

基于 CANopen 总线的数控系统高速数字通信技术研究

Study on Digital Communication with High Speed Based on
CANopen in CNC

作者姓名 黄 涛

学位类型 学 历 硕 士

学 科、专 业 机械制造及其自动化


研 究 方 向 机械制造计算机综合自动化

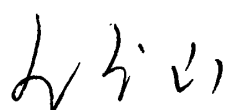
导 师 及 职 称 韩 江 教 授


2011 年 3 月

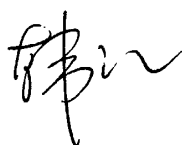
本论文经答辩委员会全体委员审查，确认符合合肥工业大学硕士学位论文质量要求。

答辩委员会签名：（工作单位、职称）

主 席：  合肥工业大学 教授

委 员：  合肥钢铁集团 教授级高工

 合肥工业大学 副教授

导 师： 

独创性声明

本人声明所呈交的学位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。据我所知，除了文中特别加以标志和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得合肥工业大学或其他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谢意。

学位论文作者签字：黄涛 签字日期： 年 月 日

学位论文版权使用授权书

本学位论文作者完全了解 合肥工业大学 有关保留、使用学位论文的规定，有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许论文被查阅或借阅。本人授权 合肥工业大学 可以将学位论文的全部或部分论文内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后适用本授权书）

学位论文者签名：黄涛 导师签名：韩江
签字日期： 年 月 日 签字日期： 年 月 日

学位论文作者毕业后去向：

工作单位：中国南车株洲电力机车研究所

通讯地址：湖南省株洲市

电话：

邮编：412001

基于 CANopen 总线的数控系统高速数字通信技术研究

摘 要

随着数控系统的发展,上位控制器与多路伺服驱动之间的高速通信成为制约其发展的关键因素。与传统数控系统相比,新型数控系统所要求的上位控制器与伺服驱动之间的通信数据量大大增多,通信速度也大大提高,此外数控系统控制精度和实时性要求也日益提高,传统模拟接口已经很难满足现代数控技术的要求。本文研究的目的是将现场总线技术、嵌入式技术和数字伺服技术结合起来,为新型数控系统上位控制器与多路伺服驱动之间的高速通信提出一种解决方案,以满足其数据传输的高速、高可靠性与实时性要求。

本文首先探讨了数控技术、数字伺服技术以及现场总线技术。详细研究了 CAN 总线的工作原理,并在此基础上研究了基于 CAN 的应用层协议 CANopen。提出了适用于数控系统上层控制器与数字伺服之间基于 CANopen 协议的高速通信系统网络架构, CANopen 主站为由 FPGA、MCP2510、MCP2551 组成的硬件结构,从站为多个 IDM640-8EIA 数字伺服驱动器。并在此基础上对主站的硬件电路进行了设计,其中 MCP2551 为 CAN 收发器, MCP2510 为 CAN 控制器,主站应用层芯片在 ALTER 公司 MAX II 系列 EP3C80 型 FPGA 中实现。

在 CANopen 主站应用层芯片设计过程中,采用 VHDL 语言,并采用有限状态机技术进行程序设计,主站应用层芯片共包含 14 个状态,其中有 8 个是常置任务,如主站应用层接口、CAN 报文接收、报文解析分发、PDO 处理、SDO 处理等,6 个非常置任务,如 SPI 接口初始化、NMT 处理和同步报文生成等。主从站之间的数据传输主要以过程数据对象(PDO)和服务数据对象(SDO)的方式进行。

经过整个系统的试验运行和调试,基本实现了基于 CANopen 协议数字伺服的运动控制,验证了基于 CANopen 协议的数控系统上层控制器与数字伺服高速通信系统软、硬件方案的可行性以及各项功能的有效性。最后对全文进行了总结,针对系统在实际应用过程中的问题与不足,为新型数控系统上位控制器与数字伺服通信问题的研究提出了建议。

关键词: CANopen 协议 ; FPGA; 数字伺服电机; 对象字典

Study on Digital Communication with High Speed Based on CANopen in CNC

Abstract

With the development of CNC system, how to realize high-speed communication between the control unit and multiple servo driver is becoming a key factor in motion control field. Compared with the traditional NC system, the data and speed of communication between the control unit and servo driver of the new type CNC system has greatly improved. In addition, the control precision and real-time requirements of new type CNC System are increasing, traditional analog interface can't meet the requirements. The purpose of this dissertation is to combine the fieldbus technology, embedded technology with digital servo technology to satisfy the requirement of new type CNC's high speed communication and real-time data transmission.

Firstly, this dissertation discusses the numerical control technology, digital servo technology, and field bus technology. Besides, the working principle of CANbus and CANopen protocol have been studied. The net frame of high speed communication has been proposed, which is used between the control unit and multiple servo driver. The master of the net is composed of FPGA, MCP2510 and MCP2551, and the slaves are several IDM640-8EIA digital servo drives. On this basis, hardware circuit of the main station has been designed, CAN transceiver is MCP2551, CAN controller is MCP2510, the application layer chip of the master is realized in FPGA with the type of EP3C80.

During the design of the application layer chip, the VHDL language and the FSM(finite-state machine) has been used. Master application layer chip contains 14 states, of which 8 states are runned frequently, such as the interface of master application layer, the receiving CAN message, parsing and distribution of message, processing of PDOs and SDOs; of which 6 states are runned not-frequently, such as the initialization of SPI Interface, processing of NMT and the generation of synchronization messages. The messages transferred between the master and slaves are PDOs and SDOs.

After the actual operation of the net system, the motion control of the digital servo has been basically realized, the feasibility and validity of the high-speed communication system based on CANopen protocol between the control unit and multiple servo driver have been verified. Finally, the whole work is summarized. Some suggestions are made on what needs further study.

Key words: CANopen protocol; FPGA; Digital servomotor; Object dictionary

致 谢

时光匆匆，三年的研究生生活转瞬即逝。我很荣幸能够成为合肥工业大学 CIMS 所的一员，在这里我不仅学习了很多专业知识，而且还锻炼了与人沟通交流的能力，为踏入社会提前做了准备。

值此论文完成之际，我首先要感谢我的导师韩江教授。论文的成稿过程凝聚了韩老师的精心点拨。韩老师渊博的知识、严谨求实的态度以及不断进取、精益求精的精神，使我受益匪浅。韩老师不仅在学术科研上给予了我悉心指导，而且还将学生视为自己孩子一般教育培养，帮助我解决工作和生活中的困惑。其谆谆教诲我一直铭记在心，在此谨向韩老师致以崇高的敬意和由衷的感谢！

还要感谢董伯麟老师，这篇论文完成的整个过程中以及平时的生活和学习中间，一直得到董老师的关心和指导。同时要感谢夏链老师、余道洋老师、何高清和丁志老师，四位老师在平时的点滴中教导我为人处事、科研求知的方法，使我受益良多。在此向四位老师谨表深深的感谢。

还要感谢和我朝夕相处的各位同学——张韬、肖传清、陈党、王国峰，王小明、肖扬、张忠铨、朱念恩、黄海金、刘凌全，李虎，王思国等同学，他们在我学习和生活上给了我许多关心和帮助，与他们的讨论使我得到了很大的进步与提高。

感谢母校合肥工业大学的培养和教育，感谢机械与汽车工程学院的老师们这些年来对我的培养和教育。

特别要感谢一直关心支持我的父母，你们不但养育了我，而且一直支持我的求学生涯，鼓励我向更高的目标前进。在此向我的父母致以崇高的敬意！

最后，感谢所有在我的人生路上曾经支持、鼓励、帮助过我的人，再次谢谢你们！

作者：黄涛

2011年3月

目 录

第一章 绪论.....	1
1.1 选题的背景和意义	1
1.1.1 数控系统的发展历史	1
1.1.2 数控系统的现状与发展趋势	2
1.1.3 数字伺服技术	3
1.1.4 现场总线技术	4
1.1.5 本课题的来源与研究意义	4
1.2 论文主要研究内容及章节安排	5
第二章 基于 CAN 的高层通信协议 CANopen 的研究	7
2.1 CAN 总线技术应用	7
2.1.1 CAN 总线技术发展及应用现状	7
2.1.2 CAN 总线的工作原理	8
2.2 基于 CAN 总线的应用层协议 CANopen	11
2.2.1 CANopen 协议简介	11
2.2.2 CANopen 协议的对象字典	13
2.2.3 CANopen 协议的预定义连接集	13
2.2.4 CANopen 协议的通信模型	14
2.3 IDM640-8EIA 型数字伺服中的 CANopen 协议	16
2.3.1 IDM640-8EIA 型数字伺服支持的通信对象	16
2.3.2 IDM640-8EIA 型数字伺服驱动设备概述 ^[27]	16
2.4 本数控系统中 CANopen 网络总体结构	17
2.5 本章小结	17
第三章 数控系统中 CANopen 主站节点的硬件设计	18
3.1 CAN 收发器	18
3.1.1 MCP2551 的功能框图	18
3.1.2 MCP2551 的功能描述	19
3.2 CAN 控制器	20
3.2.1 MCP2510 的功能概述	20
3.2.2 MCP2510 的报文发送与接收	21
3.2.3 SPI 接口的操作	22
3.3 CANopen 应用层芯片	23
3.4 CANopen 主站节点电气原理图	23
3.5 本章小结	24

第四章 主站应用层芯片总体设计与网络管理	25
4.1 CANopen 主站应用层芯片总体设计与系统状态机的实现	25
4.1.1 CANopen 主站应用层芯片总体设计与任务调度	25
4.1.2 CANopen 主站应用层芯片状态机的实现	26
4.2 CANopen 网络的启动与 NMT 管理	31
4.2.1 CANopen 网络的启动	31
4.2.2 NMT 网络管理	32
4.3 CANopen 网络的同步	34
4.3.1 CAN 网络的同步机理	34
4.3.2 CANopen 网络的同步	35
4.4 本章小结	36
第五章 主站应用层芯片主要功能模块设计与数字伺服运动控制的实现	37
5.1 CAN 控制器的初始化	37
5.1.1 CAN 控制器引脚配置	37
5.1.2 CAN 控制器内部参数设置	38
5.2 CANopen 主站应用层接口模块设计	39
5.3 FPGA 中主站报文的接收与发送	42
5.3.1 应用层芯片与 CAN 控制器之间的收发逻辑	42
5.3.2 报文的接收与处理	43
5.3.3 报文的封装与发送	46
5.4 SDO 的操作	48
5.4.1 SDO 的运作流程	48
5.4.2 SDO 的读写	49
5.5 PDO 的操作	50
5.5.1 PDO 的运作流程	50
5.5.2 PDO 的读写	52
5.6 数字伺服电机的运动控制的实现	52
5.6.1 数字伺服初始参数设定	53
5.6.2 PDO 报文参数设置	54
5.6.3 实验结果分析	55
5.6.4 本章小结	56
第六章 总结与展望	57
参考文献	58

插图清单

图 2-1	CAN 的 ISO/OSI 参考模型与分层结构	8
图 2-2	报文的数据结构帧（标准帧）	10
图 2-3	报文的数据结构帧（扩展帧）	10
图 2-4	报文的错误帧（扩展帧）	10
图 2-5	报文的过载帧	11
图 2-6	CAN、CANopen 标准在 OSI 网络模型中的位置框图	11
图 2-7	CANopen 设备模型	13
图 2-8	预定义连接集的 ID 结构	14
图 2-9	主从式通信模型	15
图 2-10	各户-服务器式通信模型	15
图 2-11	生产者-消费者式通信模型	15
图 2-12	基于 CANopen 协议的数字伺服通信系统架构	17
图 3-1	CANopen 通信模型	18
图 3-2	MCP2551 的功能框图	18
图 3-3	MCP2510 的功能框图	20
图 3-4	MCP2510 的发送缓冲器	21
图 3-5	MCP2510 的发送状态寄存器	22
图 3-6	MCP2510 的接收缓冲器原理图	22
图 3-7	EP3C80 型 FPGA	23
图 3-8	CANopen 主站控制器与收发器电气原理图	24
图 4-1	CANopen 主站任务模块图	26
图 4-2	CANopen 主站状态转移图	28
图 4-3	CANopen 网络启动流程图	31
图 4-4	CANopen 节点最小化 boot-up 状态转移图	32
图 4-5	NMT 节点保护模式图	33
图 4-6	NMT 心跳报文模式图	34
图 4-7	位时间结构	35
图 4-8	同步 PDO 典型传送模式	35
图 4-9	SYNC 触发节点 PDO 发送流程图	36
图 5-1	CAN 控制器初始化流程图	38
图 5-2	CANopen 主站应用层接口模块程序流程图	40
图 5-3	读 MCP2510 报文接收缓冲区	42
图 5-4	写 MCP2510 报文发送缓冲区	43

图 5-5 位修改操作时序.....	43
图 5-6 报文接收过程数据流.....	44
图 5-7 St 3 CAN 报文接收状态流程图	44
图 5-8 报文发送过程数据流.....	46
图 5-9 St 12 CAN 报文发送状态流程图	47
图 5-10 SDO 通信过程.....	48
图 5-11 SDO 操作在主站应用层芯片中的状态机转移图	49
图 5-12 SDO 段传输操作的一般模式.....	50
图 5-13 PDO 通信过程.....	51
图 5-14 PDO 操作在主站应用层芯片中的状态机转移图	51
图 5-15 时间计数器触发或事件触发型 PDO 下载.....	52
图 5-16 远程帧触发型 PDO 上载.....	52
图 5-17 实验系统实物图	53
图 5-18 驱动参数设置.....	53
图 5-19 电机参数设置.....	54
图 5-20 PDO 参数设置.....	55
图 5-21 电机转速图.....	56

表格清单

表 2-1 总线位的数值表示.....	9
表 2-2 映射到 CAL 服务和对象的 COB-ID(11 位 CAN 标识符).....	12
表 2-3 CANopen 子协议模块.....	12
表 2-4 CANopen 对象字典通用结构.....	13
表 2-5 预定义连接集 CAN 标识符分配表	14
表 3-1 MCP2551 引脚	19
表 3-2 MCP2551 的三种工作模式	19
表 3-3 MCP2510 的引脚说明	21
表 3-4 SPI 指令集	23
表 4-1 CANopen 主站状态转移表.....	29
表 4-2 NMT 报文格式.....	32
表 4-3 CS (命令字) 定义.....	33
表 4-4 NMT 节点保护状态字含义.....	33
表 4-5 NMT 心跳报文状态字含义.....	34
表 5-1 滤波/屏蔽寄存器真值表.....	39
表 5-2 错误中断类型.....	39
表 5-3 trans_state 所对应的数据发送状态.....	42
表 5-4 发送缓冲器优先级.....	46
表 5-5 请求帧 (主站-数字伺服)	50
表 5-6 响应帧 (数字伺服-主站)	50

第一章 绪论

1.1 选题的背景和意义

1.1.1 数控系统的发展历史

数字控制(numerical control, NC)是一种自动控制技术, 国家标准(GB 8129-87)定义为“用数字化信号对机床运动及其加工过程进行控制的一种方法^[1]”。而数控技术(numerical control technology)则是指用数字量及字符发出指令并实现自动控制的技术, 它是实现制造业自动化、柔性化和集成化的技术基础^[2]。

数控机床(numerical control machine tools)则是采用了数控技术的机床。国际信息处理联盟(International Federation of Information Processing, IFIP)第五次技术委员会, 对数控机床做了如下定义: 数控机床是一种装了程序控制系统的机床^[3]。该系统将载体上事先给定的数字量进行自动录入, 并对其译码, 然后进行必要的运算和信息处理, 最后控制机床运动实现零件加工。

1952 年世界上第一台试验性三坐标数控铣床在美国麻省理工学院被研制出来。由于早期计算机的运算速度很低, 远不能适应机床实时控制的要求, 所以只能采用数字逻辑电路来“搭”建一台专用计算机作为数控装置, 被称为电子管数控系统, 通常也称第一代数控。1959 年, 随着晶体管电路的出现, 产生了能够在工业中应用以晶体管电路为基础的第二代数控系统。1965 年小规模集成电路应用到 NC 中, 形成了第三代数控系统。这三代数控系统的一个共同特点就是各种控制功能均由硬件逻辑完成, 故又称之为硬件数控系统, 其有很多的缺点, 如功能简单、灵活性差, 系统可靠性低, 成本高。

20 世纪 70-80 年代, 随着大规模、超大规模集成电路, 半导体存储器以及微处理器的产生, 以微处理器为基础的计算机数控系统(CNC)产生了, 数控系统以前由硬件实现的功能逐步改为由软件完成。与硬件数控系统(NC)相比, 计算机数控系统的柔性大大提高, 真正实现了机电一体化; 体积大大的缩小; 生产成本也降低了。这使得数控系统产品逐步实现了标准化和系列化。

20 世纪 80 年代中期到 90 年代中期, 32 位 CPU 在 CNC 中得到应用, 其超强的数据处理能力使 CNC 系统进入了高速、高精度的时代。

进入 20 世纪 90 年代, 随着 PC 性能的快速提高, CPU 的运算位数从 8 位、16 位发展到 32 位, 乃至现在的 64 位, 这使 PC 可以满足作为数控系统核心部件的要求^[4]。PC 机的引入大大加速了数控系统的发展, 为数控系统的发展提供了广阔的平台。

1.1.2 数控系统的现状与发展趋势

在现代制造系统中,数控系统是关键技术,它集计算机、微电子、信息处理、自动检测、自动控制等高新技术于一体,具有高效率、高精度、柔性化等特点。目前,数控系统正在由专用型封闭式开环控制模式向通用型开放式实时动态全闭环控制模式发展,有单通道专用型数控系统向多轴多通道复合加工数控系统方向发展。具体表现在以下几个方面:1、数控系统运用了超薄型、超小型的集成技术;2、数控系统实现了集计算机、模糊控制、神经网络等多学科技术为一体的智能化,能够实现自动故障诊断和智能化故障处理等功能;3、数控系统采用了高速、高精、高效控制技术,在加工过程中能够进行动态补偿^[3]。

在国外高效、高速、高精度、多功能复合加工机床得到了很好的发展。例如 FANUC 公司的 30i/31i/32i 数控系统和 0iMD 数控系统、三菱公司的 M700V 数控系统、西门子公司的 828D 数控系统都具备纳米插补功能,采用以纳米为单位精确计算位置指令,使机床平滑移动并且提高加工精度。同时,使用 HRV 功能提高相应速度和控制精度。例如 0iMD 数控系统的主轴 HRV3 控制功能和 30i/31i/32i 数控系统的 HRV4 控制功能,三菱公司的 SSS 控制功能(超高平滑表面控制),适合于实行高品质的模具加工。为了方便用户调试和使用,广泛使用现代的交流伺服驱动器,能在短时间内实现进给伺服和主轴相关参数的最优化,实现高速、高精的伺服性能,纷纷推出了各种伺服调整工具。例如 FANUC 的 SERVEGULDE,三菱公司的 Servotuningtool: MsConfigurator^[5]。在多功能复合加工方面,德国 DMG 公司、INDEX 公司、奥地利 WFL 公司、日本森精机公司、MAZAK 公司、大隈公司等企业生产的复合加工中心已经逐渐形成系列产品,具有较高质量和国际市场的竞争力。

随着我国加工制造业的飞速发展,用户对于数控机床的功能要求逐渐提高。特别是为了参与国际竞争,通过与国外开展技术合作以及加强自主研发,我国国内企业在中高档数控系统的开发和生产上取得了一些成绩,主要表现在以下几个方面:1、在开放式数控系统方面,武汉华中数控公司、北京航天数控公司、西南自动化所等单位先后开发出了数控系统运动控制器、可编程控制器,还开发出了人机界面的二次开发软件包以及适合不同加工工艺要求的功能组件;2、在伺服系统方面,武汉华中数控公司、北京航天数控公司、广州数控设备厂等一批企业相继开发出交流伺服驱动系统和主轴交流伺服控制系统,完成了 20A-200A 交流伺服系统系列型谱;3、此外,北京机床研究所还开发了针对使用多种控制系统的数控机床集群控制系统,并在车间级数控机床集群控制项目中进行了示范^[3]。

随着计算机技术与微电子技术的发展，数控系统在性能上日益提高。为了满足社会经济与科技发展的需要，数控系统正向着高效，高速，高精度，低能耗，高可靠性，智能化，网络化，多功能复合加工的方向发展。

