败返回FALSE。

(2) unsigned char OpenFile(unsigned char *pBuffer)

打开一个指定文件名的文件。其实就是将描述该文件状态的结构体中的bFileOpen变量置1。pBuffer指向要打开文件的文件目录项的32个字节的首地址。在打开该文件之后，从文件目录项中读取该文件的属性，并赋给描述该文件的状态的结构体中的某些变量。如果打开成功则返回TRUE，失败返回FALSE。

(3) unsigned char WriteFile(unsigned int writeLength,unsigned char *pBuffer)

往指定的文件中写数据。此时文件必须处于打开状态，如果文件没有被打开，则无法写文件。writeLength为要写入的数据的长度，pBuffer指向要写入的数据的首地址。如果写入成功则返回TRUE，失败则返回FALSE。

3.3.5 用户应用层程序设计

USB设备应用程序层是主机系统软件的最上层，面向记录仪功能上的应用。

向 U 盘中写入行驶记录，以便导入到后台数据分析软件中。数据文件格式与后台分析软件读取的格式必须完全一致。读写文件都要通过调用 FAT16 文件系统的接口来完成。在操作 U 盘的过程中，必须有标志来描述 U 盘的状态，因此程序中定义了这样一个结构体，该结构体表明 U 盘的各种状态[50]。

```
typedef struct
{
unsigned char SLAVE_IS_ATTACHED: 1;//U 盘是否被正确读取 1:正确 0:不正确
unsigned char SLAVE_ REMOVED:1;  //U 盘是否被拔出 1:拔出 0:没有拔出
unsigned char SLAVE_FOUND:1;  //U 盘是否被插入 1:插入 0:没有插入
unsigned char SLAVE_ENUMERATED:1;//U 盘是否经过枚举 1:已枚举 0:未枚举
unsigned char SLAVE_ONLINE:1://U 盘存在标志 1:存在 0:不存在
unsigned char DATA_ STOP:1;//是否需要停止数据传输 1:停止 0:不停止
unsigned char bDatal:1;//使用 dataX 传输
unsigned char bMassDevice:1://插入设备是否是 U 盘 1 是:0:不是
}XXGFLAGS;
```

在操作文件的过程中，必须有标志来描述所操作文件的状态，因此程序中定义了这样的一个结构体，该结构体表明所操作文件的各种状态。

```
typedef struct_FILE_INFO
{
unsigned char bFileopen;//文件是否被打开
unsigned int StartCluster;//文件的起始簇号
unsigned long LengthinByte;//文件的长度，以字节为单位
unsigned int ClusterPointer;//文件中指定字节所在簇的簇号
unsigned long SectorPointer;//文件中指定字节所在扇区的扇区号
unsigned int Pointer;//文件中写了多少个字节，用于更新 LengthInByte
unsigned char SectorofCluster;//文件在一个簇中占了多少个扇区
}FILE_INFO,*PFILE_INFO;
```

利用 FAT，文件操作的程序和上面两个结构可以构成一个完整的写文件过程。如果文件名已经存在，那么就不需要再次创建文件名。用户应用层的程序直接根据需要调用相关的 FAT 层的函数，就可以实现相应的写文件功能。在本项目中，应用层是将汽车行驶记录仪中的存储单元的内容存到 U 盘中，并以一

