题目：基于 FPGA 的 RFID 读写器通讯虚拟检测系统的设计

摘　要

RFID 技术是一种新兴的自动识别技术，具有信息量大、读取距离远、可同时读取多张卡片等特点，被广泛应用于门禁、物流、管理等领域。

虚拟仪器是现代计算机技术和仪器技术深层次结合的产物。虚拟仪器充分利用了计算机的运算、存储、回放显示及文件管理等智能化功能，同时把传统仪器的专业化功能和面板控件软件化，使之与计算机结合构成一台功能完全与传统硬件仪器相同，同时又充分享用了计算机软硬件资源的全新虚拟仪器系统。

Wiegand 协议和 ABA 协议作为一种常用的通讯协议被广泛的应用于 RFID 读卡器与上位机之间的通讯以及 RFID 读卡器与控制器之间的通讯。本设计的目的是检测 Wiegand 协议和 ABA 协议的数据通信是否符合协议规定，主要包括脉冲宽度、脉冲间隔等。本设计包含 FPGA 和上位机软件两部分，FPGA 上完成对信号的采样和对采样数据的储存和缓冲，上位机完成对采样数据的处理，以及波形的显示。FPGA 上的设计应用 Verilog 语言在 Altera 公司的 Max+PlusII 平台上进行开发。上位机软件设计基于 NI 公司的图形化编程软件 LabVIEW。

关键词：RFID、Wiegand 协议、ABA 协议、Max+PlusII、LabVIEW

Design of a FPGA-based Virtual RFID Reader Communication Protocol Testing System

ABSTRACT

Radio Frequency Identification(RFID) is a new rising automatic identification technology. It has many promising features such as large information capacity, long effective distance and multiple tags operation and is widely used in access control, logistics and other management area.

Virtual Instrument (VI) is the combination of modern computer and instrument technology. By fully utilizing the computing, storage, display and file management capacity of modern computer and realizing the original professional function and deck control of traditional instruments with software, VI turns modern computer system into a full functional instrument system with abundant software and hardware resources.

The Wiegand and ABA protocol are two major communication interface format used to link the RFID reader to host PC or controller. This paper researches a testing system to verify the integrity, validity and timing conformity of Wiegand and ABA communication. The testing system completes the major characteristics parameters measure of the two protocol such as pulse width, pulse interval etc. It comprises FPGA and host software two major parts. FPGA part carries out the signal sampling, FIFO buffer and storage. Host software completes digital processing of sampled signal and displaying of the waveform. FPGA part is designed with Verilog hardware description language under Altera's Max+plus II environment. Host software is developed with National Instruments' graphic program toolkit Labview.

Keywords: RFID、Testing System 、Wiegand protocol、ABA protocol、Verilog、Max+PlusII、LabVIEW

插图清单

表格清单

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：夏乌根　签字日期：2017年5月21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：夏乌根　　导师签名：

签字日期：2017年5月21日　　签字日期：2017年5月29日

学位论文作者毕业后去向：

工作单位：　　电话：

通讯地址：　　邮编：

致 谢

首先感谢我的导师解光军教授，导师从一开始就很尊重我的兴趣发展，在学习上给我指引了前进的方向，并在各方面给予了大力的支持，使我能从对可编程逻辑器件一无所知到现在能运用到设计上，顺利完成毕业论文。研究生期间，导师尽量的给我提供实践机会，使我的动手能力得以提高。导师严谨的治学态度、渊博的知识和孜孜不倦的教诲使我终生受益。在此，向导师表示深深的感谢，谢谢导师一直以来在学习上和生活上给予我的关心和帮助！

感谢邹谊博士，在公司实习的时间里，邹博士给予了我最无私的帮助，指导我的学习和工作，使我在实习期间开阔了眼界，增长了知识。

感谢在座的各位老师，我的成长离不开各位的教诲。

感谢学习和实习期间一直给予我帮助的师兄范海秋、操礼程，同学顾云海、陈树海、王新亚、司俊丽、信磊、程松、刘炳龙、王凯、贺克军、曹冰、罗正平等。以及各位师弟师妹对我的支持。

作者：夏禹根

2007 年 5 月 10 日

第一章 绪 论

20世纪80年代美国NI公司首先提出了虚拟仪器的概念，提出了“软件即仪器”的口号，从此一种全新的测量方式进入了人们的视野[1]。经过了二十年的不断完善与发展，同时随着计算机数据处理能力的不断强大，使得虚拟仪器的发展进入了全新的篇章，处理速度不断加快，精度不断进步；随着网络的普及和笔记本电脑功能的强大，虚拟仪器正在向着便携设备和网络化发展，相信在不远的明天“将实验室建在网上”将不再是一句口号[2]。

虚拟检测仪是一种检测信号是否符合通讯协议的设备，可以全面而快捷的检测信号的各项指标，诸如脉冲宽度，脉冲间隔等，同时支持长时间不间断的开机通讯实验，满足对读卡器检测的各种要求。

1.1、引言

RFID（Radio Frequency Identification）作为一种新兴的识别方法具有信息量大、读取距离远、读取速度快、可同时读取多张卡片等特点，在门禁、物流管理、库存管理、物品追踪等领域发挥着越来越大的作用，正在渐渐走入人们生活的各个领域，以它的便捷和高效改变着人们的生活。

一个典型RFID系统有以下几部分组成，电子标签、天线、读卡器和服务器。具体结构如图1-1所示：

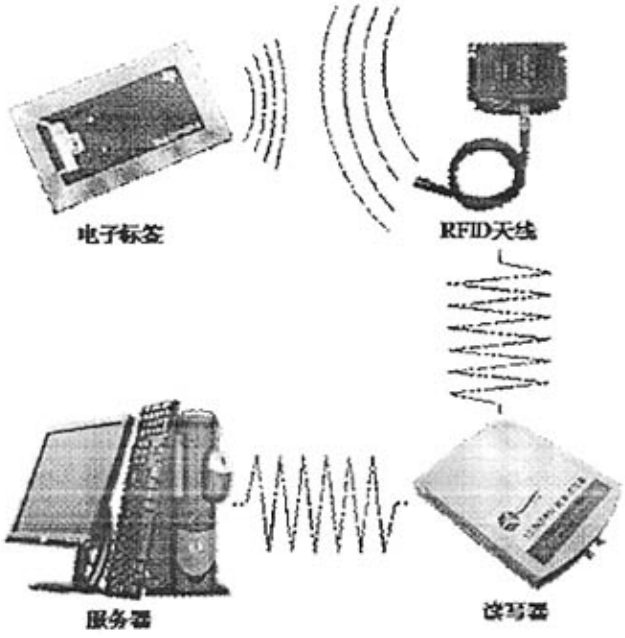


图1-1 RFID典型系统组成

其中电子标签和天线之间通过电磁耦合传递数据，读卡器和天线之间通过固定频率的载波传递数据，而读卡器与服务器之间传递的数据正是虚拟检测仪要检测的数据，它的正确性直接关系到读卡的结果和服务器的工作，是一种直接面

向用户的数据[3]。此外，RFID 系统还有一种典型结构包括

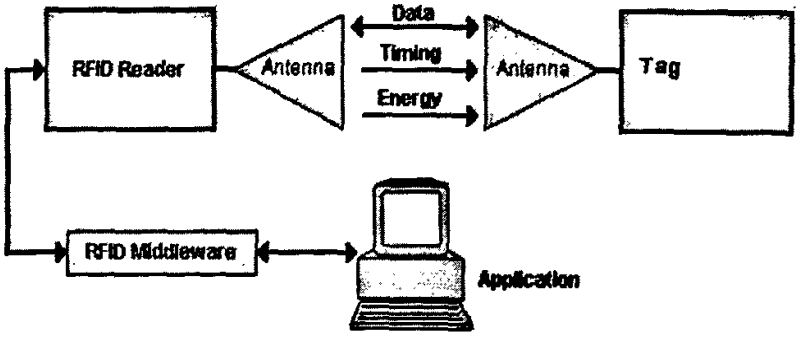


图1-2 带有中间件的RFID系统组成：

其中中间件是将底层 RFID 硬件和上层企业应用结合在一起的粘合剂，读卡器控制器就是一种典型的中间件，在读卡器和中间件之间传递的数据也是虚拟检测仪的检测对象[4]。

1.2、读卡器常用通讯协议介绍

1.2.1、韦根（Wiegand）协议[5]

韦根（Wiegand）协议又可称作韦根码，是一种 TTL 电平的串行通讯协议，它是由美国安全工业协会 SIA（Security Industry Association）规定的读写接口控制协议。在门禁、安防、考勤以及与之相关的其他行业中，韦根码作为一种读卡设备与上位机之间的通信介质得到了广泛的应用。韦根码在数据的传输中需要两条数据线，一条为 DATA0，另一条为 DATA1。协议规定，两条数据线在无数据时均为高电平，如果 DATA0 为低电平代表数据 0，DATA1 为低电平代表数据 1（低电平信号低于 1V，高电平信号高于 4V），DATA0 和 DATA1 上的数据不能重叠或同步，其数据信号波形如图 1-3 所示，另外有一条 HOLD 信号线，平时为高电平当有卡时为低电平，两张电子卡片之间间隔 0.25s 以上。

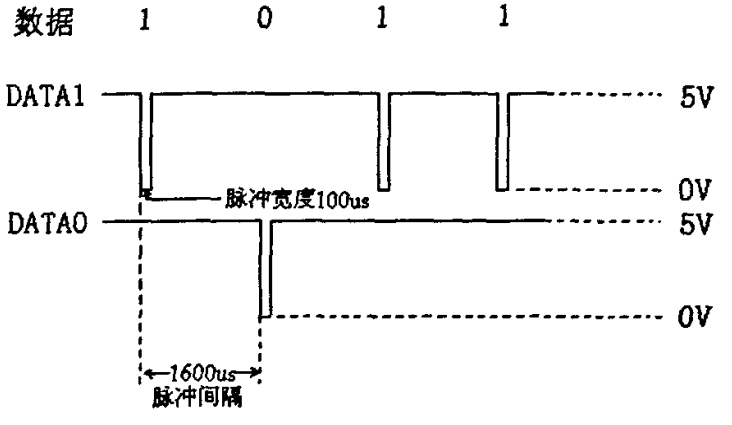


图1-3 Wiegand信号波形图

韦根码有多种数据格式，本文的检测对象是读头中较为常用的韦根 26bit 数据格式。该数据格式如表 1-1。表 1-1 中第 2 到第 9 位为分组码，分组码共有 8 个二进制位，有 256 个状态；第 10 到第 25 位为标识码，标识码共 16 个二进制位，有 65536 个状态；第 1 位是第 2 到第 13 位的偶校验位；第 26 位是第 14 到第 25 位的奇校验位。

位数	意义
第 1 位	第 2 位到 13 位的偶校验位
第 2~9 位	分组码（0~255），第 2 位是高位（MSB）
第 10~25 位	标识号（0~65535），第 10 位是高位（MSB）
第 26 位	第 14 位到第 25 位的奇校验位

表 1-1　Wiegand26bit 数据格式

1.2.2、ABA 协议[6]

ABA 协议的格式是基于传统的磁卡第二轨道输出格式（ABA TRACK2 DATA FORMAT），在一些上世纪的使用磁卡的系统中作为标准格式得到极大应用（包括 Wiegand 格式也一样）。为保证兼容，后期的非接触 IC 卡读卡设备也常使用这一格式作为输出格式，具体格式解释如下：

1、引导位：10bit“0”

2、开始位：5bit 为 16 进制“B”“11010”

3、数据位：每个数由 5 位组成，可以不止 10 个数

4、结束位：5bit 为 16 进制“F”“11111”

5、校验位：5bit

6、结尾：5bit“0”

说明：

数据（包括开始位）为 5 个 bit 一组，其中前四个为有意义的数据，最后一位为前四位的奇校验，并且在传输时数据的低位在前。

例如一次读卡传出的整个数据结构如下（全部是二进制）：

0000000000 11010 XXXXP XXXXP XXXXP XXXXP XXXXP XXXXP XXXXP

引导位 开始位 数据位

XXXXP XXXXP XXXXP 11111 LRC 00000

数据位 结束位 校验位 结尾

其中“X”为数据，“P”为前四位数据奇校验。

校验位（LRC）的算法如表 1-2：

	B0	B1	B2	B3	P
开始位	1	1	0	1	0
3个数据的数据位（123）	1	0	0	0	0
	0	1	0	0	0
	1	1	0	0	1
结束位	1	1	1	1	1
校验位	0	0	1	0	0

表 1-2　LRC 算法表

传输数据使用 3 根数据线，分别是 DATA、CLOCK、CP

CP：card present 当有卡时，会发一个下拉宽脉冲。

DATA 线和 CLOCK 线发送二进制数据，CLOCK 是间隔 1ms 的脉冲，宽度 100μs，拉低时有效，如拉低时 DATA 为高，则输出 0，反之则输出 1。

1.3、虚拟仪器及其发展现状

虚拟仪器（VI）是随着计算机技术、现代测量技术发展起来的新型高科技产品，代表着当今仪器发展的新方向。虚拟仪器的概念是由美国国家仪器（National Instruments）公司首先提出，是对传统仪器概念的重大突破。其具体概念可具体描述为“虚拟仪器是利用现有的 PC 计算机、加上特殊设计的仪器硬件和专用软件，形成既有普通仪器的基本功能，又有一般仪器所没有的特殊功能的新型仪器”[7]。虚拟仪器技术给用户一个充分发挥自己的才能和想象力的空间，用户（而不是仪器厂家）可以根据自己的需求设计自己的仪器系统，从而满足多种多样的应用需求[8]。