要实现高效、高精度、多轴多通道复合加工数控系统最主要也是最关键的一个问题就是选用何种伺服执行机构，以及如何实现控制器与伺服执行机构之间的通信，而数字伺服系统和现场总线恰能很好的解决这两个问题。

1.1.3 数字伺服技术

伺服驱动系统是数控机床的重要组成部分，伺服系统的性能在很大程度上决定了数控机床的性能。在早期电动机多采用步进电机和液压扭矩放大器，后来开始采用液压伺服系统、小惯量直流伺服电动机、大惯量直流伺服电动机、交流伺服电动机。随着科学技术的发展，带伺服系统的电动机、电路及检测装置等的技术水平都有极大的提高。近年来出现了数字伺服系统，组成伺服驱动电路的位置、速度和电流控制环节部分实现数字化，甚至可以以单片微机或高速数字信号处理器为硬件基础，进行全数字化控制，这样一来伺服系统可以与 CNC 系统的计算机进行双向通信，故能避免了零点漂移，提高了位置与速度控制的精度和稳定性，与通常的模拟伺服系统相比，数字伺服系统的脉冲当量从 $1\mu\text{m}$ 减小到 $0.1\mu\text{m}$ ，进给速度仍能达到 10m/min 。采用高速和高分辨率的位置检测装置组成半闭环和闭环位置控制系统，增量式位置检测编码达到 10000p/r （脉冲/转），绝对式编码器可以达到 1000000p/r 和 $0.01\mu\text{m}$ 的分辨率，分辨率为 $0.1\mu\text{m}$ 时，位移速度可达 240m/min ，这样便极大地提高了位置控制的精度，即机床的定位精度^[2]。

数字伺服系统的出现，将会为高精度数控机床的发展提供广阔的空间。因为在数字伺服装置内部可以完成所有实际值与指令值的计算与处理，数字伺服装置能很轻松的完成速度环控制和扭矩环控制，这都是传统伺服装置所需要完成的任务，另外数字伺服装置还能高速的实现精插补。无论是在速度控制范围方面还是在速度控制的精确度方面，数字伺服装置都远胜于传统的伺服装置，所以采用数字伺服装置的数控系统具有更高的加工精度与加工速度。此外数字伺服装置还支持适应性控制，这可以增强数控系统的故障诊断功能和自动调整以适应不同工作环境的功能，而且数字伺服系统的硬件简单，系统的复杂性和成本都大大降低了。

随着开放式数控系统和数字伺服技术的发展，如何实现控制单元与数字伺服装置之间的高速通信成为数控领域研究的重要内容之一^[6]。为了满

足不同伺服生产厂家的产品互换,全数字伺服需要有一个开放式接口,现场总线则为这个接口提供了很好的解决方案。

1.1.4 现场总线技术

根据国际电工委员会 IEC(International Electrotechnical Commission)标准和现场总线基金会 FF(Fieldbus Foundation)的定义:现场总线是连接智能现场设备和自动化系统的数字式、双向传输、多分支结构的通信网络^[7]。现场总线技术是控制技术、计算机技术、通讯技术的交叉与集成,几乎涵盖了所有连续、离散工业领域,如过程自动化、制造加工自动化、楼宇自动化、家庭自动化等等^{[8][9]}。具有总线式连接、开放性、数字与智能化、维修方便、成本低、通信具有确定性与实时性以及环境适应性等特点^[10]。

目前世界上存在着大约四十余种现场总线,如 FIP、ERA、ProfiBus、FINT、LONWorks、CAN、FF 等等^{[7][11]}。但是由于很多总线协议不对外开放,各个厂家都开发出成套产品一起出售,成本昂贵,运用起来灵活性较差。而 CAN(Controller Area Network 控制器局域网)总线却是一款完全开放的通讯协议,最早由德国 BOSCH 公司推出,其总线规范已被 ISO 国际标准组织认定为国际标准,得到了 Intel、Motorola、NEC 等公司的支持^[7]。CAN 协议共包含物理层和数据链路层两个层,其报文采用短数据帧差分信号传输,具有较强的抗干扰能力,支持多主站工作方式,采用了非破坏性总线仲裁技术,通过设置优先级可以避免总线冲突^[11]。目前已有多家公司开发了符合 CAN 协议的通信芯片。而 CANopen 则是基于 CAN 总线基础之上高层通信协议,它的物理层和数据链路层与 CAN 总线完全一样,只是在这两个层的基础上新添加了应用层。

1.1.5 本课题的来源与研究意义

本课题来源于国家重大科技专项支持项目(编号:2009ZX04004-021)和安徽省科技创新专项支持项目(编号:2010AHDS0242)。

开放式、高效、高速、高精度、多轴多通道复合加工控制策略是当今世界数控技术研究的热点也是难点。其核心问题就是上层控制器与多路伺服驱动之间的通信问题,这里的通信包括控制信号的传递、状态信号的反馈等。本文的目的就是探讨如何实现上位控制器与伺服驱动之间的高速双向通信,提出了一种基于 CANopen 总线协议的数字智能伺服驱动系统的控制方案。

通过对 CANopen 数字智能伺服驱动进行研究,总结出数字智能伺服

驱动与控制器之间的通信机理,最终建立以 FPGA 为硬件基础的主站和以数字伺服驱动为从站的主从式通信系统,并完成整个运动控制系统中 CANopen 通信部分的软、硬件平台搭建,建立基于 CANopen 协议的通信,完成数据的高速读写,实现数字控制。这改变了传统的伺服控制方式,提高了控制器与数字伺服驱动之间的通信速度,因此本课题对开放式、高速、高精度、多轴多通道复合加工数控机床的研究具有重要的意义。

1.2 论文主要研究内容及章节安排

要实现数控系统上位控制器与多路数字伺服之间的高速通信,必须选择一种高效可行的通信协议,建立安全可靠的通信节点,实现整个网络的软硬件设计,其中硬件设计包括芯片与设备的选用;软件设计包括网络任务的调度与实现,网络的管理与可靠性保证等。针对以上要求,本文对现场总线技术加以研究,最终选定 CANopen 总线作为连接数控系统上位控制器与多路数字伺服之间的高速通信协议,建立以基于 CANopen 总线协议的数字智能伺服为从站以数控系统中 FPGA 加 CAN 控制器与收发器为主站的 CANopen 通信网络,搭建由 FPGA、CAN 控制器(MCP2510)和 CAN 收发器(MCP2551)组成的 CANopen 主站硬件电路,在 FPGA 中建立 CANopen 主站任务调度有限状态机,完成各个任务模块的设计,最终实现数字伺服电机的运动控制。本文各章节安排如下:

第一章、绪论。本章节介绍了数控系统的历史、现状以及发展趋势。介绍了数字伺服技术与现场总线技术。阐述了本课题的来源和研究意义以及论文的主要研究内容。

第二章、基于 CAN 的高层通信协议 CANopen 的研究。本章详细研究了 CAN 总线技术以及基于此总线的高层通信协议 CANopen 协议。重点研究了 CAN 总线的工作原理, CANopen 的通信机制,还介绍了本课题所使用的数字伺服驱动的特性,最后对 CANopen 网络进行了总体结构设计。

第三章、数控系统中 CANopen 主站节点的硬件设计。本章研究了 CAN 控制器和 CAN 收发器的性能特点及其工作原理,完成了主站硬件电路 PCB 板原理图设计。

第四章、主站应用层芯片的 FPGA 实现与 CANopen 网络管理。本章对数控系统中 CANopen 主站应用层芯片做出了总体设计,列出了主站应用层芯片所包含的各个功能模块,以及各个功能模块之间的有限状态机调度的实现方法。研究了 CANopen 网络管理中初始化过程以及网络同步的实现方法。

第五章、主站应用层芯片主要功能模块设计与数字伺服运动控制的实

现。本章着重介绍了 CANopen 主站应用层芯片中主要功能模块的实现方法，如 CAN 控制器初始化模块、应用层芯片接口模块、服务数据对象（SDO）处理流程和过程数据对象（PDO）处理流程。最后对本文所采用的数字智能伺服进行了运动控制试验。

第六章、总结与展望。本章对本课题的相关研究进行了总结，并对进一步的研究提出了建议。

第二章 基于 CAN 的高层通信协议 CANopen 的研究

2.1 CAN 总线技术应用

CAN (Controller Area Net), 即控制器局域网, 是一种支持分布式控制和实时控制的串行通信网络。最早由德国 BOSCH 公司推出, 用于汽车执行部件之间的数据通信和内部测量。其总线规范已被 ISO 国际标准组织认定为国际标准。CAN 总线与其他一般通信总线相比, 具有可靠性高、实时性好等特点。另外 CAN 总线具有基于多主站的灵活的工作方式, 根据报文标识符分配优先级, 采用非破坏性总线仲裁技术, 可实现点对点通信、也可实现一点对多点的全局广播式报文传送接收方式。CAN 总线的技术规范包括 CAN2.0A 和 CAN2.0B 两部分, CAN 协议只有物理层和数据链路层两个底层协议, 规定了对数据通讯的成帧处理, 包括位填充、数据块编码、循环冗余检验等项的工作标准^[11]。

2.1.1 CAN 总线技术发展及应用现状

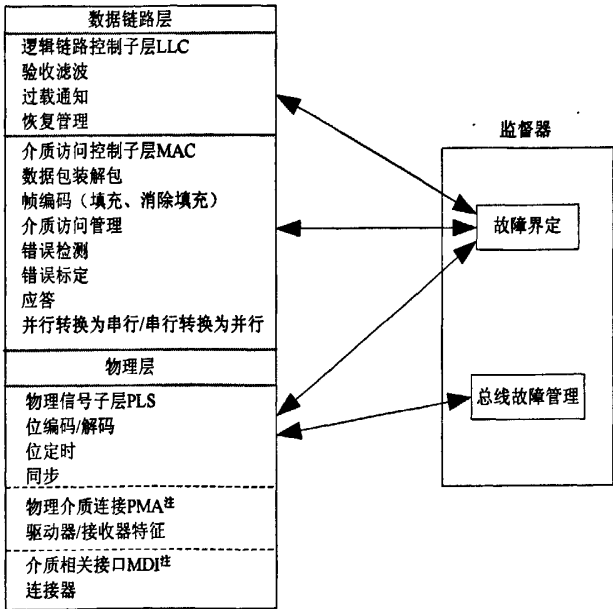
20 世纪 80 年代初, 德国的 BOSCH 公司提出了 CAN (Controller Area Net) 控制器局域网, 这是个多主站无破坏性仲裁机制的网络协议。在 20 世纪 90 年代初 BOSCH CAN 规范 2.0 版被提交作为国际标准, 到 1993 年 11 月正式公布了 CAN 的 ISO-11898 标准, 同年在 Esprit 方案范围内, 由 BOSCH 公司领导的欧洲协会在设计一种 CANopen 的原型, 它是基于 CAL 的层面, 用于生产单元的内部网络。1995 年在国际标准 ISO-11519-2 中规定了 CAN 数据传输的容错方法。同年, 完整的修改后的 CANopen 通信规范被发布, 仅仅在 5 年之内它便成为欧洲最重要的标准化嵌入式网络, CANopen 不仅定义了应用层的通信方案, 还指定了可编程系统、不同设备、接口和应用方案的结构。CAN 规范正处在标准化的过程中, ISO-11898-1 描述了 CAN 数据链路层; ISO-11898-2 定义了非容错 CAN 物理层; ISO-11898-3 规定了容错 CAN 物理层; 国际标准 ISO-11992 和 ISO-11783 都在美国标准 J1939 的基础上定义了基于 CAN 应用的协议, 但都不完整^[7]。

虽然 CAN 协议已经有 20 多年的历史, 但它仍然在不断的改进之中, CAN 在全球市场上也处于起始点。然而 CAN 总线在组网和通信功能上的优点以及它的高性价比, 决定了它在许多领域都有广阔的应用前景和发展潜力^[12]。在大型仪器设备系统中, 由于大型仪器设备系统复杂, 需要对多种信息进行采集、处理、控制、输出等操作, CAN 总线就能很好的解决这一难题, 因为 CAN 总线可以提供广播式的信息通信, 这样一来数据

就可以在各个模块之间进行交换，而且 CAN 具有低成本的总线接口，通信线路少，抗干扰能力强等优点^[13]。在工业控制领域，CAN 可以作为连接现场级设备的总线，与其他总线相比，它具有可靠性好，性价比高的优点。CAN 网络上的任何一个节点都可以作为主站发送具有一定优先级的信息帧，这非常适合实时性要求较高的场所，而且其独特的物理层与数据链路层设计技术，使其在抗干扰性与错误检测方面都具有很好的性能^[14]。德国专业委员会 BIA 和德国安全标准权威 TuV 已经对一些基于 CAN 的安全系统进行了认证，CANopen-Safety 已经获得 BIA 许可。此外，CAN 总线技术还可用于家用电器和智能楼宇以及小区建设中。总之，CAN 的应用前景很广泛，被誉为“最有发展前景”的现场总线之一^{[7][15]}。

2.1.2 CAN 总线的工作原理

CAN 总线在 ISO/OSI 七层参考模型中的分层结构如图 2-1 所示，共包含物理层跟数据链路层两个层次，物理层包含物理信号子层（PLS）、物理介质子层（PMA）和介质相关接口（MDI）。数据链路层又被划分为逻辑链路控制子层和介质访问子层^[7]。



注：CAN2.0B 规范对物理层中的驱动器、收发器、连接器、电缆的形态没有规定，但 ISO11898 和 11519-2 对物理层的 PMA 层及 MDI 层都有定义，内容不同。

图 2-1 CAN 的 ISO/OSI 参考模型与分层结构

在物理层中，物理信号子层 PLS 用来实现位编码/解码、定时与同步等功能；物理介质连接 PMA 规定了总线收发的功能电路并提供了总线故障检测方法；介质相关接口 MDI 规定了物理介质与媒体访问单元之间的

机械和电气接口。CAN总线采用差分电压传送,两条信号线分别为 CAN_H 和 CAN_L , 如表 2-1 所示, 通过两条线之间的电压差来决定传输介质上的“显性”和“隐性”两种互补逻辑状态, 这两种状态分别对应的逻辑值为“0”和“1” [16]。

表 2-1 总线位的数值表示

名称	隐性			显性		
	Min	Nom	Max	Min	Nom	Max
V _{can-H} /V	2.00	2.5	3.0	2.75	3.50	4.50
V _{can-L} /V	2.00	2.5	3.0	0.50	1.50	2.25
V _{diff} /V	-0.5	0	0.05	1.5	2.0	3.0

在数据链路层中, 逻辑链路控制子层 LLC (Logical Link Control) 可以完成验收滤波、过载通知和恢复管理等功能, 其中验收滤波是根据报文标识符确定是否接收此报文; 过载通知是指接收器因内部原因提出延迟接收下一数据帧的要求; 恢复管理是用来重新发送因仲裁失败或者其他错误而没有发送成功的数据帧。介质访问控制子层 MAC (Medium Access Control) 主要是规定传送规则, 包括控制帧结构、执行仲裁、错误检测、出错标定和故障鉴定。在帧结构控制方面主要是实现帧编码, 当发送位流中检测到 5 个数值相同的连续位时, 就在实际发送位流中自动插入一个补码位; 仲裁是发生在两个或者两个以上的单元同时开始传送报文时, 用来确定谁具有总线控制权。

CAN 总线上的信息都是以固定的报文格式发送出去的, 报文共分为数据帧 (Data Frame)、远程帧 (Remote Frame)、错误帧 (Error Frame) 和过载帧 (Overload Frame) [17]。

数据帧携带了从发送节点至接收节点的数据信息, 由 7 个位域组成: 帧起始、仲裁域、控制域、数据域、CRC 域、应答域以及帧结尾, 结构如图 2-2 (标准帧) 和图 2-3 (扩展帧) 所示 [7]。帧起始 (SOF) 标志着数据帧与远程帧的起始位, 只有在总线为空闲状态时才允许站点开始发送信号; 仲裁域中包含了报文的仲裁信息, 标准帧由 12 位组成, 而扩展帧则是由 29 位组成, 各个位域的具体信息已在图中标明, 其中包含远程帧识别位 (RTR); 控制域中包含扩展帧识别位 (IDE), 以及有效数据长度码 (DLC3—DLC0), 用于说明报文中包含的有效数据字节数; 数据域中包含将要发送的数据, 具体长度由控制域中的 DLC 决定; 循环冗余码 (CRC) 域是用来检测报文传输错误的, 转换出的多项式为 $X^{15} + X^{14} + X^{10} + X^8 + X^7 + X^4 + X^3 + 1$, 由硬件完成校验; 应答域, 所有的接收器检查报文传输的一致性, 接收正确的节点做出应答, 不正确的做出标志; 帧结尾是数据帧与远程帧结束时的标志序列界定, 由 7 个隐性位组成 [18]。

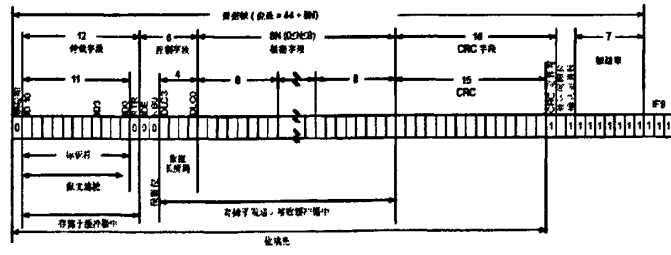


图 2-2 报文的数据结构帧（标准帧）

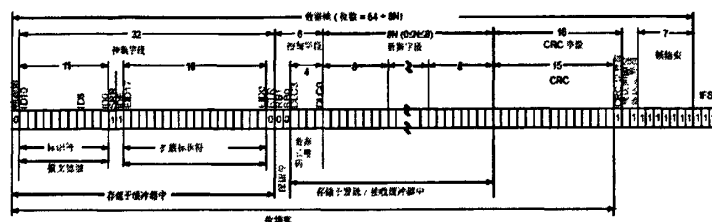


图 2-3 报文的数据结构帧（扩展帧）

远程帧是用来请求源节点送数据的，即远程数据请求。它由六个域组成：帧起始、仲裁域、控制域、CRC 域、应答域和帧结束。与数据帧相比，除了 RTR 位为隐性状态和没有数据域外，其他都与数据帧相同。

错误帧是由任何一个检测到总线错误的节点产生。如图 2-4 所示，错误帧包括错误标志和错误定界两个域。错误标志包括激活错误标志（Active）和认可错误标志（Passivity）两种，激活错误标志由 6 个连续的显性位组成，一旦出现错误标志就打破了位填充规则，其他节点在识别到所形成的位填充错误之后自行产生错误帧，即错误反射标志；而认可错误标志由 6 位连续的隐性位组成，检测到错误条件的站通过发送“认可错误”标志来指示错误，当检测到 6 个连续的相同极性的位之后“认可错误”标志就完成了。错误定界符包括 8 个隐性位，传送完错误标志以后，每个节点开始发送错误定界符，先发一个隐性位并在检测到一个隐性位后，发送其余 7 为隐性位。

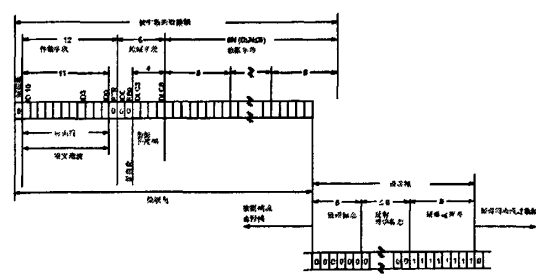


图 2-4 报文的错误帧（扩展帧）

过载帧只能在帧间间隔产生，节点最多可以使用两条连续的过载帧来延迟下一条报文的发送。如图 2-5 所示，过载帧由过载标志和过载定界符组成，过载标志由 6 个显性位构成，过载定界符包含 8 个隐性位，发送方