个文件名存在。流程图如图 3-15 所示。

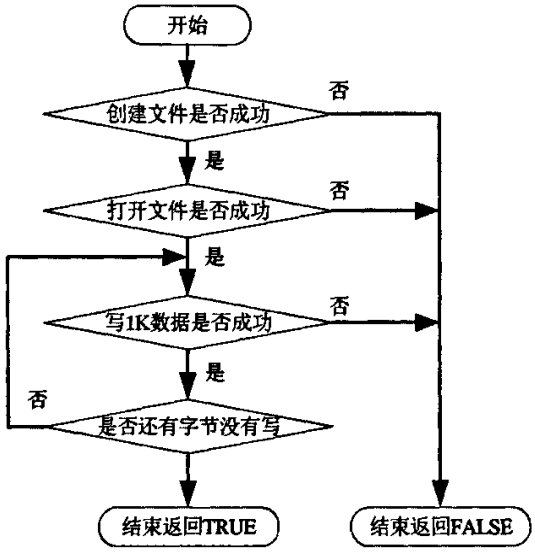


图 3-15 利用 FAT 创建文件流程图

第 4 章　系统抗干扰设计与调试

4.1 系统抗干扰设计

汽车运行时的环境一般比较恶劣，干扰很严重。所以在系统设计上，应该采取必要的软硬件措施，去除和减小各种不良因素对它的影响和损害，提高工作的稳定性和可靠性。记录仪的可靠性是由多种因素决定的，其中系统的抗干扰性能是系统可靠性的重要指标。在本系统中，系统的输出信息更为重要，所以对系统的精度要求也很高，抗干扰技术也贯穿于整个设计中[51]。

硬件抗干扰设计的基本原则是：抑制干扰源，切断干扰传播路径，提高敏感器件的抗干扰性能。

抑制干扰源就是尽可能的减小干扰源的 du/dt, di/dt。这是抗干扰设计中最优先考虑和最重要的原则，常常会起到事半功倍的效果。减小干扰源的 du/dt 主要是通过在干扰源两端并联电容来实现。减小干扰源的 di/dt 则是在干扰源回路串联电感或电阻以及增加续流二极管来实现。对于可能会影响汽车行驶记录仪工作状态的干扰源，在其源头进行屏蔽。按干扰的传播路径可分为传导干扰和辐射干扰两类。所谓传导干扰是指通过导线传播到敏感器件的干扰。高频干扰噪声和有用信号的频带不同，可以通过在导线上增加滤波器的方法切断高频干扰噪声的传播，也可加隔离光耦来解决。本系统中高频干扰噪声主要是系统采集汽车开关量时有可能产生高频信号，对此系统中采用的措施是使用光耦进行隔离。光电隔离措施主要通过光电隔离器来实现。光电隔离器以光电转换原理传输信息，它不仅使信息发出端与信息接收并输出端是带电绝缘的，从而对地电位差干扰有很强的抑制能力，而且有很强的抑制电磁干扰的能力，且速度高，价格低，接口简单，因而得到广泛应用。所谓辐射干扰是指通过空间辐射传播到敏感器件的干扰。一般的解决方法是增加干扰源与敏感器件的距离，用地线把它们隔离和在敏感器件上加蔽罩。切断干扰传播路径：切断干扰传播路径的途径主要有以下几种：

① 电路中每个 IC 在电源输入端都要并接一个去耦电容，电容型号为 0.1uF

的高频瓷片电容。

② 当使用 DSP 的 I/O 口控制或接收汽车信号时，必须使用光电隔离器进行隔离，不能直接使用 I/O 口控制，隔离外部高频信号对系统的干扰。

印刷电路板（PCB 板）是单片机应用系统中器件、信号线、电源线的高密度集合体，PCB 板设计的好坏对抗干扰能力影响很大，故 PCB 板的设计绝不单是器件、线路的简单布局安排，还必须符合抗干扰的设计原则。一个性能优良的系统，必须从软硬件设计到 PCB 板的制作都要时刻考虑系统抗干扰的设计，消除系统的大部分干扰很大程度上取决于 PCB 板的布局绘制，特别是在高频电路中，PCB 板的设计尤为重要，所以 PCB 板的设计是系统设计的重中之重。通常 PCB 板设计有下述几个抗干扰措施。

