大连理工大学

硕士学位论文

基于USB2.0和FPGA的图像采集、存储系统研究

姓名：张通

申请学位级别：硕士

专业：动力机械及工程

指导教师：孙培岩

20081201

摘　要

图像信号的采集和处理在科学研究、工农业生产、医疗卫生、公共安全等领域得到了越来越广泛的应用，而这些工作都需要一套高速的图像系统来完成。同时图像采集也是进行图像处理、图像压缩、图像识别的基础，所以图像采集系统的研制有着重要的现实意义和价值。要对图像进行采集就需要一种高速的，能进行长时间、大吞吐量数据传送的计算机接口。USB 2.0 接口就是一种符合图像采集要求的计算机接口。同时 USB 接口还具有支持热插拔、占用系统资源少、易于扩展、使用方便等优点。当前，计算机的许多外围设备都采用了 USB 接口来实现与计算机间的数据通信。

本课题“基于 USB 和 FPGA 的图像采集、存储系统研究”，就是要设计一套高速的图像采集、存储、传输系统，通过 FPGA 控制图像的采集，把数据存储到 FLASH 中，当需要查看数据时，通过该系统上的 USB2.0 接口来实现高速的数据传输，将数据送到计算机中进行储存或显示。在本课题中，设计了图像采集系统的硬件电路，制作了 PCB 板，并对图像采集系统的软件部分进行了开发。该图像采集系统由五大部分组成：CMOS 图像传感器（OV7620）电路、FPGA（EP1C6T144C8）控制电路、高速缓存（AL422B）电路，FLASH 存储器（K9F1208）电路、基于 USB2.0(CY7C68013)接口的数据传输部分。

在硬件设计方面，论文提出了图像数据采集系统总体方案，设计了图像采集系统的工作原理，介绍了采集系统中用到的主要芯片以及系统硬件电路中主要模块的设计、FPGA 内部逻辑设计和 PCB 设计等内容。

在软件设计方面，论文讲述了采集系统的 FPGA 程序的设计开发过程；还在 USB2.0 协议的基础上提出了采集系统的 CY7C68013 固件程序的设计思路。在完成采集系统固件程序开发的基础上，论文对固件程序中主程序、初始化函数、端点配置以及 GPIF 接口设计等内容进行了论述。WDM 驱动程序是实现访问和控制硬件的软件接口，应用程序为用户提供界面并最终将 USB 数据传输的结果提供给用户，论文叙述了采集系统的 USB 接口 WDM 驱动程序和应用程序的开发。论文实现了从 FPGA 到计算机的数据传输，并给出显示数据传输的结果。

关键词： 图像采集；USB2.0；FPGA；FLASH

The study of Image Acquisition and storage System Based on USB Interface and FPGA

Abstract

The acquisition and processing of image signal are used widely in the fields of scientific research, industrial and agricultural production, medicine and sanitation, public security and so on. And these works can't be completed without a set of high-speed image acquisition system. Meanwhile image acquisition is the basis of image processing, compressing and recognizing, so the research on image acquisition system is very important and valuable. In order to acquire image, a kind of high-speed computer interface supporting long time and high throughput data transferring is needed. USB 2.0 interface is a computer interface that meets the need of image acquisition. In addition, USB interface has the advantages of high transfer-rate, hot plug and play, less resource requirement, being easily extended and used. At present, many peripheral devices of computer use USB interface to accomplish data communication with computers.

The research of "The study of Image Acquisition and storage System Based on USB Interface and FPGA" is to design a high-speed image acquisition, card acquiring and processing image data, and then transferring the data to computer by USB2.0 interface, to display and store the data on computer. The hardware circuit of the system was designed and PCB board was produced. The software of the card was developed too. The image acquisition card contains the following five parts: CMOS image sensor circuit, FPGA control circuit, SRAM storage, digital signal processor and data transferring part based on USB 2.0interface.

For the hardware circuit design, the conceptual scheme of image data acquisition system was proposed, and then the principle of image acquisition card was demonstrated. Lastly, it's expatiated that the main chips used in the acquisition system, the main modules of hardware circuit, the FPGA logic design and the PCB design etc.

For the software design, the development of FIFO program in the acquisition system was introduced. At the same time, the design scheme of CY7C68013's firmware was given. And the firmware was developed. It is discussed based on firmware design that the design method, developing structure, main program, initial function, and endpoint configure

and GPIF design of CY7C68013 firmware program. WDM driver is the software interface to access and control hardware, and application provides users with interface and the USB data transferring result. So the development of WDM driver and the application of the acquisition system were introduced. Lastly, the data transferring results between the application and FIFO by USB interface were demonstrated.

Key Words: Image acquisition; USB; FPGA; FLASH

大连理工大学学位论文独创性声明

作者郑重声明：所呈交的学位论文，是本人在导师的指导下进行研究工作所取得的成果。尽我所知，除文中已经注明引用内容和致谢的地方外，本论文不包含其他个人或集体已经发表的研究成果，也不包含其他已申请学位或其他用途使用过的成果。与我一同工作的同志对本研究所做的贡献均已在论文中做了明确的说明并表示了谢意。

若有不实之处，本人愿意承担相关法律责任。

学位论文题目：基于USB2.0和FPGA的图像采集、存储系统研究

作者签名：张强　　日期：2008年12月31日

大连理工大学学位论文版权使用授权书

本人完全了解学校有关学位论文知识产权的规定，在校攻读学位期间论文工作的知识产权属于大连理工大学，允许论文被查阅和借阅。学校有权保留论文并向国家有关部门或机构送交论文的复印件和电子版，可以将本学位论文的全部或部分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和汇编本学位论文。

学位论文题目：基于USB2.0和FPGA的图像采集存储系统研究

作者签名：张通　　日期：2008年12月31日

导师签名：[illegible]　　日期：2009年1月6日

1 课题方案的选择

1.1 图像采集系统的发展现状

图像采集系统的发展已有几十年的历史。早期的图像系统处理速度较慢，分辨率较低，这给图像识别带来了困难。随着科学技术的发展，图像采集系统已有了长足进步。从处理静止图像到处理运动图像，从传送 16 位数据到传送 32 位及更多位数据，从 PC 机总线到 ISA 总线及到 PCI 总线，已都取得了明显进步。分辨率从 480×480 到 1024×1024，输入图像范围可大可小。噪声影响也有一定的改善。基本能满足实时性的要求。图像质量也有了进一步的提高。

目前，基于 PCI 的仍然是主流。基于 PCI 总线的图像采集系统向更高性能方向发展，在我国，已有许多家单位在生产图像卡。其中中国科学院大恒图像公司的 DH-VT142，四路复合图像同时输入同频显示，最大平均有效传输速率：4×768×576×24bit 同时传输，相当于 180MBps(依赖于主机传输速率)，同时 DH-VT 142 为用户提供了自行加密的手段，可以保护用户知识产权不受侵犯。DH-CGMPEG 可将两路图像按 MPEG-1 或 MOTION JPEG 压缩并全动态显示，同时可实时采集两路音频。同采集系统加压缩卡的方案相比更方便系统连接，节约了资金，增加了系统稳定性。支持两路 PAL，NTSC 图像信号同时输入同时显示。图像分辨率最高 PAL 768×576×24bit，NTSC 640×480×24bit。国外图像卡在我国占较大市场的是加拿大 MATROX 公司，数据宽度已达 64 位，分辨率达 1600×1200，卡上有 DSP 处理硬件，并可直接接在显示器上显示，但是这些图像采集系统的速度普遍的很高。

USB2.0 的数据传输速率可高达 480Mbps，虽然比 PCI 总线速率低，但是完全满足传输一路图像的要求，同时由于系统设计难度相对较低，使用方便，价格低，基于 USB2.0 的图像采集系统成为研究的热点。

1.2 课题要求

本课题要求能采集多种分辨率（640×480，800×600，1024×768，1280×1024）下的图像信号，目前国外有很多实现该功能的采集卡，而且达到的性能指标也很高。对于 1024×768 的图像，每秒钟输出的帧率可达到 16 帧。但是，国外高速图像采集卡的价格也非常昂贵，通常的费用都在万元以上，限制了其在国内的应用。课题要求，该图像采集卡不但能进行电脑视频(VGA)图像的采集，而且最关键的是要求图像能将图像传感器采集到的图片，储存到外部的 FLASH 中，在需要查看的情况下，通过 PC 控制，可将图片数据上传到上位机（PC）进行显示。

1.3 课题方案的确定

我们选用FPGA作为图像处理的芯片，USB2.0作为图像向上位机传输的接口芯片，目的是要设计一个图像采集，存储系统，将采集到的图像数据在FPGA中进行处理后，存储到大容量的FLASH中，在需要查看数据的情况下，FPGA控制从FLASH中读出图像数据，再通过USB接口送到PC机上进行存储或显示。

该系统由CMOS图像传感器OV7620、可编程逻辑器件器件EP1C6T144C8，高速缓存器件AL422B，FLASH储存器K9F1208，USB2.0控制器CY7C68013组成，系统原理图1.1如下：

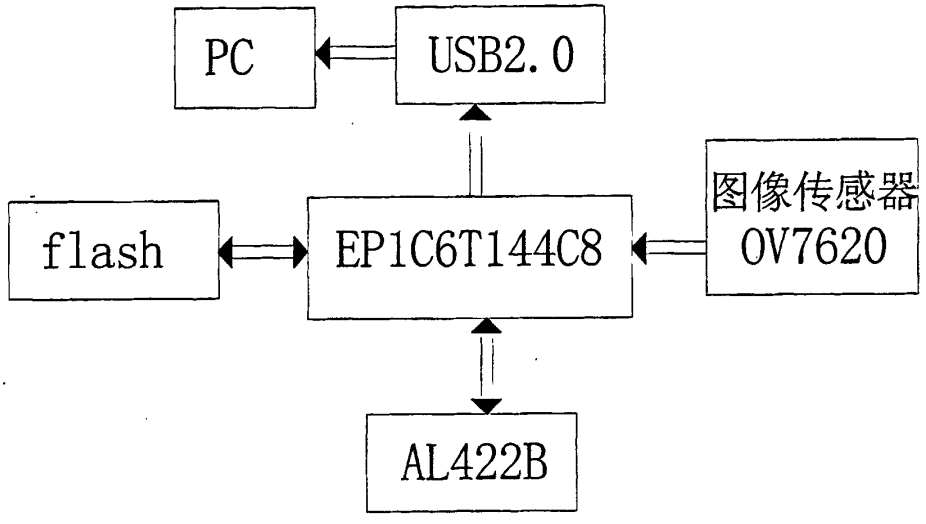


图1.1 系统原理框图

Fig. 1.1 The block diagram of the system

系统中，EP1C6T144C8的时钟使用了图像传感器的输出时钟PCLK，这是为了保证系统在时序上的同步。图像传感器HREF信号为水平窗口参考信号，高电平时有效。

系统的工作流程为：上电后，OV7620按设定方式工作，自动输出数据和行同步时信号和时钟信号，FPGA接收图像数据。OV7620每一个PCLK上升沿就有一个输出数据。当HREF为高电平时，输出的数据为有效数据。FPGA就是利用PCLK和HREF从OV7620接收数据。紧接着FPGA将接收到数据送到FIFO高速缓存内，同时FPGA(时钟为10分频后)控制将FIFO内的数据转存到大容量的FLASH中；在需要察看图像的情况下，FPGA通过USB接口，将图像数据传送到PC机，这里考虑到OV7620输出数据

和 FLASH 写数据的速度问题，由于 FLASH 的读写速度比较慢，而数据的输出速度很快，所以在设计中加入了高速的 FIFO，作为数据缓冲使用。上传图像数据时，CY7C68013 工作在 SLAVE FIFO 模式，当 CY7C68013 收到来自 FPGA 的数据后，自动将数据打包，再通过 USB 总线，将数据送给计算机进行存储或.显示，完成整个数据处理和传送过程。

1.4 课题任务和论文的内容

本课题的主要研究任务是设计一个基于 USB2.0 和 FPGA 图像采集、存储系统。课题要求在 USB 2.0 规范的基础上，对采集系统的硬件和软件进行设计和开发，通过 USB 2.0 接口将图像数据采集到 PC 机。

论文的内容：

(1) 介绍了图像采集系统的发展现状，课题要求，设计方案的确定及论文的内容。

(2) 阐述了基于FPGA和USB的硬件设计方案，对主要芯片和芯片间的连接进行介绍。

(3) 介绍了FPGA的特点和开发流程，主要对FPGA控制FLASH的状态机进行了介绍。

(4) 介绍了本系统中USB的固件程序设计。

(5) 介绍了USB设备的驱动程序和应用程序设计。

2 采集系统的硬件电路设计

2.1 主要应用芯片选择

2.1.1 FPGA芯片的选择

在目前市场上的图像采集处理系统中，用作系统的核心器件一般有FPGA，数字信号处理器(DSP，Digital Signal Processor)以及特定用途集成电路(专用ASIC，Application Specific Integrated Circuit)芯片。