式与错误帧中的错误定界符相同。

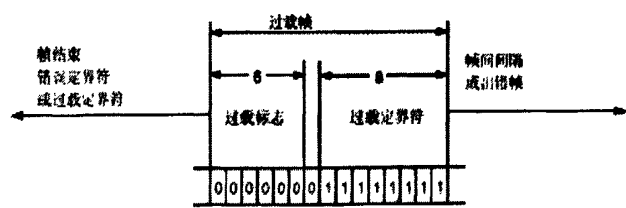


图 2-5 报文的过载帧

2.2 基于 CAN 总线的应用层协议 CANopen

CAN 协议只规定了物理层和数据链路层，所以 CAN 协议能适用于不同的应用对象，但是它不能直接满足工业控制网络的产品互连和组态的要求，在实际工程应用中应用层协议是必须需要的，开发者根据工程需要，自行规定应用层的相关内容，由于没有统一的规范，这种应用层协议与设计者本身的习惯与思路有很大关系。协议内容的不严谨、不统一会使开发人员在每个新项目中都要进行应用层协议设计，重复劳动大，产品互换性差，维护也不方便。这便需要一种统一的技术规范来供大家遵循，而 CANopen 协议则正是这样的一种应用层协议规范，它已经在机械制造、船舶、车辆、铁路、食品加工、制药等领域得到广泛的应用^[19]。

2.2.1 CANopen 协议简介

CANopen 协议是 CAN-in-Automation(CiA)定义的标准之一，它是基于 CAN 的高层协议，如图 2-6 所示，CANopen 协议的物理层与数据链路层与 CAN 协议完全相同，并且在 CAN 的基础上对 CAN 报文中的 11/29 位标识符、8 字节数据进行了定义，另外 CANopen 协议还采用了 CAL 提供的所有的网络管理服务 and 报文传送协议，使用了 CAL 通讯和服务协议子集，并提供了一种分布式控制系统的实现方案^[20]。

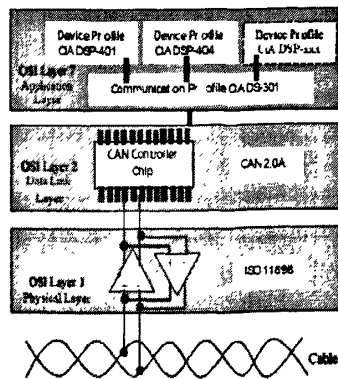


图 2-6 CAN、CANopen 标准在 OSI 网络模型中的位置框图

CAL (CAN Application Layer) 协议是基于 CAN 的高层通讯协议中的一种，最早由 Philips 医疗设备部门制定^[20]。CAL 共提供了 4 种应用层服务功能，分别为 CMS(CAN-based Message Specification)、NMT (Network Management)、DBT (DistriBuTor)、LMT (Layer Management)。其中 CMS 用于实现用户的应用，提供了开放的、面向对象的环境；NMT 提供网络管理服务，包括网络的初始化、节点的启动和停止、失效节点的侦测等；DBT 用于提供动态分配的 COB-ID (Communication Object Identifier)；LMT 提供修改层参数的服务，允许通过一个节点设置另外一个节点的某一层的参数。其中 CMS 为它的消息定义了八个优先级别，如表 2-2 所示，每个优先级拥有 220 个 COB-ID，其范围为 1-1760，其余的 COB-ID (0,1761-2031) 保留给 NMT、DBT 和 LMT^[20]。

表 2- 2 映射到 CAL 服务和对象的 COB-ID(11 位 CAN 标识符)

COB-ID	服务和对象
0	NMT 启动/停止服务
1-220	CNS 对象 优先级0
221-440	CNS 对象 优先级1
441-660	CNS 对象 优先级2
661-880	CNS 对象 优先级3
881-1100	CNS 对象 优先级4
1101-1320	CNS 对象 优先级5
1321-1540	CNS 对象 优先级6
1541-1760	CNS 对象 优先级7
1761-2015	NMT 节点保护
2016-2031	NMT、LMT、DBT 服务

CANopen 是在 CAL 基础上开发的，使用了 CAL 通讯和服务协议子集。但是 CAL 只提供了网络管理服务和报文传送协议，没有规定具体发送的内容是什么，而 CANopen 则定义 CMS 对象的内容或者通讯的对象的类型。CANopen 包含多个子协议模块，用来连接不同的通信器件，如表 2-3 所示。

表 2- 3 CANopen 子协议模块

协议类型	描述的设备
DS-401	数字输入输出设备
DS-402	伺服驱动设备
DS403	操作和显示设备
DS404	传感器和调节器
DS405	可编程控制器
DS406	位置编码器
DS-...	保留

2.2.2 CANopen 协议的对象字典

对象字典（OD: Object Dictionary）是 CANopen 的核心概念，如图 2-7 所示，对象字典是连接通信层和应用层的桥梁，在报文信息的解释和生成过程中起着重要的作用^{[21][22]}。通过它可以确定网络的通讯模式，得知各节点的状态以及报文的具体含义等信息。

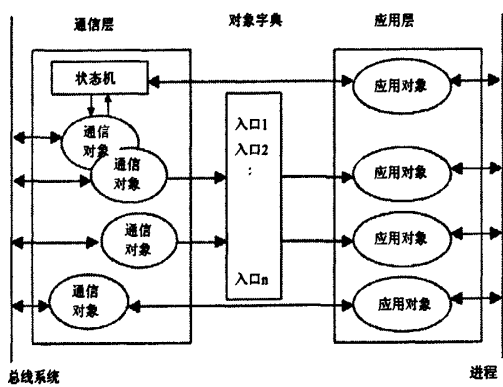


图 2-7 CANopen 设备模型

实质上讲对象字典就是一个有序的对象组，如表 2-4 所示，里面存储着各个网络节点的信息对象，这些信息可以是数据、参数或者是设备的功能等。每个对象可以通过 16 位的索引进行访问。另外，如果某对象包括多个子对象的话，可以通过 8 位的子索引进一步访问子对象，这些子对象中优先级别最高的子对象索引号为 00h^[23]。

表 2- 4 CANopen 对象字典通用结构

索引	对象
0000	未使用
0001-001F	静态数据类型（标准数据类型，如 Boolean, Integer 16）
0020-003F	复杂数据类型（预定义由简单类型组合成的结构如 PDOCommPar, SDOParameter）
0040-005F	制造商规定的复杂数据类型 0060-007F 设备子协议规定的静态数据类型
0080-009F	设备子协议规定的复杂数据类型
00A0-0FFF	保留
1000-1FFF	通信子协议区域（如设备类型，错误寄存器，支持的 PDO 数量）
2000-5FFF	制造商特定子协议区域
6000-9FFF	标准的设备子协议区域(例如“DSP-401 I/O 模块设备子协议”： Read State 8 Input Lines等)
A000-FFFF	保留

2.2.3 CANopen 协议的预定义连接集

对于简单的小型网络，可以使用预定义连接集的方式减小网络组态的工作量，CANopen 协议预先定义了一些强制性的缺省标识符（CAN-ID），其结构如图 2-8 所示，缺省的 CAN-ID 将 11 位基本标识符划分为功能码和节点 ID^[24]。其中 4 位的功能码决定了通信对象的优先权；7 位节点 ID 则对应着网络中的不同节点的地址。这些标识符只有在预操作状态下才是可用的，而且是动态分配的。

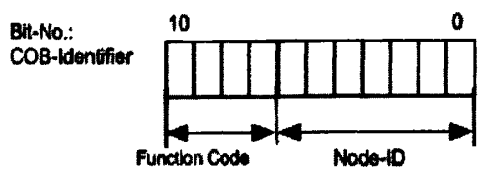


图 2-8 预定义连接集的 ID 结构

预定义连接集可以通过设定节点 ID 为 0 的方式开启广播模式，通过此种方式可以完成网络管理、时间标记、网络同步等功能。此外预定义连接集还定义了一个服务数据对象（包括请求与应答，占用两个 CAN-ID）、4 对过程数据对象（分别为 4 个发送 TPDO 和 4 个接收 RPDO）、1 个节点错误控制对象和 1 个紧急报文对象的标识符^[25]。这些被预定义的标识符如表 2-5 所示，根据总线仲裁规则，他们的功能码值越小，也就是 CAN-ID 的值越小，其对应的优先级就越高，其中优先级最高的为 NMT 模块控制对象报文，它由 CANopen 主站发出，用来进行网络管理，更改节点状态；同步对象（SYNC）报文用于数据同步；时间标记对象（TIME STAMP）由主站发出，对所有从站内部时钟进行同步；紧急报文对象是用于处理网络中的紧急情况，由从节点发出，整个网络中任何具备紧急事件监控与处理能力的节点都参与报文的接收并处理紧急报文；PDO 报文与 SDO 报文将会在接下来的几章中讲到；NMT 错误控制对象作为优先级最低的报文由从节点发出，专为从节点状态检测而设计的，包括节点保护（Node Guarding）和心跳报文（Heartbeat）。

表 2-5 预定义连接集 CAN 标识符分配表

对象类型	对象	功能码(二进制)	COB-ID 通讯参数	在对象字典中的索引
广播类型	NMT 模块控制对象	0000	000h-	
	同步对象(SYNC)	0001	080h	1005h,1006h,1007h
	时间标记对象(TIME STAMP)	0010	100h	1012h,1013h
对等类型	紧急报文对象	0001	081h-0FFh	1014h,1015h
	PDO1(发送)	0011	181h-1FFh	1800h
	PDO1(接收)	0100	201h-27Fh	1400h
	PDO2(发送)	0101	281h-2FFh	1801h
	PDO2(接收)	0110	301h-37Fh	1401h
	PDO3(发送)	0111	381h-3FFh	1802h
	PDO3(接收)	1000	401h-47Fh	1402h
	PDO4(发送)	1001	481h-4FFh	1803h
	PDO4(接收)	1010	501h-5FFh	1403h
	SDO(发送)	1011	581h-5FFh	1200h
	SDO(接收)	1100	601h-67Fh	1200h
	NMT 错误控制对象	1110	701h-77Fh	1016h,1017h

2.2.4 CANopen 协议的通信模型

针对不同的数据对象与控制方式，为了节省报文传输时间，提高 CANopen 网络的通信效率，CANopen 协议规定了三种不同的通信模型，分别为主从式通信模型、客户与服务器式通信模型、生产者消费者式通信模型^{[20][26]}。

主从式通信模型，体现的是网络中节点之间控制与被控制的关系，如图 2-9 所示，网络中的任何一种操作都是由主站发起，主站可以直接向从站发送数据，也可向从站发出远程请求来获取从站中的相关信息。CANopen 协议中的 NMT（网络管理）和 LSS（层设置）都是基于这种通信模型定义的。

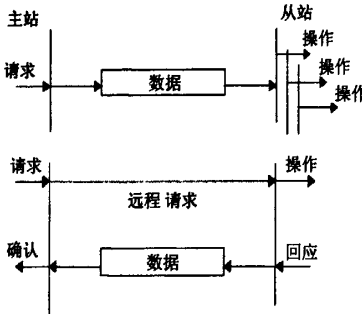


图 2-9 主从式通信模型

客户与服务器式通信模型，是一种一求一答的通信模式，如图 2-10 所示，客户端发出请求，服务器端做出相应的应答，客户在得到应答后做出验证，这种通信方式的可靠性很高。在网络中的任何一个节点都可以作为客户端去获取其他节点的信息。

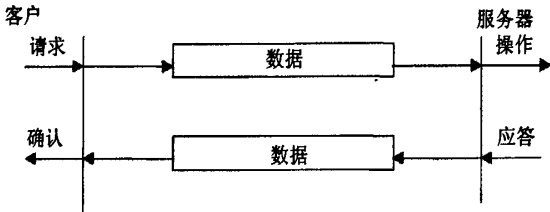


图 2-10 客户-服务器式通信模型

生产者消费者式通信模型，是一种单方向、无证实的服务方式，如图 2-11 所示，生产者只需将自己产生的报文发送到网络上，消费者们根据自己的需要从网络上获取报文信息，而不需要向生产者反馈任何信息，这种通信模式具有很高的传输效率。

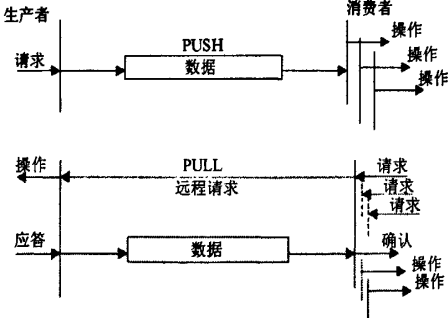


图 2-11 生产者-消费者式通信模型

2.3 IDM640-8EIA 型数字伺服中的 CANopen 协议

本研究所采用的数字智能伺服驱动器型号为 IDM640-8EIA(CANopen 版本), 它是一款基于最新 DSP 技术的全数字智能伺服驱动器, 适用于控制驱动直流无刷电机、交流无刷电机(也称永磁同步)、直流有刷电机、步进电机。可接收正交增量式编码器、绝对式编码器(SSI 接口为交流无刷和直流有刷)作为位置反馈信号接口^[27]。IDM640-8EIA 支持 CiA 拟定的标准 DS-301 v4.02 CANopen 应用层通讯规范, 也支持 CiA 拟定的标准 DSP-402 v2.0 CANopen 设备规范。

2.3.1 IDM640-8EIA 型数字伺服支持的通信对象

IDM640-8EIA 接收和支持 DS-301 v4.02 标准中的通信对象类型包括服务数据对象(SDO)、过程数据对象(PDO)、同步对象(SYNC)、网络管理对象(NMT)和急停对象(EMCY), 不支持时间标记对象(Stamp TIME)^[27]。

1、服务数据对象(SDOs)是用于 CANopen master 接收来自驱动器对象字典中的任何对象, 支持加速与分段 SDO 传输。SDOs 应用于驱动器上电后的参数配置、如设定 PDOs 通信参数和映射参数等。

2、过程数据对象(PDO)用于 CANopen master 与驱动器间的高优先级、实时数据传送, 发送过程数据对象(TPDOs)用于驱动器向主站发送数据, 接收过程数据对象(RPDOs)用于驱动器接收来自主站的数据, IDM640-8EIA 预定义连接集支持 4 个发送 PDO 和 4 个接收 PDO, PDOs 的目录可根据应用需求用动态 PDO-映射参数设置, 此操作也可以在用 SDOs 配置驱动器期间进行。

3、同步对象(SYNC)提供基本的网络时钟, 但是该驱动器不能作为 SYNC 同步发生器产生周期同步信号, 只能响应其他所有 SYNC 用户和发生器的同步信号。

4、网络管理对象(NMT)用于执行 NMT 服务, 通过 NMT 服务, 主站可以对驱动器进行初始化、启动、监控、复位和停止操作。

5、急停对象(EMCY)在驱动器发生内部错误时被触发, 每个“错误事件”所触发的急停对象仅被传送一次, 如果长时间无新错误发生, 驱动器将不再进一步传送急停对象。

2.3.2 IDM640-8EIA 型数字伺服驱动设备概述^[27]

IDM640-8EIA 支持 CiA DSP402 v2.0 操作模式如下:

1、位置曲线模式;

- 2、速度曲线模式；
- 3、回原点模式；
- 4、插补位置模式。除此之外还有几种附加模式，这是由几家厂商指定的模式：外部参考量模式（位置、速度或转矩）；
- 5、电子齿轮位置模式；
- 6、电子凸轮位置模式。

2.4 本数控系统中 CANopen 网络总体结构

本研究的系统架构如图 2-12 所示，在数控系统中嵌入 FPGA 模块，用来完成插补计算，并实现 CANopen 网络的管理与通信。CANopen 网络以 FPGA 中的应用层芯片加 CAN 控制器与收发器为主站，以 IDM640-8EIA 型数字伺服驱动为从站建立的通信网络。CANopen 主站负责整个网络的运行和管理，监视数字伺服驱动上的状态信息，并将数控系统插补器中得出的数据信息传给数字伺服驱动。IDM640-8EIA 型数字伺服驱动内置 CANopen 通信对象字典，使用时需对对象字典中的参数进行配置设定。IDM640-8EIA 型数字智能驱动器可将主站发送来的报文中的有效数字信息提取出来，在驱动器内部对数据进行加工，计算出电机运动的位移、速度、加速度等运动参数，执行相应的运动，以实现数控系统的运动控制。

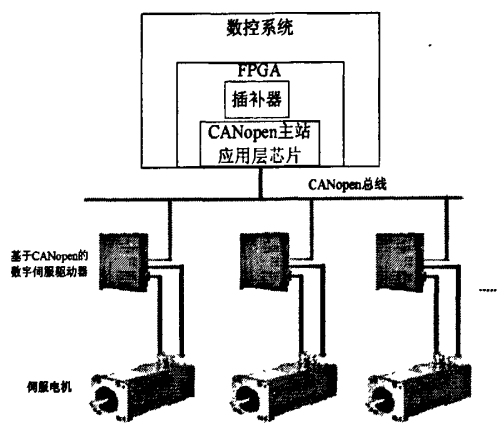


图 2- 12 基于 CANopen 协议的数字伺服通信系统架构

2.5 本章小结

本章对基于 CAN 的应用层协议 CANopen 进行了研究，阐述了 CANopen 协议的通信机理；介绍了本研究所用的 IDM640-8EIA（支持 CANopen 协议）型智能数字伺服驱动；最后提出了本研究的系统架构。

第三章 数控系统中 CANopen 主站节点的硬件设计

CANopen 主站节点主要由 CAN 收发器、CAN 控制器和应用层控制芯片组成,CANopen 的通信模型如图 3-1 所示,CAN 收发器对应着 CANopen 协议的物理层,主要担任网络电平处理,硬件接口定义等功能;CAN 控制器对应着 CANopen 的数据链路层,总线的仲裁、错误校验以及位填充等功能都在此控制器中完成;CANopen 应用层控制芯片中实现报文的处理,对象字典访问和应用层控制算法的调度等功能。

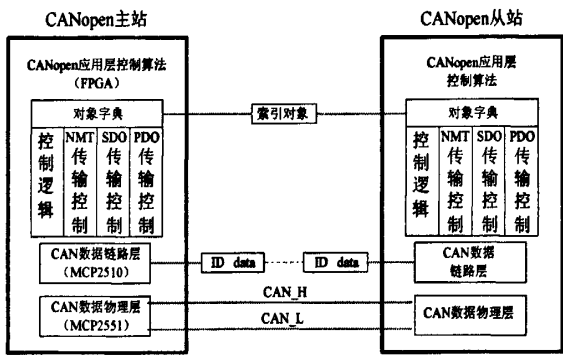


图 3-1 CANopen 通信模型

3.1 CAN 收发器

CAN 收发器的主要功能是将控制器生成的数字信号转化成为适合总线传输的 (差分输出) 信号。本研究采用的收发器为美国微芯公司的 MCP2551 芯片,它采用 ISO-11898 标准物理层标准,支持外部斜率控制,具有欠电压保护和热关断等安全保护性能,最多支持 112 个节点连接^[28]。

3.1.1 MCP2551 的功能框图

MCP2551 的功能框图如图 3-2 所示,主要包含驱动控制器、接收器、发送器、显性检测以及热关断保护电路等功能部分^[28]。

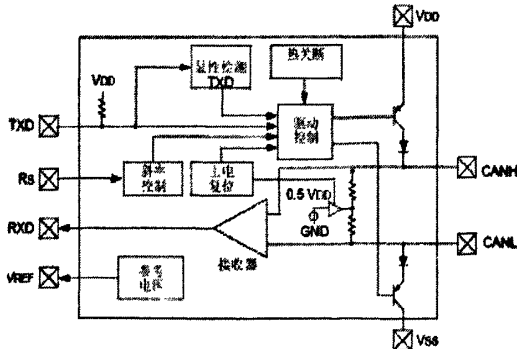


图 3-2 MCP2551 的功能框图

3.1.2 MCP2551 的功能描述

MCP2551 具有自我保护功能，当节点温度超过标定值 165°C 的时，其热关断电路就促使输出驱动器停止工作，如此器件就能免受过多负载电流的影响，这一保护措施对于由短路引起的总线损坏是必需的。此外，MCP2551 还具有稳定显性检测功能，如果 MCP2551 检测到有持续低电平输入，它将停止 CANH 和 CANL 之间的数据输出，以避免 CAN 总线上数据混乱。