(1) 电源线与地线设计

在整个 PCB 设计过程中，电源线和地线的设计布局尤为重要。为减小干扰，本系统 PCB 板中的电源线都要尽量的粗，为数字 IC 供电的电源线宽度应该不小于 20mil，对大电流器件更要加宽电源线，比如系统中为打印机供电的电源线宽度为 80mil。本系统中的地线主要有电源地、数字地、模拟地，地线的处理是 PCB 设计中的重要环节。抑制干扰的重要方法就是接地线的布局，如能将接地和屏蔽结合起来正确使用则可解决大部分的干扰问题。本系统中采用了以下方法：

① 数字、模拟电路分开

因为本设计中，既有系统外的模拟电路信号接入，也有逻辑电路，因此，在制作电路板时，将模拟电路用地线与数字电路隔离，并分别用两个独立的电源对它们供电，以增强抗干扰能力。

② 接地线尽量加粗

若接地线条很细，接地电位则随电流的变化而变化，致使微处理器的定时信号电平不稳，抗噪声性能变坏。因此，应将接地线加粗，使它能通过三倍于印刷电路板上的允许电流，并且使电源线、地线的走向与数据传递的方向一致，将有助于增强抗噪声能力。

③ 接地线构成闭环回路

根据电路设计工程师的经验，对于只用数字电路组成的印刷电路板。接地时，将其做成闭环回路大多明显地提高抗噪声能力。其原因是：一块印刷电路

板上有很多集成电路，尤其遇有耗电多的元件时，因受到线条粗细限制，地线产生电位差，引起抗噪声能力下降，如连成环路，则其差值缩小。因此，将系统的地线设计成回路。

(2) 去耦电容配置

在印刷电路板的各个关键部位配置去耦电容是印刷电路板设计的一个常规做法。电源输入端跨接电解电容器，本系统中为两个电源输入端各配置了一个2200uF/16V 的铝电解电容。每个集成电路芯片都应安置一个 0.1uF 的陶瓷电容器用来去耦，而且在 PCB 设计时要注意使去耦电容靠近 IC，并且电源线最好先经过去耦电容再供给 IC。因 FLASH 的抗噪声能力弱、关断时电流变化大，因此，在芯片的电源线和地线间直接接入较大容量的去耦电容。

(3) 其他措施

在进行布线的时候要注意使总线走线平行且基本等长，顶面和地面的线应该尽可能相互垂直。芯片的晶振应该尽可能靠近所属芯片，并且两个脚的布线应尽量等长，而且 USB 口的两条数据线 D+和 D-更是要注意使其基本等长，这可以大大减小读写数据出错的可能。CMOS 芯片的输入阻抗很高，易受感应，对其不用的引脚要接地或接正电源。

为了抑制空间电磁干扰，在系统的电源线上加入一个磁环，电源线和地线在磁环上缠绕两圈，这就极大的减少了空间电磁干扰及汽车打火干扰等。实践还表明，元器件的质量对系统影响很大，应选择正品元器件而且生产年限最好是近两年之内的。使用前还要进行必要的测试和筛选。对于接插件，应选择抗震性能好，接合可靠，防松的接插件。传输电缆应具有性能良好的屏蔽层，耐老化，抗损伤，不易断线。外接的连接线最好加一个防火的波纹管。同时系统的性能与结构的设计有很大的关系，比如定位孔的合理，PCB 板上对定位孔的设计，定位孔是要与 PCB 板绝缘还是要与 PCB 的底线连通，这都是设计中应该注意的地方。

4.2 系统调试

产品设计完成后，首先必须保证系统样机硬件上没有非常明显的错误。这

一步需要做的工作如下：

(1) 检查 PCB 板上是否有明显的走线短路或断路的情况。

(2) 检查 PCB 板上元器件的封装是否与实际的元件相同，对于不太合适的要在下次改板时修改。

(3) 检查 PCB 板上的元器件的焊接是否存在虚焊、断路方向错误等非常明显的错误。

(4) 检查电源部分是否断路，断开电源与系统之间连接的情况下，使用万用表单独检测电源是否可以正确输出 5V 和 3.3V 的电压。

(5) 连接好电路后，监测 PCB 板上各供电点的电压是否正常。在检查电源的过程中，即在样机前几次上电时必须要非常小心，一发现有异常的情况（比如芯片短时间急剧发热）必须马上断开供电电源，然后检查电路哪里存在问题。