FPGA作为当今主流的大规模可编程专用集成电路，可以实现系统集成，具有实现宏函数的嵌入式阵列及实现普通功能的逻辑阵列，提供异步的“乘积项”或者“和项”构成的寄存器的置位/复位信号；对于I/O驱动能力强的器件，可以在印刷线路板上直接驱动总线，每个I/O引脚都有一个独立的三态输出使能控制，从而可以方便地实现总线结构，另外每个I/O引脚都有漏极开路选择，还有单独的、可编程的输出电压摆率控制位，可配置成高摆率或低摆率输出，大多数的FPGA器件都具有符合IEEE 1149.1技术规范的边界扫描测试CTAG电路，可以方便地进行芯片和板级的测试。还可以根据后期现场的需要进行现场编程，可擦写多次。

图像信号处理类的低层信号处理运算主要面向图像象素操作，核心算法部分是乘加等算术逻辑运算和最值运算、二值分割、绝对值等非线性运算，特点是数据量大但运算简单机械，适于采用FPGA并行操作流水线方式实现，虽然DSP有专用的硬件乘法器，但与FPGA相比较，在此类算法上，FPGA有更大的数据吞吐量。

本课题中，实时图像采集和处理对系统性能的要求极高，数据流量比较大，因此几乎所有只具最简单功能的通用DSP都不能在图像采集处理系统中达到所需的性能要求。

基于FPGA的算法具有专用集成芯片的运算速度，可以满足高速图像处理的需要；FPGA现场可编程，使检测系统具有良好的柔性，满足不同检测需求；FPGA的编程技术发展成熟，算法实现简单，编程容易，具有很好的可移植性和继承性。

因此，结合国内图像采集和处理系统的现状，我们采用FPGA作为核心器件代替DSP或者ASIC芯片来实现图像采集和处理，研发出基于FPGA的高速高分辨率图像采集处理系统。

EP1C6T144C8是ALTERA公司Cyclone系列FPGA中的一款，是基于成本优化的，全铜工艺的1.5V SRAM工艺，相对竞争对手的FPGA仅一半的成本，依然提供强大的功能。最高达20，060个逻辑单元和288K位的RAM，除此之外，Cyclone系列的FPGA

还集成了许多复杂的功能。Cyclone 管理和专用 I/O 接口，这些接口用于连接业界标准的外部存储器件。

Cyclone 系列器件提供低成本 FPGA 解决方案，具有无与伦比的灵活性和高性能的系统集成性，具体表现在以下几个方面：

(1) 成本优化的架构：Cyclone 器件具有多达 20060 个逻辑单元，容量是以往低成本 FPGA 的四倍。

(2) 嵌入式存储器：Cyclone 器件中 M4K 存储块提供 288Kbit 存储容量，能够被配置来支持多种操作模式，包括 RAM、ROM、FIFO 及单口和双口模式。

(3) 外部存储器接口：Cyclone 器件具有高级外部存储器接口，允许设计者将外部单数据率（SDR）SDRAM，双数据率（DDR）SDRAM 和 DDR FCRAM 器件集成到复杂系统设计中，而不会降低数据访问的性能。

(4) 支持 LVDS：Cyclone 器件具有多达 129 个兼容 LVDS 的通道，每个通道数据率高达 640Mbps。

(5) 支持单口 I/O：Cyclone 器件支持各种单端 I/O 接口标准，如 3.3V、2.5V、1.8V、LVTTL、LVCMOS、SSTL 和 PCI 标准，满足当前系统要求。

(6) 时钟管理电路：Cyclone 器件具有两个可编程锁相环（PLL）和八个全局时钟线，提供健全的时钟管理和频率合成功能，实现最大的系统性能。Cyclone 的 PLL 具有多种高级功能，如频率合成、可编程相移、可编程延迟和外部时钟输出。这些功能允许设计者管理内部和外部系统时序。

(7) 接口和协议：Cyclone 器件支持诸如 PCI 等串行、总线和网络接口，可访问外部存储器和以以及拥有多种通信协议如以太网协议。

(8) 热插拔和上电顺序：Cyclone 器件具有健全的片内热插拔和顺序上电支持，确保上电顺序无关的正常工作。这一特性在上电前和上电期间起到了保护器件的作用并使 I/O 缓冲保持三态，让 Cyclone 器件成为多电压系统及需要高可用性和冗余性应用的理想选择。

(9) DSP 实现：Cyclone 器件位在 FPGA 上实现低成本数字信号处理（DSP）系统提供了理想的平台。

(10) 串行配置器件：Cyclone 器件能用 Altera 新的串行配置器件进行配置。

(11) Nios II 系列嵌入式处理器：Cyclone 器件的 Nios II 系列嵌入式处理器能降低成本，增加灵活性，非常适合于替代低成本的分离微处理器。

(12) 支持工业及温度：部分 Cyclone 器件提供工业级温度范围-40 摄氏度至＋100 摄氏度的产品，支持各种工业应用。

(13) 扩展温度支持：部分 Cyclone 器件达到扩展温度范围－40 摄氏度至＋125 摄氏度。这些器件是需要扩展温度范围和高质量产品的 In-cabin automotive 应用的理想选择。

在本系统中，系统硬件电路板需要实现以下功能：

(1) 控制图像数据的采集：FPGA 将图像传感器输出的图像数据采集出来。

(2) 处理算法的实现：FPGA 对图像传感器采集到的图像数据进行预处理和可行的后期处理

(3) 控制图像数据的储存：FPGA 将处理过的图像数据储存到 FLASH 中。

(4) 图像数据向上位机的传输：通过 FPGA 将处理后的图像数据通过向 USB 接口芯片传输，实现向上位机的图像数据的传输。

(5) FPGA 配置芯片的设计：实现 FPGA 的自动上电配置。

2.1.2 USB 芯片的选择

数据采集系统的发展离不开数据传输方式的发展。外设与 PC 机通信的方式很多，如 ISA 接口、PCI 接口、PS/2 接口、串行接口、并行接口等。表 2.1 是 USB 和其他常用计算机接口的比较。

表 2.1 常用计算机接口的性能对比（典型值）[1]

Tab. 2.1 Comparison of PC interfaces in common use (Typical Value) [1]

接口类型	数据格式	传输速率（b/s）	最大设备数(个)	电缆长度（m）	是否支持热插拔
USB 串行	串行	1.5M、12M、480M	126	3 或 5	是
RS232	串行	20K	2	15~30	否
RS485	串行	10M	32	1200	否
IEEE-1394	串行	400M、3.2G	63	4.5	是
以太网	串行	10M、100M、1G	1024	500	否
并行端口	并行	8M	2 或 8	3~9	否
ISA	并行	128M	— —	— —	否
EISA	并行	256 M	— —	— —	否
—PCI	并行	1056M、2112M	— —	— —	否
AGP	并行	≥2112M	— —	— —	否

对于本课题的设计，由于图像的数据量很大，要进行图像数据的传输必须使用 480Mbps 的高速传输才能满足要求，所以要选择能够支持高速传输的 USB 芯片。在硬件资源上，要易于和 FPGA 实现连接和控制，所以选择了带有 MCU 的 USB 芯片，较大

的存储器也可以方便系统的开发。本课题选择 Cypress 公司的 EZ-USB FX2 系列芯片 CY7C68013，符合传输速度和硬件资源上的要求。

Cypress 公司的 EZ-USB FX2 系列芯片产品是世界上第一款支持 USB2.0 协议的微控制器。它支持 12Mbps 的全速传输和 480Mbps 的高速传输，可使用全部的 USB 传输方式。图 2.1 为 CY7C68013 芯片的内部结构图。

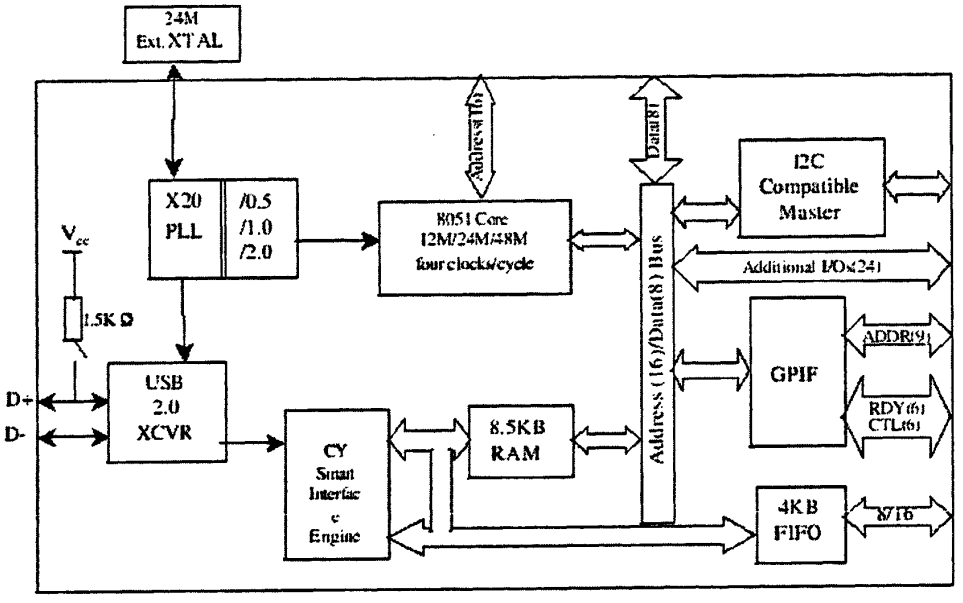


图 2.1 CY7C68013 内部结构

Fig.2.1 Block Diagram of CY7C68013

CY7C68013 是 Cypress 公司推出的 EZ-USB FX2 系列产品之一。CY7C68013 内部集成了 USB2.0 收发器、智能串行引擎（SIE）、增强型 8051 控制器、通用可编程接口(GPIF)、8.5KB 的 RAM 和 4KB 的 FIFO 存储器，具有很高的性价比。FX2 系列的独创性设计最大限度的满足了 USB2.0 的总线带宽，而它的内部使用的仍是低成本的 8051 控制器。下面对 CY7C68013 芯片内的主要部分简单介绍[4]。

(1) 增强型 8051 内核，CY7C68013 中的增强型 8051 与标准的 8051 相比，具有更快的速度、更强的功能，且其指令集和标准 8051 完全兼容；该内核的一个总线周期为 4 个时钟周期，而标准 8051 的一个总线周期由 12 个时钟周期组成；增强型 8051 的默认工作频率为 48MHz，需要在芯片外部接 24MHz 晶振，经过内部振荡电路和锁相环（PLL）倍频电路，产生 48MHz 的时钟；CY7C68013 内部 8.5KB 的 RAM 可以兼做程序存储器

和数据存储器，增强型8051的固件代码就存储在该区域内；固件代码可以通过USB总线进行下载，也可以通过I2C总线从外部EEPROM下载。

(2) 智能串行引擎（SIE），CY7C68013芯片内串行接口引擎负责完成USB协议的有关功能，包括数据的编解码、差错控制、位填充等。这样就将MCU（增强型8051）解放出来，从而简化了固件代码的开发。

(3) 通用可编程接口（GPIF），在CY7C68013中还有一个通用可编程接口（GPIF），它支持几乎所有通用的总线标准，可以由软件来编程输出读写控制波形。

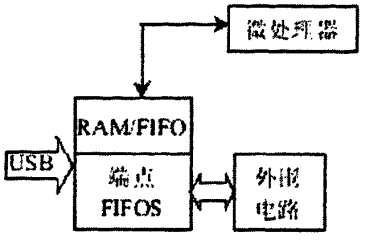


图 2.2 CY7C68013的数据传输模式

Fig.2.2 Data Transfer Mode of CY7C68013

(4) FIFO芯片中有一个大小为4K的FIFO存储器，USB接口和外围电路直接共享这个FIFO存储器。图2.2为FIFO的传输模式示意图。

从图中可以看出，与外围电路的数据传输可以在没有8051固件干涉的情况下完成，8051也可以通过FIFO或RAM的方式访问所传输的数据。FIFO存储器与USB之间的传输是以数据包的形式实现的，而不是一次只传输1字节，这种处理被称为"量子FIFO"，它较好地解决了USB高速模式下的带宽问题[1]，非常适合于大容量的数据传输。如果芯片时钟频率为48MHz，使用FIFO存储器进行数据传输，最高速率可以达到96Mbyte/s。