MCP2551 的引脚如表 3-1 所示，其中 RXD 为输出引脚，通常与 CAN 控制器的输入相连，反映的是 CANH 和 CANL 之间的电压值，当二者之间的电压高于 1.2V 时总线为显性，当二者之间电压低于 0V 时总线为隐性^[28]。TXD 为发送数据输入引脚，通常与 CAN 控制器的输出相连，当 TXD 为低电平时，CANL 和 CANH 之间的显示为隐性；反之则为显性。

表 3- 1 MCP2551 引脚

引脚编号	引脚名称	引脚功能
1	TXD	发送器数据输入
2	VSS	接地
3	VDD	提供电压
4	RXD	接收器数据输出
5	VREF	参考输出电压
6	CANL_CAN	低电压 I/O
7	CANH_CAN	高电压 I/O
8	RS	斜率控制输入

RS 引脚可选择三种操作模式，分别为高速模式、斜率控制模式和待机模式，各个模式所对应的引脚电气参数如表 3-2。高速模式可以将 RS 引脚与 VSS 引脚相连来实现，在这个模式下，发送器的输出驱动具有快速的输出上升和下降时间，可以满足 CAN 总线的高速要求；斜率控制模式可以通过改变 RS 和 VOL（通常接地）之间的外接电阻（REXT）来实现不同的斜率要求，REXT 阻值的不同其对应的转换率也不同；如果把 RS 与高电平相连，器件就被置为休眠模式，在此式下，CAN 收发器的发送器将被关闭，控制器侧的接收引脚（RXD）仍然可以工作，但是工作在低速率状态^[28]。

表 3- 2 MCP2551 的三种工作模式

模式	Rs 引脚电流	RS 引脚上的电压
待机	-IRS < 10 μ A	VRS > 0.75 VDD
斜率控制	10 μ A < -IRS < 200 μ A	0.4 VDD < VRS < 0.6 VDD
高速	-IRS < 610 μ A	0 < VRS < 0.3VDD

3.2 CAN 控制器

CAN 控制器中固化了数据链路层协议，其功能为处理总线上所有报文的发送和接收。本文采用的 CAN 控制器为美国微芯公司的 MCP2510 芯片，它是一款控制器局域网络(CAN) 协议控制器,完全支持 CAN 总线 V2.0A/B 技术规范。能够发送和接收标准帧或扩展帧。带有高速的 SPI 接口,可与上位机实现高速通信,共包含三个发送缓冲区和两个接收缓冲区,报文在进入接收缓冲区之前需要通过六个完全验收滤波器和两个完全验收屏蔽滤波器所组成的屏蔽码的检测。在调试过程中支持环回模式,以便用户进行自检^[29]。

3.2.1 MCP2510 的功能概述

如图 3-3 所示，MCP2510 共由四大功能模块组成，分别为 CAN 协议引擎、控制逻辑与算法、SRAM 寄存器、SPI 协议模块^[29]。总线上报文的接收与发送都是由 CAN 协议引擎来进行处理，通过将发送使能引脚电平拉低或者由 SPI 接口修改发送控制寄存器中的数据都可以触发已经装载在发送缓冲器中数据报文的发送。CAN 控制器时刻监测着总线上的报文，一旦发现错误报文，便启动错误处理方式，若没有错误报文，则时刻将总线上报文的 ID 与自己的滤波器进行匹配，将符合要求的报文接收到接收缓冲器中，以供上位机读取。

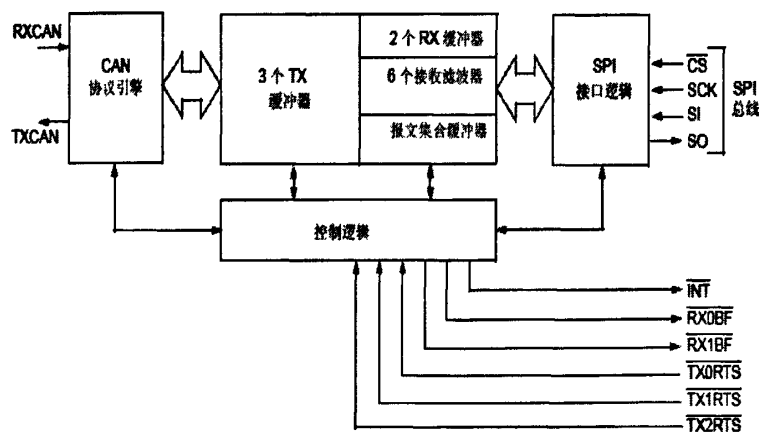


图 3-3 MCP2510 的功能框图

CANopen 应用层芯片与 CAN 控制器的信息交换是通过高速 SPI 接口实现的。CAN 控制器的 INT 引脚可以配置成所有类型中断的通用引脚，用以报告 CAN 控制器内部的中断信息,此外 CAN 控制器还有一些专用的中断引脚,如 RX0BF、RX1BF 等,这样用户就可以根据自己的需要灵活的指定中断指示形式。存储在三个发送缓冲器中的报文的触发可以通过

拉低 TX0RTS、 TX1RTS 和 TX2RTS 引脚电平来触发，也可以通过修改控制寄存器中的数据来触发。

表 3- 3 MCP2510 的引脚说明

名称	SOIC引脚	I/O/P类型	说明
TXCAN	1	O	连接到CAN发送器
RXCAN	2	I	连接到CAN控制器
CLKOUT	3	O	带可编程预分频器的时钟输出引脚
TX0RTS	4	I	发送缓冲器TXB0请求发送或通用数字输入引脚
TX1RTS	5	I	发送缓冲器TXB1请求发送或通用数字输入引脚
TX2RTS	6	I	发送缓冲器TXB2请求发送或通用数字输入引脚
OSC2	7	O	振荡器输出
OSC1	8	I	振荡器输入
VSS	9	P	逻辑和I/O引脚的参考地端
RX1BF	10	O	接收缓冲器RXB1的中断引脚或通用数字输出引脚
RX0BF	11	O	接收缓冲器RXB0的中断引脚或通用数字输出引脚
INT	12	O	中断输出引脚
SCK	13	I	SPI接口时钟输入引脚
SI	14	I	SPI接口数据输入引脚
SO	15	O	SPI接口数据输出引脚
CS	16	I	SPI接口片选输入引脚
RESET	17	I	低电平有效器件复位输入引脚
VDD	18	P	逻辑和I/O引脚的正电源
NC	-	-	无内部连接

3.2.2 MCP2510 的报文发送与接收

MCP2510 共包含三个占据 14 字节的 SRAM 发送缓冲器。如图 3-4 所示，与报文发送缓冲器相关的控制寄存器存放在第一字节中。通过设定该字节可以设定报文的发送条件，而且报文的发送状态也存储在该字节中。报文的仲裁信息和有效的报文数据占据其后的十三个字节。

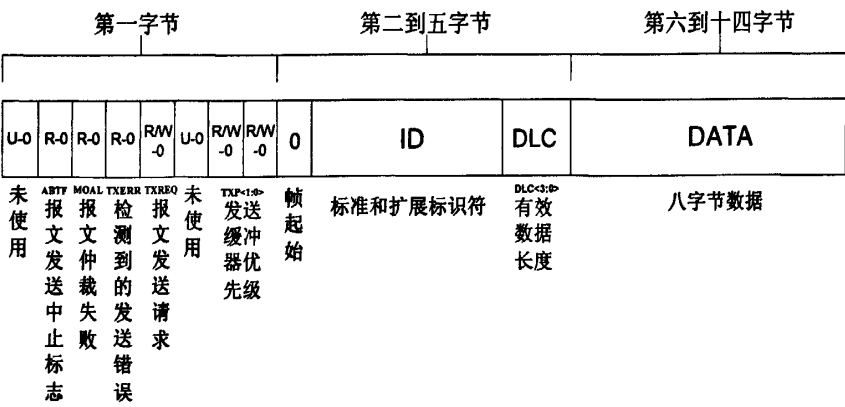


图 3- 4 MCP2510 的发送缓冲器

发送报文时只需通过 SPI 接口对第一字节中的控制寄存器进行相应的操作即可，报文发送的状态都在如图 3-5 所示的状态寄存器中，可以通过读取该状态寄存器来了解当前报文发送的情况。此外，还可以通过操作 TXNRTS 引脚来控制报文的发送，此操作需要事先对 TXRTSCTRL 寄存器进行设定。

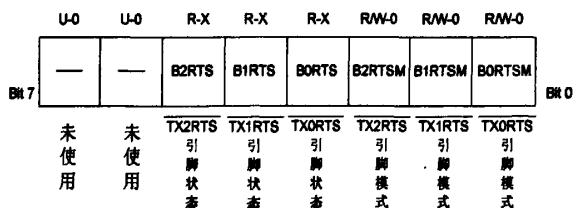


图 3- 5 MCP2510 的发送状态寄存器

MCP2510 具有 RXB0 和 RXB1 两个接收缓冲器，其中 RXB0 具有较高优先级，并配置有 2 个报文验收滤波寄存器。RXB1 优先级较低，配置有 4 个验收滤波器寄存器。此外，MCP2510 还包含一个报文集成缓冲器 (MAB)，如图 3-6 所示，无论如何 CAN 总线上的报文总是能被存储到报文集成缓冲器之中并进行错误检测。如果报文集成缓冲器之中的报文能通过验收滤波器的筛选，则该报文就会被存储到 RXB0 或 RXB1 中空闲的一个缓冲器上，如此，报文就被成功接收到该节点。

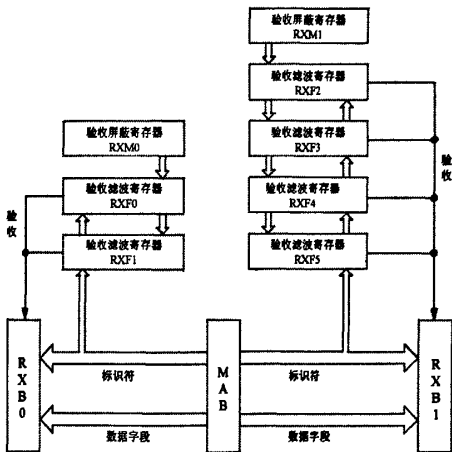


图 3- 6 MCP2510 的接收缓冲器原理图

当 RXB0 或 RXB1 中有新接收到的报文时,CAN 控制器就会通过 INT 引脚或者 RXB₀BF 和 RXB₁BF 引脚（被配置为终端引脚模式）向上位机发出中断。上位机可以通过 SPI 接口或 RXB₀BF 和 RXB₁BF 引脚（被配置为数据输出引脚模式）对接收缓冲区中的报文进行读操作。

3.2.3 SPI 接口的操作

MCP2510 可与微控制器的串行外设接口（SPI）直接相连，支持 0, 0 和 1, 1 运行模式。外部数据和命令通过 SI 引脚传送到器件中，而数据在 SCK 时钟信号的上升沿传送进去。MCP2510 在 SCK 下降沿通过 SO 引脚发送表 3-4 列出了所有操作的指令字节，RTS（发送请求）指令中后三位 nnn 对应着 TXB2、TXB1 和 TXB0 的发送请求。

表 3- 4 SPI 指令集

指令名称	指令格式	功能介绍
复位	11000000	将内部寄存器复位为缺省状态，并将器件设定为配置模式
读	0000 0011	从以指定地址起始的寄存器读取数据
写	0000 0010	向以指定地址起始的寄存器写入数据
RTS（发送请求）	1000 0nnn	设置TXBnCTRL.TXREQ位以启动一个或多个发送缓冲器的报文发送
状态读	1010 0000	读取MCP2510的状态（包括发送接收中断标志位和各请求发送位）
位修改	0000 0101	对指定寄存器进行位修改

3.3 CANopen 应用层芯片

本文所采用的 CANopen 应用层芯片是在 FPGA（Field Programmable Gate Array）中实现的。FPGA 是在 PAL、GAL 等逻辑器件的基础上发展起来的一种特殊的 ASIC 芯片，具有丰富的逻辑资源与 I/O 引脚。此外，FPGA 还具有规模大、体积小、可靠性高、计算速度快等优点^{[30][31]}。

生产 FPGA 的厂商很多，其中最具代表性的是 Altera 公司和 Xilinx 公司。它们的产品广泛而全面，可以满足不同设计层次的需求，而且具有非常完备的开发工具。如图 3-7 所示，本文采用的是 Altera 公司的 Cyclone III 系列 EP3C80 型 FPGA，数控系统的插补计算模块也可设计在该 FPGA 中，由于 CANopen 网络所要传输的正是数控系统插补器的计算结果，所以 CANopen 应用层模块就可以与插补计算模块共用数据存储器，用来存放插补运算的结果，并等待 CANopen 网络将其发送到各个从节点。

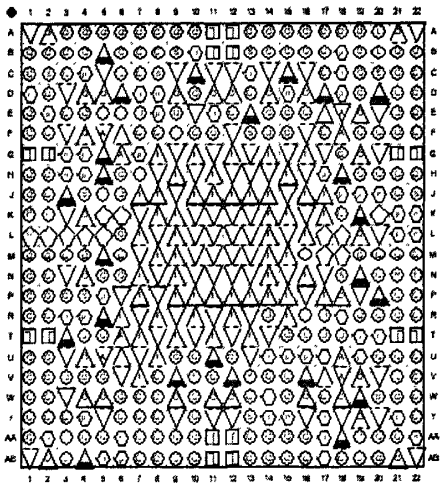


图 3- 7 EP3C80 型 FPGA

3.4 CANopen 主站节点电气原理图

CANopen 主站节点硬件电路主要由以下几部分组成：

- 1、CAN 控制器与应用层芯片接口；
- 2、CAN 控制器时钟；

3、CAN 收发器工作模式跳线:

4、CAN 总线物理接口。

图 3-8 为 CANopen 主站节点电气原理图, 其中只截取出了 CAN 控制器与 CAN 收发器部分。CAN 控制器所有与应用层芯片的接口都连接到 FPGA 上去, 方便使用多种控制模式对 CAN 控制器加以操作。此外, 还为 CAN 控制器设计了振荡器输入, 通过 HEAD3 可以进行跳线操作^[32]。

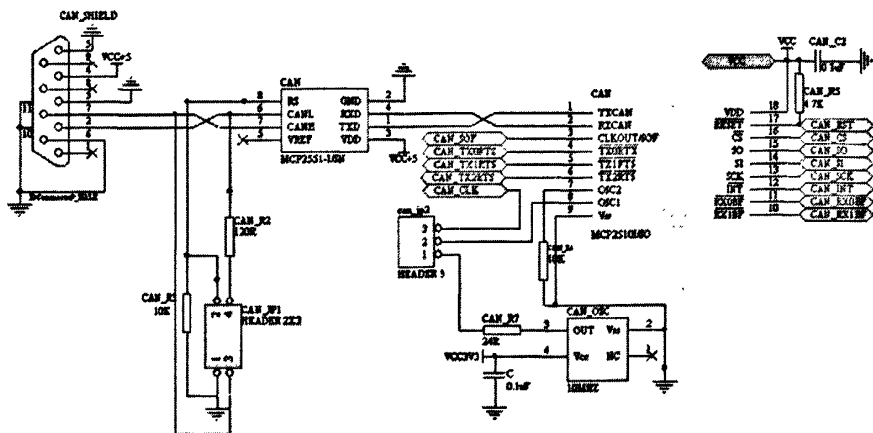


图 3-8 CANopen 主站控制器与收发器电气原理图

对于 CAN 收发器, 其 RXD 与 TXD 分别接到控制器的 RXCAN 与 TXCAN 引脚, CANH 与 CANL 通过 9 针 D 形接头连入 CAN 网络, 若作为网络的一个终端节点, 还需接入一个 120Ω 的终端电阻。RS 引脚可用于 CAN 收发器工作模式设定。

3.5 本章小结

本章对 CAN 控制器与 CAN 收发器进行研究,分析了它们的功能与控制方式;对基于 FPGA 的 CANopen 应用层芯片引脚加以设计;定义了 CAN 控制器与 FPGA 的接口;最后设计出 CANopen 主站节点的硬件电路原理图。

第四章 主站应用层芯片总体设计与网络管理

由第三章 CANopen 主站硬件结构可知, 主站中最复杂最难实现的部分就是应用层芯片的设计。本研究应用层芯片是在 FPGA 中实现, 主要用于实现整个网络的任务调度与运行管理。其中任务调度就是各种通信对象之间的协调以及应用层芯片内部各个任务模块之间的逻辑关系处理等; 网络管理包括网络的初始化, 网络的容错处理以及网络的同步处理等, 这些都是保证网络可靠性与系统安全所必须需要的。

4.1 CANopen 主站应用层芯片总体设计与系统状态机的实现

CANopen 主站应用层芯片的主要功能有网络管理 (NMT)、从节点状态监视与控制、服务数据对象 (SDO) 处理, 过程数据对象 (PDO) 处理, 伺服电机运动控制参数的封装与发送等, 所需要完成的任务很多, 芯片内部功能结构复杂^{[33][34]}。

CANopen 协议定义了最基本的定时规范, 通信周期是以 μs 进行测量和计算的, 这就要求应用层芯片必须具备很高的实时性, 很短的循环周期。当 SPI 接口满负荷工作在 5M/s 时, 所需要的位处理周期是 200ns。

使用 FPGA 作为主站应用层芯片的硬件, 并使用有限状态机进行程序设计就能很好的解决上述问题。首先 FPGA 进行逻辑运算的速度很高; 其次, 采用有限状态机能很好的解决芯片内部功能结构复杂的问题。

4.1.1 CANopen 主站应用层芯片总体设计与任务调度

如何在主站中建立一个高效的任务调度机制是主站设计的难点, 这个调度机制需要适时合理的对各种通信对象加以处理, 产生相应的响应, 并将响应数据和需要传输的控制数据进行报文封装, 然后发送出去。

如图 4-1 所示, CANopen 主站应用层芯片包含 12 个任务状态, 其中主站应用层接口、CAN 报文接收、报文解析分发、PDO 处理、SDO 处理、新报文数据生成、报文识别封装和 CAN 报文发送为 8 个常置任务, SPI 接口初始化、NMT 处理、同步报文生成和错误处理只有在网络初始化、网络同步过程中以及网络产生错误时才启动, 属于非常置任务^[35]。

CANopen 应用层接口是 CANopen 主站连接 CAN 控制器的接口任务, 负责监视 CAN 控制器的中断信息与运行状态, 从而去启动应用层芯片内部的其他任务。报文接收任务启动后, 通过操作 SPI 接口将接收到的报文存储在接收报文缓冲区中, 然后根据 COB-ID 区分报文类型, 此操作需要参照对象字典。区分过的报文被分别送往各自的报文处理模块进行处理,

并产生响应数据，而后按照报文格式进行封装，并送入发送报文缓冲区，最后通过 CAN 报文发送模块进行发送。

作为 CANopen 主站，它还担任着网络初始化与网络参数配置的任务，网络初始化是通过 NMT 报文实现的，在初始化时，新报文数据生成任务启动，并产生 NMT 报文，对整个网络进行配置；在本文的应用中，从站伺服运动控制所需的参数也是经新报文数据生成任务进行创建并加载到 PDO 中的。

同步报文是用来对网络进行同步管理的，由同步报文生成任务创建，在 4.3 中会重点介绍。

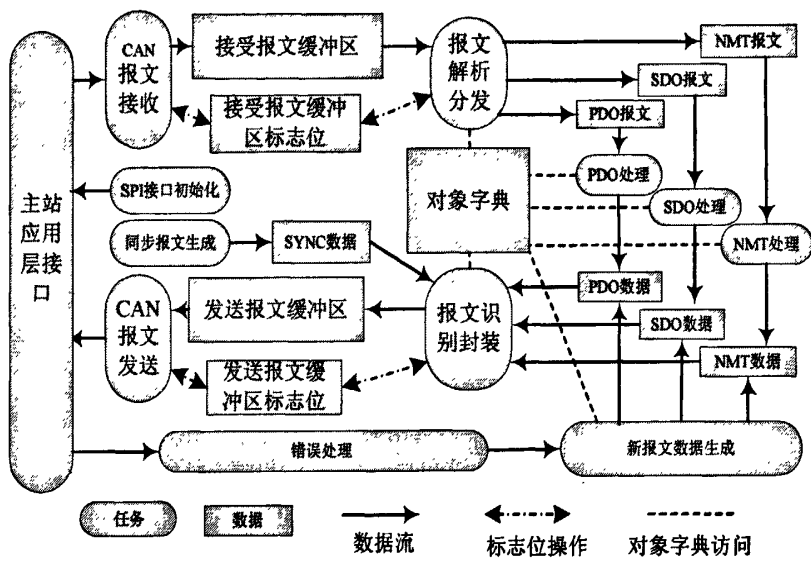


图 4-1 CANopen 主站任务模块图

4.1.2 CANopen 主站应用层芯片状态机的实现

有限状态机(Finite-State-Machine,简称 FSM)在数字集成电路中是指输出取决于过去输入部分和当前输入部分的时序逻辑电路，它将一个时序逻辑抽象成一个同步有限状态机，有限个状态以及在这些状态之间的转移和动作等行为的数学模型就是状态机的调度^{[36][37]}。