以上过程在硬件调试过程中都比较简单，也只能发现最简单的电路问题，并不能发现由于系统设计失误而发生的问题，但这是样机调试必不可少的，必须加以重视，而且检查过程中要非常细心仔细，任何一点小的问题都不能放过在确认样机的硬件没有问题之后就可以开始进行软件调试。

在这个过程中就要在整体上对系统的各个部分的功能进行详细的调试，检查电路的每个部分是否都能正常工作，确认各部分功能是否达到，解决由于系统硬件设计或软件设计方面存在的缺陷、错误等。在这个过程中，最主要的任务就是调试驱动程序，通过各部件的运行和整体运行发现软件中的问题并一步步完善系统软件。同时，在发现问题后首先要分析是软件上的问题还是硬件上的问题，然后有针对性的进行解决。

在完成了系统的整体测试，样机的各部分都能正常工作，功能都达到之后，就需要进行系统的性能测试。要测试系统的各个方面的功能是否都达到了既定的要求，各种性能是否都满足了需要。在这个阶段就不但要在实验室中进行模拟测试，还必须进行实际状况下的测试，毕竟记录仪是需要安装在汽车上使用的，而不是放在实验室的桌面上使用的模拟测试就是在实验室中对系统的各项功能及要达到的性能进行完整的测试。本系统进行模拟测试的过程，是严格按照行车记录仪国家标准进行的。

对于系统基本功能的测试，包括开机自检、LCD 显示、实时时钟、速度信号检测、开关量检测、数据存储、驾驶员身份识别(IC 卡)、数据打印、数据通讯(RS232 和 USB)等在系统调试阶段基本就可以确认功能已经达到。而对于速

度、里程的检测存储是否准确，记录误差是否达到要求，数据存储是否可以长期保持、系统长期运行时可能出现的问题，系统的环境适应性等都需要在实验室中模拟测试才能确认。对于记录仪的环境适应性，本系统按照国标给出的测试方法对记录仪进行了测试，必须符合国标要求。由于汽车工作环境恶劣，一些元件在常温下工作正常，但当环境改变时时候，可能出现问题。同时还要对采集模块进行模拟采集。行车记录仪最终是要安装在汽车上运行的，所以实车测试对记录仪来说是非常重要的。这个阶段主要测试系统在汽车上各项功能是否可以正常工作，电源、打印机是否受到影响，速度、开关量及里程的测试记录是否准确等。总之，对系统的测试是个重要的不可或缺的部分，它对记录仪的改进和升级是必须的。

第 5 章 总结与展望

5.1 总结

作为规范驾驶员行为、保障道路交通安全的一把利器，汽车行驶记录仪在汽车电子产品中有着其无法替代的地位。本论文着重阐述了记录仪主机模块的软硬件设计和开发，对系统方案的确定、芯片的选型、软件模块的设计等进行了详细论述。本记录仪主机模块具有如下功能：

(1) 自检功能。

(2) 实时时钟、日期的输出显示功能。

(3) 疲劳驾驶时段及超速驾驶时段的记录功能。

(4) 车辆行驶速度的测量、记录和存储功能。

(5) 车辆行驶里程及里程小计的测量、记录和存储功能。

(6) 驾驶员身份记录和识别功能。

(7) LCD 液晶显示功能。

(8) 按键操作功能。

(9) 数据打印输出功能。

(10) 数据 RS232 串口通信功能。

(11) CAN 总线接口通信功能。

(12) USB 主机功能。

对于汽车行驶记录仪而言，能够对大量的实时数据进行及时采集、处理和存储至关重要。因此，较高的实时性是汽车行驶记录仪一个十分重要的指标要求。本记录仪主机模块采用高效的 16 位 DSP 为核心，能够有效地提高系统实时性。优化编写的代码对多任务进行了统筹兼顾的合理调度，有效地保证了实时性。此外，DSP+CPLD 的构架供了方便，减轻了处理器负担，也间接提高了系统的器实时性。丰富通信方式为数据的下载提供了有效和多样的渠道。