2.1.3 图像传感器芯片的选择

图像传感器（Imaging Sensor：IS)的功能是把光学图像信号转换为电信号，即把入射到传感器光敏面上按空间分布的光强信息(可见光和不可见光)，转换为按时序串行输出的电信号—视频信号，而视频信号能再现入射的光辐射图像。

固体图像传感器主要有三大类型，一种是电荷耦合器件(Charge Coupled Device：CCD)、第二种是CMOS图像传感器，又称为自扫描光电二极管列阵（Self Scanned Photodiode Array：SSPA)、第三种是电荷注入器（Charge Injection Device：CID)。目前，前两种用得比较多。

CMOS 图像器根据像元的小同又分为:线列阵、面列阵以及特殊列阵等。

另外，与 CCD 相比，CMOS 的体积小，耗电量小到 CCD 的 1/10，售价也比 CCD 便宜 1/3 的优点。CMOS 是标准工艺流程，可利用现有半导体设备，不需要额外投资设备，且品质可随着半导体技术的提升而进步。同时，全球晶圆厂的 CMOS 生产线较多，以后进行批量生产时也有利于成本的降低。

CMOS 传感器的最大优势是已具有高度系统整合的条件。理论上，所有图像传感器所需的功能，例如垂直位移、水平位移寄存器、时序控制、CDS，ADC 等，都可放在集成在一颗晶片上，甚至于所有的晶片包括后端晶片(Back-end Chip、闪存(Flash Memory)等也可集成到一块品片上构成片上系统。

CMOS 图像传感器一般由光敏像元阵列、行选通逻辑、列选通逻辑、定时控制电路和片内模拟信号处理器(ASP)等组成。数字式 CMOS 图像传感器还集成有片内的模/数转换器(ADC)。CMOS 图像传感器的光敏像元有无源像素结构和有源像素结构两大类。后者主要有光敏二极管和光栅型两种。行选通逻辑和列选通逻辑可以是移位寄存器，也可以是译码器。定时和控制电路决定信号的读出模式、设置积分时间、控制数据输出速率等。在片模拟信号处理器完成信号积分、放大、取样和保持、相关双取样、双△取样等功能。在片模拟/数字转换器是在片数字成像系统所必需的，CMOS 图像传感器可以是整个成像阵列有一个 ADC 或几个 ADC(每种颜色一个)，也可以是每个成像阵列各有一个 ADC。

OV7620 是 OmniVision 公司推出的一款高集成度、高分辨率（640×480）、逐行/隔行扫描、彩色/黑白 CMOS 数字图像传感器芯片。它的数字视频口支持多种输出格式，包括：60Hz 的 16 位/8 位 YCrCb 4：2：2 格式，ZV 口输出格式、RGB 原始数据 16 位/8 位输出格式和 CCIR601/CCIR656 格式。可以很容易地通过内置的 SCCB(Serial Camera Control Bus)接口来控制芯片的功能。其原理框图见图 2.3。

OV7620 主要的性能参数有：

——单片数字式彩色/黑白图像传感器；

——逐行扫描/隔行扫描方式；

——成像面积：4.86 mm×3.64mm；

——像素尺寸：7.6μm×7.6μm；

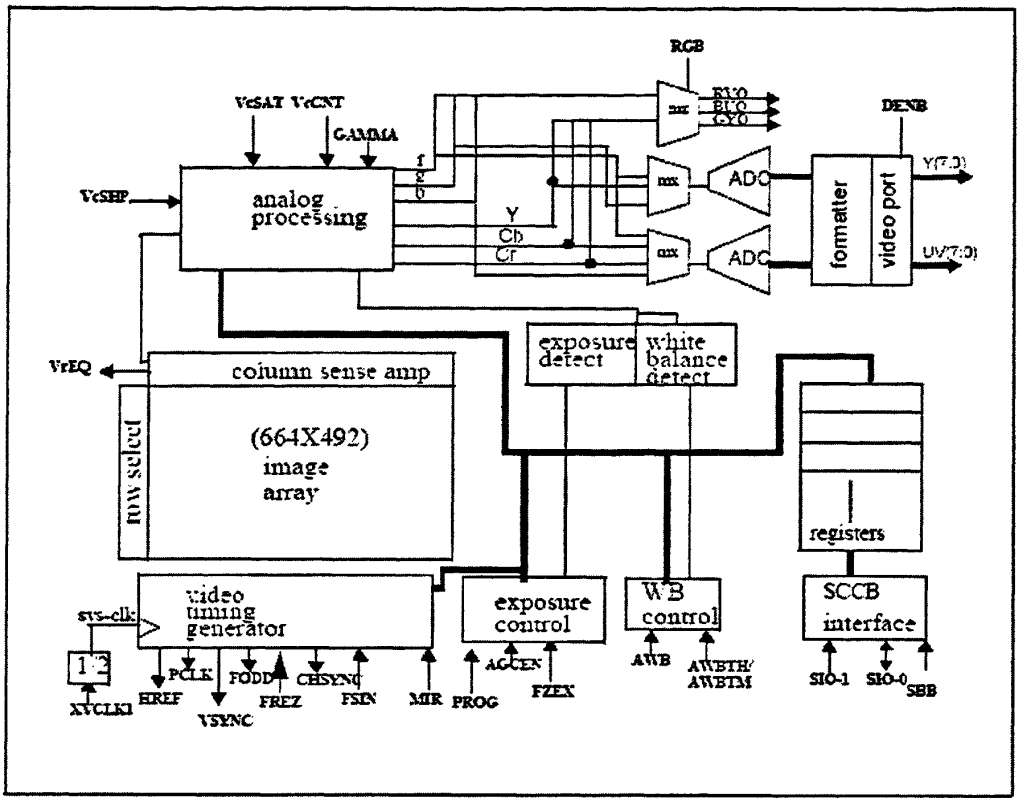


图 2.3 OV7620 原理框图

Fig.2.3 OV7620 Block Diagram

——总有效像素单元：664(水平方向)×492(垂直方向)；

——缺省像素单元：640(水平方向)×480(垂直方向)；

——10 位双通道 A/D 转换器，8 位/10 位数据输出；

——数字视频输出格式：可选择 YCrCb 的 16Bit 模式或 8Bit 模式

60Hz 16Bit YCrCb 4：2：2-640×480

60Hz 8Bit YCrCb 4：2：2-640×480

——CCIR601/CCIR656 标准视频格式；

——支持 QVGA(320×280)格式；

——SCCB 总线接口，支持 400Kbit/s 快速模式；

——自动增益和自动白平衡控制；

——能对亮度、对比度、饱和度、Y 校正等多种功能进行调节；

——输出帧频可编程（范围为 0.5 到 30 帧/秒）；

——48 脚 LCC 封装。

在进行 SCCB 寄存器的设置时，要注意如下问题：OV7620 支持单片或多片（最多可支持 8 片）工作，可以通过在复位时设置它的引脚 CS[2：0]来设置每一个芯片的 ID 值。如果 OV7620 是单片工作，那么它的 ID 值为 42H(写 SCCB 寄存器)和 43H(读 SCCB 寄存器)。在进行读写 SCCB 寄存器时，SCCB 线上第一个字节为 OV7620 的 ID 值。在写周期，SCCB 总线上的第二节为用来选择片上寄存器的子地址，第三个字节是写入寄存器的数据;在读周期,第二个字节返回的是上次访问的子地址的数据。如图 2.4 为 OV7620 工作在 8 位 YUV 输出格式下的时序图。

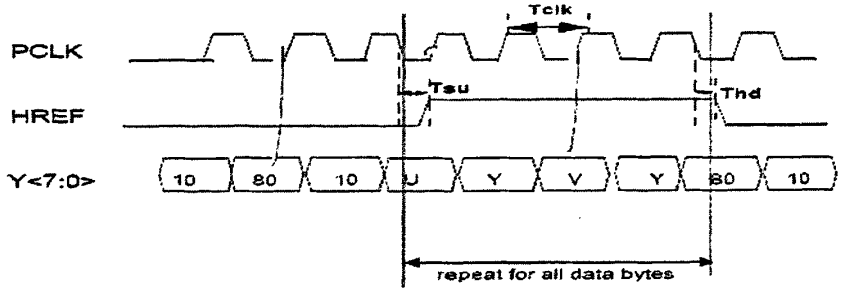


图 2.4 位像素数据输出时序(Y 格式输出)

Fig.2.4 Pixel Data 8-bit Timing(Y Output)

其中，在 PCLK 的上升沿锁存数据。图中的 PCLK 为像素时钟周期，当 OV7620 的系统时钟为 27MHz 时，Tclk=74ns；Tsu 是 HREF 的建立时间，最大为 15ns；Thd 为 HREF 的保持时间，最大为 15ns。

2.1.4 K9F1208 的结构与工作原理

K9F1208 是三星公司生产的 64M×8bit NAND FLASH 存储器，工作电压为 2.7-3.6V，内部存储结构为 528 字节×32 页×4096 块，可实现页的读取、写入、擦除等操作。K9F1208 由存储阵列、输人输出缓冲及锁存、命令寄存器(保存外部设备对存储器操作的类型)、地址锁存与译码(保存被访问的地址并产生相应的译码选通信号)、控制逻辑及高电压发生器等五部分组成，其结构如图 2.5 所示。外部电路通过 8 位 I/0 端口分时访问 K9F1208 的命令寄存器、地址寄存器和数据寄存器，完成对芯片内部存储体的访问。

K9F1208 采用 48-PIN TSOP 封装形式，但有效引脚只有 17 个，其功能如下。

CLE：命令允许(锁存)，平时为低电平。当 CLE 为高电平时，在 WE 的上升沿将 I/0 端口上的命令信息输人并锁存到命令寄存器。

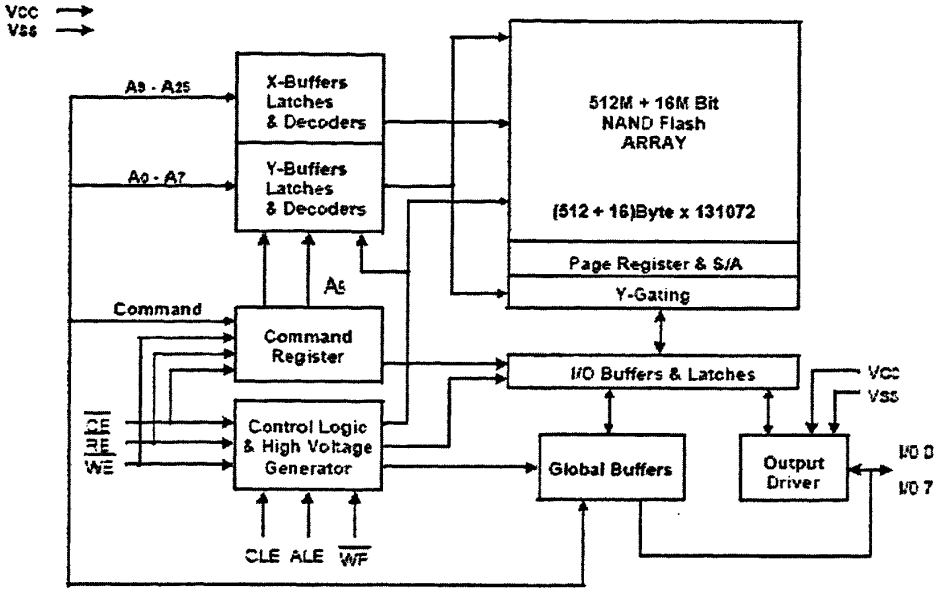


图 2.5 功能结构图

Fig2.5 Functional Block Diagram

ALE：地址锁存，平时为低电平。当 ALE 为高电平时，在 WE 的上升沿将 I/O 端口上的地址信息输入并锁存到地址寄存器。

CE：片选，低电平有效，用于器件的选择控制。当 CE 变为高电平时，器件返回到备用(未选中)状态.注意：当器件处于写操作或擦除操作的“忙”状态时，CE 的变高将被忽略，不会返回到备用状态。