有限状态机及其设计技术是数字系统设计中的重要组成部分，状态机是实现高效率、高可靠性逻辑控制的重要途径^[37]。相比其他设计方案或者可以完成相似功能的 CPU，有限状态机设计技术都有着无可比拟的优越性，主要表现在一下几个方面^[38]：

- 1、状态机可以克服纯硬件数字系统顺序控制方式不灵活的缺点。
- 2、状态机的结构模式简单，设计方案比较固定。
- 3、使用状态机可以构成性能良好的同步时序逻辑模块，可以很好的

解决大规模逻辑电路设计过程中的竞争冒险现象。

4、使用状态机的 VHDL 程序层次分明，结构清晰，易读易改，可移植性好。

5、一个使用状态机设计的结构体可以包含多个状态机，而每个状态机都可以包含多个进程，每个单独的状态机（或多个并行运行的状态机）以顺序方式所能完成的运算和控制方面的工作与一个 CPU 类似。所以，在高速运算和控制方面，状态机设计方式有着无可比拟的优越性。

6、状态机能解决 CPU 不可能实现圆满的容错的缺点，具有很高的可靠性。

VHDL 的英文全名是 VHSIC (Very High Speed Integrated Circuit) Hardware Description Language(超高速集成电路硬件描述语言)，诞生于 1982 年美国国防部提出的超高速集成电路计划，其目的是为了在各个承担国防部订货的集成电路厂商之间建立一个统一的设计数据和文档交换格式，1987 年底被 IEEE 和美国国防部确认为标准硬件描述语言^[36]。

在 VHDL 语言中，状态机设计有多种形式，从输出方式上分为 Mealy 型和 Moore 型状态机。其中 Mealy 型状态机属于同步输出状态机，其输出是在输入变化之后立即发生，不依赖于时钟同步，其速度很高，但是在状态译码比较复杂时很容易在输出端产生大量毛刺；而 Moore 型状态机属于一步输出状态机，在接收到上一状态输入之后，当前状态的变化还必须等待下一时钟到来之后才发生变化，因此 Moore 型状态机的输出与时钟同步，具有很好的稳定性^[37]。本研究中所需要的应用层芯片的运算速度并不是很高，再考虑到系统的稳定性，故选用了 Moore 型状态机作为程序设计的基础。

为了便于程序设计，将新报文数据生成任务进行细化，分为 Sync 生成、PDO 生成、SDO 生成和 NMT 生成。这样一来 CANopen 主站应用层芯片共包含 14 种状态。如图 3-10 所示，当主站上电以后，首先需要对 CAN 控制器与 CANopen 网络进行初始化，包括 SPI 接口初始化，引脚功能配置，CAN 控制器工作模式设定等。而后状态机跳转到主站应用层接口任务状态，此任务状态的控制逻辑最为复杂，上接四个状态输入，下启五个输出状态。该状态对 CAN 控制器进行监控，根据 CAN 控制器的中断类型，进行相应的操作，如读到接收报文中断时，便启动 CAN 报文接收状态；若长期无报文接收则新 PDO 生成状态进行数据发送；若满足其他要求则进入其他相应状态。

当第一次进入 st1 时，需跳入新 Sync 生成状态以产生同步报文，使整个网络进行同步。第三个状态为报文接收，在该状态被启动以后，CAN

控制器接收缓冲区的报文被取出存放在 FPGA 的接收缓冲区中,同时 CAN 控制器的相关状态寄存器将被复位。接下来是报文解析分发状态,在此模块根据接收报文的 COB-ID 将报文分类,并相应地启动 PDO 处理、SDO 处理和 NMT 处理三个状态中相应的状态。随后报文将在此三个处理器中进行处理。新 PDO 生成状态模块用于产生新的 PDO 数据,数控系统插补器计算的数据结构都是由此模块发送给相应的轴节点。新 SDO 生成状态模块是进行对象字典读取与改写的场所。新 NMT 生成状态模块用于网络管理指令的下达。报文封装状态模块是将各个状态产生的报文按照 CANopen 协议进行封装。封装好之后的报文需通过 CAN 报文发送状态模块进行发送。主站应用层接口在接收到错误中断时,便启动错误处理,经过分析之后若需要通过 NMT 报文进行管理的则跳至新 NMT 报文生成状态以产生网络管理报文,若无需响应则直接返回主站应用层芯片接口进行其他处理。

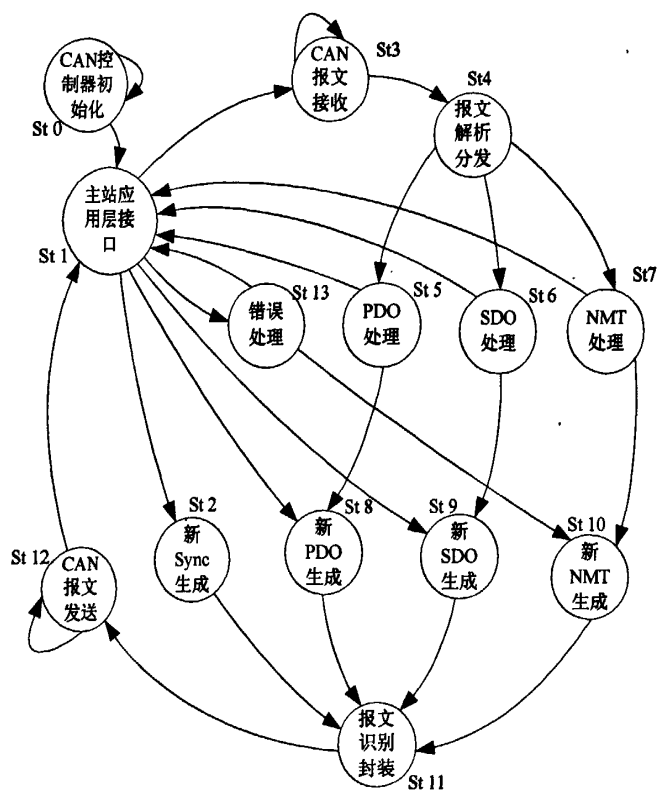


图 4-2 CANopen 主站状态转移图

CANopen 应用层芯片内部各个功能模块之间的状态转移条件是非常复杂的逻辑关系,如表 4-1 所示,本状态机共包含 14 种状态,28 种状态转移条件,St 1 是内部各个状态转移的起点也是终点,各个种循环流程以此作为始终节点。

表 4- 1 CANopen 主站状态转移表

转移代号	前一状态	后一状态	转移条件
1	St 0	St 0	CAN 控制器初始化失败
2	St 0	St 1	CAN 控制器初始化完成
3	St 1	St 2	第一次进入 St 0
4	St 2	St 11	Sync 生成成功
5	St 0	St 3	接收报文中断
6	St 3	St 3	CAN 报文接收失败
7	St 3	St 4	CAN 报文接收成功
8	St 4	St 5	解析到 PDO 报文
9	St 4	St 6	解析到 SDO 报文
10	St 4	St 7	解析到 NMT 报文
11	St 5	St 8	PDO 报文处理过后, 需要产生新 PDO 报文
12	St 8	St 11	新 PDO 报文生成完成
13	St 6	St 9	SDO 报文处理过后, 需要产生新 SDO 报文
14	St 9	St 11	新 SDO 报文生成完成
15	St 7	St 10	NMT 报文处理过后, 需要产生新 NMT 报文
16	St 10	St 11	新 NMT 报文生成完成
17	St 11	St 12	报文封装成功
18	St 12	St 12	报文发送不成功
19	St 12	St 1	报文发送成功
20	St 1	St 8	主站点产生原始 PDO 报文数据
21	St 1	St 9	主站点产生原始 SDO 报文数据
22	St 1	St 10	主站点产生原始 NMT 报文数据
23	St 1	St 13	应用层接口收到错误中断
24	St 5	St 1	PDO 报文处理完毕, 无需产生应答报文
25	St 6	St 1	SDO 报文处理完毕, 无需产生应答报文
26	St 7	St 1	NMT 报文处理完毕, 无需产生应答报文
27	St 13	St 1	错误中断处理完成需要跳至主站应用层芯片
28	St 13	St 10	错误中断处理完成需要 NMT 报文对错误进行网络处理

整体程序的顶层控制程序片段如下所示，程序列出了所调用的函数库，并对 CANopen 应用层芯片的引脚进行了定义。在 FPGA 中，各个功能模块都是作为有限状态机中的一个状态而存在，如下面一段 VHDL 程序代码所示，先定义一个枚举数据类型，其元素为各个状态的名称。状态变量被定义为信号，以便于状态信息的传递。状态机共包括时序进程、组合进程以及一些辅助进程，时序进程和组合进程部分代码如下所示，辅助进程将在第五章中加以介绍。

```
LIBRARY IEEE;                                --调用库
USE IEEE.STD_LOGIC_1164.ALL;
```



```

USE IEEE.STD_LOGIC_UNSIGNED.ALL;
LIBRARY SIMPRIM;
USE SIMPRIM.VCOMPONENTS.ALL;
USE SIMPRIM.VPACKAGE.ALL;
ENTITY CANopen IS PORT          --定义 CANopen 应用层芯片的引脚
    (CAN_SO: IN STD_LOGIC;
     INT: IN STD_LOGIC;
     CAN_RX0BF, CAN_RX1BF: IN STD_LOGIC;
     CAN_CS: OUT STD_LOGIC;
     CAN_TX0RTS, CAN_TX1RTS, CAN_TX2RTS: OUT STD_LOGIC;
     CAN_SOF: OUT STD_LOGIC;
     CAN_RST: OUT STD_LOGIC;
     CAN_SI: OUT STD_LOGIC;
     CAN_SCK: OUT STD_LOGIC);
END CANopen;
ARCHITECTURE behave OF CANopen IS
    TYPE FSM_ST IS(st0, st1, st2, st3, st4, st5, st6, st7, st8, st9, st10,
st11, st12, st13, );          --定义 FSM_ST 数据类型和状态名
    SIGNAL current_state, next_state: FSM_ST;    --定义状态变量的数据类型为 FSM_ST
BEGIN
    PROCESS(rest,clk)          --时序进程
    BEGIN
        IF reset= '1' THEN present_state<=s0;
        ELSIF clk= '1' AND clk'EVENT THEN
            Present_state<=next_state;
        END IF;
    END PROCESS;
    PROCESS(present_state)     --组合进程
    BEGIN                      --对暂存信号赋值和确定状态机的次态
        CASE present_state IS
        WHEN s0=> next_state<=s1;
        WHEN s3=> next_state<=s4;
        ...
    END CASE;

```

END PROCESS;
...

4.2 CANopen 网络的启动与 NMT 管理

网络的启动与管理是保证网络数据顺利传输的前提与先决条件，它包括网络的配置，通信参数的设置，网络错误管理，网络状态监控等，维护着网络的运行，确保数据的顺利传输。

4.2.1 CANopen 网络的启动

1、CANopen 网络启动流程

CANopen 网络在上电以后，各个节点都会按照相应的程序回归到初始状态。网络启动的过程如图 4-3 所示，首先是主站启动、初始化通信配置，此操作包括 CANopen 主站 CAN 控制器工作模式配置，SPI 接口初始化，应用层芯片接口初始化等操作；接下来是从站启动，如果从站对象字典中的参数需要配置，则主站通过 SDO 对其进行相应的配置，如果对象字典无需修改，则直接运行主站和从站，至此整个网络便进入正常运行模式，PDO 传输便可以进行。在整个网络运行过程中，如果某个或某几个从站节点运行过程中出现错误，则整个网络跳转到从节点启动状态，对出错的从节点重新进行配置和启动，从节点重新配置和启动成功后再启动整个网络。若整个网络出错或者重新上电，则需要跳转到 CANopen 主站启动状态，整个网络从新开始启动^[39]。

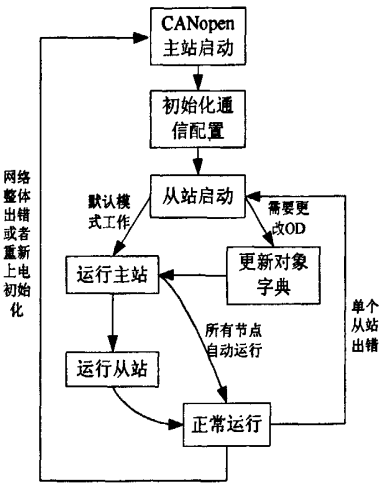


图 4-3 CANopen 网络启动流程图

2、CANopen 网络节点的启动与运行状态

在网络初始化过程中，CANopen 支持最小化 Boot-up 过程。如图 4-4 所示，节点在上电以后自动进入初始化状态，在此过程中，节点内部各种

参数恢复到缺省值，如通信参数、对象字典实体参数等。初始化状态过后节点便自动进入，此时可以进行 SDO 对象通信，通过 SDO 对象通信从节点可以得到配置，需要修改的对象字典参数也可以进行相应的修改，如进行 PDOs 通信参数与映射参数的修改，在必要的时候，也可以用 SDOs 来更改节点 ID，在从节点预操作状态下 PDO 传输是被禁止的。节点参数配置完成之后，便可以对节点进行正常操作了，正常运行模式存在于操作状态中，此状态支持所有的通信对象，如 PDOs、同步对象等。状态转移图中的停止状态包含节点关闭功能，一旦进入此状态，该节点便处于关闭状态。

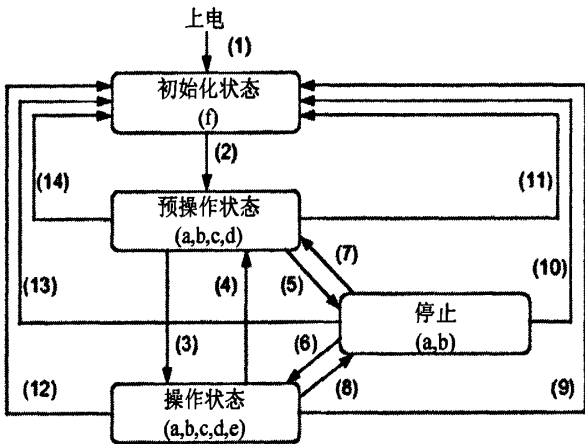


图 4-4 CANopen 节点最小化 boot-up 状态转移图

4.2.2 NMT 网络管理

NMT 网络管理除了网络启动之外，还包括模块控制（Module Control）、节点保护（Node Guarding）、心跳报文（Heartbeat）^{[20][40][41]}。

1、模块控制（Module Control）

在 CANopen 网络中 NMT 报文只能由主站发出来对从站进行控制，而且网络中的从站必须支持 NMT 模式，本文所采用的智能数字伺服支持 CANopen 的 NMT 管理功能。如表 4-2 所示，NMT 报文分为 COB-ID、命令字和 Node-ID 三个部分，其中 COB-ID 的功能码为 0000；命令字如表 4-3 所示，Byte0 中不同的数值预示着对节点不同的操作^[42]；Node-ID 是目标节点的 ID,如果 Node-ID=0 表示这条 NMT 报文采用的是广播模式，报文信息将发送给所有从站的。

表 4-2 NMT 报文格式

COB-ID	Byte0	Byte1
0X00	CS	Node-ID

表 4- 3 CS（命令字）定义

命令字	含义
1	启动节点
2	停止接点
128	进入预操作状态
129	复位节点
130	复位通信

2、节点保护（Node Guarding）

通过节点保护服务，MNT 主节点可以检查每个节点的当前状态，主站通过发送远程帧的方式来实现这一功能，当这些节点没有数据传送时这种服务尤其有意义。如图 4-5 所示，NMT 主站节点发送远程帧（无数据）的 COB-ID 为 0X700+Node-ID，在接收到此报文后从站节点将自己的状态返回给主站，应答报文格式如表 4-4 所示，数据部分包括一个在节点保护应答中交替置“0”或置“1”的触发位（bit7）；如表 4-4 所示，0 到 6 位数据中存放着节点的状态。

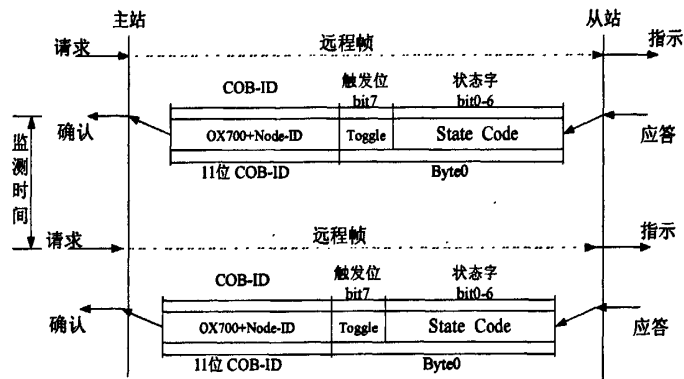


图 4-5 NMT 节点保护模式图

表 4- 4 NMT 节点保护状态字含义

状态值	含义
0	初始化状态
1	未连接状态
2	连接状态
3	准备状态
4	停止状态
5	操作状态
127	预操作状态

3、心跳报文（Heartbeat）

节点也可被配置为产生周期性的被称作心跳报文(Heartbeat)的报文。如图 4-6 所示，从节点定期向主站节点以心跳报文格式汇报其当前状态。被配置为 Heartbeat 的节点在启动后，它的第一个 Heartbeat 报文是 Boot-up

报文。报文格式跟节点保护应答报文相似，COB-ID 也是 0X700+Node-ID，后面也有一个字节的命令字，命令字含义如表 4-5 所示。每个 Heartbeat 节点都被设置了一个超时值，当超时发生时采取相应的动作。

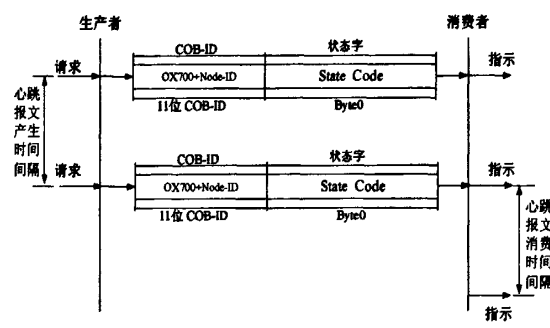


图 4-6 NMT 心跳报文模式图

表 4- 5 NMT 心跳报文状态字含义

状态值	含义
0	初始化状态
4	准备状态
5	操作状态
127	预操作状态

4.3 CANopen 网络的同步

在 CANopen 网络信息传递的过程中，发送节点按照一定的波特率往总线上发送自己的报文，接收节点必须以相同的采样频率去接收，但是如何能在正确的时间窗口采集到每一位上的信息是一个必须解决的问题，这就需要 CANopen 网络的同步功能，发送节点跟接收节点之间必须建立同步机制，这样才能保证接收节点准确无误的接收到来自发送节点的报文信息。

4.3.1 CAN 网络的同步机理

CAN 总线的标准位时间（也称额定位时间）的结构如图 4-7 所示，由同步段（SYBC_SEG）、传播段（PROP_SEG）、相位缓冲段 1（PSEG1）和相位缓冲段 2（PSEG2）四部分组成^[7]。所以额定的时间份额 $t_{NBT}=t_{SYNC_SEG}+ t_{PROP_SEG}+ t_{PSEG1}+ t_{PSEG2}$ ，位周期中的这些段的时间长短都是可以通过编程进行设置。不管是发送节点发送一位数据还是接收节点接收一位数据都是从同步段（SYBC_SEG）开始；报文在网络中传播的物理延时可以用传播段（PROP_SEG）来进行补偿， $t_{PROP_SEG}=（信号在总线上传播的时间+输入比较器延时+输出比较器延时）\times 2$ ；采样点（Sample