其基本体系结构可表示为图 1-4[9]：

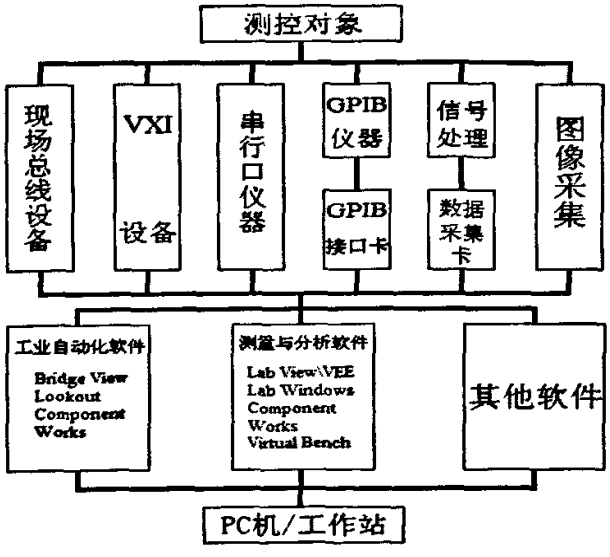


图1-4　虚拟仪器体系结构图

1.4、开发虚拟检测仪的必要性

读卡器与上位机或中间件之间的数据通讯直接影响着整个RFID系统工作，影响着应用层对卡片数据的获取和处理，若 RFID 系统应用在门禁，物流管理等领域，对卡片信息的误操作可能带来很大的损失。

同时，RFID 系统应用的应用环境一般比较恶劣而且多数情况需要整个系统长时间不间断的工作，这就对 RFID 系统的稳定性提出了很高的要求，本文设计的虚拟检测仪在完成对数据脉冲宽度、脉冲间隔等指标的检测的同时，还可以完成对读卡器的稳定性测试，可以实现长时间不间断的自动检测，保证了整个 RFID 系统的稳定。

1.5、本文的主要工作

本文主要工作可分为 FPGA 设计和上位机软件设计两部分，在 FPGA 上设计检测仪的主要功能和通讯模块，通过上位机软件对采样数据进行分析处理并显示出来，同时设计一个友好的人机界面。

FPGA 的设计基于 Altera 公司的开发平台 Max+Plus II，主要的开发语言是 Verilog 语言。

上位机软件设计主要应用 NI 公司开发的虚拟仪器软件 LabVIEW。

第二章　可编程逻辑器件

可编程逻辑器件（Programmable Logic Device，PLD）是当前数字系统设计的主要硬件基础，是硬件编程语言的物理实现工具，对数字系统设计自动化起着重要作用。

当今社会是数字化的社会，是数字集成电路广泛应用的社会。数字集成电路本身在不断进步。它由早期的电子管、晶体管、小中规模集成电路发展到超大规模集成电路以及许多具有特定功能的专用集成电路。但是，随着微电子技术的发展，设计与制造集成电路的任务已不完全由半导体厂商来独立承担。系统设计师们更愿意自己设计专用集成电路（ASIC）芯片，而且希望 ASIC 的设计周期尽可能的短，最好是在实验室里就能设计出合适的芯片，并且立即投入实际应用之中，因而出现了现场可编程逻辑器件（FPGA），其中应用最广泛的当属现场可编程门阵列（FPGA）和复杂可编程逻辑器件（CPLD）[10]。

2.1、简单 PLD 的基本结构[11]

大部分简单低密度 PLD 器件的主体是“与”阵列和“或”阵列，根据这两个阵列的可编程性，简单 PLD 器件又分为 3 种，即“与”阵列固定“或”阵列可编程；“与”阵列可编程“或”阵列固定；“与”、“或”阵列均可编程。

利用布尔代数我们可以将任意的逻辑表达式表示成积之和的形式，因此通过提供可编程与/或阵列，逻辑可以被定制来实现任意特定的应用。如图 2-1 所示，带有反馈的输入项。每个输入变量及其反变量都可连接到与门上。与门阵列的输出列接到独立的或门，从而形成用户定义的布尔表达式。

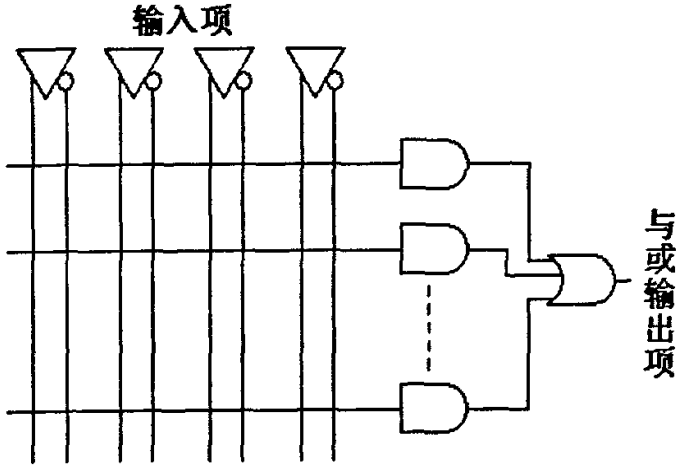


图2-1　带有输入反馈与或阵列

2.2、现场可编程逻辑门阵列（FPGA）

现场可编程逻辑门阵列（FPGA）的基本属性是一个细粒度架构，包含一个小逻辑单元的阵列。与 PAL、GAL 器件相比，它的优点是可以实时地对外加或内置的 RAM 或 EPROM 编程，实时地改变器件功能，实现现场可编程（基于 EPROM 型）或在线重配置（基于 RAM 型），是科学实验、样机研制、小批量产品生产的最佳选择器件[12]。

2.2.1、基本结构及分类[13]

FPGA 在结构上包含 3 类可编程资源：可编程逻辑功能块（Configurable Logic Block，CLB），可编程 I/O 块（I/O Block，IOB）和可编程互连（Interconnect Resource，IR）。如图 2-2 所示，可编程逻辑功能块是实现用户功能的基本单元，它们通常排列成一个阵列，散布于整个芯片；可编程 I/O 块完成芯片上逻辑与外部封装脚的接口，常阵列于芯片四周，可编程内部互连包括长度的线段和编程连接开关，它们将各个可编程逻辑快或 I/O 块连接起来，构成特定功能的电路。不同厂家生产的 FPGA 在可编程逻辑块的规模、内部互连线的结构和采用的可编程元件上存在较大的差异。

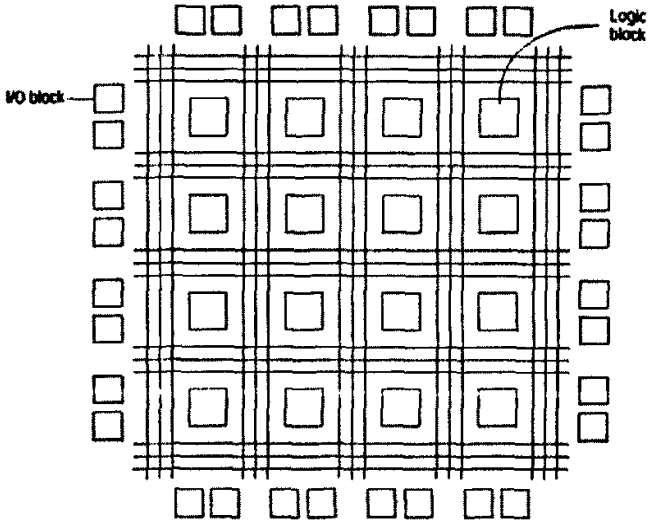


图2-2　FPGA基本结构

常见的 FPGA 可分为以下 3 种结构：查找表结构、多路开关结构、多级与非门结构。

1、查找表型 FPGA

查找表型 FPGA 可编程逻辑功能块是查找表 LUT（Look Up Table），有查找表构成的函数发生器，通过查找表来实现逻辑函数。LUT 本质上就是一个 RAM，n 个输入项的逻辑函数可以由一个 2^n 位容量的 RAM 实现，函数值存放在 RAM 中，RAM 的地址线起输入线的作用，地址即输入变量值，RAM 输出为逻辑函数值，由连线开关实现与其他功能块的连接。目前 FPGA 中多使用 4

输入的 LUT，所以每一个 LUT 可以看成一个有 4 位地址线的 16×1 的 RAM。当前用户通过原理图或硬件描述语言描述了一个逻辑电路后，FPGA 开发软件会自动计算逻辑电路的所有可能的结果，并把结果事先写入 RAM，这样，每输入一个信号进行逻辑运算就等于输入一个地址进行查表，找出地址对应的内容，然后输出即可。查找表型 FPGA 产品有 Altera 的 ACEX、APEX、FLEX 10K 系列，Xilinx 的 Spartan、Virtex 系列等。

2、多路开关型 FPGA

在多路开关型 FPGA 中，可编程逻辑功能块是可配置的多路开关。利用多路开关的特性对多路开关的输入和选择信号进行配置，接到固定电平或输入信号上，从而实现不同的逻辑功能。

3、多级与非门型 FPGA

多级与非门型 FPGA 结构基于一个与-或-异或逻辑块，该基本电路可以用一个触发器和一个多路开关来扩充。多路开关选择组合逻辑输出、寄存器输出、锁存器输出。异或门用于增强逻辑块功能，当异或门输入端分离时，其作用相当于或门，可以形成更大的或函数，用来实现其他算术功能。

2.2.2、现状及发展[12]

可编程逻辑器件正处于高速发展阶段。新型的 FPGA 规模越来越大，成本越来越低。高性价比使可编程逻辑器件在硬件设计领域扮演着日益重要的角色。低端 CPLD 已经逐步取代了 74 系列等传统的数字元件，高端的 FPGA 也在不断的夺取 ASIC 的市场份额，特别是目前大规模 FPGA 多数支持可编程片上系统（SOPC），与 CPU 或 DSP Core 的有机结合使 FPGA 已经不仅仅是传统的硬件电路设计手段，而逐步升华为系统级实现工具。

下一代可编程逻辑器件硬件上有 4 大发展趋势：最先进的 ASIC 生产工艺将被更广泛的应用于以 FPGA 为代表的可编程逻辑器件；越来越多的高端 FPGA 产品将包含 DSP 或 CPU 等处理器内核，从而 FPGA 将由传统的硬件设计手段逐步过渡为系统级设计平台；FPGA 将包含功能越来越丰富的硬核（Hard IP Cord），与传统 ASIC 进一步融合，并通过结构化的 ASIC 技术加快占领部分 ASIC 市场；低成本 FPGA 的密度越来越高，价格越来越合理，将成为 FPGA 发展的中坚力量。

2.3、FPGA 器件的选型

本文系统设计采用了 Altera 公司的 FLEX 10K 系列器件 EPF10K100E。

2.3.1、器件特点及基本结构[14]

FLEX 10K 是 Altera 公司研制的第一个嵌入式的 PLD，它具有高密度、低成本、低功率等特点。它采用了重复可构造的 Cmos SRAM 工艺，并把连续的快速通道互连与独特的嵌入式阵列结构相结合，同时可结合众多可编程器件来

完成普通门阵列的宏功能。每一个 FLEX 10K 器件均包括一个嵌入式阵列和一个逻辑阵列，因而设计人员可轻松地开发集存贮器、数字信号处理器及特殊逻辑等强大功能于一身的芯片。

EAB（embedded array blocks）的概念源于门阵列的嵌入式功能，为了使复杂的功能在尽可能小的硅片上得以实现，通常需把定制的硅片放在门阵列基片之上。Altera 公司首先把这一技术应用于 FLEX 10K 器件系列。FLEX 10K 中的嵌入式阵列由一系列具有实现逻辑功能和存贮功能的 FAB 组成。EAB 是在输入、输出口上带有寄存器的 RAM 块，利用它可以非常方便地实现一些规模不太大的 ROM、RAM、双端口 RAM 和 FIFO 等功能。

FLEX 10K 器件的特性如下[15]：

1、嵌入式可编程器件可提供集成系统与单个可编程逻辑器件性能；

2、密度高，可提供 1 万～25 万个可用门、6144～40960 位内部 RAM；

3、功耗低：多数器件在静态模式下的电流小于 0.5mA；

4、速度高：器件内含树形分布式低失真时钟，并具有快速建立时间和时钟到输出延时的外部寄存器；

5、具有灵活的互连方式，采用快速、互连延时可预测的快速通道连续式分布结构；

6、支持多电压 I/O 接口，遵从 PCI2.2 总线标准；

7、具有多种配置方式和多种封装形式。

FLEX 10K 的体系结构类似于嵌入式的门阵列。不过嵌入式门阵列不能进行用户定制，从而限制了设计者的选择范围。相反，FLEX10K 器件是可编程的，它提供给设计者对嵌入式多功能以及通用逻辑的控制，方便了在调试过程中一遍一遍的设计修改。

FLEX 10K 器件结构简图如图 2-3 所示：

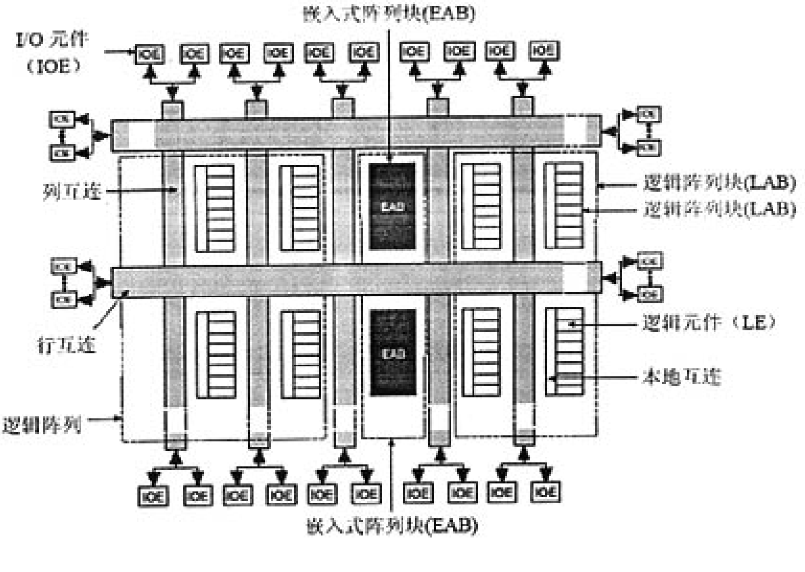


图2-3 FLEX 10K器件结构简图

每个 FLEX 10K 器件包含嵌入式阵列和逻辑阵列两部分。嵌入式阵列用来实现大量的内存功能和复杂的逻辑功能（例如数字信号处理、微控制器、数据转换功能等）；逻辑阵列用来实现通用逻辑（例如计数器、加法器、状态机和多路复用器等等）。 FLEX 10K 器件的配置是在系统加电的时候完成的，通过 Altera 的串行配置设备（例如 EPC2,EPC1,EPC1441）通过串行数据流的方式对 FLEX 10K 进行配置。配置数据也可以从系统内存下载或者通过 Altera 的 BitBlaster 串行（或 ByteBlaster 并行）下载电缆进行下载[16]。