WE：写控制，平时为高。在 WE 的上升沿，将 I/O 端口上的命令、地址和数据信息分别输入到相应寄存器。

RE：读控制，平时为高，在它的下降沿 tREA 时间后，I/O 端口上的输出数据有效，同时内部列地址自动加 1。

I/O 端口：用于命令、地址和数据的输入及读操作时的数据输出。当芯片未选中时，I/O 端口为高阻态。

WP：写保护，低电平有效。当它为低电平时，内部的高电压发生器将停止工作，实现禁止写和擦除操作。

R/B(准备好/忙)：当前器件的工作状态，当它为低电平时，表示正在进行写、擦除以及随机读操作。当它为高电平时，表示这些操作已经完成。

2.2 硬件连接设计

数据采集系统基本原理已经在前面章节作了叙述，本节主要目的是更具体地说明系统的重要电路的原理和设计。

2.2.1 OV7620 与 EP1C6T144C8 的连接

OV7620 与 EP1C6T144C8 连接示意图如图 2.6。

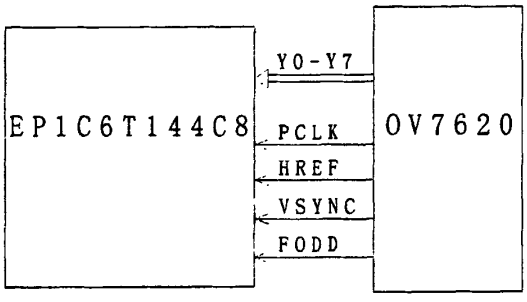


图 2.6 OV7620 与 EP1C6T144C8 连接示意图

Fig.2.6 Connection diagram of the OV7620 and EP1C6T144C8

OV7620 的数据输出 Y[7：0]（注：由于设置了图像传感器的寄存器，图像数据是 8 位输出模式）和连接到 EP1C6T144C8 的 I/O 引脚，同时帧同步信号 VSYNC 和行同步信号 HREF 也连接到 EP1C6T144C8 的 I/O 引脚。OV7620 的 PCLK 引脚输出为点同步信号，当 OV7620 正常工作时，PCLK 会不断输出与 VSYNC 和 HREF 号无关的固定时钟。这里必须注意的是 OV7620 的像元输出时钟 PCLK 要连接到 EP1C6T144C8 的全局时钟引脚（Pin83），作为 EP1C6T144C8 的全局时钟。这样才能保证 OV7620 与 EP1C6T144C8 保持同步。OV7620 输入时钟为 27MHz。

当 HREF 为高电平，PCLK 的处在上升沿的时候，就有一个有效数据出现在数据线 Y[7：0]上，这时 EP1C6T144C8 开始读取数据。

OV7620 的 SCCB 接口有 SCL 与 SDA 两个引脚，用于设定 OV7620 的寄存器，这两个引脚与 CY7C68013 连接。这样就可以通过 CY7C68013 对 OV7620 的工作方式进行设定。

2.2.2 EP1C6T144C8 与 AL422B 的连接

EP1C6T144C8 与 AL422B 的连接示意图如图 2.7 所示：

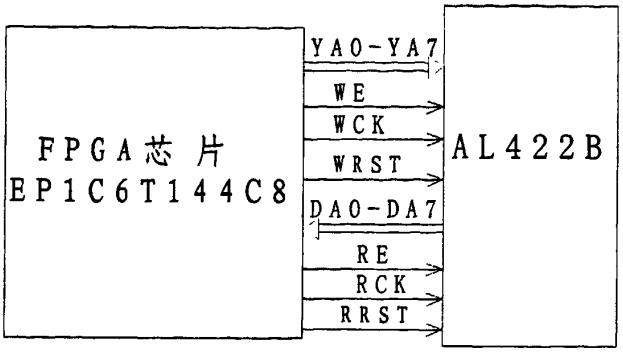


图 2.7 EP1C6T144C8 与 AL422B 连接示意图

Fig.2.6 Connection diagram of the EP1C6T144C8 and AL422B

FPGA 接收到 OV7620 送来的数据后，通过图 2.7 所示的方式将数据送到 AL422B 内的存储器中。如上图所示，在数据写入时，FPGA 的输出数据线（8 个 I/O 引脚）连接到 AL422B 的 8 位数据线（DI0-DI7）上，FPGA 的写入控制线（2 个 I/O 引脚）分别的 FIFO 的 WE 和 WRST 上。FPGA 的一个 I/O 引脚连接到 AL422B 的 WCK，可以向 AL422B 提供写入时钟。当 WE 为低电平时，在 WCK 信号的上升沿，数据将通过 DI7～DI0 写入到写寄存器。按照 WCK 的输入周期，其写入的数据必须满足建立和保持时间的要求。当 WE 为高电平时，写操作被禁止，写地址指针停在当前位置上；当 WE 再次变为低电平时，写地址指针从当前位置开始。

数据读出时，FPGA 的输出数据线（8 个 I/O 引脚）连接到 AL422B 的 8 位数据线（DO0-DO7）上，FPGA 的写入控制线（3 个 I/O 引脚）分别接到 AL422B 的 RE，OE 和 RRST 上。FPGA 的一个 I/O 引脚连接到 AL422B 的 RWCK，但这个时钟是经过 FPGA 分频后的时钟，可以向 AL422B 提供写入时钟。当 RE 和 OE 为低电平时，在 RCK 信号的上升沿，数据由 DO7～DO0 输出。当 RE 为高电平时，读地址指针停在当前位置上；当 RE 再次变为低电平时，读地址指针从当前位置开始。执行读操作时，OE 应为低电平。若 OE 为高电平，则数据输出端也为高阻态，且读地址指针仍然同步加 1。RE 和 OE 应按照 RCK 的输入周期来工作，同时应满足建立和保持时间的要求。

通常，复位信号可在任何时候给出而不需考虑WE、RE及OE的状态；但是它们仍然要参照时钟信号的输入情况来满足建立时间和保持时间的要求。如果在禁止时钟周期内给出复位信号，那么，就必须等到允许周期到来后才会执行复位操作。当WRST和RRST均为低电平时，数据的输入和输出将从地址0开始。

2.2.3 EP1C6T144C8 与 K9F1208 的连接

EP1C6T144C8 与 K9F1208 的连接连接示意图，如图 2.8 所示：

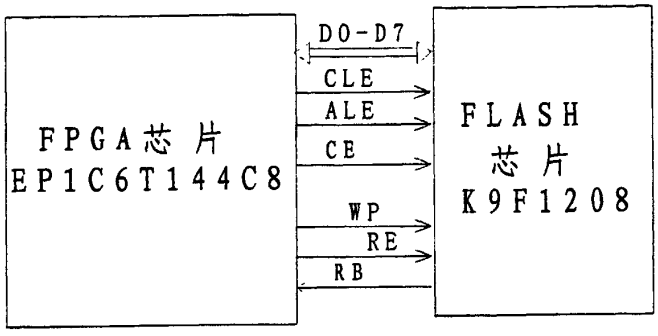


图 2.8 EP1C6T144C8 与 FLASH 连接示意图

Fig.2.8 Connection diagram of the EP1C6T144C8 and FLASH

前面已经对 NAND FLASH 性能和使用方法做了介绍，这里只对 K9F1208 与 EP1C6T144C8 的连接做介绍.

K9F1208 的数据总线引脚 I/0~I/07 连接到 EP1C6T144C8 的 I/O 引脚上，进行数据的双向传输。

K9F1208 的控制引脚 CLE，ALE，CE，WP，RE，RB 分别与 EP1C6T144C8 的 I/O 引脚相连，FPGA 发出控制信号对 NAND FLASH 进行控制，控制数据的输入和输出。

2.2.4 EP1C6T144C8 与 CY7C68013 的连接

连接示意图，见图 2.9 所示。

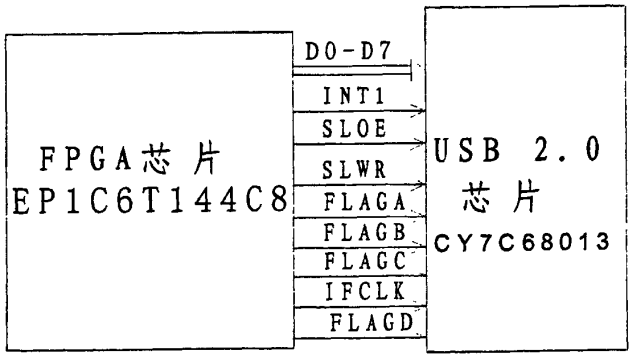


图 2.9 EP1C6T144C8 与 CY7C68013 连接示意图

Fig.2.9 Connection diagram of the EP1C6T144C8 and CY7C68013

其硬件连接示意图如图所示：D0-D7 位数据总线，用于传输图像数据，INT1 位中断标志，当传输完一幅图像时产生一次中断。其中，IFCLK 为外部时钟输入；FIFOADR[1:0]选择四个 FIFO 中的一个，本设计中 FIFOADR[1]直接接+3.3V，FIFOADR[0]直接接地，选择用 EP6 传输数据，4 倍缓存；FULL 为 FIFO 满状态标志，低电平有效；SLWR 为数据写入 FIFO 使能，低电平有效；FD[7：0]为 8 位数据线；PKEND 为包结束标志，低电平有效。在同步从机模式下，只有当缓冲区满时数据才会被转发，而传送的数据只能是我们制定的缓冲区大小的整数倍，因而，当最后一帧没有填满缓冲区时，我们需要将 PKEND 拉低，表示这是最后一帧，这样，控制器就会将这些数据转发。

如上所述，我们选择了FIFO的同步被动模式进行数据传输的方式。同步被动模式的时序工作图如图2.10所示，在IFCLK的上升沿，如果SLWR有效(即为低电平)，而数据有满足建立保持时间，数据就被写入FIFO，同时，如果这时，内部控制器发现FIFO已满，经过一定延时(大约为10ns)，FULL标志会被置成有效。

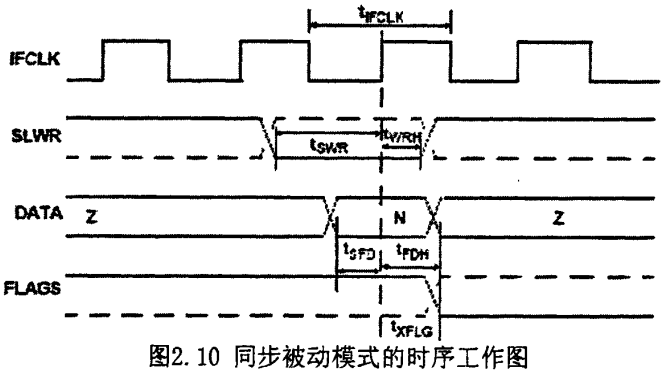


图2. 10 同步被动模式的时序工作图

Fig2.10 Slave FIFO Synchronous Write Timing Diagram

2. 2. 5 CY7C68013 外围电路及接口电路设计

图 2.11 为 CY7C68013 外围电路及接口电路图。图中 J1 为设备到主机的 USB 接口。本系统中 USB 设备的供电方式采用自供电和外供电两种方式，一种是主机的电源线+5V 电源，一种是外接的+5V 电源。由于 CY7C68013 需要的电压位+3.3V，所以设计中设计了+5V 转换位+3.3V 的电压转化电路，芯片型号为 ASM1117-3.3V。J1 中的 DPLUS 接 D+，DMINUS 接 D-，这是 USB 接口的串行数据线。J1 的 GND 端接到电路板的地，使主机与 USB 设备电平能够匹配。同样，SCL、SDA（无论是否使用这两个引脚）也必须通过 2.2 欧姆电阻上接到+3.3V 电源才能使 CY7C68013 正常工作，外接 EEPROM 信号

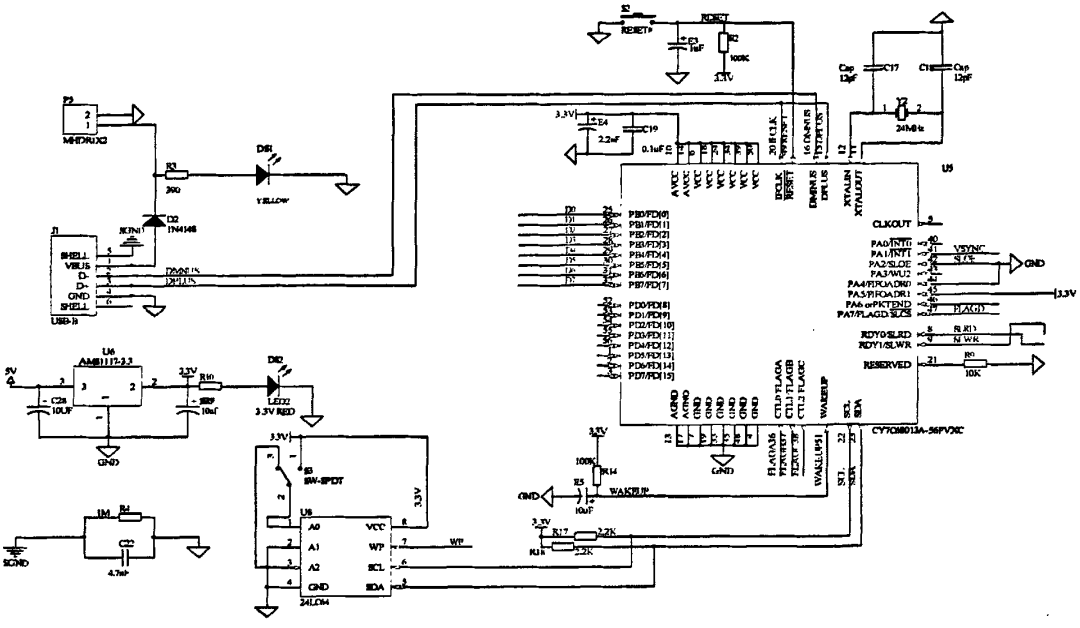


图 2.11 CY7C68013 外围电路及接口电路

Fig2.11 The external and interface circuit of CY7C68013

为 24LC64，用于存储固件程序。XTALIN 和 XTALOUT 两个引脚接 24M（CY7C68013 的推荐时钟频率）晶振。RESET 为复位引脚，低电平有效。E3 和 R2 构成上电复位电路。同样，WAKEUP 的作用是使 CY7C68013 的 8051 内核退出挂起状态。当 8051 内核处于挂起状态，如果这时让 WAKEUP 有效，则晶振开始启动并且向 8051 发出中断，让其退出挂起状态。