Point) 是读取总线电平并转换为一个对应的位置的一个时间点, 位于相位缓冲段 1 结尾^[43], 可以通过加长或者缩短相位缓冲段 1 (PSEG1) 和相位缓冲段 2 (PSEG2) 的时间来改变采样点的位置, 从而补偿边沿阶段的误差。

CAN 协议规定了两种类型的同步: 硬同步和重同步。硬同步只在总线空闲时通过一个下降沿 (帧起始) 来完成, 此时不管有没有相位误差, 所有节点的位时间重新开始。硬同步后, 位时间由每个位定时逻辑单元以 SYNC_SEG 重新启动^[7]; 在报文的随后位中, 每当有从隐性位到显性位的跳变, 而且当该跳变落在了同步段之外时, 就会引起一次重同步, 重同步可以根据跳变沿发生的位置来增长或者缩短位时间以调整采样点的位置, 保证正确采样^{[7][44]}。

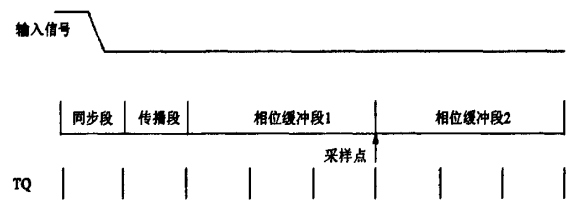


图 4-7 位时间结构

4.3.2 CANopen 网络的同步

CANopen 网络的同步 (SYNC) 是管理各节点同步数据收发的一种方法, 相当于网络节拍, 由网络中某个节点按照固定频率向网络中发出广播模式的同步报文, 网络中有且只有一个 SYNC 生产者, 一般有多个消费者, 并不是所有节点都需要 SYNC, 需要 SYNC 的节点(消费者)将其接收, 其网络标识符优先级很高, 设置为 0x80^{[25][45]}。网络中的报文发送情况如图 4-8 所示, 需要发送数据的节点要在规定时间内将数据发送, 这个时间段叫做时间窗口(由对象字典索引号为 1006H 的实体指定), 虽然同步报文的优先级很高, 但是由于它不携带数据信息, 所以占用总线的时间很短, 紧随同步报文之后的其他数据对象才是占据网络的重要部分。

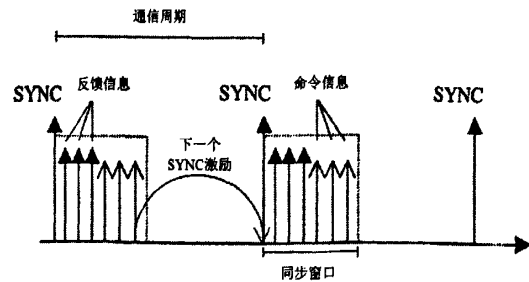


图 4-8 同步 PDO 典型传送模式

同步报文的信息配置可以在相应的对象字典中进行，如在索引号为 1005H 的对象字典中可以通过设定它的 32 位无符号实体来确定该设备是否是 SYNC 对象的生产者，支持的是标准报文帧还是扩展报文帧，也可以在相应的对象字典中设置同步报文的发送周期等。

图 4-8 是同步 PDO 的运行模式，需要 SYNC 的节点(消费者)将来自 SYNC 生产者的同步报文接收并计数，当满足节点相应 TPDO 发送要求时，该 TPDO 映射的数据就被发送。在此过程中 SYNC 生产者提供网络节拍，为消费者提供报文发送的触发条件。同步 PDO 又可分为周期性同步 PDO 和非周期性同步 PDO，如图 4-9 所示，周期性同步 PDO 在对来自生产者的 SYNC 报文数量到达设定值时便触发 PDO 报文发送，PDO 报文的发送周期是 SYNC 报文周期的 N 倍，N 的数值可以在对象字典中设定；对于非周期性同步 PDO 来说，SYNC 报文只是其 PDO 发送的一个必要的触发条件，在接到 SYNC 报文后，该节点还需要判断内部的其他触发条件，若都满足则触发 PDO 的发送。本研究中主站插补器计算出的数据结果就是以非周期性同步 PDO 进行发送的。

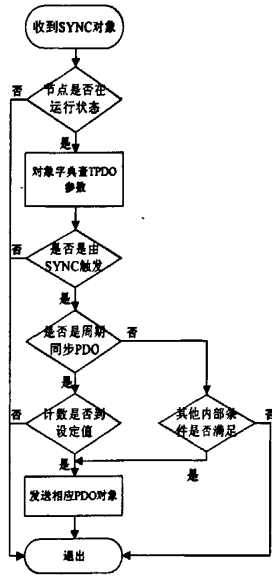


图 4-9 SYNC 触发节点 PDO 发送流程图

4.4 本章小结

本章节对 CANopen 主站应用层芯片的总体任务调度进行了研究，设计了主站应用层芯片 FPGA 实现的状态机。此外还研究了 CANopen 网络的启动与管理机制，着重研究了网络错误控制机制和网络同步管理的实现方法。

第五章 主站应用层芯片主要功能模块设计与数字伺服运动控制的实现

CANopen 应用层芯片内部是由多个任务模块构成，高质量地实现各个任务模块的功能是实现应用层芯片的前提和保障。由于篇幅限制，本章节挑选了几个重要功能模块加以介绍，如 CAN 控制器的初始化模块、CANopen 主站应用层接口模块，本章还对 SDO 与 PDO 的运作流程做了研究，介绍了与之相关的各个功能模块。此外，还建立了一个最小化的 CANopen 网络系统，实现了单个数字伺服电机的启停运动控制。

5.1 CAN 控制器的初始化

CAN 控制器的初始化是主站上电后要做的第一件事，是整个主站状态机的起始状态。当主站遇到重大错误时，状态机也会跳到该状态进行重新启动，重新配置 CAN 控制器参数，而后对网络进行初始化。

5.1.1 CAN 控制器引脚配置

与报文发送有关的引脚包括 SPI 接口与 INT 引脚，首先对 TXRTSCTRL 寄存器进行设定，将报文的发送触发模式设定成只由 SPI 接口控制，而非采用 TXNRTS 控制。当检测到发送缓冲区为空的时候，应用层芯片通过 SPI 接口将芯片内的待发送报文写进发送缓冲区，待发送的报文可以设定发送优先级以便将三个发送缓冲器中的待发送报文进行比较，择取优先级最高的先发送，无论发送成功还是发送失败都会通过 INT 通知应用层芯片得知，应用层芯片通过 SPI 接口访问 CAN 控制器内部状态寄存器来查询报文发送的具体状态。

与报文接收有关的引脚包括 SPI 接口和 RXBF0、RXBF1、INT 引脚，RXBF0 和 RXBF1 可以被配置成接收报文信息中断，当 RXB0 或 RXB1 成功接到报文之后，通过 RXBF0 或 RXBF1 向应用层芯片发出中断信号，通知应用层芯片通过 SPI 对 RXB0 或 RXB1 中的信息进行读取。当报文接收出错或者报文溢出时，INT 引脚会产生中断，可通过 SPI 接口访问相应的寄存器去读取中断类型。

此外，CAN 控制器的复位信号引脚 RESET 需与应用层芯片相接，以便复位时使用。其他引脚在第三章硬件电路中已经做过安排。

5.1.2 CAN 控制器内部参数设置

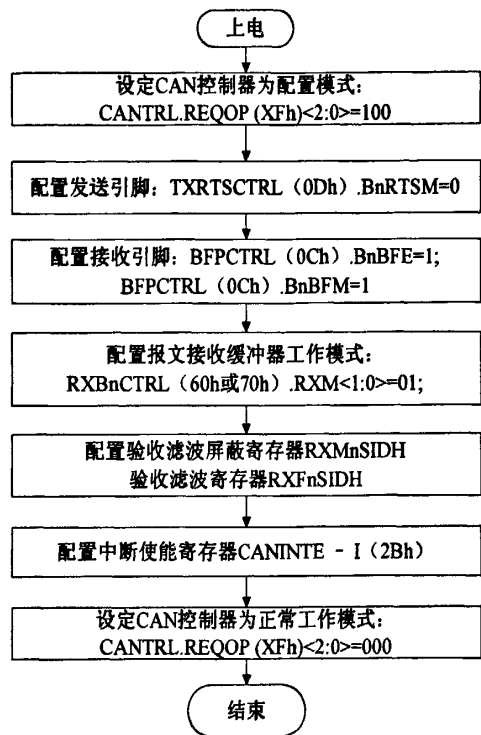


图 5-1 CAN 控制器初始化流程图

通过配置地址为 0Dh 的 TXRTSCTRL 寄存器中的 BnRTSM 位可以选择是否启用 TXnRTS 引脚，在本文中，为了减小报文发送模块的复杂程度，决定不启用 TXnRTS 引脚来触发报文发送，这样一来可以通过设定 TXBnCTRL（30h、40h、50h）寄存器中的 TXP<1:0>来设定发送缓冲区中待发送报文的优先级，在 5.2 中将会重点介绍。

接收报文引脚配置可以通过设定 BFPCTRL 寄存器来实现，BFPCTRL 寄存器的地址为 0Ch，当配置 B0BFE=1 和 B1BFE=1 时，RX0BF 与 RX1BF 引脚被使能，此时若配置 B0BFM=1 和 B1BFM=1，则 RX0BF 与 RX1BF 引脚就被设置成当有有效报文载入相应接收缓冲器时就产生低电平，以告知应用层芯片来读取报文信息。

如图 5-1 所示，报文接收缓冲器工作模式的配置可以通过设定报文接收缓冲器控制寄存器来实现，RXBnCTRL（60h 或 70h）.RXM<1:0>=01 表明只接收符合滤波器条件的带标准标识符的有效报文。

通过设定验收滤波屏蔽寄存器 RXMnSIDH 和验收滤波寄存器 RXFnSIDH 来确定是否应当将报文集成缓冲器（MAB）中的报文载入 RXB0 或者 RXB1。表 5-1 所示的真值表显示了标识符中每一位数据与验收屏蔽器和滤波器进行比较的过程。报文标识符中的哪些位需要与验收滤波器进行比较可以通过设定屏蔽寄存器来实现，如果某屏蔽位为 0，对应

的标识符位将被自动接收而不被滤波。

表 5-1 滤波/屏蔽寄存器真值表

屏蔽位n	过滤位n	报文标识符n	接受或拒绝位n
0	X	X	接收
1	0	0	接收
1	0	1	拒绝
1	1	0	拒绝
1	1	1	接收

配置中断使能寄存器可以决定哪些中断将被启用，在遇到相应状态时会向 INT 引脚发出中断以告知应用层芯片。在该寄存器中可以设置的中断类型有报文错误中断、唤醒中断、错误中断、发送缓冲器 2 空中断、发送缓冲器 1 空中断、发送缓冲器 0 空中断、接收缓冲器 1 满中断、接收缓冲器 0 满中断。

5.2 CANopen 主站应用层接口模块设计

由图 4.2 可知，主站应用层接口模块在整个主站状态机中起着至关重要的作用，它上承 6 个输入状态接口，下启 5 个输出接口，各种循环流程以此作为始终节点。如图 5-2 所示，它是基于中断工作模式，当 St 1 状态启动时，首先读取数据信息发送完成标志信号，若有未发送完成的信号，则直接跳到继续发送报文的程序分支。此时，应用层接口模块对来自 CAN 控制器的中断信号进行监视，当监听到 INT 引脚发送过来的中断信号时，就通过 SPI 接口访问控制器 CANSTAT 寄存器的 ICOD<2:0>位，以探求其中断类型。如表 5-2 所示，ICODE<2:0>位一共可以反映 8 种中断源。如果是错误中断，则跳至 St 13 错误处理状态，同时应用层接口状态模块关闭；若是发送缓冲器为空中断则对上次未发送完成的数据对象种类进行判断，之后继续转入相关数据对象生成状态进行相关数据对象的发送。

表 5-2 错误中断类型

ICODE<2:0>	状态数值中断类型
000	无中断
001	出错中断
010	唤醒中断
011	TXB0中断
100	TXB1中断
101	TXB2中断
110	RXB0中断（未使用）
1111	RXB0中断（未使用）

如果在 St1 状态启动时没有未完成的待发送数据信息，则转为检测报文接收缓冲区是否有新接收的报文，如果 RX0BF 或 RX1BF 产生低电平跳变则表明有新的报文被写入 CAN 控制器的接收缓冲区，此时启动 St3 报

文接收状态，进行 CANopen 应用层芯片报文的接收；如果没有报文接收中断产生，则进入 INT 引脚中断扫描，当出现 INT 引脚中断时，其处理方法与有待发送数据信息时的 INT 中断相同，先判断是否是错误中断，而后看是否是报文发送完成中断。由于没有待发送数据信息等候发送，当发送缓冲器为空时，需要判断网络的运行状态，以便将主站状态转移到相应的新报文对象生成状态。假如网络处于从节点配置状态，则主站自动将状态跳转至 St9 新 SDO 生成状态。新 Sync 生成状态是由同步计数器计数已满触发的，在其触发后，同步计数器被清零。

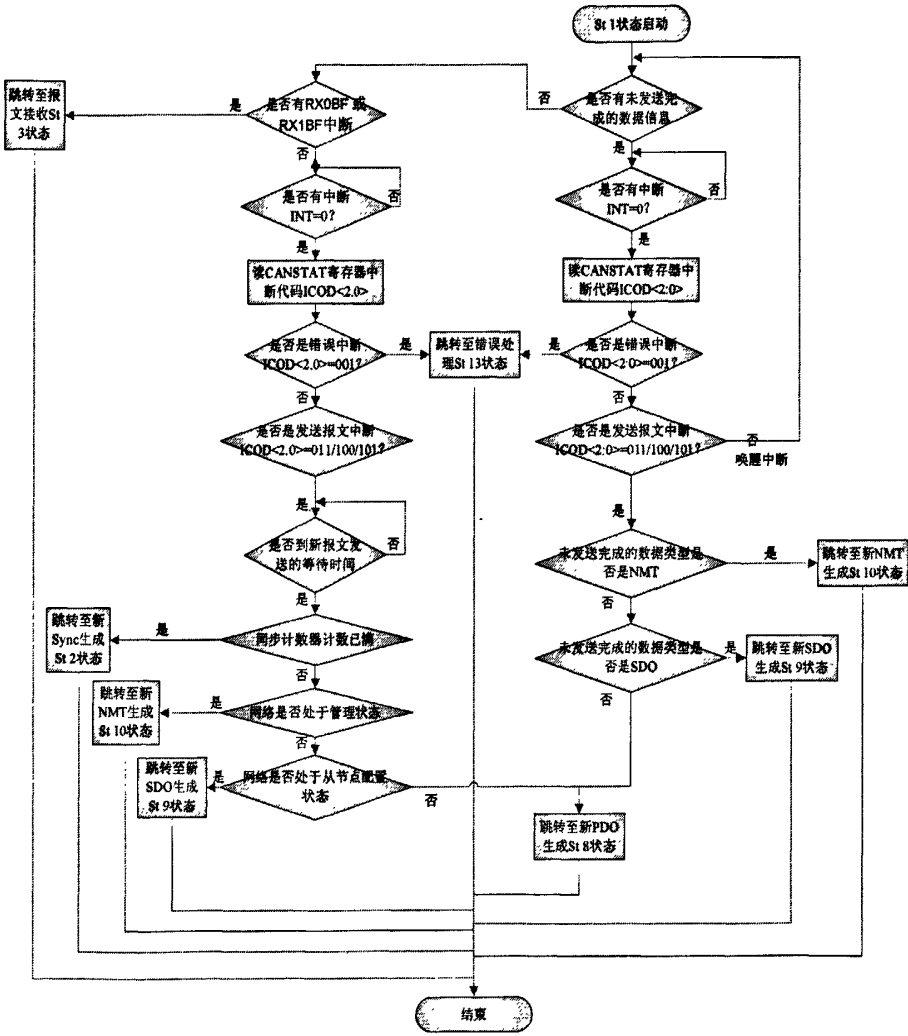


图 5-2 CANopen 主站应用层接口模块程序流程图

CANopen 主站应用层接口模块的状态机转移程序用 VHDL 语言在 FPGA 中实现的程序段如下，其中 trans_state 信号表示主站没有发送完成的报文数据种类，如表 5-3 所示，第一位状态显示是否有没有发送完成的报文，后两位显示了未发送完成的是哪种报文。

```

...
IF (trans_state (0) = '1') THEN          --判断是否有未发送完的报文
    IF (INT= '0') THEN
        IF (ICOD<2:0>= '001') THEN next_state<=st13; --跳转至错误处理状态
        ELSE IF (ICOD<2:0>= '011' OR  ICODE<2:0>= '100' OR ICODE<2:0>=
'101') THEN                                --发送缓冲器有空闲
            IF (trans_state= '100') THEN next_state<=st10; --跳转至新 NMT
生成状态
            IF (trans_state= '101') THEN next_state<=st9; --跳转至新 SDO
生成状态
            IF (trans_state= '110') THEN next_state<=st8; --跳转至新 PDO
生成状态
            ELSE next_state<=st13;
            END IF;
        ELSE next_state<=st1;              --跳转至主站应用层接口
状态
        END IF;
    ELSE next_state<=st1;
    END IF;
    ELSE IF (RX0BF= '0' OR RX1BF = '0') THEN next_state<=st3; --判断
是否是接收报文中断
    ELSE IF (INT= '0' ) THEN
        IF (ICOD<2:0>= '001') THEN next_state<=st13;
        ELSE IF (ICOD<2:0>= '011' OR  ICODE<2:0>= '100' OR ICODE<2:0>=
'101') THEN WAITUNTIL (time_full= '1' );
        IF (Sync_conter= '1' ) THEN next_state<=st2; --跳转至同步报文生成
状态
        ELSE IF (NMT_state= '1' ) THEN next_state<=st10;
        ELSE IF (SDO_state= '1' ) THEN next_state<=st9;
        ELSE next_state<=st8;
        END IF;
    END IF;
END IF;
END IF;
...

```

表 5- 3 trans_state 所对应的数据发送状态

Trans_state	含义
0XX	没有未发送完成的数据信息
100	有未发送完成的数据信息为NMT报文
101	有未发送完成的数据信息为SDO报文
110	有未发送完成的数据信息为PDO报文
111	未使用

5.3 FPGA 中主站报文的接收与发送

FPGA 中主站报文的接收与发送是 CAN 控制器与应用层芯片之间进行数据通信的桥梁，应用层芯片通过报文接收模块将 CAN 控制器接收缓冲器中的报文读取到应用层芯片进行处理，然后将经过处理之后所产生的报文再通过报文发送模块存储到 CAN 控制器发送缓冲器中。

5.3.1 应用层芯片与 CAN 控制器之间的收发逻辑

应用层芯片与 CAN 控制器之间的报文传递是通过 SPI 接口实现，SPI 接口的具体操作指令如表 3-4 所示，可实现寄存器读写，而发送报文缓冲器与接收报文缓冲器也都是寄存器的一种，所以可以通过寄存器读写指令来进行报文缓冲器操作。

在读接收缓冲器时，如图 5-3 所示，片选信号 CS 引脚为低电平，而后，发送的依次是读指令和 8 位地址码，MCP2510 在处理后这些数据之后将需要的寄存器数据通过 SO 引脚反馈回来给上位控制器。由于在数据发送之后 MCP2510 内部的地址指针会自动指向下一位的地址，所以 MCP2510 支持连续读写操作，当所需数据读取完成之后，通过将 CS 引脚置“1”就能技术此次操作。

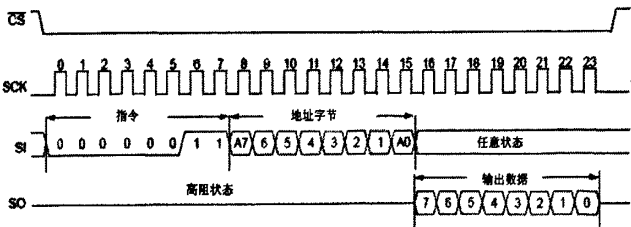


图 5- 3 读 MCP2510 报文接收缓冲区

在写发送缓冲区时，如图 5-4 所示，同样的将 CS 引脚置“0”启动操作，然后输入写指令以及需要写入数据的寄存器地址，随后就是将要写入的数据。与读指令相同，写指令也支持连续写操作。每一位数据的写入发

生在 SCK 引脚电平的上升沿，最先写入的是数据字节的 D0 位。

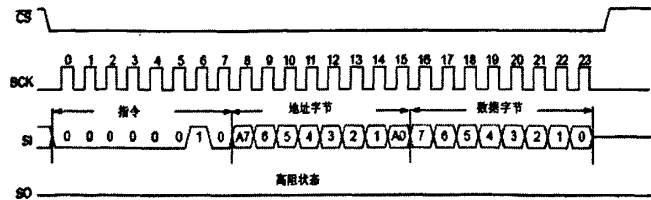


图 5-4 写 MCP2510 报文发送缓冲区

写 MCP2510 报文发送缓冲区的程序如下所示。

```
...
SIGNAL DOWN_BUFFER: STD_LOGIC_VECTOR(63 TO 0);
SIGNAL i: INTEGER RANGE 63 DOWNT0 0;
...
i<= 0;
IF SCK AND SCK'EVENT AND i<64? THEN
    SI <= DOWN_BUFFER(i);
    i<= i+1;
END IF;
CS <='0' after 40 ns;
SI <='1';
SI <= '1' AFTER 55 ns ;
...