2.4、FPGA 的典型开发流程[11]

使用 FPGA 器件为主要器件实现的数字系统的开发设计的流程主要有以下三种：

- 自顶向下设计法（Top-Down）
- 自底向上设计法（Bottom-Up）
- 混合式设计法（Inside-Out）

本设计采用自顶向下设计法。

在自上而下的设计中，将设计分成几个不同的层次：系统级、功能级和门级等。然后，按照自上而下的顺序，在不同的层次上对系统进行设计、描述与仿真。

自上而下的设计须经过“设计——验证——修改设计——再验证”的过程，不断反复，直到得到的结果能够完全实现所要求的逻辑功能，并且在速度、功耗、价格和可靠性方面实现较为合理的平衡为止。

自上而下的设计也并非是绝对的，在设计的过程中，有时也需要用到自下而上的方法。这种方法是在系统划分和分解的基础上，先进行底层单元的设计，然后再逐步向上进行功能块和子系统的设计，直至构成整个系统。

第三章　虚拟检测仪的具体设计

本文所述虚拟检测仪的设计可以从总体上分为两个部分，即 FPGA 部分和上位机软件部分。本章的叙述重点为 FPGA 部分的设计，这也是整个虚拟检测仪的核心部分.本设计应用时下比较流行的硬件描述语言 Verilog 编写,在 Altera 公司开发的 Max+PlusII 平台上进行设计。Max+PlusII 是一款集设计、仿真、编程于一体的工具软件，它提供了无可比拟的灵活性和高效性，具有丰富的图形界面和完整的可即时访问的在线文档，同时支持 Synopsys、Viewlogic、Menter Graphic、Cadence、OrCAD 等公司提供的开发工具接口[11]。

3.1、虚拟检测仪的系统结构

虚拟检测仪的 FPGA 部分主要实现了数据的采样、解码、数据流缓冲、存储以及串行通信功能。上位机软件为使用者提供了一个人性化的操作界面，同时对 FPGA 采样得到的数据进行分析和储存，并显示给使用者看。利用计算机日渐强大的功能对采样数据进行一系列的分析，得到脉冲间隔，脉冲宽度等信息，以供使用者参考[17]。

FPGA 上系统电路图 3-1 所示：

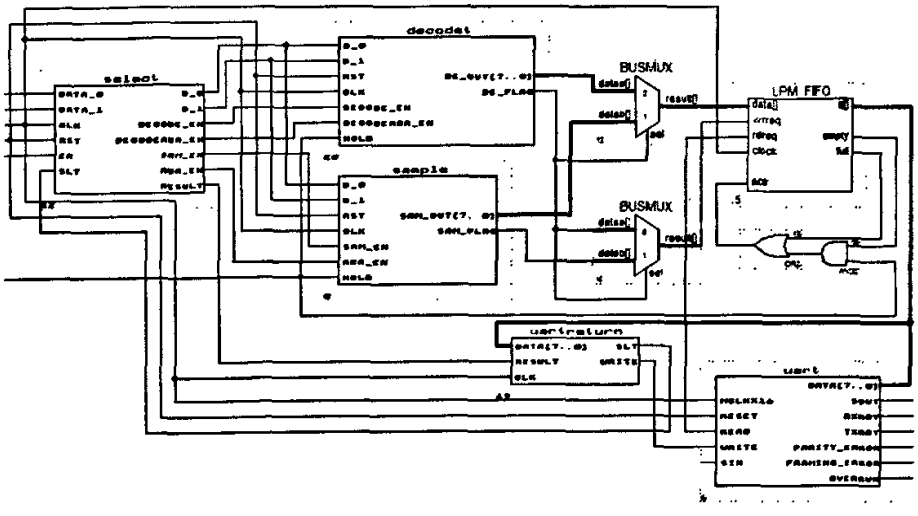


图3-1　FPGA部分系统设计图

3.2、虚拟检测仪的功能模块

如图 3-1 所示，这个设计可以分为以下四个部分：

1、检测功能模块：主要用于对输入信号采样，同时完成对 Wiegand 信号和 ABA 信号的解码。

2、FIFO 模块：主要用于数据缓冲和数据存储，由采样和解码得到的数据先存在 FIFO 中，然后经由 UART 模块传送到上位机。

3、UART 模块：主要用于与上位机之间的串行通信，采用通用通讯格式即：起始位+8 位数据+结束位。

4、控制模块：用于总线中数据流的控制，同时完成 Wiegand 检测和 ABA 检测模式之间的切换控制。

各个模块之间的数据流以及时钟流动如图 3-2 所示：

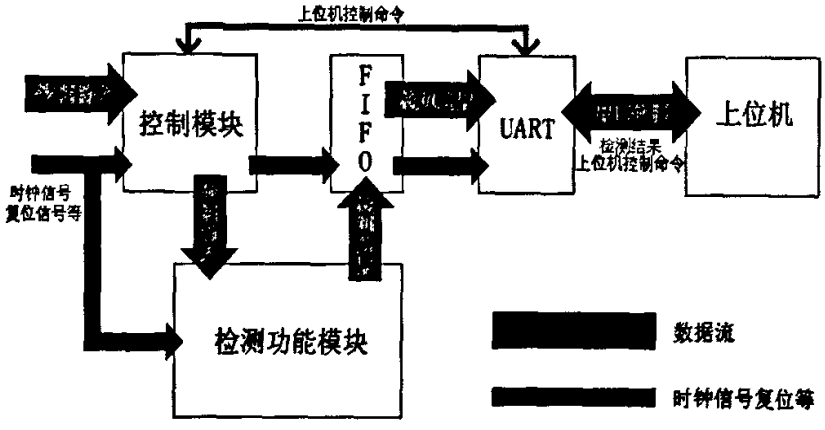


图3-2　数据及时钟流动图

系统上电时通过 UART 接受上位机的配置信息，根据配置信息决定对 Wiegand 信号还是 ABA 信号进行检测。配置控制模块结束时，控制模块根据配置信息将对应得数据输入检测功能模块。这时检测功能模块对数据进行采样，同时进行数据的解码，将解码结果暂时存放在解码模块中。采样结果实时的存入 FIFO 中，解码数据在一帧采样数据结束后再存入 FIFO 中。FIFO 中的数据经由 UART 以串行通信方式传递给上位机。各模块的时钟信号和复位等信号由外部输入。

下面将分别叙述个功能模块的 Verilog 实现。

3.3、检测功能模块的 FPGA 设计

检测功能模块主要用于对输入信号进行采样，采样得到的结果传输到上位机，通过软件的重建和计算、分析得到脉冲宽度、脉冲间隔等所需要的数据。

解码功能由单独的解码模块来完成，由于 Wiegand 信号和 ABA 信号解码方式的不同，解码模块分别由 Wiegand 解码和 ABA 解码组成。

3.3.1、主要功能

检测功能模块包括两部分功能：

1、对输入信号进行采样输出采样结果；

2、对信号进行解码将解码结果输出。

检测功能模块的两个功能对应着两种功能模块，各功能模块的 Max+PlusII

模块图如图 3-3 所示，图中显示了两种模块的输入输出引脚图。其中解码模块由 Wiegand 解码和 ABA 解码两个模块组成。

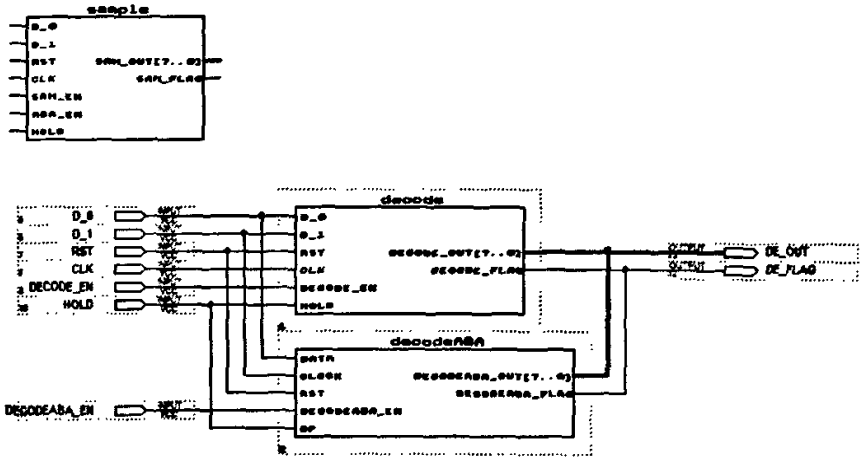


图3-3　检测功能模块结构图

其中 D_0、D_1、HOLD 为数据输入，CLK 为时钟输入，RST 是外部复位端低电平有效，SAM_EN 和 DECODE_EN 分别为两个模块的使能端高电平有效，DECODEABA_EN 为 ABA 信号解码使能，高电平有效。

输出部分，SAM_OUT 和 DE_OUT 分别为 Sample 模块和 Decode 模块的数据输出，均为 8 位输出，SAM_FLAG 和 DE_FLAG 分别为各自模块的输出标志位。

两个模块采用相同的时钟输入和外部复位，其时钟频率（采样频率）为串口通讯波特率的 4 倍，19200Hz×4。由外部输入的时钟分频得到。下面将详述两个模块的 Verilog 设计。

3.3.2、采样模块

采样模块的设计基于可综合的同步有限状态机（FSM）技术。有限状态机是功能强大的设计元件，可用于用硬件实现算法。一个有限状态机包括一个状态向量、一组寄存器，以及一些能够根据外部输入和当前状态计算下一个状态的相关逻辑。这些逻辑就像是每个时钟周期的未来状态变化依赖外部的输入和当前状态变化。在这点上，有限状态机就好像用微处理器的程序计数器控制排序的一系列指令。其中每一个状态都被设计成可以完成任意不同的行为，同时可以跳转到其他不同的状态，这与根据算法的指示软件跳转到不同的程序部分的概念相同。如果一个问题能够被分解成多个确定的逻辑步骤，那么这个算法可以用一个有限状态机实现。

下面以 Wiegand 信号为例，说明采样模块的工作，当被检测信号为 ABA 信号时，ABA_en 置高，这时 hold 信号线接在 ABA 信号中的 CP 信号线上，D_0 和 D_1 信号线分别接在 ABA 信号中的 DATA 和 CLOCK 上。当 CP 信号出现下跳变时表示一帧数据开始，当 CP 信号出现上跳变时表示一帧数据结束。

采样模块的工作与 Wiegand 信号时相同。

采样模块的状态图如图 3-4 所示：

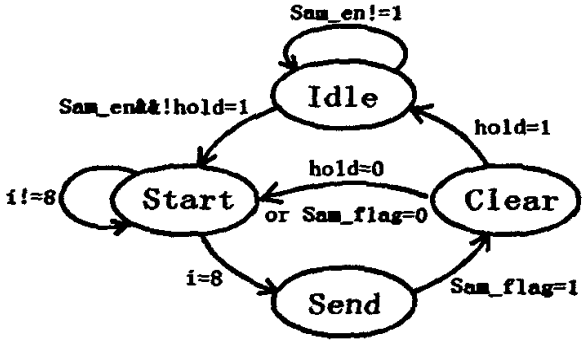


图3-4　采样模块状态图

由图中可知，采样模块状态机由四个状态组成：Idle、Start、Send、Clear。

①Idle 状态：

模块的起始状态，当开机或复位之后，模块进入 Idle 状态，当模块的使能端 Sam_en 置高时，同时 Hold 输入线出现下跳变，并保持下调变 8 个时钟周期不变，这时认为 Hold 信号有效，模块由 Idle 状态转入 Start 状态。

②Start 状态：

模块的工作状态，在时钟上升沿对两个输入 D_0 和 D_1 进行采样，将结果按顺序存放在 8 位寄存器中，用一个计数器 i 负责对采样的位数计数，当 i 不等于 8 时表示 8 位寄存器还没有填满，状态机继续停留在 Start 状态，直到 i 等于 8，8 位寄存器填满为止。当 i 等于 8 时，模块进入下一个状态 Send。

③Send 状态：

在 Send 状态中 8 位寄存器中的数据被送到输出线上，由于 Wiegand 信号的特殊性，D_0 和 D_1 信号线上的数据被分别采样后，在 Send 状态中经过同或运算后将最后的结果输出到输出线上，同时输出标志位 Sam_flag 置高，标志着输出数据线上出现了有效的数据，通知下面的 FIFO 模块将数据存储。由于在状态机内部各语句是用顺序语句编写的，所以将 Sam_flag 置高的语句出现在状态的最后，同时也标志着模块进入了最后一个状态 Clear。

④Clear 状态：

对在前面使用过的计数器，寄存器清零，准备迎接下一个 Byte 数据的到来，Clear 状态同样是用顺序语句编写，在状态的最后将 Sam_flag 置低，同时检测 Hold 线上的数据输入，若 Hold 线保持低电平不变，则说明一帧数据还没有结束模块由 Clear 状态回到 Start 状态继续下一个 Byte 的采样，若 Hold 线出现上跳变，则说明一帧数据结束，模块由 Clear 状态回到 Idle 状态等待新一帧数据的到来。

下面给出状态机转换的 Verilog 代码[18]：

```
module sample(D_0,D_1,Rst,clk,Sam_en,ABA_en,hold,Sam_out,Sam_flag);
       input D_0,D_1,Rst,clk,Sam_en,hold,ABA_en;
       output[7:0] Sam_out;
       output Sam_flag;
       reg Sam_end;
       reg Sam_flag;
       reg[7:0] a,b;
       reg[7:0] Sam_out;
       reg[3:0] i;
       reg[2:0] state;

parameter Idle  = 'b000;
          Start = 'b001;
          Send  = 'b010;
          Clear = 'b011;

always@(posedge clk)
    if(!Rst)
        begin
          state = 'b000;
          i = 4'b0000;
          a = 0; b = 0; Sam_out = 0;
          Sam_flag = 0;
          Sam_end = 0;
        end
    else if(!hold||ABA_en)
      case(state)
        Idle: begin
                if(Sam_en) begin
                           state = Start;
                           i = 4'b0000;
                           end
                else state = Idle;
                end
        Start: begin
                if(i==8)
                  state = Send;
                else
                  state = Start;
                end
        Send:  begin
                 if(Sam_flag)
                    state  = Clear;
                 end
        Clear: begin
                  i = 1;
                  a = 0; b = 0;
                  if(hold)
                    state = Idle;
                  else
                    state = Start;
                end
      endcase

endmodule
```