2.2.6 EP1C6T144C8 外围电路设计

EP1C6T144C8 外围电路，如图 2.12 所示：

2.2.6.1 配置部分

器件工作期间，EP1C6 器件将配置数据保存在 SRAM 中。因为 SRAM 数据是易丢失的，SRAM 单元必须在器件每次加电后重新装入配置数据。当 EP1C6 器件完成配置后，它的 I/O 引脚及输入/输出寄存器被初始化，然后，器件进入用户模式，系统开始正常工作，如图 2.13 所示。

EP1C6 芯片在加载中利用四个控制信号：nCONFIG，nSTATUS，CONF_DONE，DCLK。nCONFIG 是 EP1C6 芯片的输入信号。当此信号为低时，EP1C6 芯片处于复位

状态。当 nCONFIG 出现由低到高的跳变时，芯片开始加载。nSTATUS 是双向漏极开

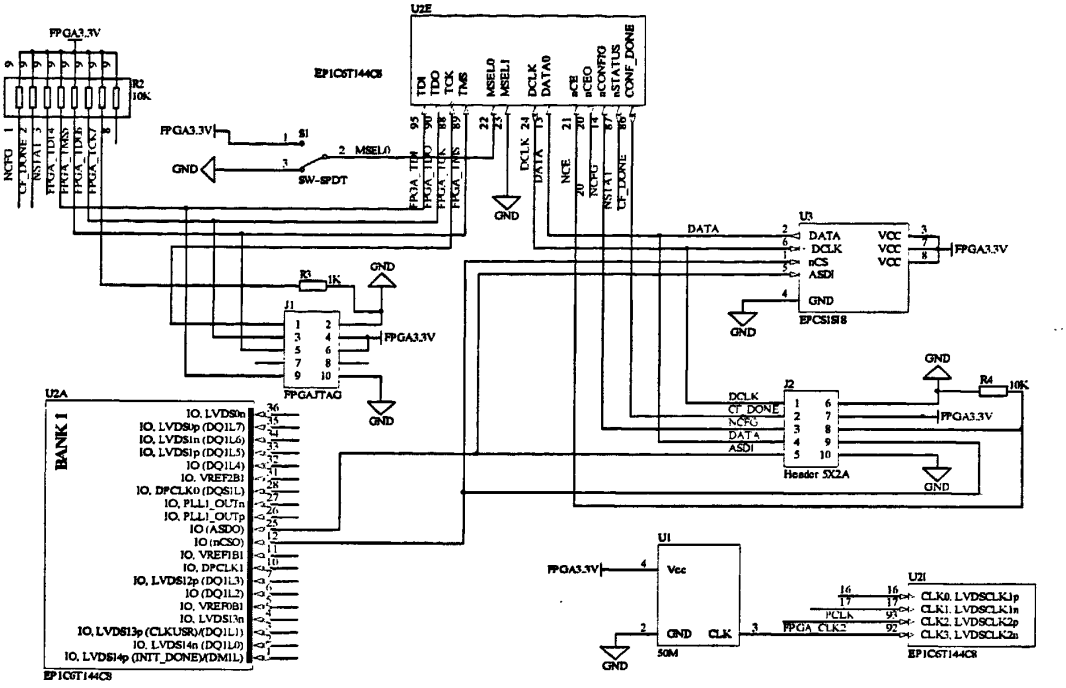


图 2.12　EP1C6T144C8 外围电路

Fig2.12 The external circuit of EP1C6T144C8

路信号。此信号需经上拉电阻与 VCC 相连。上电后 EP1C6 芯片将此信号拉低，并保持 100ms 的时间。若加载中出现错误，芯片将此信号拉低。因而 nSTATUS 可用于错误指示。若外部器件将 nSTATUS 拉低，则 EP1C6 芯片将进入错误状态。

CONFIG-DONE 是双向漏极开路信号。此信号需经上拉电阻与 VCC 相连。作为输出时，CONFIG-DONE 可作加载结束标志。在加载期间，EP1C6 芯片将此信号拉低，芯片正确加载后，此信号被释放。但如果配置器件输送完所有的配置数据而 CONFIG-DONE 没有一变高，则标志配置失败。在这种情形下，配置器件会将它的 OE 端拉低数微秒，从而使得 EP1C6 器件的 nSTATUS 端变低，进行重新配置。

CONFIG- DONE 信号作为输入时，此信号变高将使 EP1C6 芯片初始化；进入用户模式。DCLK 是时钟输入信号。此信号需经上拉电阻与 VCC 相连。此信号将被用于锁存外部数据。过配置过程中，当此信号为低时，锁存外部数据；当此信号为高时，数据线状态改变。当配置结束时，配置器件将 DCLK 信号拉低。

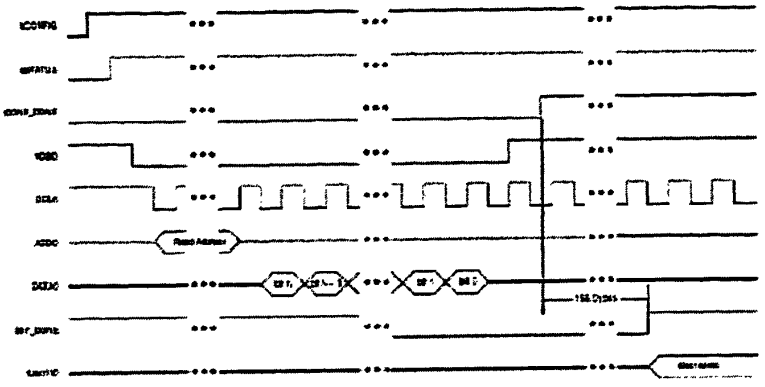


图 2.13 主动配置波形图

Fig.2.13 Active configuration wave Diagram

EP1C6T144C8 的配置方案有三种：主动串行配置 AS（Active Serial Configuration）、被动串行配置 PS（Passive Serial）、基于 JTAG 的配置，用户可以根据设计需要通过 EP1C6T144C8 片上的 MSEL0 和 MSEL1 选择适当的配置方案，所有配置方案下，MSEL1 均接低电平，当 MSEL0 接高电平时，系统采用被动串行配置（Passive Serial），当 MSEL0 接低电平时，系统采用主动串行配置方案（Active Serial Configuration），使用 JTAG 配置时，不考虑 MSEL0 和 MSEL1 的状态。Cyclone 的配置模式的关系如表 2.2 所示：

表 2.2 Cyclone 的配置图

Table 2.2 Configuration schemes of Cyclone

MSEL1	MSELO	配置表
0	0	AS
0	1	PS
0	0 或 1	JTAG-based

主动串行配置（AS）采用低成本非易失性配置芯片 EPCS1 和 EPCS4，四个引脚的配置接口，这个方案之所以被称为主动模式是因为与被动串行配置模式（PS）配置芯片控制配置接口相比较，主动串行配置（AS）过程中 Cyclone 系列 FPGA 控制配置接口。主动串行配置模式下配置芯片的四个配置引脚为：串行时钟输入（DCLK）、串行数据输出（DATA）、主动数据输入（ASDI）和低有效芯片选择，分别与 Cyclone 系列芯片上的 DCLK、DATA0、ASDO、nCSO 相连接。采用主动串行配置模式时，MSEL0、MSEL1 均接低电平，FPGA 的芯片使能信号 nCE 也必须接低电平 JTAG 配置模式中，配置芯片与 EP1C6T144C8 构成 JTAG 菊花链，配置芯片的 TDO 引脚连接到 FPGA 的 TDI 引脚，

FPGA的TDO引脚与JTAG下载口的TDO引脚相连接。这样的JTAG配置模式，可以方便的利用下载电缆对配置芯片和EP1C6T144C8进行配置。

2.2.6.2 复位部分

EP1C6T144C8有二种方式复位：一种是直接硬件复位，代码重新配置；第二种是软件复位，编写复位的程序；

硬件复位，硬件复位接到FPGA的nCONFIG引脚上，按下此键，FPGA的代码重新从eeprom（EPCS1）中配置，该引脚为低电平是代码重新开始配置。

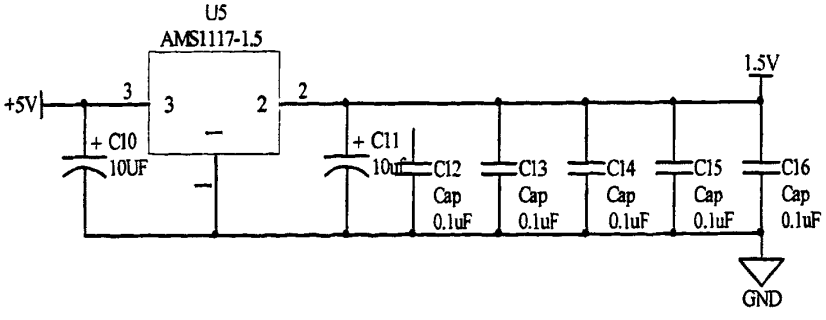
软件复位，软件复位是用户在编写代码的时候的那个reset引脚，只是按照代码来进行复位。

2.2.6.3 外部时钟

EP1C6T144C8一共有4个全局时钟，如图4.12所示：分别是FPGA的16、17、92、93号引脚，92号引脚接到外部晶振上，本电路板采用50M有源晶振，外部晶振提供时钟。在具体使用的过程中，当单独使用FPGA时，只能用外部晶振作为全局时钟，把引出来的那些IO引用户用来扩展。93号引脚与图像传感器输出时钟PCLK相连，当图像传感器有信号输出时，作为FPGA的主时钟或参考时钟。

2.2.7 电源模块设计

系统中使用了+5V、+3.3V、+1.5V三种电压的电源。图像传感器OV7620工作在+5V的电源电压；EP1C6T144C8使用了两种电源电压，一种是为片上外设供电的+3.3V电源电压，另一种是为CPU核心供电的+1.5V电源电压。其他芯片采用+3.3V电源电压。系统中+5V的电源电压可以从外部电源输入（包括Vbus线上的+5V电源），而+3.3V和+1.5V的电源电压则是利用电源芯片ASM1117将+5V的电源电压转换成+3.3V和+1.5V电源电压。转化电路图，如下图2.13所示。



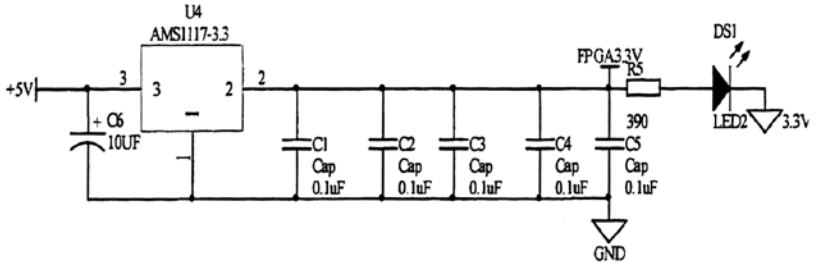


图 2.13 3.3V 和 1.5V 电源转化电路

Fig2.13 Power conversion circuit of 3V and 1.5V

2.3 PCB 电路板设计：

本设计中的电路板原图如图 2.14 所示：

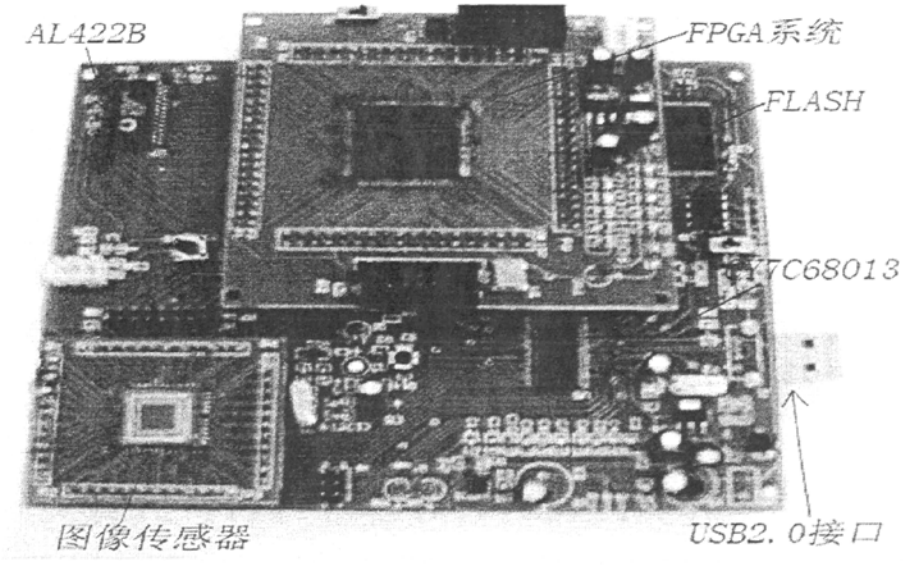


图 2.14 本系统的 PCB 图

Fig2.14 The diagram of the PCB

2.3.1 电源、地的设计

任何一块电路板都要将电源线和地线连接到芯片的电源和地引脚。由于电源线和地线的考虑不周全而引起的干扰，会使设备的性能下降，或导致设备不能正常工作。本系统 PCB 设计采用的是两层板，没有专门的电源层和地层，而系统中又有三种供电电压值，所以在 PCB 板的布局上的考虑是将使用相同供电电压的器件尽量放在一起。这样做的目的是为了防止不同电源线混在一起，增加布线的难度；同时也可以避免因电源线