```

对于发送缓冲器的控制寄存器，有些时候需要修改其中的某些位，这需要用到位修改指令，该指令并非对所有寄存器都有效。如图 5-5 所示，在命令字节发送后，其后面是寄存器地址，屏蔽字节以及数据字节。屏蔽字节决定寄存器中的哪一位将被修改，屏蔽字节中的‘1’表示允许对寄存器相应的位进行修改，‘0’则表示禁止修改^[29]。

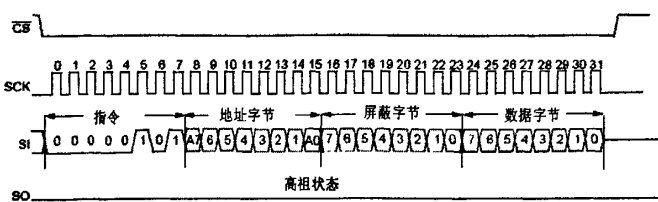


图 5-5 位修改操作时序

5.3.2 报文的接收与处理

报文经由 SPI 接口接收之后便存放在应用层芯片之中，以供应用层芯片状态机对其进行调用与处理。如图 5-6 所示，当报文接收状态被启动时，CAN 控制器接收缓冲区中的报文通过 SPI 接口被迅速转移至 FPGA 的报

文接收缓冲区,因为 CAN 控制器接收缓冲区中装载着一个完整的数据帧,所以此处 FPGA 中的报文接收缓冲区也需要一个包括 13 个字节的存储区域 Receive_Data。而后数据经过报文解析分发状态模块进行报文类型识别,为了节省 FPGA 资源,此处报文分发只做为启动 PDO 处理、SDO 处理和 NMT 处理三个状态的触发条件,报文仍然存放在 Receive_Data 中,在进行相关报文处理时对其进行访问即可。

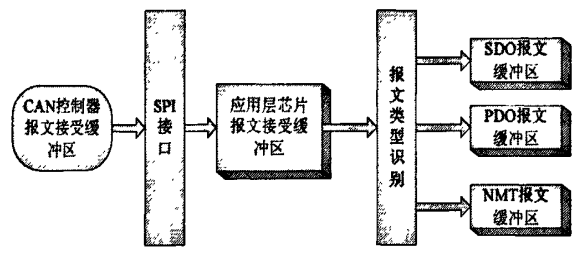


图 5-6 报文接收过程数据流

CAN 报文接收状态 St 3 的程序流程图如图 5-7 所示,在 St 3 状态启动之后首先需要判断 CAN 控制器的哪个接收缓冲器接收到了新的报文信息,然后将相应接收缓冲区的首字节地址放入读指令之后的 8 位地址码(A7 至 A0)中,然后通过 SPI 接口将数据依次读取到 Receive_Data 中,在数据接收的过程中如果出现错误则重新跳转到 St 3 的启动状态重新进行数据读取,如果成功接收则将 CAN 控制器的中断标志位 CANINTF.RXNIF 置 0,然后跳转至报文解析分发状态 St 4。

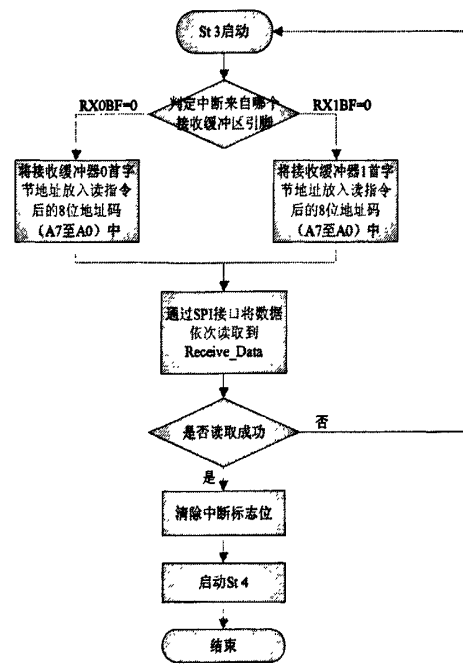


图 5-7 St 3 CAN 报文接收状态流程图

在报文解析分发状态 St 4 中,报文类型的区分是根据表 2-4 中的功能

码区分的，将 Receive_Data<0:3>中的数据分别与相关类型的代码进行比较，如下面一段程序段所示，比较过后启动相应的报文处理状态。

```

...
if(clk'event and clk='0')then
    if(sgnl1_1='0')then --判断 sgnl1 的状态。sgnl1=0 表示还没有接
        收数据，sgnl1=1 表示在接收数据
        if(rx='0')then --此 if 语句用于判断是否起始位，是则 sgnl1
            置位‘1’
            if(sgnl2=7)then --对应于起始位的处理
                sgnl2<=0;
                sgnl1_1<='1';
                sgnl4<='1'; --一旦 sgnl4=1,就要采用一次数据。
            else
                sgnl2<=sgnl2+1;
                sgnl4<='0';
            end if;
        else
            sgnl2<=0;
        end if;
        else if(sgnl2=15)then --对应于数据位的处理（每 16 个时钟处理
            一次）
            sgnl2<=0;
            if(sgnl3=8)then --是否接收完一组数据，若接收完，则 sgnl1
                置‘0’
                sgnl3<=0;
                sgnl1_1<='0';
                sgnl4<='0'; --每 16*64 个时钟处理一次，数据组长为
                64。
            Else
                sgnl3<=sgnl3+1;
                sgnl4<='1'; --sgnl4 的周期为 16 个时钟
            end if;
        else
            sgnl2<=sgnl2+1;
            sgnl4<='0';

```



```

end if;
end if;
end if;
...

```

5.3.3 报文的封装与发送

新接收过来的数据对象经过应用层芯片处理过后会产生相应的新数据对象，或者主站应用层芯片根据需要新产生新数据对象，这些数据对象如图 5-8 所示，包括 Sync 对象、SDO 对象、PDO 对象和 NMT 对象，它们经过报文识别之后被封装成为标准的数据帧存放在应用层芯片报文发送缓冲区 Transmit_Data 中，当 CAN 控制器的发送缓冲区为空时，该数据帧就通过 SPI 接口被写入到 CAN 控制器的发送缓冲器中。

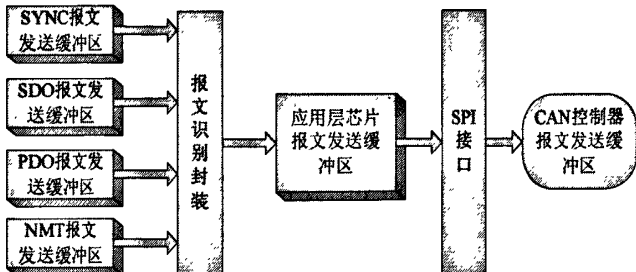


图 5-8 报文发送过程数据流

报文识别封装状态模块（St 11）的功能是将来自四种新报文生成状态模块所产生的报文按照其各自的报文对象格式进行封装。CAN 控制器的每个报文发送缓冲器都有一个控制寄存器，对控制寄存器 TXBNCTRL 的 TXP<1:0>位进行设定可设置该缓冲器报文发送的优先级^[29]，如表 5-4 所示，Sync 报文的优先级应当最高，它用于对网络进行同步，为网络提供时间节拍；其次是 SDO 报文，用于节点的配置；PDO 报文用于传递过程数据，最后是 NMT 报文，进行网络状态监控。报文识别封装状态模块（St 11）在报文封装完成之后将报文转存到应用层芯片报文发送缓冲区，并触发 CAN 报文发送状态模块（St 12）。

表 5-4 发送缓冲器优先级

TXP<1:0>	发送缓冲器优先级	报文对象
11	最高的报文发送优先级	Sync
10	中偏高的报文发送优先级	SDO
1	中偏低的报文发送优先级	PDO
0	最低的报文发送优先级	NMT

CAN 报文发送状态 St 12 的程序流程图如图 5-7 所示，在 St 12 状态

启动之后首先需要判断 CAN 控制器的发送缓冲器是否有空置状态，若没有则直接跳转到 St 12 的启动状态，若有则需要判断是哪个发送缓冲器为空，然后将 St 11 产生的发送缓冲器控制寄存器信息写入到相应的缓冲器控制寄存器中，将相应发送缓冲区的首字节地址放入写指令之后的 8 位地址码（A7 至 A0）中，通过 SPI 接口将 Transmit_Data 中的数据依次写入到发送缓冲区中。此过程如果出现错误，则需要跳转到是哪个发送缓冲器为空的判断状态，若成功写入，则清除发送缓冲器的中断标志位，并启动 St 1 状态，关闭 St 12 状态。

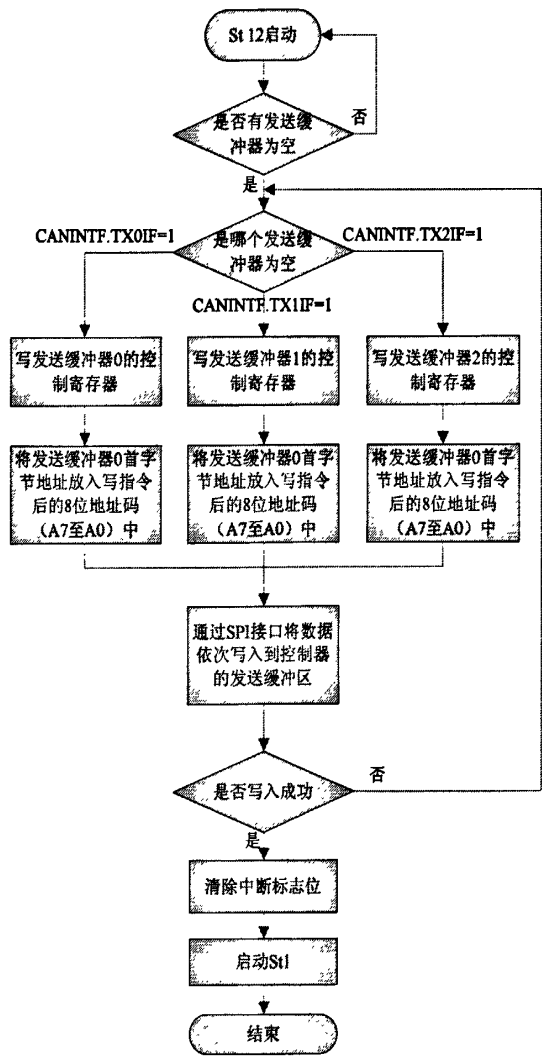


图 5-9 St 12 CAN 报文发送状态流程图

FPGA 与 CAN 控制器之间的数据交换是通过 SPI 接口实现的。截取代码如下：

```
...
TX_Buffer: OUT STD_LOGIC_VECTOR(63 DOWNT0 0); --定义发送缓冲区
...
L1: FOR I IN 0 TO 63 LOOP -循环，将发送缓冲区中的数据在 CLK
上升沿发送出去
    IF CLK'EVENT AND CLK=1 THEN
        SI<= TX_Buffer(I);
    END IF;
END LOOP [L1];
...
```

5.4 SDO 的操作

SDO 数据对象是用于网络配置的数据对象，在网络初始化或者需要对网络进行配置的时候，可通过 SDO 修改从节点对象字典（OD）中的数据信息，以完成相应的配置，如从节点的运行模式、PDO 的通讯参数或匹配参数等^[46]。

5.4.1 SDO 的运作流程

SDO 在 CANopen 网络中的通讯过程如图 5-10 所示，在网络配置的过程中，首先由主站应用层芯片作为客户生成 SDO 报文，经由 CAN 控制器与收发器发送到 CANopen 网络上，传输到相应的从节点，对其对象字典参数进行操作，从节点在接收到主站的 SDO 之后产生相应的 SDO 报文，并通过 CAN 总线反馈给 CANopen 主站，主站在接收到反馈 SDO 之后对其进行分析，如果对该节点的配置已经完成或出错，则停止下一个 SDO 的发送，如果对象字典一次操作没有完成，则继续发送 SDO，如此循环直至所有从节点都配置完成。

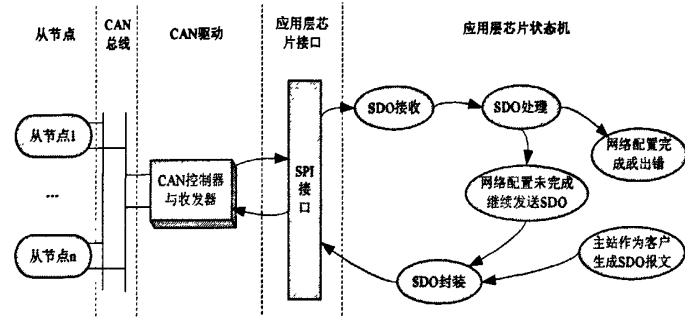


图 5-10 SDO 通信过程

SDO 的操作在主站应用层芯片的状态机中共涉及 7 个状态，SDO 操作的启动是由 St 1 状态触发的，当主站需要对网络进行配置时，便将相应的配置交由 St 9 进行操作生成对应的 SDO 报文，经封装（St 11）后发送（St 12）出去，由于 SDO 报文是基于应答模式，当主站向 CANopen 网络发出 SDO 报文后，消费节点在收到后会给一个 SDO 作为回应，St 1 便对回来的 SDO 报文进行接收（St 3），经报文解析（St 4）之后送给 SDO 处理（St 6）状态单元，St 6 可以继续刚才上一个 SDO 报文没有完成的工作，如此循环，直到该项配置结束，配置结束的返回报文运转到 St 6 后，直接跳至 St 1，如此便进行其他的任务状态。

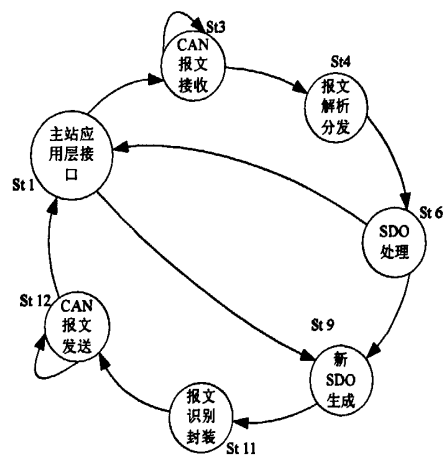


图 5-11 SDO 操作在主站应用层芯片中的状态机转移图

5.4.2 SDO 的读写

SDO 的读写是基于客户/服务器通信模式的，此处的客户就是 CANopen 主站，服务器则是从站数字伺服，主站通过 SDO 的读写去访问从站的对象字典。SDO 包含着索引和子索引，用来访问从站对象字典中的具体实体^[47]。由于对象字典实体的长度是任意的，有些不可能通过一次 SDO 读写操作就能完成，所以 SDO 的读写可分为三种操作方式^{[20][25]}：

- (1) 段传输，一般传输手段，可用于任何用户；
- (2) 加速传输，用于小段数据的传输；
- (3) 块传输，用于大段数据的传输。

本文所采用的 SDO 读写方式是段传输模式，如图 5-12 所示，主站首先要向从站发送读/写 SDO 的初始化报文以确定需要传输的数据和传输的方式，等响应成功之后便可进行 SDO 片段的读/写操作，直到最后一个片段传输完成为止。

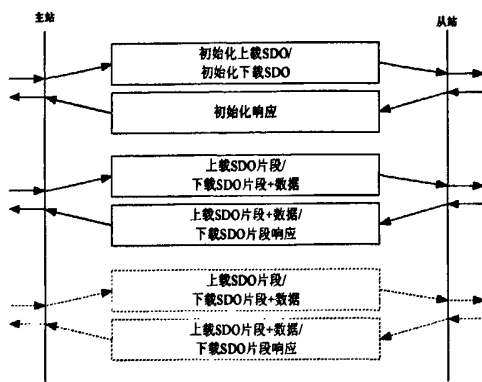


图 5-12 SDO 段传输操作的一般模式

要修改对象字典中的参数配置，则需要运用 SDO 对对象字典进行操作，SDO 的报文格式如表 5-5 和表 5-6 所示，CAN 报文包括帧起始、仲裁字段、数据段、CRC 校验以及帧结束几大部分组成。CANopen 协议则是将自己的数据信息按照协议要求封装好放在 CAN 报文的 8 字节数据当中。如表中所示 SDO 将 8 字节数据段分为请求码、对象和子对象索引以及 4 字节数据^[48]，从而进行对象字典数据的读写。

表 5-5 请求帧（主站-数字伺服）

帧起始	仲裁字段		8字节数据字段								校验	帧结束
			请求码	对象索引 (index)		对象索引 (sub-index)	数据 (4 Byte)					
	COB-ID (数字伺服从站ID)	RTR	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	CRC	

表 5-6 响应帧（数字伺服-主站）

帧起始	仲裁字段		8字节数据字段								校验	帧结束
			响应码	对象索引 (index)		对象索引 (sub-index)	数据 (4 Byte)					
	COB-ID (CANopen主站ID)	RTR	Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	CRC	

5.5 PDO 的操作

PDO 数据对象是用于过程数据传输或控制的数据对象，在网络正常运行时，可通过 PDO 读取各个节点的状态，或者向各个节点下达控制指令以完成相应的控制，如数字伺服的位置控制，速度控制等。

5.5.1 PDO 的运作流程

PDO 在 CANopen 网络中的通讯过程如图 5-13 所示，网络配置完成之后，系统进入正常运行状态，各从节点的状态信息以及主节点中的控制信息都在此过程中进行传递。从节点通过远程帧的形式从主站中获取其需要的信息，主站在接到远程帧之后产生相对应的 PDO 报文并发送回去。此外，从节点的状态信息也需要经由 PDO 传送给主站，主站在接收到状态

PDO 之后将其状态信息提取并存入主站中相关存储区域，以供主站及时掌握各个从站的状态，主站还有一个重要的任务就是将数控系统插补器计算出来的数据信息传递给相应的从站，此进程采用同步 PDO 触发模式，在插补结果传递过程中，根据同步周期将插补数据结果中的数据依次传送给相应的从站，即数字伺服驱动。

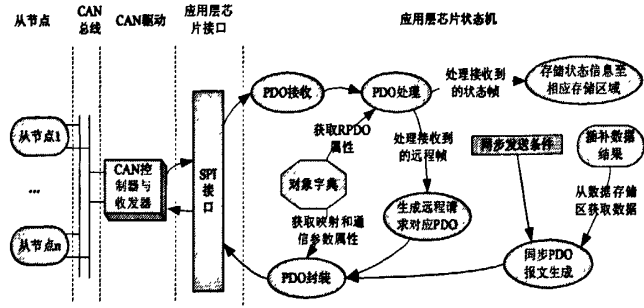


图 5-13 PDO 通信过程

PDO 的操作在主站应用层芯片的状态机中共涉及 7 个状态，如图 5-14 所示，当主站需要通过 PDO 向从站发送过程数据时，主站应用层接口（St 1）便启动新 PDO 生成（St 8），经封装（St 11）后发送（St 12）出去；从节点发送过来的 PDO 分为两种，一种是远程帧，用来从主站中获取相应的信息，远程帧经过主站应用层接口（St 1）与 CAN 报文接收（St 3）后经报文解析（St 4）与 PDO 处理（St 5），启动新 PDO 报文生成（St 8），将其所请求的信息放入 PDO 报文发送回去，另外一种状态 PDO，包含着从站向主站发送的状态信息，与远程帧相同，经由主站应用层接口（St 1）与 CAN 报文接收（St 3）后经报文解析（St 4）与 PDO 处理（St 5），在 PDO 处理的过程中将从节点的状态信息存入主站的相应存储区域。