图 3-5 是采样模块 Max+PlusII 仿真截图：

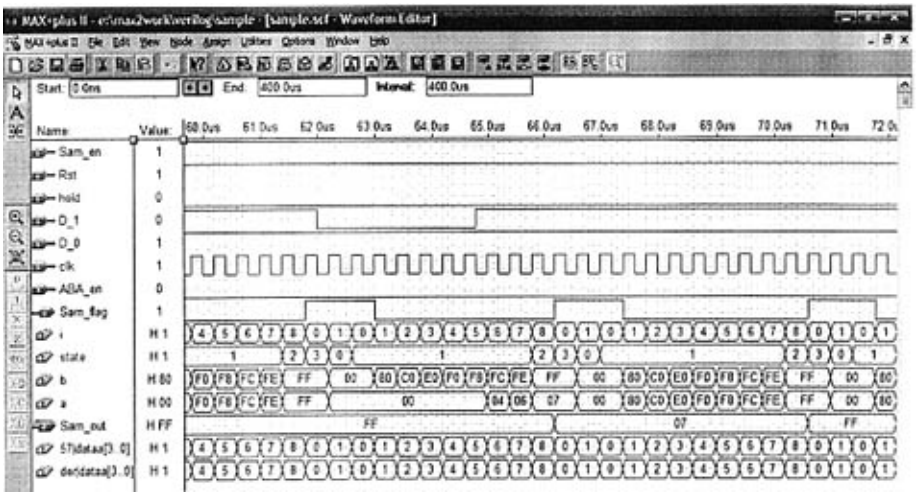


图3-5　采样模块仿真图

由上图可知采样模块实现了预先设计的功能，完成了对输入信号的采样。

3.3.3、解码模块

解码模块包含 Wiegand 解码和 ABA 解码两部分，其模块图如图 3-6 所示：

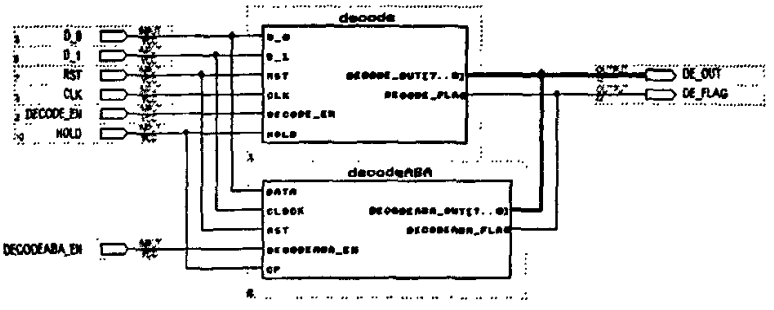


图3-6　解码模块结构图

下面分别叙述两部分的 FPGA 设计。

3.3.3.1、Wiegand 解码模块

Wiegand 解码模块的主要功能是对 Wiegand 信号进行解码，并将解码结果输出。在解码模块内部定义了一个存储器容量为 4×8Byte，用于存储解码结果，当一帧数据全部结束时 Hold 输入会置高，代表采样工作已经完成，这时存储器中的解码数据会被提取出来，存放在 FIFO 中。也就是说，当采样工作全部完成后，数据总线的控制权交给解码模块，将解码数据放在采样数据之后存放在 FIFO 中。在采样数据和解码数据之间模块会自动加上一个 Byte 的采样结束符用以告诉上位机采样已经结束，下面的数据是解码数据。采样结束符是 16 进制的“AA”“10101010”

模块图如图 3-7 所示：

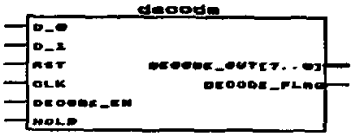


图3-7 Wiegand解码模块图

解码模块同样应用有限状态机技术实现，解码模块状态图如图 3-8 所示：

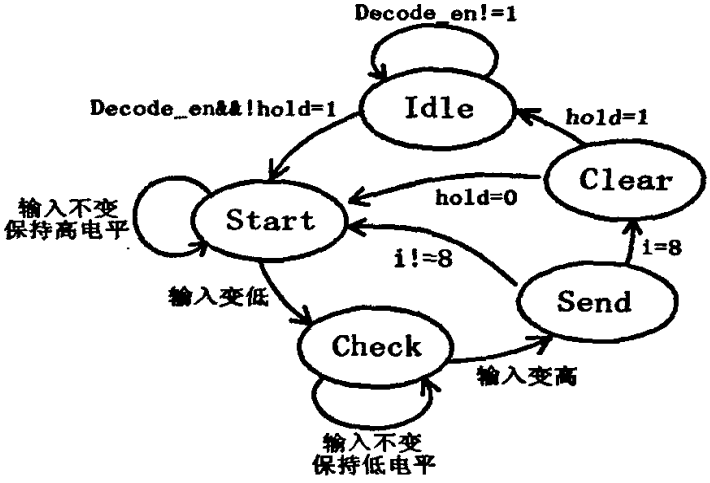


图3-8 Wiegand解码模块状态图

解码模块共分为五个状态：Idle 状态、Start 状态、Check 状态、Send 状态和 Clear 状态。

①Idle 状态：

模块的起始状态，当 Decode_en 置高同时 Hold 置低并有效时，状态转移到 Start，否则将保持 Idle 状态。

②Start 状态：

开始对两条输入信号线 D_0 和 D_1 进行采样，在平时两条输入线都是高电平，当有数据时变为低电平，D_0 线出现低电平代表数据 0，D_1 上出现低电平代表数据 1。当两条输入线上检测到低电平时进入 Check 状态，否则将保持 Start 状态。

③Check 状态：

用于解码的采样时钟频率较快，不能一出现低电平就认定是数据，必须出现由低电平到高电平的脉冲上跳变才能成为数据，在 Check 状态中当输入保持低电平时状态不发生改变，当输入变高，也就是出现上跳变时，模块进入到下一个状态 Send。

④Send 状态：

按位得到的数据按照先后顺序存储到一个 8 位寄存器中，用一个计数器 i 来计位的个数，当不足 8 位时（i!=8）模块跳回到 Start 状态进行下一位的解码，当 8 位寄存器填满后（i=8）模块转入 Clear 状态。

⑤Clear 状态

完成了两个功能，一是对使用过的计数器、寄存器清零；二是将 8 位寄存器中的数据转移到存储器中按顺序储存起来，等待一帧数据结束时将这些数据放在总线上存入 FIFO 中。在 Clear 状态中若 Hold 线变高说明一帧数据结束模块跳转到 Idle 状态等待下一帧数据的到来，若 Hold 线保持不变说明一帧数据没有结束模块跳转到 Start 状态继续解码。

Wiegand 解码模块状态机转换的 Verilog 代码如下所示：

```
module decode(D_0,D_1,Rst,clk,Decode_en,hold,Decode_out,Decode_flag);
       input  D_0, D_1, Rst, clk, Decode_en, hold;
       output [7:0] Decode_out;
       output Decode_flag;
       reg a;
       reg [7:0] n;
       reg [7:0] i;
       reg [7:0] Decode_out;
       reg [3:0] state;
       reg [3:0] now;
       reg [7:0] memdecode0;
       reg [7:0] memdecode1;
       reg [7:0] memdecode2;
       reg [7:0] memdecode3;
       reg [2:0] k;

parameter Idle  = 'b0000;
          Start = 'b0001;
          Check = 'b0010;
          Send  = 'b0011;
          Clear = 'b0100;

always@(posedge clk)
    if(!Rst)
        begin
            state = Idle;
            Decode_out = 0;
            Decode_flag = 0;
            a = 0, i = 0, n = 0, k = 0;
        end
    else
      case (state)
        Idle: begin
              if (Decode_en) begin
                             state = Start;
                             end
              else state = Idle;
              end

        Start: begin
                    if (! D_0) state = Check;
                    else if (! D_1) state = Check;
                    else state = Start;
               end

        Check: begin
                    if (D_0) state = Clear;
                    else if (D_1) state = Clear;
                    else state = Check;
               end

        Send: begin
                    if (i==8) state = Clear;
                    else state = Start;
              end
        Clear: begin
                    If (hold) state = Idle;
                    else state = Start;
               end
      endcase

endmodule
```

Wiegand 解码模块 MAX+PLUSII 仿真图如图 3-9 所示：

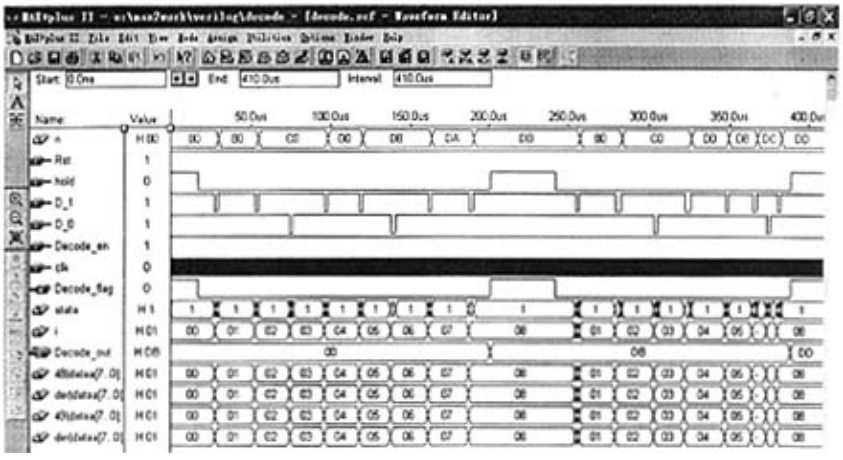


图3-9　Wiegand解码模块仿真图

如图所示，检测功能模块实现了预定的功能。

3.3.3.2、ABA 解码模块

ABA 输入与 Wiegand 输入同样采取两线制，一根为 Data 线，一根为 Clock 线，当 Clock 出现下跳变时，Data 上的数据即为有效数据。

模块图如图 3-10 所示：

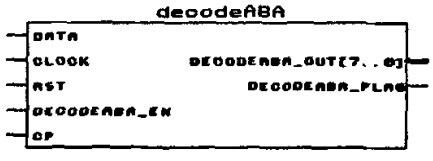


图3-10　ABA解码模块图

其中输入方面，DATA 和 CLOCK 为输入，分别与 Wiegand 信号中的 D_0、D_1 复用同一根信号线。RST 为复位端低电平有效 DECODEABA_EN 为 ABA 解码模块的使能端高有效，CP 的功能相当于 HOLD 线的功能即标志一帧数据的开始和结束，DECODEABA_OUT 为 8 位数据输出，DECODEABA_FLAG 数据输出标志位。

ABA 解码模块同样采用有限状态机技术设计，各状态功能与 Wiegand 解码模块相近。具体状态图如图 3-11 所示：

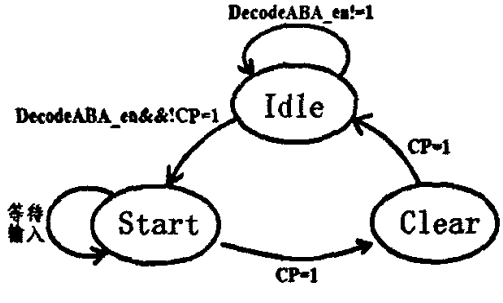


图3-11　ABA解码模块状态图

ABA 解码模块的仿真图如图 3-12 所示：

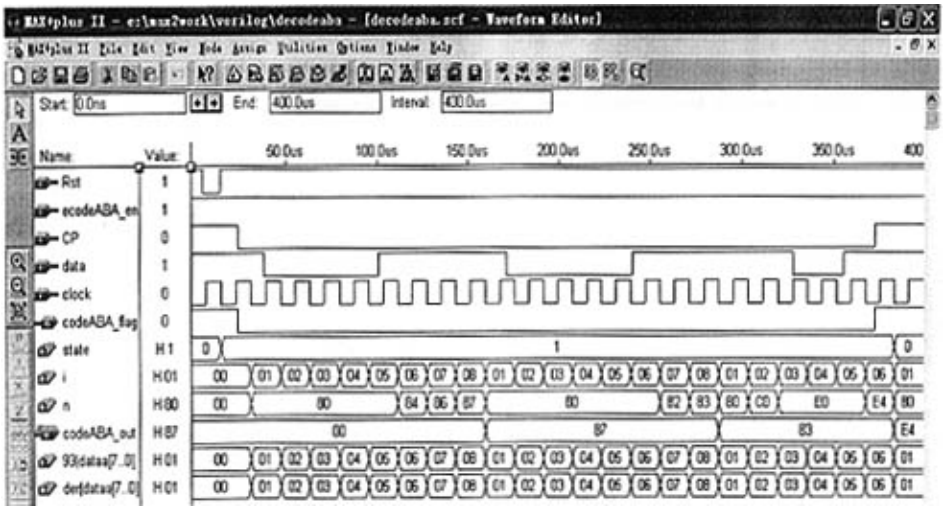


图3-12　ABA解码模块仿真图

3.4、控制功能模块的 FPGA 设计

控制模块包含两种控制：其一是根据用户需要和输入信号的种类，控制检测模块进行何种检测；其二是总线控制即决定总线何时由何种模块控制。所以控制模块由两部分组成：

一是 Select 模块，其作用是根据用户选择决定对何种信号进行采样，当用户选择 Wiegand 信号时，对信号进行采样，同时对通过 Wiegand 解码模块对信号进行解码；当用户选择 ABA 信号时，则通过 ABA 解码模块对信号进行解码。同时生成配置返回值，告诉用户所选配置已生效。

二是 Busmux 模块，其作用是根据需要决定总线控制权归什么模块所有。下面分别详述两种模块的 Verilog 实现。

3.4.1、Select 模块

Select 模块如图 3-13 所示：

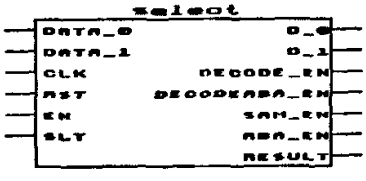


图3-13　Select模块图

其中 DATA_0、DATA_1 为数据输入，CLK 为外部时钟输入，RST 为外部复位，EN 为使能端。SLT 为选择端默认为低进行 Wiegand 信号检测，当置高时进行 ABA 信号检测。