走线过长，使电源线的阻抗增加，而造成的电源噪声增大以及电源驱动能力不足。另外，在整个 PCB 板的设计中，还专门对电源线和地线的宽度作了加宽处理，以减小线路的阻抗。一个 USB 设备有两种供电方式，一种是总线供电，即通过 USB 总线获得电源，还有一种供电方式是自供电，即 USB 设备有自己的供电电路，不使用 USB 总线的电源。由于 CY7C68013 和 OV7620 两块芯片都有模拟电源和模拟地引脚，所以在 PCB 设计时，要将模拟地与数字地分开布线。最后只是在 PCB 与外接电源和地的连接处，将数字地与模拟地一点连接。这样做的目的是为了防止模拟地与数字地之间相互干扰。另外，在系统的 PCB 设计中电源的走线避免了突然拐弯，在拐弯处采用了大于如 90 度的钝角。

2.3.2 去耦电容的设计

在直流电源回路中，增加去耦电容是必要的。这是因为在直流电源回路中，负载的变化会引起电源噪声。特别是在数字电路中，信号从一个状态转换到别一个状态时，在电源线上会有很大的尖峰电流，形成电源噪声。去耦电容的作用就是起电荷池的作用，可以抑制电源噪声。在系统中 PCB 板上去耦电容主要放在了以下几个位置：

(1) 在电源输入端跨接了一个 470μF 和一个 0.1μF 的电容。

(2) 在芯片的电源线和地线的输入端接入 0.1μF 去耦电容，以平滑电源的波动。这类去耦电容的放置原则采用就近原则，即电容放置的位置与引脚的距离尽量的近。

(3) 在 PCB 板上均布大小为 10μF 的电容。

2.3.3 电磁兼容性（EMC）设计

电磁兼容性是指电子设备在各种电磁环境中仍能够协调、有效地进行工作的能力。为了让电子设备能够抑制各种外来电磁干扰，同时又能减少其本身其它电子设备的干扰，所以要在进行 PCB 设计时要考虑设备的电磁兼容性。提高设备的电磁兼容性主要从以下几个方面来考虑：

(1) 采取合理的布线策略，由于长距离的平行走线会增加导线间的互感和分布电容，使用得两条线之间形成耦合而相互干扰，所以在系统设计中尽量避免长距离的平行走线，而是采取井字形网状布线结构。即一层采取横向布线的方法，而另一层采取纵向布线的方法，然后在交叉处用过孔连接。

(2) 选择适当的导线宽度以及导线长度设备工作时，PCB 板上信号线的电平状态一直都在变化，由此而产生的瞬变电流会在信号线上产生冲击干扰。信号线的电感量越大，则冲击干扰也越大。所以减少信号线的电感量可以有效地减少冲击干扰。信号线的电感

量与信号线长度成正比，与信号线宽度成反比。因此，在进行系统的 PCB 设计时，尽可能地减少了导线长度并且增加导线的宽度。

(3) 减少回路的数目，电流回路是电磁波的辐射源，要减小回路的电磁干扰，就要减小回路的数量以及回路的天线效应，不要过多地造成电流回路。但是回路在 PCB 设计中是不可避免的，所以在 PCB 设计有意识地减少了回路的形成。

3 FPGA的特点及开发流程

3.1 FPGA程序开发流程和工具介绍

3.1.1 Quartus II设计流程

Altera公司的Quartus II软件提供了可编程片上系统设计的一个综合开发环境，是进行EDA设计的基础工具。Quartus II集成开发环境包括以下内容：系统级设计，嵌入式软件开发，可编程逻辑器件（PLD）设计，综合，布局和布线，验证和仿真。

Quartus II设计软件提供完整的多平台设计环境，它可以轻易满足特定设计的需要，拥有FPGA和CPLD设计的所有阶段的方案。Quartus II软件的典型设计流程一般可分为设计输入、设计实现和器件编程三个设计步骤及相应的功能仿真、时序仿真和器件测试三个设计验证过程：

(1) 设计输入：

设计输入有多种方式，目前最常用的有电路图和硬件描述语言两种。对于简单的设计，可采用原理图或ABEL语言设计。对于复杂的设计，可使用原理图或硬件描述语言(如VHDL，Verilog语言等)，或两者混用，采用层次化设计方法，分模块、分层次地进行描述。设计输入软件在设计输入时会检查语法错误，产生网表文件，供设计实现和设计校验用。

(2) 设计实现：

设计实现是指从设计输入文件到位流文件的编译过程。在该过程中，编译软件自动地对设计文件进行综合、优化并针对所选中的器件进行映射、布局和布线，产生相应的位流数据文件。

(3) 器件编程：

器件的编程也称为器件的配置，就是将位流数据文件配置到相应的FPGA器件中。FPGA器件的配置方式分为两大类：主动配置方式和被动配置方式。主动配置方式是由GAL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；而被动配置是由外部机或控制器控制配置过程。

(4) 设计校验：

对应于功能仿真、时序仿真、器件测试组成的设计验证过程。功能仿真验证设计的逻辑功能。在设计输入过程中，对部分功能或整个设计均可进行功能仿真。时序仿真是

在设计实现后，针对器件的布局、布线方案进行的时延仿真，分析定时关系。器件测试是在器件编程后，通过实验或借助于测试工具，测试器件最终功能和性能指标。

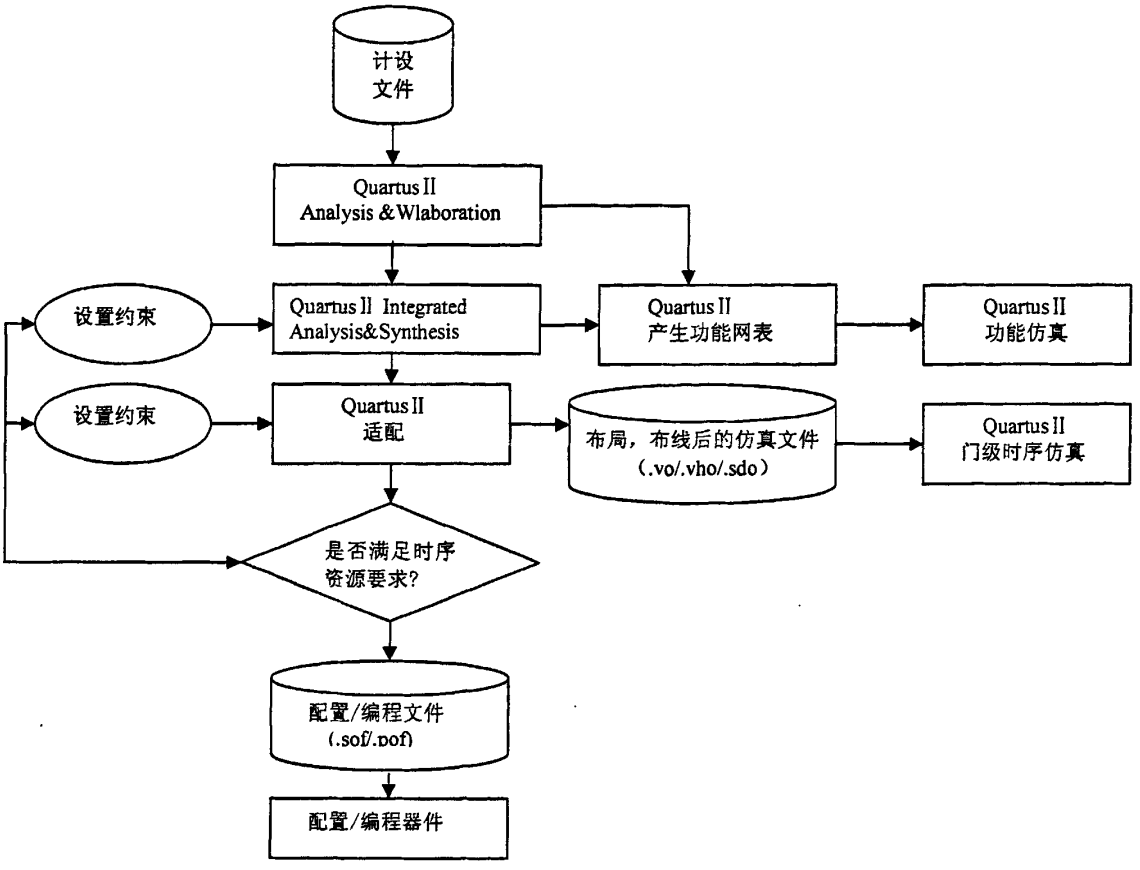


图 3.1 Quartus II 软件的典型设计流程图

Fig.3.1 Flow Diagram of Design with Quartus

3.1.2 硬件描述语言(VHDL)

VHDL 是电子系统设计制造各阶段的一种正式的注释语言。因为它既是一种机器能识别的语言，也是一种人可以阅读的语言，所以它支持硬件设计的开发、验证、综合和测试；支持硬件设计数据的通信；支持硬件的维护、修改与生成。主要用于描述数字系统的结构、行为、功能和接口，非常适用于可编程逻辑芯片的应用设计。与其它的 HDL 相比，VHDL 具有更强的行为描述能力，从而决定了它成为系统设计领域最佳的硬件描述语言。强大的行为描述能力是避开具体的器件结具体的工艺技术和硬件结构无关，所