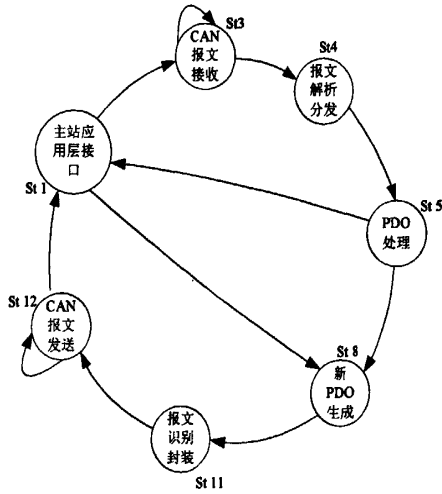


图 5-14 PDO 操作在主站应用层芯片中的状态机转移图

5.5.2 PDO 的读写

PDO 的读写是基于生产者与消费者模式的数据传输方式，生产者只需要向 CANopen 网络发送报文，消费者根据报文 COB-ID 接收自己需要的报文即可^[49]。PDO 发送的触发方式分为三种^{[20][25][50]}：（1）事件触发模式；（2）时间计数器触发模式；（3）远程帧触发模式。这三种触发模式在一个节点中可以并存。如图 5-15 所示，事件触发模式和时间计数器触发模式的 PDO 报文传输方式只需要主站向从站不断的发送 PDO 报文片段，直至 PDO 数据发送完毕，在数据传输的过程中不需要从站给出反馈信息。本文主站中经过插补器计算出的需要传送给数字伺服的数据就需要采用时间计数器触发模式。而远程帧触发模式如图 5-16 所示，需要主站先向从站发送远程帧以告知其需要的数据，从站在接收到来自主站的远程帧之后将相应的反馈信息以 PDO 报文发送给主站，在本文研究中，主站用此模式对从站的运行状态进行监视。

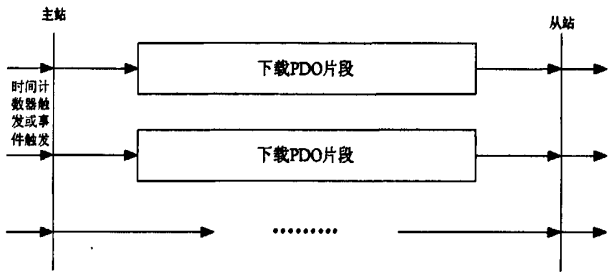


图 5-15 时间计数器触发或事件触发型 PDO 下载

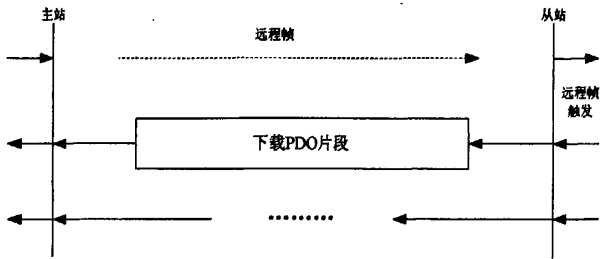


图 5-16 远程帧触发型 PDO 上载

5.6 数字伺服电机的运动控制的实现

如图 5-17 所示，根据第三章中的硬件设计搭建了一个基于 CANopen 协议的数字伺服电机控制平台，应用层芯片在 Alter 公司 Cyclone III 系列型号为 EP3C80 的 FPGA 中实现，从站采用 IDM640-8EIA 型 CANopen 数字智能伺服驱动，编程环境为 QuartusII，采用 VHDL 语言。

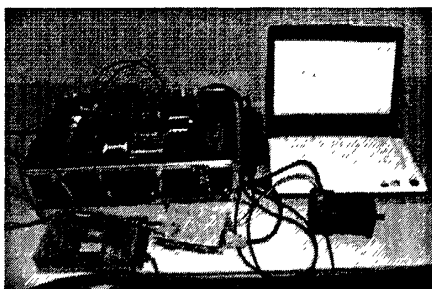


图 5-17 实验系统实物图

在系统调试过程中，通过 PC 中的 QuartusII 编程环境编写 VHDL 程序编译后下载到 FPGA 中，并在 FPGA 中设置存储区，将网络初始化和从节点控制数据存在存储区中，按照主站协议状态机将数据转换成报文发送给从站实现控制。

5.6.1 数字伺服初始参数设定

在进行实验之前，需要对数字伺服驱动的参数和电机参数进行设置，这些都是在 IDM640-8EIA 型 CANopen 智能伺服驱动的专用设置软件 EasySetUp 中进行的。在进行驱动器参数设置时，可以设定 CANbus 的波特率，驱动器的供电参数，控制模式，补偿方式，电机保护等参数信息^[51]，具体参数如图 5-18 所示。

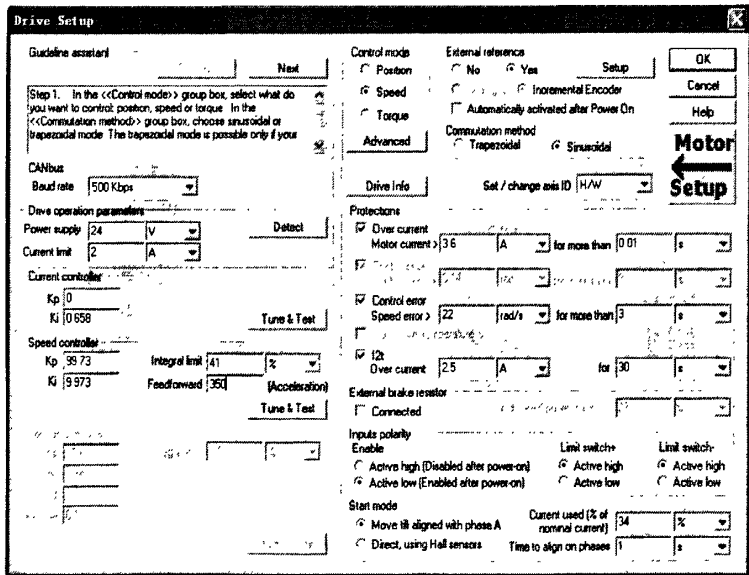


图 5-18 驱动参数设置

在进行电机参数设置时，可以选择电机类型，设置电机参数，编码器参数等，具体参数设置如图 5-19 所示。

图 5-19 电机参数设置

5.6.2 PDO 报文参数设置

要对基于 CANopen 协议的数字伺服电机进行运动控制，CANopen 主站需要首先通过 SDO 对对象字典进行配置，设定 PDO 报文参数，然后按照相应的参数对控制主站中的控制参数加以封装，通过 CANopen 协议发送给从站伺服；同时监控来自从站伺服所传输回来的状态信息，并进行相应的操作。PDO 参数的配置包括两个不同的步骤^[50]：（1）设置 PDO 中的通信参数，如定义 PDO 的标识符与传输类型等；（2）设置 PDO 的映射参数，如定义哪些数据将会通过 CAN 报文的哪些数据位进行传输，这些被传输的数据又具有什么意义等，CANopen 主站和从站需要同时知道其映射参数，以便于二者进行数据交换^[52]。

PDO 通信参数存储在索引号为 1400H -- 15FFH (RPDO) 和 1800H to 19FFH (TPDO) 的对象字典中，其中前四个 PDO 通讯参数是预定义的，后面的可以根据自己的需要去定义^[53]。如图 5-20 所示，访问 PDO 通讯参数实体，可以设置 PDO 的 COB-ID 和传输类型等参数。

PDO 映射参数存储在索引号为 1600H--17FFH (RPDO) 和 1A00H--1BFFH (TPDO) 的对象字典中，其索引值是在其相应的 PDO 通信参数的索引值的基础上加上 200H^[54]。PDO 映射参数描述了 PDO 中携带了多少位有效数据（1-64 位），以及各有效数据位所对应的具体含义，这些信息都在如图 5-20 中 PDO 映射参数表中。从 PDO 映射对象所指引的索引号与子索引号，可以找到对象字典中映射对象的具体参数，从参数

CANopen 总线波特率：500kb/s
电机型号：MT8N57-02E；
驱动器型号：IDM640-8EI CANopen；
驱动电压：24V；驱动电流 2A；
电机驱动模式：速度控制模式。

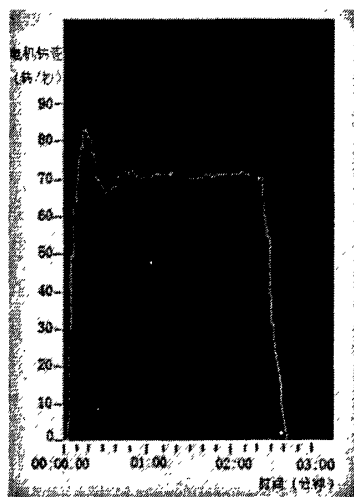


图 5-21 电机转速图

用测速仪对电机转速进行测试结果如图 5-21 所示，电机在 CANopen 主站的控制下启动达到预定速度 70r/s，经过 2.5 分钟后电机停转。实验结果表明 CANopen 主站能成功控制基于 CANopen 协议的数字伺服电机运转。

5.6.4 本章小结

本章着重讨论了 CANopen 主站应用层芯片主要功能模块设计，如 CAN 控制器引脚初始化模块、CAN 报文的接收与解析模块以及 CAN 报文的封装与发送模块；研究了 SDO 与 PDO 的操作流程；实现了数字伺服电机的运动控制。

第六章 总结与展望

基于现场总线的数字伺服技术是自动控制领域发展的一个热点,将其应用于高效、高精度、多轴多通道复合加工数控系统中,能很好的满足其高信息通信速度、高实时性和高可靠性的要求。基于现场总线的数字伺服技术为高效、高精度、多轴多通道复合加工数控系统的实现提供了广阔的发展前景和技术支持。本文通过对基于 CANopen 协议数字伺服装置的运动控制进行研究,提出了一种基于 CANopen 协议数字伺服的数控系统高度通信系统,并对其进行尝试建立了以 FPGA 加 CAN 控制器与收发器为主站以数字伺服为从站的主从通信系统。CANopen 协议是基于 CAN 的应用层协议,在国内该方面的研究和应用还较少,生产基于 CANopen 协议的从站的和主站的公司也不多,这在很大程度上限制了基于 CANopen 协议的总线在国内工业上的使用^[39]。本文主要是对基于 CANopen 协议数字伺服的数控系统速数字通信进行研究,主要完成的工作如下:

(1) 研究了数控系统的国内外发展现状以及当今流行的数字伺服技术和现场总线技术,提出了适用于数控系统上位控制器与多路数字伺服之间高速通信的实施方案。

(2) 深入研究了 CAN 总线技术以及基于 CAN 总线的应用层 CANopen 协议。

(3) 设计了 CANopen 主站的硬件系统,完成了主站电气原理图和 PCB 设计,对系统硬件进行了调试。

(4) 完成 CANopen 主站应用层芯片的功能设计,在 FPGA 中建立了主站应用层芯片的有限状态机调度机制,实现了 PDO、SDO、NMT 以及 Sync 对象的有效调度,完成了基本的通信和网络管理功能。

(5) 建立了以基于数字伺服为从站以 FPGA 加 CAN 控制器与收发器为主站的 CANopen 网络,实现了基于 CANopen 协议数字伺服电机的控制,为其在高效、高精度、多轴多通道新型数控系统中的应用提供了事实依据。

本文只是为高效、高精度、多轴多通道复合加工数控系统的上位控制器与数字伺服之间通信提出了一种通信方案,并对其进行了设计和研究。由于时间有限,该通信系统还存在很多不足的地方,比如 CANopen 网络的可靠性不高,总线数据通信能力有限等,这些都是下一步工作的改进重点。此外,还可以进一步开发 CANopen 网络,将 CAN 网通过网关连入局域网和互联网,这样各从节点的数字伺服电机的状态可以得到更大程度的共享。

参考文献

- [1] 杜君文, 邓广敏.数控技术[M].天津:天津大学出版社, 2002.2.14
- [2] 韩建海.数控技术及装备[M].武汉:华中科技大学出版社, 2007
- [3] 张吉堂等.现代数控原理及控制系统[M].北京:国防工业出版社,2009
- [4] 童教陞.浅谈以个人计算机为基础的数控系统[J].制造技术与机床, 1997, (10):5-7
- [5] 孔亿宾. 浅析数控系统的现状及发展趋势[J]. 机电信息, 2010.18: 47-48
- [6] 郇极, 尹旭峰.数字伺服通讯协议 SERCOS 驱动程序设计及应用[M].北京, 北京航空航天大学出版社, 2005.9
- [7] 杨春杰, 王曙光, 亢红波.CAN 总线技术[M].北京:北京航空航天大学出版社, 2010.2
- [8] 阳宪惠.网络化控制系统[M]. 北京:清华大学出版社,2009.09
- [9] 张剑.基于 SERCOS 总线的数控系统高速数字通信技术研究[D]. 南京航空航天大学硕士学位论文: 3-4
- [10] 王慧锋, 何衍庆.现场总线控制系统原理及应用[M].北京:化学工业出版社工业装备与信息工程出版中心,2006.1
- [11] 奖建文, 林勇, 韩江红.CAN 总线通信协议的分析 and 实现[J].计算机工程, 2002.28(2):219-220
- [12] 卢光宝.基于 CAN 总线的嵌入式多轴数控磨得开发与研究[D].中南大学硕士学位论文: 7-9
- [13] 基于 CAN 的较高层协议和子协议.广州周立功单片机发展有限公司.1-15
- [14] Prof.Dr.-Ing.Gerhard Gruhler,CANopen based Distributed Control System[J]. 1998 The Institution of Electrical Engineers,Print and published by The IEE,Ssvvoy Place,LondonWC2R 0BL,UK.
- [15] 王黎明, 夏立, 邵英等.CAN 现场总线系统的设计与应用[M].北京:电子工业出版社,2008
- [16] 饶运涛, 邹继军, 郑勇芸.现场总线 CAN 原理与应用技术[M].北京:北京航空航天大学出版社,2003
- [17] Duan Jian min, Jin jun, Zhang Ming jie. Frameword of CANopen Protocol for a Hybrid Electric Vehicle [C].MIEEE ,2007, 6: 906- 911.
- [18] 杜尚丰, 曹晓钟, 徐津等.CAN 总线测控技术及其应用[M]. 北京:电子工业出版社,2007
- [19] 孙健, 陶维青.CAN 应用层协议 CANopen 浅析[J].仪器仪表标准化与计量, 2006, 02: 22-24

- [20] CANopen Application Layer and Communication Profile[S]. CiA Draft Standard 301. Version 4.02. 13 February 2002
- [21] 吴爱国, 刘莉. CAN 总线控制系统的应用层协议 CANopen 剖析[J]. 微机计算机信息, 2003, 19 (3) :27-28.
- [22] 宋晓强.CAN bus 高层协议 CANopen 的研究以及在模块化 CAN 控制器上的实现[D].天津大学硕士学位论文
- [23] M .Farsi,K .Ratcliff, Manuel Barbosa.An introduction to CANopen[J] . Computer&Control Engineering, Journal,A ugust 1999: 161-168
- [24] 姜盼.CANopen 协议概述及其在车辆控制系统中的应用[J]. 电脑学习, 2009 .10 , 5
- [25] Mohammad Farsi, Manuel Bernardo Martins Barbosa. CANopen Implementation: applications to industrial networks [M]. Research Studies Press Ltd. 2002
- [26] CANopen Communication Profile for Industry System Based on CAL [S].CiA Draft Standard 301 October 1996
- [27] ID640-8EIA CAN 智能伺服驱动器用户手册 (版本 V1.2) . 泰科智能.12/2008
- [28] MCP2551 Data Sheet [Z]. Microchip.2002
- [29] MCP2510 Data Sheet [Z]. Microchip.2004
- [30] 赵小凤.CAN 在开放式数控系统中的应用[D].兰州大学硕士学位论文
- [31] 刘延飞, 郑锁利, 王晓戎等.基于 Altera FPGA/CPLD 的电子系统设计与工程实践[M]. 北京:人民邮电出版社,2009
- [32] 邵明.多轴嵌入式数控系统中的 CAN 通信系统设计[J]. 福建工程学院学报, 2009 年 6 月第 7 卷第 3 期: 283-284
- [33] 宋威. CANopen 现场总线应用层协议主站的开发与实现[D].北京工业大学硕士学位论文
- [34] 邓遵义, 宁祎. 基于 CANopen 协议的主节点通讯实现[J]. 微机计算机信息, 2008, 8-2 (34): 62—64
- [35] 宋威, 方穗明, 张明杰, 徐喆.任务调度机在 CANopen 主站设计中的应用 [J].计算机测量与控制 2008 年 16 期: 558-560,581
- [36] 张丕状, 李兆光.基于 VHDL 的 CPLD/FPGA 开发与应用[M]. 北京:国防工业出版社,2009
- [37] 赵艳华, 曹丙霞, 张睿. 基于 Quartus II 的 FPGA/CPLD 设计与应用[M]. 北京:电子工业出版社,2009.09
- [38] 潘松, 王国栋.EDA 技术丛书 VHDL 实用教程 (修订版) [M].成都: 电子

科学技术大学出版社, 2001.07

- [39] 孟诏.基于 CANopen 协议的 CAN 总线控制系统研究[D]. 北京工业大学硕士学位论文
- [40] 蒋智康.基于 CANopen 协议的分布式控制系统的研究[D].广西大学硕士学位论文
- [41] 高旭. Windows CE 嵌入式操作系统下的 CANopen 主站实现方案及其应用研究[D]. 河北工业大学硕士学位论文
- [42] 饶怡新.基于 CANopen 协议的智能电动执行机构监控系统主站的研究[D]. 华南理工大学硕士学位论文
- [43] 杜尚丰, 曹晓钟, 徐津. CAN 总线测控技术及其应用[M]. 北京:电子工业出版社,2007
- [44] 郭宽明. CAN 总线原理和应用系统设计[M]. 北京:北京航空航天大学出版社,1996
- [45] 赵飞, 陈冰, 陈幼平.基于 CANopen 协议的同步运动控制器[J].机械与电子, 2010(12): 54-56
- [46] 李华嵩, 李小兵, 董全义.基于 CANOPEN 的智能消防水炮系统设计[J].微计算机信息, 2007, 5-2 (23): 32—34
- [47] 张先庭, 邓洪峰, 陈琼.基于 CANOPEN 协议的印刷机张力控制系统[J].微计算机信息, 2007, 11-2 (23): 51—51
- [48] 刘磊,李长春,田颖,卢青春.基于 CANopen 协议的交流测功机控制系统研究[J].车用发动机 2010 年 6 月第三期(总期 188 期): 21-22
- [49] 陈在平, 王 峰.基于 CANopen 协议从节点研究[J].制造业自动化 2010 年 2 月第二期第 32 卷: 27-28
- [50] CANopen Programming User Manual[Z]. Switzerland:Technosoft S.A.2007
- [51] IDM640-8EIA CAN 智能伺服驱动器用户手册(版本 V1.2). 泰科智能.12/2008:52-58
- [52] 李澄,赵辉.CANopen 协议及在电动机控制系统中的应用[J].微电机 2009 年第 42 卷第 4 期: 24-25
- [53] 胥布工,程俊,匡付华.基于 CANopen 协议的电动执行机构设计[J].控制工程 2010 年 5 月第 3 期第 17 卷: 369-370
- [54] 尹洪胜, 于宁宁, 赵宗平等.基于 CANopen 的煤矿压力在线监测系统[J].煤矿安全, 2010.03: 73-74

攻读硕士学位期间发表的论文

1. 韩江, 黄涛, 董伯麟, 夏链. 基于 CANopen 协议的数字伺服电机通讯主站研究[J]. 轻工机械. 2012.1