输出方面 D_0、D_1 输入数据提供给后面的采样和解码模块，DECODE_EN

为 Wiegand 信号解码使能， DECODEABA_EN 为 ABA 信号解码使能均为高有效。SAM_EN 为采样使能，ABA_EN 为 ABA 信号输入检测使能端，RESULT 用于向用户返回配置有效状态值，高电平为配置生效，低电平为配置出错。

Select 模块应用组合逻辑方法实现，具体实现代码如下所示：

```
module select(data_0,data_1,clk,Rst,en,slt,d_0,d_1,Decode_en,DecodeABA_en,Sam_en,ABA_en,Result);

input data_0,
      data_1,
      clk,
      Rst;
input en,slt;

output d_0,d_1,Decode_en,DecodeABA_en,Sam_en,ABA_en,Result;

reg d_1,d_0,Decode_en,DecodeABA_en,Sam_en,ABA_en,Result;

always@(posedge clk)
    begin
        if(!Rst)
            begin
                d_1 = 0;
                d_0 = 0;
                Decode_en = 0;
                Sam_en = 0;
                ABA_en = 0;
            end
        else if(en)
                d_1 = data_1;
                d_0 = data_0;
                Decode_en = slt^en;
                Sam_en = 1;
                ABA_en = slt^~en;
                DecodeABA_en = slt^~en;
                Result = 1;
    end

endmodule
```

当用户选择对 Wiegand 信号进行检测时，将 Wiegand 信号的两根输入接到 Data_0 和 Data_1 上，这时 SLT 信号保持默认值不变。Sam_en 置高、ABA_en 置低、Decode_en 置高、DecodeABA_en 置低。

当用户选择对 ABA 信号进行检测时，将 DATA 和 CLOCK 分别接到 Data_0 和 Data_1 上，这时 SLT 信号置高，各标志位的值与 Wiegand 信号时相反，Sam_en 和 ABA_en 同时置高。

Select 模块仿真结果如图 3-14 所示：

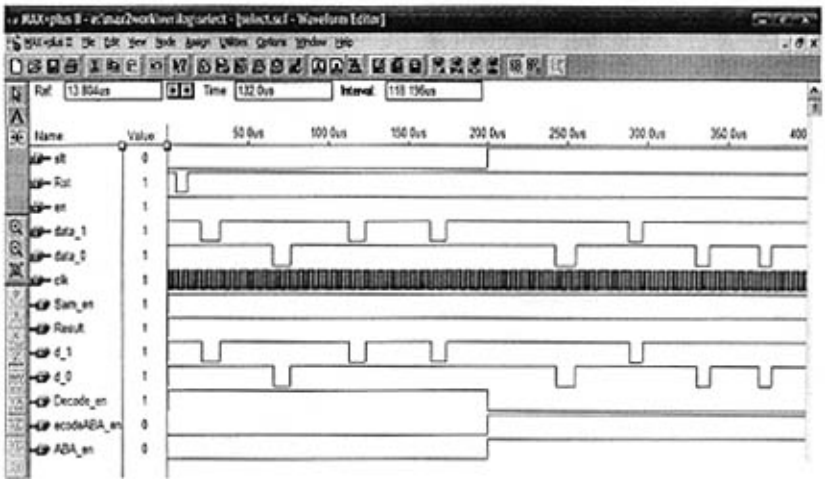


图3-14　Select模块仿真图

3.4.2、总线分配模块

总线是运算部件之间数据流通的公共通道。通过对控制端的电平控制来确定在某一时间内，总线归哪些部件使用。

总线分配模块由两个 8 位二选一多路器（Multiplexer）实现。8 位二选一多路器使用了 LPM 宏单元库。LPM 是参数化的模块库（Library of Parameterized Modules）的英文缩写，它是优秀的版图设计人员和软件设计人员智慧的结晶。模块的各个参数是由设计者为适应电路设计的要求而定制的，只要通过修改 LPM 模块的某些参数，就可得到适合自己需要的设计[13]。

使用 LPM 宏单元库还有如下优点：

1、LPM 设计出来的电路是与结构无关的。

2、设计者在利用 LPM 宏单元进行设计时，不用担心芯片的利用率和效率问题，也不需要自己再用基本的逻辑单元构造逻辑功能。同时使用 LPM 时，设计输入和模拟仿真都独立于物理结构，因此设计者即使到设计流程的最后也无需考虑最终的结构。

多路器应用的 LPM 名称为：Busmux，模块图如图 3-15 所示：

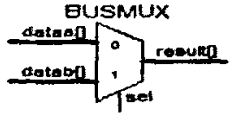


图3-15　Busmux模块图

*Dataa[7:0]*和 *Datab[7:0]*为 8 位输入，*sel* 为选择端，为高时输出 Dataa[7:0]，为低时输出 Datab[7:0]，Result[7:0]为输出结果。

所需要自定义的参数为位宽，这里选择 8 位位宽。

Busmux 模块在设计中的应用如图 3-16 所示：

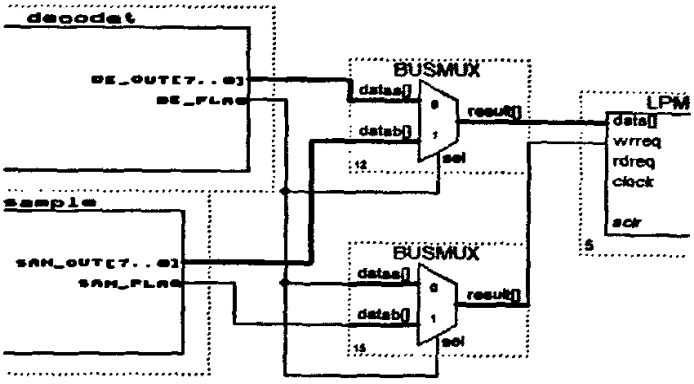


图3-16 Busmux模块应用图

两个 Busmux 分别用来控制 FIFO 的数据输入和写使能信号，他们的 Sel 端都接在 DE_FLAG 上。由于数据流结构如图 3-17 所示：

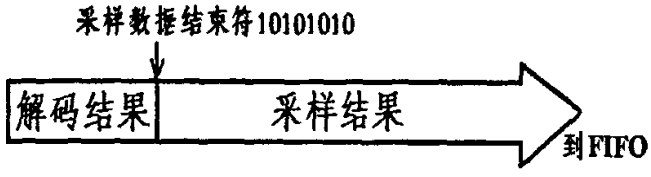


图3-17 数据流结构图

当一帧数据结束时才取出解码模块存储的解码结果传送到 FIFO 储存。DE_FLAG 在一帧数据结束时置高，这时上面的 Busmux 输出变为解码的输出，下面的 Busmux 输出变为解码标志位的输出。

3.5、FIFO 的 FPGA 设计

先进先出堆栈 FIFO 实现了队列结构的有特殊用途的存储器。它广泛应用在计算机和通信系统中。与其他存储器器件不同，一个典型的 FIFO 具有两个单一方向的端口并且没有地址的输入：一个用于写，一个用于读。就像它名字所说的那样，最先写入的数据也是最先被读出的数据。FIFO 中的数据只能读一次，当一个数据被读出，第二次读操作返回的就是下一个数据了。由于 FIFO 是由队列结构构成，所以它也有上溢和下溢的情况。它们的有限大小，通常称为深度。当向一个满的 FIFO 中在企图写入数据时，就发生上溢，同样如果对一个空的 FIFO 进行读操作时，就会发生下溢[16]。

FIFO 的另一个常用领域是处理时钟域交叉。这种应用用于要从一个逻辑部件传输一定的数据到另一个逻辑部件，并且这两个逻辑部件的工作频率不一样。通常一个双时钟的同步 FIFO 可以解决这个问题。

在 FIFO 中，双端口存储器是一个异步器件，可以被每一个时钟域中的逻辑模块访问。在一个双时钟的同步 FIFO 中，一个或多个字节按照时钟 A 写入后，写指针的信息可以通过 FIFO 内部的时钟域同步逻辑，安全的传递给时钟 B

使用。这个信息能够告诉读控制逻辑现在有数据等待被读出去，在时钟域 B 中的逻辑可以经过足够长的时间，等写入的数据稳定后再读出来。如图 3-18 所示[19]：

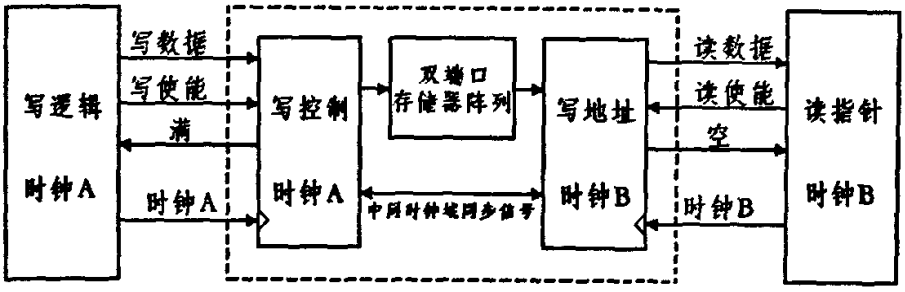


图3-18　采用同步FIFO实现的时钟域交叉

此外 FIFO 还可以用来处理速率匹配问题。当一个数据源发送数据非常快，而接收方的速率没法跟上时，就需要使用 FIFO。

FIFO 有一个存储器快、一个读端口、一个写端口和一个控制逻辑模块组成。存储器块通常采用的是 SRAM，但是对某些应用也可以使用 DRAM。它的结构示意图如图 3-19 所示：

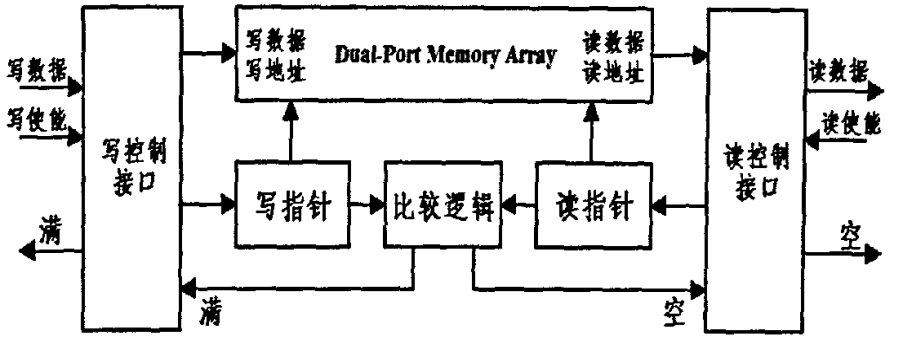


图3-19　基本的FIFO结构

FIFO 的数据读写操作与 SRAM 的数据读写操作基本相同，只是 FIFO 没有地址。所以用 SRAM 实现 FIFO 的关键点是如何产生正确的 SRAM 地址。

我们可以借用软件中的方法，将 FIFO 抽象为环形数组，并用两个指针，即读指针（fifo_rp）和写指针（fifo_wp）控制对该环形数组的读写。其中，读指针 fifo_rp 指向下一次读操作所要读取的单元，并且每完成一次读操作，fifo_rp 加一；写指针 fifo_wp 则指向下一次写操作时存放数据的单元，并且每完成一次写操作，fifo_wp 加一。由 fifo_rp 和 fifo_wp 的定义可知，当 FIFO 被读空或写满后，fifo_rp 和 fifo_wp 将指向同一单元，但在读空和写满之前 FIFO 的状态是不同的，所以如果能区分这两种状态，再通过比较 fifo_rp 和 fifo_wp 就可以得到 nempty 和 nfull 信号了。FIFO 工作状态的示意图如图 3-20 所示：

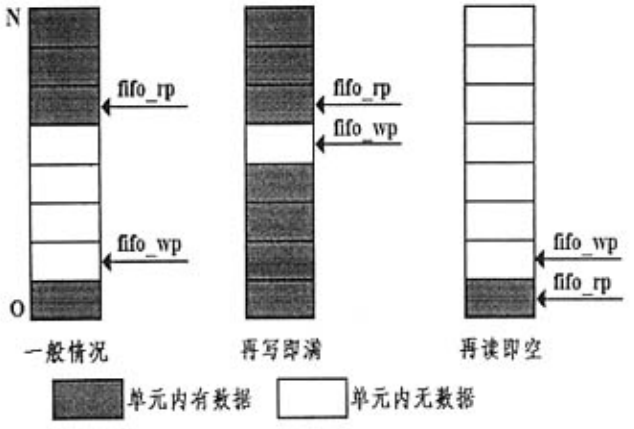


图3-20　FIFO的工作状态

一般情况下，在写 SRAM 时，先建立地址和数据，然后置写使能信号 WR，使其有效。在 WR 保持一定时间的有效后，先复位 WR，再释放地址总线和数据总线。而读取 SRAM 时，则先建立地址，然后置读使能信号 RD 有效，在 RD 维持一定时间有效后，复位 RD，同时读取数据总线上的值，然后再释放地址总线。

FIFO 模块图如图 3-21 所示：

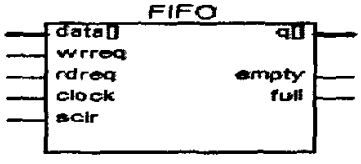


图3-21　FIFO模块图

FIFO 模块的 Max+PlusII 仿真图如图 3-22 所示：

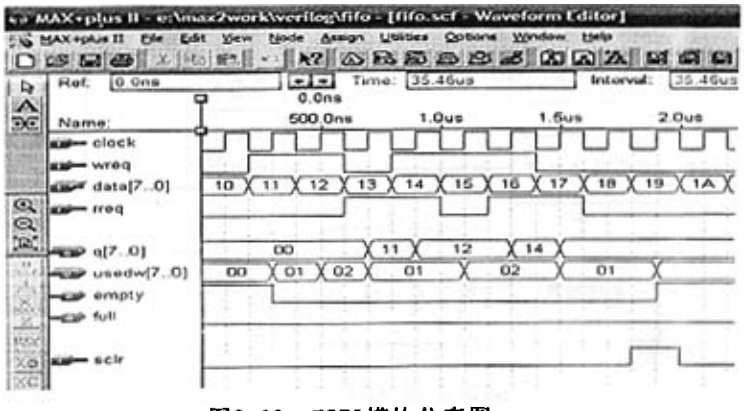


图3-22　FIFO模块仿真图

3.6、UART 的 FPGA 设计

UART（Universal Asynchronous Receiver Transmitter，通用异步收发器）是通信领域和计算机领域中数据通信流行和广泛使用的一种接口设备，它的主要