5.2 展望

虽然本文针对车辆行驶记录仪做了一定的研究工作，并取得了一些研究成果，但仍有一些问题值得进一步探讨。结合本文的所介绍内容，今后需要从以下几方面做进一步的研究：

(1) 将 GPS 定位技术和记录仪结合，从而可以实时地加强对车辆的全过程动态安全管理，提高车辆事故与安全的预防和分析能力，同时可以结合汽车防盗装置，实现车辆防盗跟踪的功能。

(2) 通过在记录仪上加入无线数据采集模块，利用无线发送装置来完成对数据的采集功能。

(3) 集成 GIS 地理信息系统，从而使汽车行驶记录仪具有电子地图功能，能提供无极缩放，任意平移和漫游的电子地图、城市交通图，方便车主查询各类地理信息。

(4) 汽车工作环境恶劣，如何进一步提高系统抗干扰能力，使行驶记录仪达到更稳定的工作状态是系统升级改进的重点。

(5) 为了方便任务的管理，可以引入实时操作系统 DSP/BIOS 或其他操作系统来完成任务的管理与运行，这样可提供系统工作效率，也方便管理。

参考文献

[1] 李奕薇．我国行车记录仪市场调研．中国交通信息产业，2003，(1)：22~24

[2] 王征平，王小惠．行车记录仪市场分析．中国交通信息产业，2003，(1)：90~95

[3] 史鑫一，陈秀红．正确驾驶客车，降低运输成本—行车记录仪在车辆技术管理中的作用．客车技术与研究，2001，23(2)：156~162

[4] 翟永红．设置最高限速用好行车记录仪．中国道路运输，2004，(4)：156~162

[5] GB/T19056-2003 汽车行驶记录仪起划草工作组．GB/T19056-2003《汽车行驶记录仪》实施指南．北京：中国标准出版社，2003

[6] TMS320C54x DSP Refence Set Volume 1: CPU and Peripherals, Texas Instruments Incorporated, 2001

[7] TMS320C54x DSP Reference Set Volume 4: Applications Guide, Texas Instruments Incorporated, 1996

[8] TMS320VC5416 Fixed-Point Digital Signal Processor Data Manual, Texas Instruments Incorporated, 2003

[9] 张福学．传感器应用及其电路精选．北京：电子工业出版社，1992

[10] 何希才．传感器及其应用．北京：国防工业出版社，2001

[11] Niloy Bhadra, P Hunter Peckham, Michael W Keith, Kevin L Kilgore etc. Implementation of an implantable joint-angle transducer. Joumal of Rehabilitation Research and Development, 2002. 39(3): 411~422

[12] Togawa Kiyoshi, Sanbonsugi Hideaki, Lapicki Adam etc. High-Sensitivity InSb Thin-Film Micro-Hall Sensor Arrays for Simultaneous Multiple Detection of Magnetic Beads for Biomedical Applications. IEEE Transactions on Magnetics, 2005. 41(10): 3661~3663