以VHDL设计程序的硬件实现目标器件有广阔的选择范围，其中包括各种系列的CPLD、FPGA及各种门阵列器件。

由于VHDL具有类属描述语句和子程序调用等功能，对于完成的设计，在不改变源程序的条件下，只需改变类属参量或函数，就能轻易地改变设计的规模和结构。

VHDL基本设计流程可用图3.2表示。

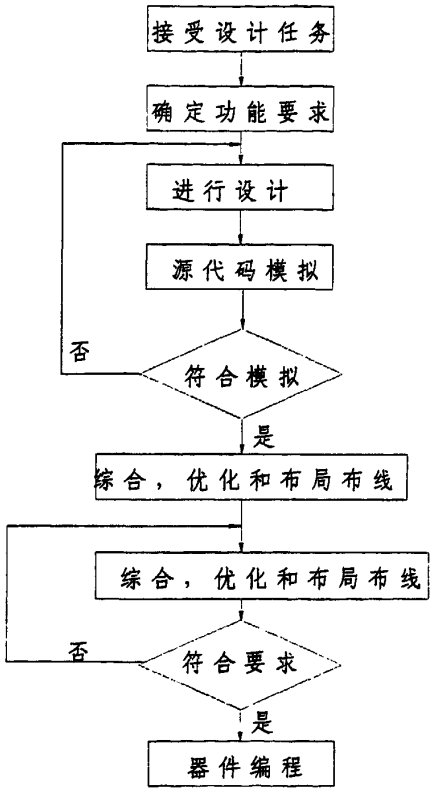


图 3.2 VHDL设计流程图

Fig.3.2 Flow Diagram of Designing with VHDL

3.2 FPGA实现的模块框架

3.2.1 FPGA系统功能框图和整体设计

FPGA芯片作为中央控制器控制整个视频信号处理过程，可以将FPGA系统划

分为以下几个模块：图像采集模块、图像缓存模块，图像存储模块，接口模块和 FPGA 配置电路模块。FPGA 系统内部各模块连接框图,如图 3.3 所示。

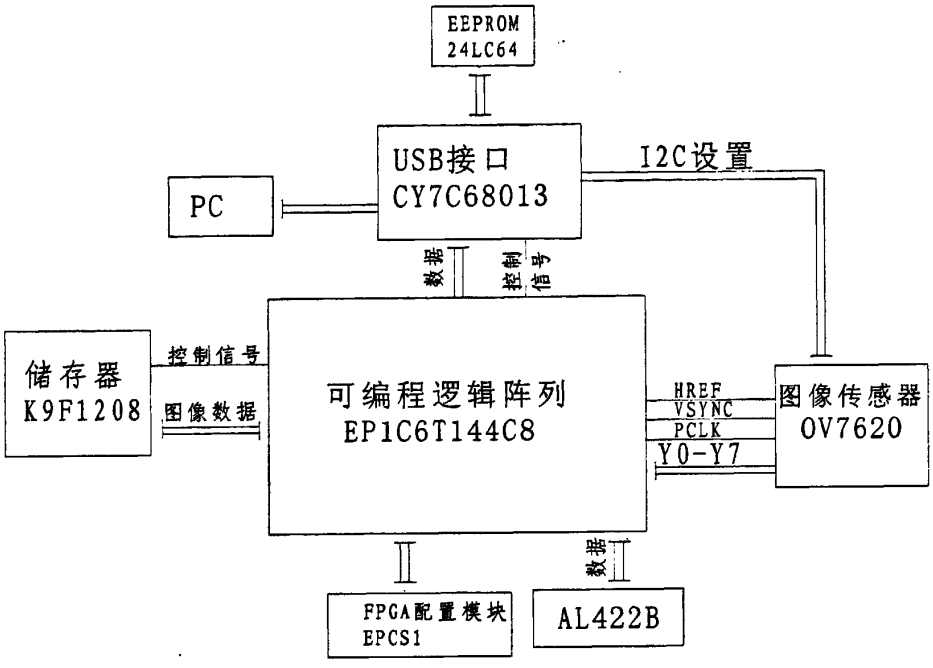


图 3.3 FPGA 系统内部各模块连接整体框图

Fig.3.3 Whole diagram of FPGA modules

系统内各模块功能简要描述已经在第三章简述，这里不再介绍，在 FPGA 实现的模块中，FLASH 的存储器 K9F1208 是最重要的难点，下面的章节将主要对 K9F1208 的编程做简要的介绍。

3.2.2 FPGA 对 FLASH 芯片 K9F1208 的读写控制

3.2.2.1 主状态机

FLASH 控制器的主要功能是响应输入命令，并根据命令产生的相应时序来实现对 FLASH 的操作。控制器可完成擦除操作(即对 FLASH 进行格式化)、写操作(响应一次命令写人一页数据)、读操作(响应一次命令读出一页数据)。

一般情况下，命令的优先级为：擦除操作高于写操作，写操作高于读操作。FLASH 控制器主要由状态机完成，其主状态机的转换示意图，如图 3.4 所示。

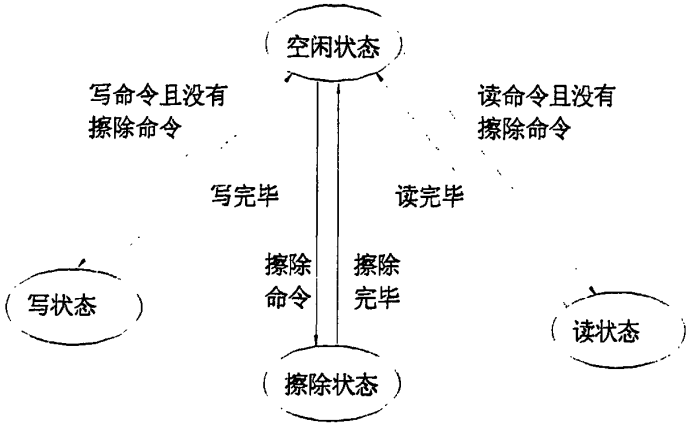


图 3.4 主状态机转化示意图

Fig.3.4 Translating diagramof the main state machine

系统复位后，状态机将进入空闲状态，等待命令输人。而在接收到命令后，主状态机会根据优先级进入相应的状态，同时启动对应的子状态机。子状态机执行完毕命令以后，就会发出执行完毕信号，主状态机收到该信号后，又进人空闲状态，以等待下一次命令输入。

当状态机转到相应的状态(非空闲状态)以后，系统会启动对应的子状态机。

3.2.2.2 擦除状态机

擦除操作是以块为单位进行的，执行一次擦除命令要将所有 2048 个块擦除一遍。擦除一个块的时序如图 3.5 所示。

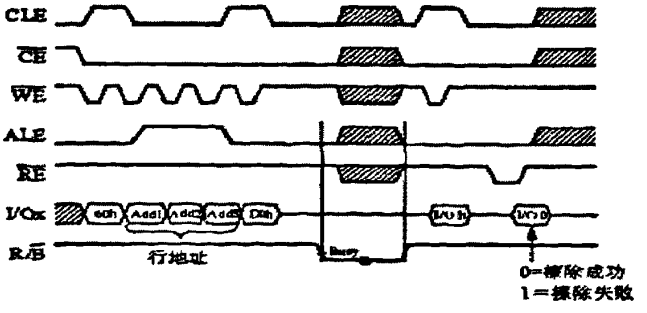


图 3.5 擦除一个块的时序图

Fig.3.5 Timing chart of erasing a block

从图 5.5 所示的时序可以看出，对于一个擦除操作，其 FLASH 的操作过程是，首先发送前“半个”命令(60H)，然后再分三个周期从低位到高位发送行地址。由于擦除是以块为单位进行的，而芯片只有 2k 个块，所以行地址只有 A28-A18 有效，其余位为 0，最后再发送后“半个”命令(D0H)。发送操作完成以后，FLASH 会响应该命令以擦除相应的块。此时，FLASH 会发出“忙”信号(rb 信号为低)，并等待 FLASH 将命令执行完毕后(rb 信号为高)发送读状态命令，同时查看擦除是否成功。根据上述时序设计的擦除操作状态机，如图 3.6 所示。

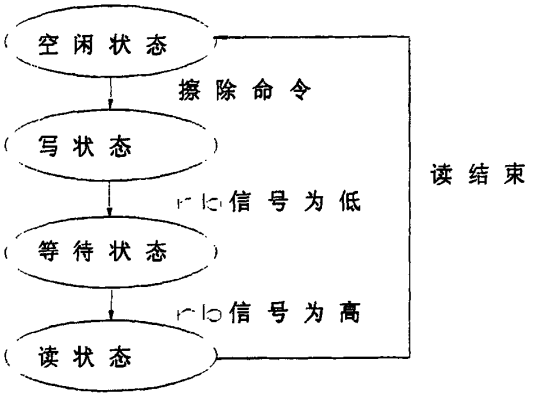


图 3.6 擦除状态机

Fig.3.6 Erasing state machine

擦除状态机的默认状态为空闲状态，在该状态下，状态机检测主状态机是否进入擦除状态，一旦进入擦除状态，子状态机就转入写状态。在写状态下，控制电路将向 FLASH 发写命令和块地址，发送完毕后检测 FLASH 是否进入忙状态(rb 信号为低)，当检测到 rb 信号为低时，说明 FLASH 已经执行该命令，状态机进入等待状态以等待 FLASH 将命令执行完毕，执行完毕后，状态机进入读状态并读出 FLASH 状态字，同时查看擦除是否成功。通过这样的状态循环和不断变换块地址，擦除状态机便可将整个 FLASH 擦除完毕，在所有块擦除完毕后，系统将发出擦除完毕信号，以指示主状态机退出擦除状态。

3.2.2.3 写状态机

写操作是以页为单位进行的，写一个页的时序, 如图 3.7 所示。

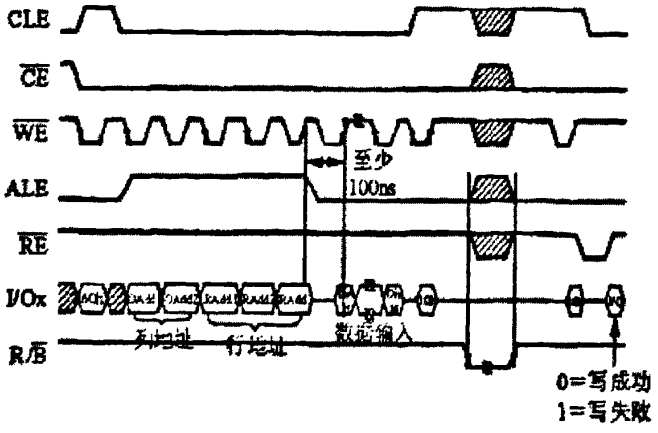


图 3.7 写一页的时序图

Fig.3.7 Timing chart of writing a page

对于一个写操作，其 FLASH 的操作过程是：首先发送前“半个”命令(80H)，再分五个周期从低位到高位发送页地址(先发列地址后发行地址)。由于每次写入一页，所以列地址 A11—A0 为 0，然后等待一段时间(至少 100ns)再将一个页的数据发出，之后再发送后“半个”命令(10H)。发送操作完成以后，FLASH 会响应该命令并写相应的页，此时 FLASH 会发出“忙”，信号(rb 信号为低)，并等待 FLASH 将命令执行完毕后(rb 信号为高)发送读状态命令，最后查看写操作是否成功。根据上述时序设计的写操作状态机程序，如图 3.8 所示。

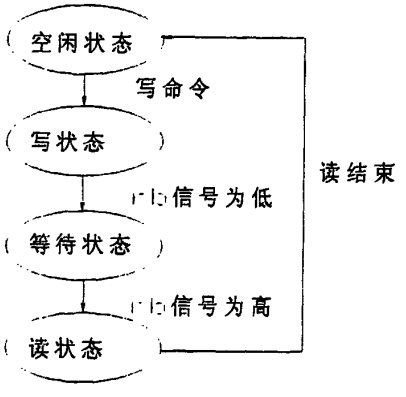


图 3.8 写状态机

Fig.3.8 Writing state machine

写状态机的默认状态为空闲状态，在该状态下，状态机将不断地检测主状态机是否进入写状态，一旦进入写状态，子状态机也随之转入写状态。在写状态下，控制电路首先向 FLASH 发页地址，并在等待至少 100ns 后发送 2k 字节数据，发送完毕后，将检测 FLASH 是否进人忙状态(rb 信号为低)，当检测到 rb 信号为低时，说明 FLASH 已经执行该命令，同时状态机进人等待状态，以等待 FLASH 将命令执行完毕。执行完毕后，状态机再进人读状态并读出 FLASH 状态字，同时查看写人是否成功。读结束后将发出写完毕信号，以指示主状态机退出写状态。

3.2.2.4 读状态机

读操作也是以页为单位进行的，读一个页的时序如图 3.9 所示。

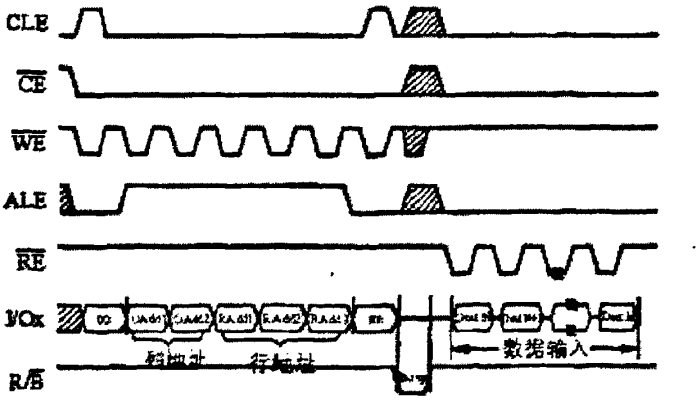


图 3.9 读一页的时序图

Fig.3.9 Timing chart of reading a page

对于一个读操作，其 FLASH 的操作过程是：首先发送前“半个”命令(00H)再分五个周期从低位到高位发送页地址，最后发送后“半个”命令(30H)。发送操作完成以后，FLASH 会响应该命令并读相应的页，此时 FLASH 会发出“忙”信号(rb 信号为低)，同时等待 FLASH 将命令执行完毕后(rb 信号为高)，再使能 FLASH 的读信号，以将数据依次读出。根据上述时序设计的读状态机程序，如图 3.10 所示。

读状态机的默认状态亦为空闲状态，在该状态下，状态机将检测主状态机是否进人读状态，一旦进人读状态，子状态机也转人读状态。在读状态下，控制电路首先向 FLASH 发读命令和页地址，发送完毕后再检测 FLASH 是否进人忙状态(rb 信号为低)，当检测