作用是用来控制计算机与串行设备的接口操作，负责在发送时把并行数据转换成为串行数据，在接收时把串行数据转换成为并行数据。一般来说，UART 提供了 RS-232C 数据终端设备接口。

通信领域和计算机领域中常用的 UART 含有一个接收模块和一个发送模块，其中接收模块的作用是接收从串行数据输入端口送来的异步数据并进行串/并转换；发送模块的作用是对接收的 8 位数据进行并/串转换。因此为了使异步串行数据同步，并保证数据的完整性，串行数据上需要添加必要的开始位、停止位和奇偶校验位等附加位。通常，一个典型的 UART 的帧格式如图 3-23 所示[13]：

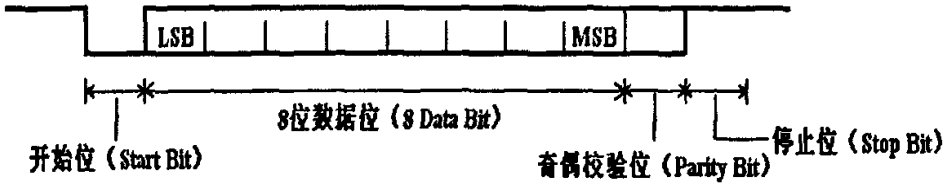


图3-23　典型UART帧格式

在 UART 中，接收模块和发送模块共同使用一个内部的 CLK16X 时钟信号，这个时钟信号是 UART 接口波特率时钟的 16 倍，它是通过外部的输入时钟分频得到的。

3.6.1、UART 的顶层设计

一个基本的 UART 结构如图 3-24 所示：

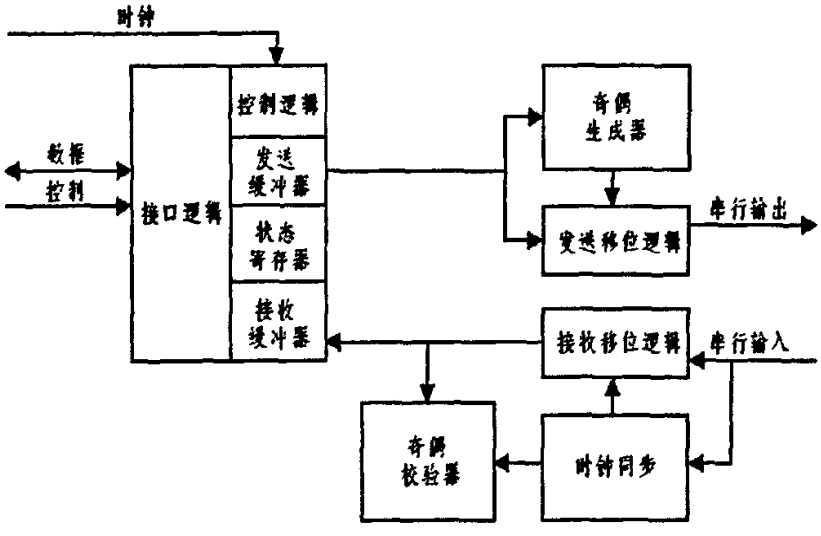


图3-24　UART基本结构

它可以分成 3 个基本的模块：接口模块、传输模块和接收模块。

接收模块：

在 UART 中，接收模块的作用是接收从串行数据输入端口送来的异步数据，并进行串/并转换。接收模块通过串行输入端口接收外来的输入数据，首先将其

存放在接收缓冲寄存器(RBR),接收模块通过内部逻辑判断是否可以接收数据,一旦接收一帧数据完毕后,就送出接收数据就绪的标志。在 RBR 读选通信号的上升沿时将数据从 RBR 中读出。

发送模块:

在 UART 中,发送模块的作用是对接收到的 8 位数据进行并/串转换。发送模块通过写信号来对发送保持寄存器(THR)进行操作。与接收模块类此,THR 的写操作也是在 THR 数据写选通信号的上升沿时来进行的。

另外,发送模块也包括模块控制和模块状态配置功能。模块控制配置主要用来设置发送数据帧的属性;发送模块的状态标志为 THRE 和 TEMT,当 THR 空时 THRE 置为逻辑 1,而当 THR 和 TSR 都为空时 TEMT 置为逻辑 1。

接口模块:

用于接收上位机对串口的配置命令并进行译码,同时根据命令配置串口[20][21]。

3.6.2、UART 的底层设计

接收模块:

串行接收模块包括一个接收缓冲寄存器和一个接收移位寄存器,它们的具体长度为 8 位。由于串行数据帧异步于接收时钟,因此输入线上的有高电平到低电平的变化将被认为是一个数据帧的开始位,为了避免由于噪声所造成的接收错误,有求开始位必须在接收波特率时钟周期的 50%以上为低电平。由于内部时钟信号是接收/发送波特率时钟频率的 16 倍,因此开始位至少在 8 个时钟周期内保持低电平才被认为有效。

一旦一个有效的开始位被接收,数据比特和奇偶校验位将会每 16 个 CLK16X 时钟周期采样一次。如果检测到任何帧错误时,UART 将假设错误源于后面的数据帧的开始位,同时重新同步,采用刚才方法的两倍进行开始位的检测。

图 3-25 是接收模块状态图:

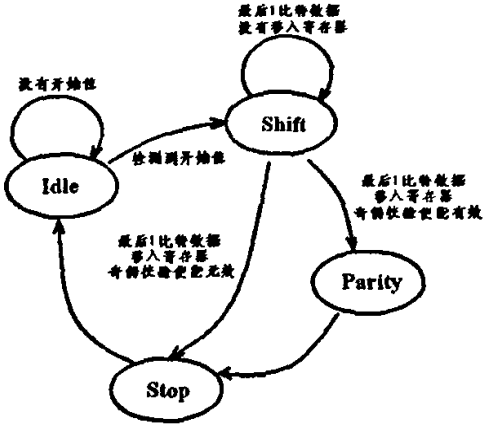


图3-25 UART接收模块状态图

发送模块：

串行发送模块包括一个发送保持寄存器（THR）和一个发送移位寄存器（TSR），它们的具体长度为 8 位。当发送数据装载到 THR 后，串行数据传输将会自动使能从而进行数据传输。首先一个开始位被传送出去，同时 THR 的数据将自动并行装载到 TSR 中；然后 TSR 中的数据将会按照线性控制寄存器（LCR）中约定的字长度并加上奇偶校验位从 TSR 中移出；最后停止位将加在数据帧的后面标志该帧的结束。

在发送模块中，上面形成的数据将以时钟 1/16 的速率进行发送。串行数据帧在发送的过程中，THR 同时在进行着写数据的操作。这样当一个串行数据帧发送完后，只要 THR 不为空，那么就可以立即发送下一个数据帧。这种传输方式可以增加传输带宽，它通常称作“back-to-back”传输。在没有串行数据帧发送的时候，串行数据输入输出端口将保持高电平[22]。

图 3-26 是发送模块状态图：

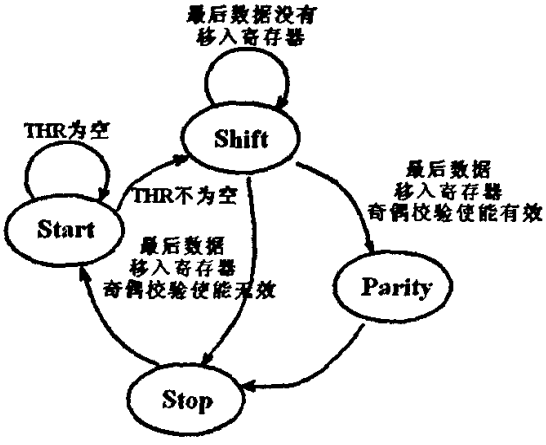


图3-26　UART发送模块状态图

3.7、器件的编程与配置

3.7.1 器件的编程[23][24]

Altera 器件的编程可通过专门的编程器、JTAG 在系统变成等方式进行。并口下载线 ByteBlaster 是用来对 FLEX,MAX 等器件进行在系统编程时使用的连接线。

ByteBlaster 并口下载电缆通过标准并口与 PC 机相连，实现在系统配置。它的构成为：与 PC 机并口相连的 25 针插座、与目标 PCB 板插座相连的 10 针插头和 25 针到 10 针的变换电路。如图 3-27 所示：

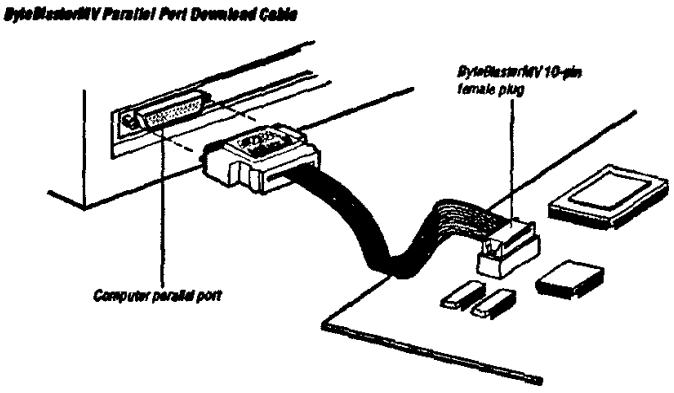


图3-27　ByteBlaster下载电缆连接图

ByteBlaster 编程电缆的内部实际上有一个 74LS244 和一些电阻，其原理如图 3-28 所示：

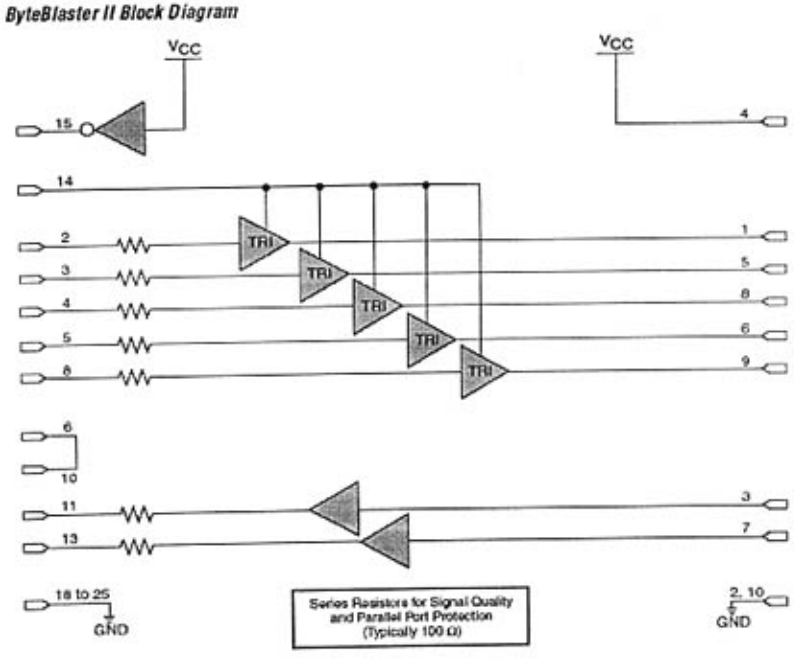


图3-28　ByteBlaster下载电缆原理图

ByteBlaster 可用来对 MAX7000、MAX9000、FLEX10K 和 FLEX8000 等器件进行编程下载。它有两种配置方式：被动串行模式（PS）和边界扫描模式（JTAG）。

采用被动串行（PS）方式，配置数据将通过 ByteBlaster 电缆串行发送到 FLEX 器件，配置数据的收发同步由 ByteBlaster 时钟提供。在 MAX+PLUSII 开发环境下，可直接对 FLEX 器件进行 PS 配置。此种配置方式使用的配置文件为 SOF 文件，此文件是在设计综合过程中自动形成的。

JATG 编程方式同时支持 MAX 器件和 FLEX 器件，但用于 MAX 器件的下载文件为 POF 文件，而用于 FLEX 器件的下载文件为 SOF 文件。

3.7.2 器件的配置

Altera 的 FPGA 器件分为两类配置方式：主动配置和被动配置。主动配置方式由 FPGA 器件引导配置操作过程，它控制着外部存储器和初始化过程；而被动配置方式则由外部计算机或控制器控制配置过程。

在 FPGA 器件正常工作时，它的配置数据储存在 SRAM 中。由于 SRAM 的易失性，每次加电时，配置数据都必须重新构造。在实验系统中，常用计算机或控制器进行调试，因此可以使用被动配置方式。而实用系统部都带有计算机控制，大多数情况下必须由 FPGA 器件引导配置过程，这时 FPGA 器件将主动从外围存储芯片中获得配置数据。

第四章　虚拟检测仪的软件开发

美国国家仪器公司曾经提出一个著名的口号：The Software is the Instrument（软件就是仪器）。可见软件的设计在虚拟仪器设计中的重要性。本设计的软件开发基于最新的 LabVIEW8.0 版本。

LabVIEW（laboratory virtual instrument engineering workbench）是由美国 NI 公司开发的一种图形化的编程语言和开发环境，它广泛地被工业界、学术界和研究室所接受，被公认为是标准的数据采集和仪器控制软件。LabVIEW 不仅提供了与遵从 GPIB，VXI，RS-232 和 RS-485 协议的硬件及数据采集卡通信的全部功能，还内置了支持 TCP/IP，ActiveX 等软件标准的库函数，而且其图形化的编程界面是编程过程变得生动有趣。其创始人 James Truchard、Jeffrey Kodosky 和 William Nowlin 开发该软件的目的就是要把广大工程师和科学家从繁重的编程工作中解放出来，其创新性的图形化开发平台，突破了基于文本的传统高级语言的框架，使编写测控程序变得直观、方便、高效。

以 LabVIEW 为代表的图形化程序语言，又称为“G”语言。使用这种语言编程时，基本上不需要编写程序代码，而是“绘制”程序流程图。LabVIEW 尽可能的利用工程技术人员所熟悉的术语、图标和概念，因而它是一种面向最终用户的开发工具，可以增强工程人员构建自己的科学和工程系统的能力，可为实现仪器编程和数据采集系统能够提供便捷途径。

利用 LabVIEW，可产生独立运行的可执行文件。LabVIEW 是真正的 32 位编译器。像其他软件一样，LabVIEW 提供了 Windows，UNIX，Linux 和 Macintosh 等多种版本[25][26][27]。