[13] 康华光．电子技术基础数字部分．北京：高等教育出版社，2000

[14] 阳子轩．汽车行驶记录仪的研究与开发：[硕士学位论文]．武汉：武汉理工大学，2006

[15] 崔立超．汽车行驶记录仪及其后台数据分析软件的设计与实现：[硕士学位论文]．西安：西北工业大学，2005

[16] 刘兵，徐家恺，刘阳．DSP 的嵌入式 USB 主机接口设计．微型机与应用，2004，(8)：21~22

[17] 王仁龙，肖忠炳，魏宇，于春花，崔允红．TMS320VC5416 与 CAN 总线的接口设计及软件编程．应用科技，2006，(9)：11~13

[18] 唐冰，周勇，骆云志，王峰，赖春强．全功能异步收发器与 DSP 的 SPI 接口技术．单片机与嵌入式，2003，(5)：34~37

[19] T6963C 控制器图形液晶显示模块使用手册．北京：北京精电蓬远显示技术有限公司，2004

[20] 吴大中．逻辑加密卡 SLE4442 及其应用．电子技术，1997，(7)：42 - 46

[21] 点阵字符型液晶显示模块使用手册．北京精电蓬远技术有限公司，2001

[22] 吴文杰．汽车行驶记录仪：[硕士学位论文]．南京：南京理工大学，2000．6

[23] 邓洁清，郑建勇，宋明．基于 DSP 和 CPLD 的人机接口实现．工业控制计算机，2004，(17)：27~29

[24] 陈晓明，苗世洪．张军民．微机保护中 DSP 与时钟 DS12CR887 的接口设计．国外电子元器件，2007，(3)：49~52

[25] Jason Hansen, Jill Hill.Switching regulator charges NiMH batteries. EDN, 2000. 45(26): 95~96

[26] Christophe Basso, Francois Lhermite. Switch-mode supply draws 43-mW standby power. EDN, 1999. 44(12): 116~117

[27] Handa Hiroshi, Sandhu Adarsh. IC has a controlling interest. Computer Design's: Electronic Systems Technology&Design, 1997. 36 (12): 41~49

[28] 彭启琮，管庆．DSP 集成开发环境 CCS 及 DSP/BIOS 的原理与应用．北京：电子工业出版社，2004

[29] 钟文政，柯鸿禧．DSPTMS320C50 原理与应用．北京：中国水利水电出版社，2003

[30] 刘益成．TMS320C54xDSP 应用程序设计与开发．北京：北京航空航天大学出版社，2002

[31] 颜友钧，朱宇光．DSP 应用技术教程．北京：中国电力出版社，2002

[32] 刘慧，林海虹，刘智．多核 DSP 的 Bootloader 程序的实现．电子技术应用，2003，(6)：72~75

[33] 刘蓉晖．DSP 的自举引导方法的应用研究．世界电子元器件，2004，(5)：49~51

[34] 郎岩梅，唐文彦，赵军．基于 DSP 的嵌入式系统中 BOOTLOADER 程序的设计方法．电测与仪表，2003，(4)：35~37

[35] 陈喆，张福洪．数字信号处理器引导装载方式研究．杭州电子工业学院学报，2002，22(6)：65~68

[36] Sha Zhanyou, Meng Zhiyong. The communication between TMS320C54x DSP and CPLD through host port interface. Proceedings of SPIE - The International Society for Optical Engineering, 1998: 1755-1757

[37] Chen Guilin, Kandemir Mahmut. Optimizing address code generation for array-intensive DSP applications. Proceedings of the 2005 International Symposium on Code Generation and Optimization, 2005: 141-152

[38] Jiangbo Sun, Houjin Chen. Optimization of DSP virtual machine in embedded DSP. Proceedings of International Conference on Signal Processing Proceedings, 2004:136-139

[39] Tao Huang, Yunfei Zhou, Ming Zhong. Adaptive Real-Time Network Design of Embedded System. Adaptive Real-Time Network Design of Embedded System. Proceedings of the 2006 IEEE/ASME International Conference on Mechatronic and Embedded Systems andApplications, 2006: 560-565

[40] Jiangbo Sun, Houjin Chen. Optimization of DSP virtual machine in embedded DSP. International Conference on Signal Processing Proceedings, 2004:136-139

[41] Huang Tao, Zhong Ming, Zhou Yunfei. Adaptive Real-Time Networks Design and Simulation Based on IC Wedge-Bonding Control System. Proceedings of the International Technology and Innovation Conference 2006 — Advanced Manufacturing Technologies, 2006: 253-256