4.1、FPGA 与计算机通讯

FPGA 通过 UART 与计算机中的 COM 口进行通信，COM 口是个人计算机上，异步串行通信口的简写。由于历史原因，IBM 的 PC 外部接口配置为 RS232，成为实际上的 PC 界默认标准。所以，现在 PC 机的 COM 口均为 RS232。若配有多个异步串行通信口，则分别称为 COM1、COM2... 。

RS232 是异步串行通信中应用最早，也是目前应用最广泛的标准串行总线接口之一。是美国电子工业协会 EIA（Electronic Industry Association）制定的一种串行物理接口标准。采用 150pF/m 的通信电缆时，最大通信距离为 15m；若每米电缆的电容量减小，通信距离可以增加。传输距离短的另一原因是 RS-232 属单端信号传送，存在共地噪声和不能抑制共模干扰等问题，因此一般用于 20m 以内的通信。一般 RS232 端口都是 25 针的 D 型插头座，也有简化的 9 针 D 型插头座。

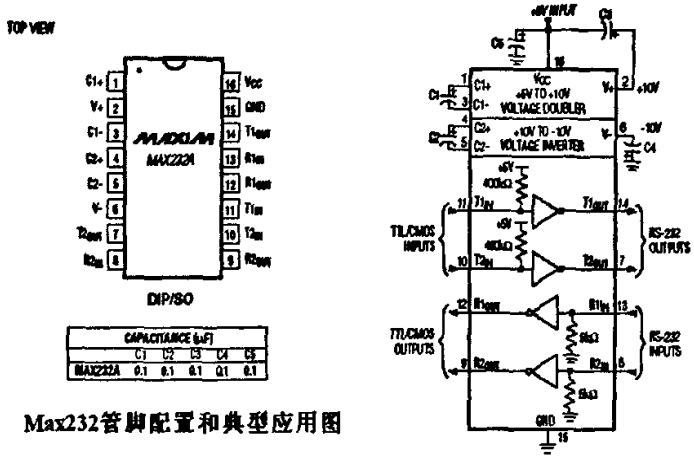
计算机中的 COM1 和 COM2 都是 RS232 串行通信标准接口。一个是 9 针 D

型连接头，一个是 25 针 D 型连接头，从而便于用户随意连接 9 针或 25 针的调制解调器等外部设备。

4.1.1、硬件电路设计

RS-232C 规定了自己的电气标准，由于它是在 TTL 电路之前研制的，所以它的电平不是+5 V 和地，而是采用负逻辑，即逻辑“0”：+5 V～+15 V；逻辑“1”：-5 V～-15 V。（TTL 电平：逻辑“0”：<0.4V；逻辑“1”：+3 V～+5 V ）因此，RS-232C 不能和 TTL 电平直接相连，使用时必须进行电平转换。

一种常用的电平转换电路是使用 Maxim 公司生产的电平转换芯片 Max232。芯片的典型应用电路如图 4-1 所示：



Max232管脚配置和典型应用图

图4-1　MAX232封装以及典型电路图

其中 RS-232 连接计算机的 COM 口，TTL/CMOS 连接 FPGA 上的 UART 端口[28]。

4.2、LabVIEW 软件设计[29][30][31]

LabVIEW 程序由前面板（front panel）和流程图（block diagram）两部分组成，整个程序是基于多线程的设计，前面板和流程图各占用一个线程。前面板是 LabVIEW 程序的图形用户接口，此接口集成了用户输入和显示程序的输出，相当于传统仪器的面板。前面板包括旋钮、按钮、图形和其它的控制（controls）与显示对象（indicators）。流程图包含虚拟仪器程序的图形化源代码。在流程图中对 VI 进行编程，以控制和操纵定义在前面板上的输入和输出功能。流程图包括内置于 LabVIEWVI 库中的函数（functions）和结构（structures），还包括与前面板上的控制对象、显示对象对应的连线端子（terminals）。

一个 VI 包括 3 个基本元素：前面板（front panel）、流程图（block diagram）、图标以及连接器（icon and connector pane）。流程图由数据终端（Terminal）、节点（Nodes）、连线（Wires）和结构（Structure）4 类元素构成。节点可以拥有

多个输入和/或输出，并在 VI 运行时完成一定的操作。有点类似于文本编程语言中的声明、操作符、函数或是子程序。

LabVIEW 的程序运行采用的是数据流编程模式，数据流编程也被称为数据依赖性，其原理是：任何一个函数、子 VI 或者其他程序节点必须获得所有输入数据之后才能运行，这些函数、子 VI 或者其他程序节点必须完全运行完毕之后，才能在输出端子上输出数据。Visual Basic、C++、JAVA 等常规编程语言，其程序运行采用的是控制流模式。在控制流模式下，每条程序在程序中的先后顺序决定其执行的顺序。

4.2.1、程序结构及其流程

上位机软件的作用主要是：从 FPGA 接收采样数据和解码数据；对采样数据和解码数据按照要求进行分析和储存；将分析后的结果显示给用户，同时将分析后的结果以文本文件的形式保存起来方便用户查询。上位机软件可分为三个主要功能模块：

1、串口通讯模块：负责与 FPGA 进行通讯接收数据，发送配置命令

2、数据处理模块：对接收到的数据进行处理

3、显示模块：将处理后的结果分别以图像和文本的形式显示出来。

软件各模块间的结构如图 4-2 所示：

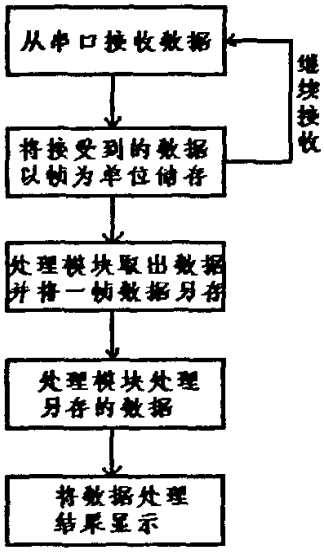


图4-2　软件结构图

从串口接收模块接收到的数据以帧为单位存储在文本文件中，处理模块从文本文件中将数据全部提取并储存到另一个地方，这样可以空出储存帧数据的文本文件，然后串口可以接收下一帧数据，同时保证处理模块有足够的时间来完成对一帧数据的处理和显示（处理模块可利用的时间为两帧数据之间的间隔和从串口接收一帧数据所用的时间）。最后处理模块将采样信号的分析结果以文本文件的方式显示出来，同时利用得到的完整的一帧采样数据将其还原为图形。

这时处理模块再取出下一帧数据进行处理，如此循环直到所有数据全部接收完成。

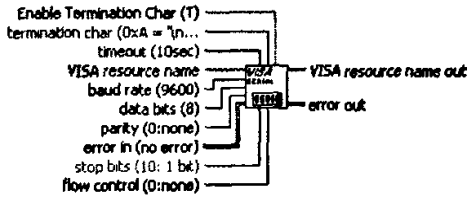
FPGA 通过 hold 或 cp 信号来判断一帧数据的到来，如果这两个信号再也不出现了说明全部数据结束，如果检测系统配套了一些可以通过上位机控制的机械设备用来自动刷卡，可以通过上位机软件设定刷卡次数来决定什么时候结束。

4.2.2、串口通讯模块

主要用于与 FPGA 的通讯，接收数据以及向 FPGA 发送配置命令，配置命令包括检测模式配置和串口配置两部分，两部分的配置都可以在界面上完成。

针对串行通讯 LabVIEW 共提供多个串行通讯节点，分别实现串口设置、串口写、串口读、检测串口缓存和中断等功能。在本设计中应用了如下几个节点：

①串口设置：输入为串口号、波特率、奇偶校验使能、停止位、停止位使能等串口参数。输出为串口数据流和错误代码。其中 VISA resource name 为 LabVIEW 的 I/O 数据端口，输入串口数据流。



error in 和 error out 为 LabVIEW 特有的一类特殊的预定义簇。LabVIEW 中的簇与数组一样都是复合数据类型。但与数组不同的是，簇可以包含任意数目的任意类型的元素，元素类型可以相同也可以不同，而数组只能包含同一类型的元素；另外簇不能在运行时添加新元素，而数组的长度在运行时可以自由改变。簇类似于 C 语言中的结构。

error in 和 error out 这两个参数是具有相同结构的簇，都由布尔类型数据 status、整形数据 code 和字符串类型数据 source 组成。status 为 True 时表明在前面的执行中已经出错；为 False 时，表明在前面执行中没有错误。code 为错误代码，其值为 0 时表示没有错误；非零时表示出现错误。source 包含对错误的简短描述和引发错误的函数名称。

②串口读和串口写：

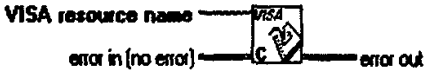
实现串口读功能

实现串口写功能

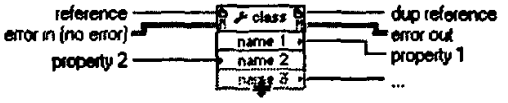
③I/O Buffer 大小设定

其中 Mask 值是决定设定哪个 buffer 的大小，16 为 I/O 接收 buffer，32 为 I/O 发送 buffer。

④关闭串口



⑤属性节点：



属性节点用于对对象的属性（reference）进行读取和设定的。属性节点的最上面有两个输入参数和两个输出参数，输入参数 reference 用于介入对象的 reference，输入的 reference 参数在被复制之后从 dup reference 输出，可以再接入下一个使用 reference 作为输入参数的节点或函数。其他两个输入、输出参数用于错误处理。当接入某一个 reference 后，在 class 位置会出现与该 reference 对应的对象类别名称。name1、name2 等端子为属性端子，name1 和 name2 代表属性名。

程序结构如图 4-3 所示：

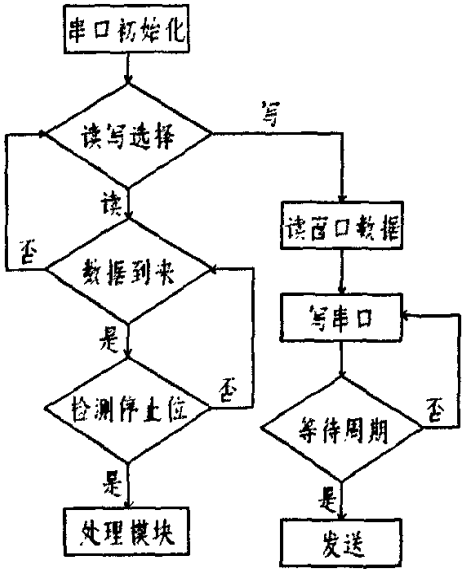


图4-3 串口通讯模块程序结构

通过串口设置节点对所选择的串口进行初始化，通过串口读和串口写节点

完成窗口读写工作，通过属性节点完成停止位的检测。

串口通讯模块 LabVIEW 程序流程如图 4-4 所示：

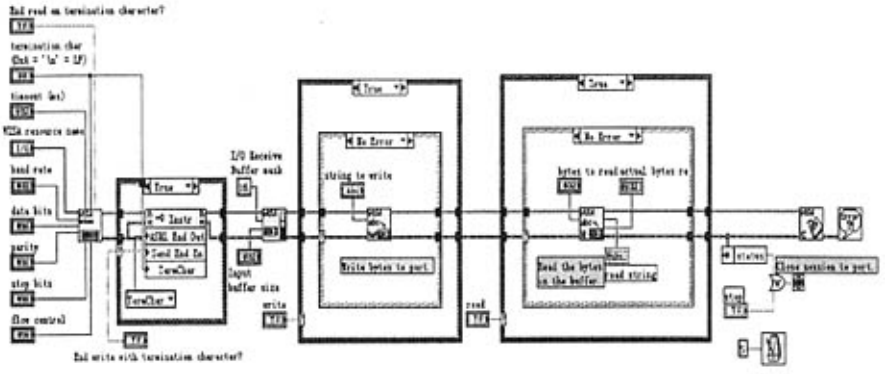


图4-4　串口通讯模块程序流程图

4.2.3、数据处理模块和显示模块

数据处理模块从串口得到采样数据并进行分析，从中得到脉冲宽度、脉冲间隔等信息。再通过显示模块分别以图形和文本的形式显示出来。

作为一个完整的编程语言，LabVIEW 支持的数据类型相当丰富，包括数值性、布尔性、字符串型、波形、数字波形、簇、数组、I/O、路径、时间等等。

本设计中应用到数值性、布尔性、字符串型、数字波形、簇、数组、I/O 型数据，LabVIEW 为没用数据类型都提供了全面而丰富节点，包含日常数据处理的各个方面，同时还包含了别的编程语言所不具备的数多特有的数据处理节点，为编程人员提供了最大限度的方便。

下面对本设计中应用的部分数据处理节点作一个简要说明：

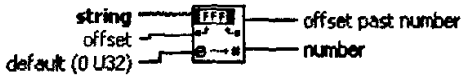
①从字符串中取出制定长度的字符

offset 是取出字符的起始位置默认为零，即从字符串的最开始取，substring 为取出后的字符。

②判断两个输入是否相同

对 X、Y 的类型没有要求，只要相同就行，X、Y 时输出为 1，不同时输出为 0。

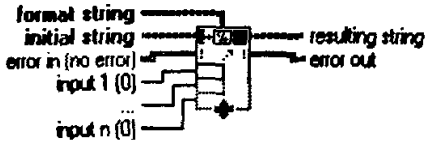
③16 进制字符串数据到数值型数据的转换节点



其中 default 控制输出数值型数据的类型，默认为 0 输出 32 位数值型数据。

④数值型数据向布尔数组的转换节点

⑤将数据转换成制定格式的字符串型数据



format string 用于设定输出字符串的格式

⑥在字符串数据中寻找指定的字符，并在指定字符的位置将字符串分割开

软件的串口接收模块将从串口得到的数据现存放在一个文件中，每次存放一帧数据的采样和解码结果，然后数据处理模块打开保存的数据进行处理和显示。

4.2.3.1、显示模块

显示包含图形显示和文本显示两部分。文本显示通过文件 I/O 函数来实现。LabVIEW 提供的文件 I/O 函数是一组功能强大的文件处理工具。使用文件 I/O 函数可以进行所有有关文件输入输出的操作，主要包括：打开关闭数据文件，在文件中读取和写入数据，读取和写入数据到电子表格格式文件，转移和重新命名文件与目录，改变文件属性，创建、修改和读取配置文件。

LabVIEW 中的文件 I/O 函数大致分为 3 个层次：High-level VIs、Low-level VIs、 Advanced VIs。使用高层 VI 可以简单的实现常用的 I/O 操作。使用底层 VI 和函数可以控制文件 I/O 的每一步操作，编制比较复杂的文件管理程序。

一个典型的文件 I/O 操作包括以下 3 个步骤：

①创建或打开一个文件。用户需要指明现有的文件在哪里，或自己指明路径并创建新的文件，文件打开之后 LabVIEW 会自动创建一个 refnum。

②对已经打开的文件进行读取或写入操作。

③关闭文件，同时 refnum 会被自动释放。

Refnum（可称为参考数或标识号）是一种特殊的数据类型。当用户打开一个文件时，LabVIEW 将返回一个与此文件相关联的 refnum，此后所有与该文件有关的操作，都依据该 refnum 来进行。当该文件关闭后，与之相对应的 refnum 就会被释放。Refnum 的分配是随机的，同一个文件被多次打开时，其每次分配的 refnum 一般是不同的。

多数文件 I/O 函数只负责完成其中一个步骤。然而，一些为常用的文件 I/O 操作定制的高层文件 I/O 函数可以一次完成所有 3 个步骤。

显示的另一个内容为图形显示。在一帧数据被保存之后，程序应用图形控件，将一帧数据的图形显示出来。LabVIEW 最吸引人的特性之一就是对数据的图形化显示提供了丰富的支持。强大的图形显示功能增强了用户界面的表达能力，极大的方便了用户对虚拟仪器的学习和掌握。

4.2.3.2、数据处理模块

数据处理模块利用了 LabVIEW 中子 VI 的概念，LabVIEW 应用程序具有分层的特点。编辑者可以将已编好的程序用一个较小的图标来表示，将此图标放在另外的框图程序中，作为该程序的一部分被调用。这个被调用的程序称为子 VI；调用此子 VI 的程序成为此子 VI 的高层框图程序。LabVIEW 中的子 VI 相当于文本编程语言中的函数、过程和子程序，可以将任何一个定义了图标和连接器的 VI 作为另一个 VI 的子 VI 进行调用。所谓图标是这个子 VI 的图形描述；而连接器定义了子 VI 的输入输出端口，子 VI 通过连接器与高层框图程序进行数据交流。应用子 VI 可以使程序简洁、明了，易于理解、调试和维护；对于逻辑复杂、计算量大的程序更为有益。

数据处理模块子 VI 软件结构如图 4-5 所示：

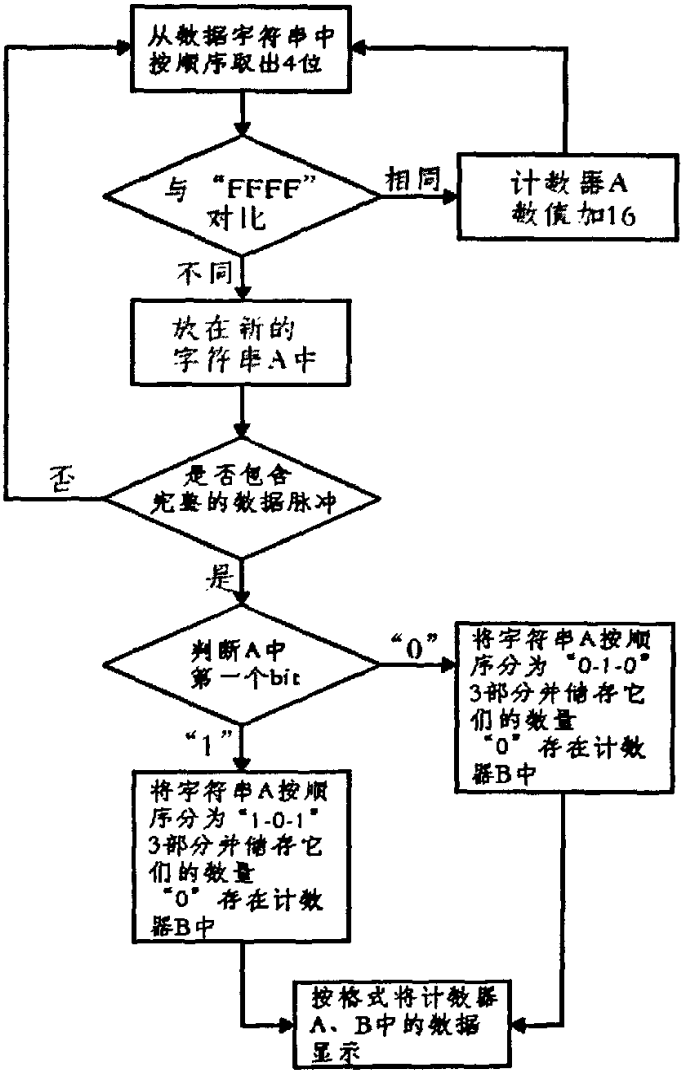


图4-5　数据处理模块子VI软件结构

对采样数据的处理就是找出采样数据中的“0”和“1”并分别记下他们的数量，然后再和采样频率一起进行计算就能准确地得出，在数据流的传输过程中出现的脉冲宽度、脉冲间隔等指标。

为了能够快速的找出采样数据流中“0”出现的位置和数量，对采样数据流进行了分段处理，将很长的数据流按顺序的一段一段的提取出来，并与事先设定好的字符串对比。因为输入数据线的常态为高电平，只有当出现数据时才变为低电平，所以对采样得到的数据进行如下处理：

首先，以四位字符为单位提取采样数据，将提取出来的字符与“FFFF”进行对比，如果相同说明在数据线上采样到连续的 16 个“1”，于是将“1”的个数存放在专门用来存放“1”的个数的计数器 A 中。然后再从采样数据中取出四位字符作同上的操作，直到出现与“FFFF”不相同的字符为止。

当出现与“FFFF”不相同的字符时，说明在取出的字符中含有“0”也就是数据脉冲，它的宽度正是我们要检测的指标之一。所以下一步回到采样数据中取出下一个四位字符重新对比，若与“FFFF”相同，则说明上一个字符包含了完整的数据脉冲可以进行下一步的处理，否则继续从采样数据中取数据直到取出的数据包含完整的数据脉冲为止。

当取出的字符串包含了完整的数据脉冲时，判断这个字符串的第一个 bit 是“0”还是“1”，然后应用 LabVIEW 中的节点寻找字符串中不同的字符并根据他们的位置将字符串分割开，再分别统计它们的数量得到所需要的指标。

得到的“0”的数量存放在专用计数器 B 中，当得到为“0”的字符时将之前的计数器 A、B 中的数值乘以采样周期，写入存放检测结果的文本文件中，以便用户查阅。

当检测完一帧数据时，处理软件回到初始状态等待从串口接收的下一帧数据的到来，同时将上一帧数据显示出来。

文本文件显示格式如图 4-6 所示：

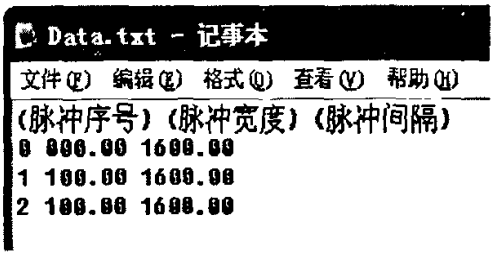


图4-6 文本文件显示格式

LabVIEW 软件界面如图 4-7 所示：

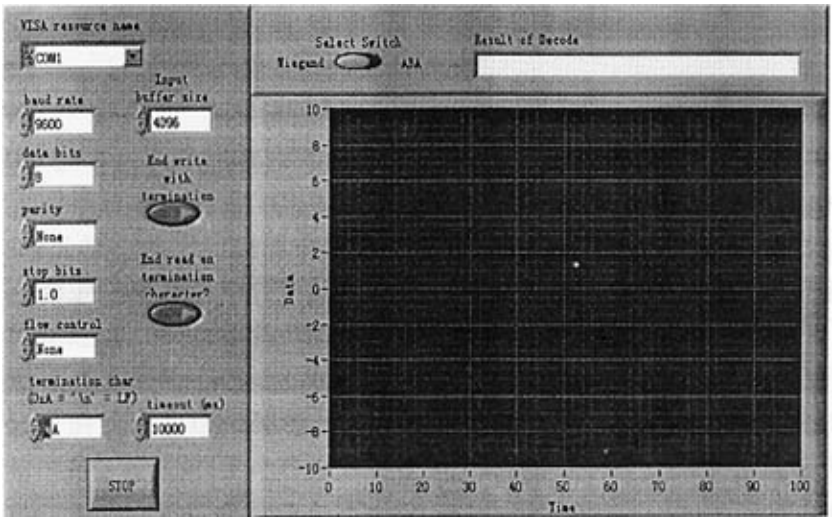


图4-7　LabVIEW软件界面

第五章　结论与展望

5.1、结论

RFID 技术是现今比较流行的一种自动识别技术。广泛应用于门禁、仓储、物流、管理等领域。随着 RFID 自身技术的不断完善和统一的国际化标准的出台，RFID 必将在我们的生活中扮演越来越重要的作用。所以对 RFID 读卡器的检测就成为了一个重要的任务。虚拟仪器是在计算机基础上通过增加相关硬件和软件构建而成，具有可视化界面的仪器。

本设计的目的是为了检测从读卡器传输出来的 Wiegand 和 ABA 信号是否协议所规定的要求。这两种协议信号在 RFID 读卡器中应用广泛，尤其是在门禁系统的 RFID 中。本设计可分为两个部分即 FPGA 部分和上位机软件部分，FPGA 部分主要用于对信号的采样和对采样信号的储存，同时在 FPGA 部分设计了一个 UART 提供与上位机之间的串行通信。上位机部分主要负责对采样结果进行分析并将分析结果显示出来。本设计中的检测系统在检测两种协议通讯真确正确性的同时，还可以进行 RFID 工作的稳定性测试，可以检测在读卡器长期工作时传输数据有没有发生错误。

5.2、展望

本设计提出了一种检测系统的设计思路，对一些情况进行了理想化的简化，距离工业化的应用实现还有很长的路要走。

在未来如果在本设计的基础上加以改进和优化实现工业用的检测系统，结合一定的机械装置，让上位机可以控制刷卡的次数，同时将显示部分做成独立的装置，实现一套便捷的检测系统，不失为 RFID 提供了一个良好的检测平台，对提高 RFID 产品的稳定性，提高用户对产品的信任度，以及对 RFID 生产企业的质量控制具有一定的积极意义。

参考文献

[1] 李青霞、任焕晞，虚拟仪器综述．现代科学仪器，1999，4

[2] 程虎，虚拟仪器的现状和发展趋势．现代科学仪器，1999，4

[3] 刘铮、章兢，非接触式IC卡中的射频识别技术．信息技术，2002.4

[4] 李锦涛、郭俊波、罗海勇、曹岗、冯波、陈益强，射频识别（RFID）技术及其应用．信息技术快报，2004.11

[5] 岳云峰、王睿、孙海涛，韦根（Wiegand）协议及其应用．齐齐哈尔大学学报，2002，第18卷第2期

[6] 许丹、徐平，维根及ABA磁卡编码产生器．金卡工程，2004，5

[7] 张小牛、侯国屏、赵伟，虚拟仪器技术回顾与展望．测控技术，2000，第19卷第9期

[8] Kang Jitao、Gan Yadong、Quan Qingquan，The method of developing Virtual Instrument Platform，Autonomous Decentralized Systems，International Workshop on，2000.9

[9] McQuiston, B.K，Virtual instruments for use in test systems development，AUTOTESTCON '93，IEEE Systems Readiness Technology Conference，1993.9

[10] Robert K. Dueck 编著、张春等译、王志华主审，数字系统设计——CPLD应用与VHDL编程，清华大学出版社，2005

[11] 刘昌华，数字逻辑EDA设计与实践—MAX+plus II与Quartus II双剑合璧，国防工业出版社，2006.8

[12] 王金明、杨吉斌编著，张雄伟审校，数字系统设计与Verilog HDL电子工业出版社，2002.1

[13] 赵鑫、蒋亮、齐兆群、李晓凯编著，VHDL与数字电路设计，机械工业出版社，2005.4

[14] Altera FLEX 10KE Embedded Programmable Logic Device Datasheet，2003-1，ver 2.5

[15] 李琳、陈勇生，FLEX10K系列EAD的应用，国外电子元器件

[16] 张蕾、黄国策、宋明，用Altera FLEX 10K可编程逻辑器件实现复用器的设计，国外电子元器件

[17] 邓云祥，基于FPGA的虚拟逻辑分析仪的开发．西南交通大学硕士论文，2005.3

[18] 夏宇闻，Verilog数字系统设计教程，北京航空航天大学出版社，2003.7

[19] Mark Balch 著、李兆麟译，完整的数字设计，清华大学出版社，2006.5
[20] Cast Altera H16450S UART with synchronous CPU Interface Megafunction，2006-1
[21] 张若岚、杨南生，UART 的 Verilog HDL 实现及计算机辅助调试，电子产品世界，2002，1，B
[22] ALATEK AL16450 IP Core Application Note，2001.11
[23] Altera ByteBlaster II Download Cable User guide，2004.12
[24] Altera ByteBlasterMV Download Cable User guide，2004.7
[25] 周求湛、钱志鸿、刘萍萍、戴宏亮编著，虚拟仪器与 LabVIEW™7 Express 程序设计，北京航空航天大学出版社，2004.6
[26] 候国屏、王坤、叶齐鑫编著，赵伟、吴静审校，LabVIEW7.1 编程与虚拟仪器设计，清华大学出版社，2005.2
[27] 杨乐平、李海涛、肖相生等编著，LabVIEW 程序设计与应用，电子工业出版社，2001.7
[28] Maxim +5V-Powered Multi-Channel RS-232 Drivers/Receivers Datasheet，1997
[29] NI. Measurement and Automation Catalog，2004
[30] NI. LabVIEW Advanced Performance&Communication Course Manual，2001
[31] National Instruments Corporation LabVIEW User Manual, 2003.4

攻读硕士学位期间发表的论文

1、基于 FPGA 的韦根（Wiegand）协议通讯检测系统．智能卡与电子标签．2006.10