[42] Xin Hua-Feng, Yu Feng, Tan Jian, Wang Wen-Li. Embedded USB host/slave application module. Journal of Jilin University, 2005, 35(2): 166-169

[43] Depari A., Flammini A, Marioli D., Taroni A. USB sensor network for industrial applications. Proceedings of the 21st IEEE Instrumentation and Measurement Technology Conference, 2004: 1203-1207

[44] Gereaux, Dean A. USB Device Drivers. Dr. Dobb's Journal, 2004, 29(4): 60-64

[45] Kim Yong-Seok, Kim Hee-Sun, Lee Chang-Goo. The development of USB Home Control Network System. Proceedings of 2004 8th International Conference on Control, Automation, Robotics and Vision, 2004: 289-293

[46] Popa M., Marcu M., Popa A.S.. A microcontroller based data acquisition system with USB interface. Proceedings of 2004 International Conference on Electrical, Electronic and Computer Engineering, 2004: 206-209

[47] Yao Aiqin, Sun Yunqiang, Shi Xiling. Data acquisition systems based on USB. Proceedings of the International Symposium on Test and Measurement, 2003: 627-630

[48] Yao Aiqin, Sun Yunqiang, Shi Xiling. QoS support for USB 2.0 periodic and sporadic device requests. Proceedings of the 25th IEEE International Real-Time Systems Symposium, 2004: 395-404

[49] Morrow Michael G, Welch Thad B., Wright Cameron H.G. Real-time DSP data acquisition for high speed host transfer. Proceedings of the 49th International Instrumentation Symposium, 2003: 525-530

[50] Morrow Michael G, Welch Thad B., Wright Cameron H. G. Enhancing the TMS320C6713 DSK for DSP education. Proceedings of 2005 ASEE Annual Conference and Exposition: The Changing Landscape of Engineering and Technology Education in a Global World, 2003: 525-530

[51] 卢存伟．计算机控制系统实用抗干扰技术．自动化与仪表，1989，2：36-39

致 谢

回顾三年来的硕士学习生活，我身边的老师、同学和家人给了我许多帮助和鼓励，在此我对他们表示最真挚的感谢！

首先要感谢我的导师黄涛教授。在学习上，黄老师渊博的理论知识、丰富的实践经验屡屡帮助我在课题的关键时刻找到解决办法。在生活中，黄老师对我关怀备，教育我如何去正确对待工作生活和为人处世，努力去做一个具有全面素质的人。同时也要感谢廖传书老师和卢珞先老师三年对我的指导，他们严谨的工作作风和渊博的知识及和蔼可亲的态度，令我终身受益。

同时也要感谢实验室和我一起工作的师兄姐妹们，我们一起建立了良好的学习氛围。特别鸣谢管璞同学对我的论文方面提供的帮忙。

最后要感谢的是我的父母和弟弟，他们在我的生活和学习上给我无微不至的关怀，让我懂了亲情的宝贵，促使我更加努力的学习和工作。

本文在某些方面难免存在不足，请各位老师和专家批评指正。

钟明

2007 年 4 月

武汉理工大学

攻读硕士学位期间发表的学术论文

[1] 黄涛, 钟明. 车载 CAN 总线故障诊断仪的设计. 武汉理工大学信息与管理工程版, 2006 年, 第 28 卷, 第 11 期

[2] Tao Huang, Yunfei Zhou, Ming Zhong. Adaptive Real-Time Network Design of Embedded System. Proceedings of the 2006 IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications. 2006, 8. （EI 收录）

[3] Huang Tao, Zhong Ming, Zhou Yunfei. Adaptive Real-Time Networks Design and Simulation Based on IC Wedge-Bonding Control System. Proceedings of the International Technology and Innovation Conference 2006 — Advanced Manufacturing Technologies. 2006,11. （EI 收录）