到rb信号为低时，说明FLASH已经执行该命令，状态机进入等待状态同时等待FLASH将命令执行完毕，执行完毕后，状态机将进入读状态读出相应的2k字节数据。同时在读结束后发出读完毕信号，以指示主状态机退出读状态。

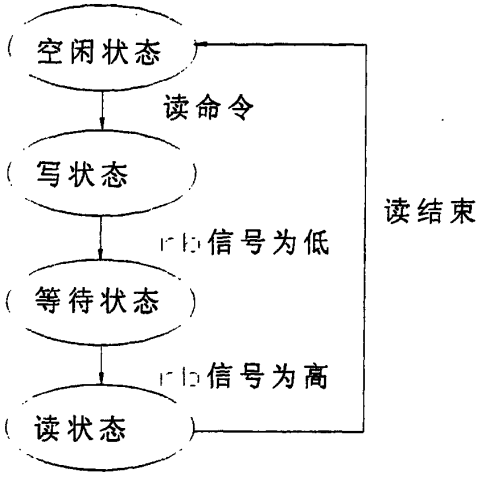


图 3.10 读状态机

Fig.3.10 Reading state machine

3.3 FPGA设计中的常见问题及注意事项

在FPGA的设计过程中，为了提高系统中FPGA运行性能，需要注意一些问题，比如避免毛刺的产生，异步电路之间的通讯以及如何实现三态等。

任何组合电路、反馈电路和计数器都可能是潜在的毛刺发生信号。毛刺并不是对所有的输入都有危害，如触发器的D输入端，只要毛刺不出现在时钟的上升沿并满足数据的建立保持时间，就不会对系统有危害。而当毛刺信号成为系统的启动信号、控制信号、握手信号、触发器的清零信号(CLEAR)、预置信号(PRESET)、时钟输入信号(CLK)或锁存器的输入信号就会产生逻辑错误。任何一点毛刺都可能使系统出错，因此消除毛刺信号是FPGA设计中的一个重要问题。毛刺问题在电路连线上是找不出原因的，只能从逻辑设计上采取措施加以解决。消除毛刺的一般方法有以下几种：

(1) 采用冗余项消除毛刺；

(2) 取样法，在信号稳定后加入取样脉冲，只有取样脉冲作用期间输出的信号才有效。这样可以避免毛刺影响输出波形；

(3) 吸收法，增加输出滤波，在输出端加小电容可以滤出毛刺，该方法在电路模块终端才能使用；

(4) 延迟法，毛刺延迟造成的，可通过找出产生延迟的支路，加上毛刺宽度的延迟可以消除毛刺；

(5) 锁存办法，在输出端加 D 触发器可以消除毛刺。

其他的注意事项还包括在状态机设计中，输入都要用同步信号，输出采用寄存器输出，设计不能产生死锁状态；采用寄存器和触发器设计电路，不要用锁存器，锁存器对信号的毛刺非常敏感；充分利用 FPGA 的全局资源，如全局时钟，全局清零，全局置位等，可减少电路设计中的毛刺，并大大提高所设计的电路的工作性能。

4 USB2.0固件程序设计

4.1 固件程序的作用

所有基于微控制器及其外围电路的功能设备的正常工作都离不开固件的参与，固件程序是基于微控制器设备运行的核心。固件设计的目标就是控制硬件来完成预期的设备功能。没有固件的参与和控制，硬件设备只是芯片的简单堆砌，不可能完成预期的功能。USB设备的正常工作同样也离不开固件的参与。虽然USB的控制芯片已经相当智能化了，但是也不可能完成所有的工作，所以还需要编写固件程序来辅助硬件完成USB的通信任务。CY7C68013芯片的固件程序负责处理PC机发来的各种USB设备请求，并负责控制CY7C68013与外围电路进行数据传输。

主要包括以下五项工作：

(1) 初始化工作，主要是给一些特殊功能寄存器的赋值以使设备具有某种属性或处于某种状态；

(2) 对设备进行重新列举（ReNumeration），主要的工作有：模拟设备的断开与重新连接、对接收到的设置包进行分析判断、对主机的设备请求作出适当的响应，完成主机对设备的配置任务；

(3) 响应中断，并对中断作相应的处理。USB接口在设备列举、响应主机标准请求和厂商请求、数据传送等动作过程中都会向微控制器申请中断，因此微控制器必须正确处理这些中断才能保证USB传输的正常进行；

(4) 数据的接收与发送。USB设备的主要任务就是进行数据的接收和发送，因此CY7C68013的固件程序必须要对接收和发送数据的端点进行正确的配置及控制，才能正确地接收和发送数据。

(5) 外围电路的控制。

如果CY7C68013接收数据，在收到数据后需要进行相应的操作，这就要求CY7C68013对其外围电路进行控制。如果CY7C68013发送数据，那么它也要控制外围电路，从外围电路获得数据，并将数据发送给主机，所以固件程序中包含对外围电路进行控制的代码也是必要的。不同的USB设备实现的功能不同，固件代码的复杂程度也有所不同。但是上面这五部分的功能，是每一台USB设备的固件程序必须具有的。Cypress公司为方便用户的开发，向用户提供了FX2系列USB芯片的Firmware库。Firmware库向用户提供了常量、描述符、数据结构、宏、函数、中断跳转表等资源，可

以简化固件程序的开发。用户只需要将 fx2.h 和 fx2regs.h 包含到项目中即可，另外还要把 Ezusb.lib 和 USBJmpTb.obj 添加到项目中，就可以对这些资源进行使用。

4.2 寄存器、缓冲区以及描述符的定义

在进行系统的 USB 固件开发时使用了 Cypress 公司提供的 Firmware 库对寄存器、缓冲区以及描述符的定义。Firmware 库中寄存器和缓冲区的定义放在 fx2regs.h 头文件中，fx2.h 头文件中存放的是描述符定义。

CY7C68013 内有非常丰富的寄存器和缓冲区，开发人员可以非常方便地通过对这一系列寄存器和缓冲区的控制来实现对 CY7C68013 的外部逻辑控制。这些寄存器种类包括特殊功能寄存器（SFR）、通过用配置寄存器、中断寄存器、端点配置寄存器、输入/输出寄存器、USB 控制寄存器、端点工作寄存器、GPIF 波形描述符表寄存器、GPIF/FIFO 寄存器、端点缓冲寄存器。

CY7C68013 的特殊功能寄存器（SFR）的寻址范围为 0x80~0xFF，其中有的是标准 8051 寄存器，有的是非标准 8051 寄存器。SFR 是可以直接寻址的，因此固件程序可以很快地对这些寄存器进行访问。对于 SFR 寄存器，使用专门的 C 语言指令 sfr 进行定义。除 SFR 之外的其它寄存器和缓冲区的存储地址在 0xE600~0xFFFF 范围之内，8051 内核不能对些寄存器和缓冲区进行直接访问，必须把它们作为外部储存器，使用外部储存器访问指令进行访问。这部分寄存器和缓冲区非常多，可以对 CY7C68013 绝大部分功能进行控制。

由于 8051 内核不能直接访问除 SFR 寄存器以外的其它的寄存器和缓冲区，所以这些寄存器和缓冲区的定义比 SFR 寄存器定义要稍微复杂一些。

8051 内核是将寄存器和缓冲区作为外部存储器来访问的，所以要使用 xdata 指令。在 xdata 后加上 volatile 关键字是为了防止编译器对该变量进行优化。

为了避免寄存器和缓冲区的重复定义，在定义寄存器和缓冲区的头文件的开始处使用了常量 ALLOCATE_EXTERN 指明如果在这之前未定义寄存器和缓冲区，那么需要在该文件中定义。如果在这之前已经定义了寄存器和缓冲区，那么就使用关键字 extern 声明，并使用“；/##/”代替“_AT_”（“_AT_”的功能是给变量指定一个绝对地址）以屏蔽掉“_AT_”后面的地址值。在第三章中，我们对 USB 描述符作了简单的介绍。在进行固件程序开发时，就要自己根据 USB 设备的情况来对 USB 描述符进行定义。可以将 USB 设备的所有描述符放在一个文件内。为方便寻址，描述符要按一定的顺序进行存放。具体的存放顺序如图 3.1 所示：

设备描述符

配置1描述符
接口1描述符
端点1描述符
端点2描述符
.....
接口2描述符
端点1描述符
.....
配置2描述符
.....
字符串描述符1
字符串描述符2
.....
类描述符1
.....
NULL描述符

图4.1 USB描述符存放顺序
Fig.4.1 Storing sequence of USB Descriptor

4.3 USB中断

CY7C68013的增强型8051内核对标准8051的中断系统进行了扩展，共新增了8个中断：INT2~INT6、Resume、Tx1&Rx1、Timer2。INT2负责处理32种USB中断，CY7C68013使用了自动向量跳转的方法来响应这32种USB中断。

当寄存器中USBBAV中的AVEN位为1（即允许自动向量跳转），CY7C68013的内核将用寄存器AVEC的值代替地址0x0045处的值，以便跳转到不同的中断服务子程序中去，如图4.2所示。

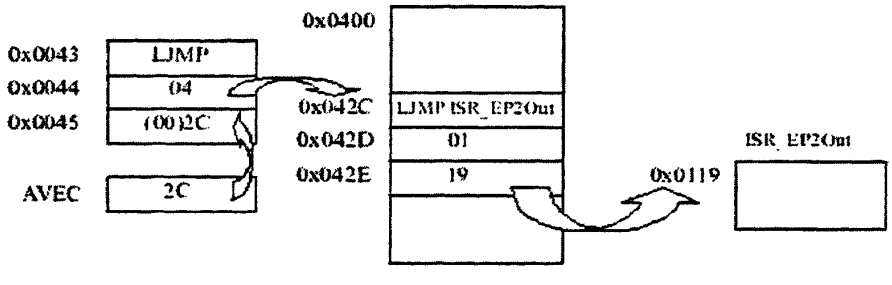


图4.2 中断的自动向量跳转举例
Fig.4.2 Example of USB autovector Mechanism

Cypress 公司的 Firmware 库中的 USBJmpTb.obj 存放了就是中断自动向量跳转表，用户可以直接使用这个向量表。有了跳转表的帮助，就可以将中断服务程序放在程序存储器的任何地方。固件的中断服务程序的编写也非常简单，其所要做的工作包括设置一个响应标志位，清除 USB 中断（INT2），最后清除相应中断请求标志位。

4.4 固件程序的设计思路及固件架构

本设计中系统的功能是将采集到的图像数据送到计算机，可见，USB 固件程序设计主要完成的工作有：使用 SLAVE FIFO 方式从 FIFO 的数据存储器读取数据、传送数据到计算机。要控制图像传感器 OV7620 的工作方式，则要通过 CY7C68013 的 I2C 接口来进行，这就要求 CY7C68013 对其内部的 I2C 寄存器进行操作。要实现与计算机之间的数据传送，就要对 CY7C68013 的端点以及端点缓冲区进行配置。在 USB 固件的开发过程中，使用固件架构来进行固件的开发。

USB 固件架构是按照结构化程序设计方法，将整个程序按功能划分成几个功能模块构成的。其主要工作是完成初始化、USB 标准设备请求的处理和 USB 挂起电源管理服务。用户在这个固件架构的基本上只需要再提供一个 USB 描述符表，配置一些相关的寄存器，添加所使用的端点接收和发送数据的通信代码，以及控制外围电路的程序代码就可以完成 USB 设备的固件开发工作。

CY7C68013 固件程序架构会按顺序完成如下 4 个步骤的操作，如图 4.3 所示：

(1) 设置所有的内部状态变量，即对这些变量进行初始化；

(2) 调用用户的初始化函数 TD_Init()，对用户自定义的变量进行初始化。

为了控制 OV7620 的工作方式，要对 I2C 接口控制寄存器进行操作；因为要使用 SLAVE FIFO 模式从 FIFO 读取数据，所以还要对 SLAVE FIFO 进行初始化；在完成 TD_Init()函数后，固件程序要设置 USB 接口为未配置状态，重新分配描述符表，使能要使用到的中断；

(3) 然后固件程序会在 1 秒钟间隔内开始重新列举（ReNumerate）设备，直到端点 0 收到 SETUP 令牌包为止；

(4) 一旦端点 0 收到一个设置包，固件架构就会启动任务分配器。

图 4.3 中，虚框内的部分为任务分配器所做的工作。

从图 4.3 中可以看出，任务分配器的工作主要由三部分组成：

(1) 调用用户函数 TD_Poll()。该函数会被反复调用，不断地从 FIFO 读取数据；

(2) 判断是否有等待处理的标准设备请求。如果有，固件代码的任务分配器会分析收到的这个请求，并作出响应；

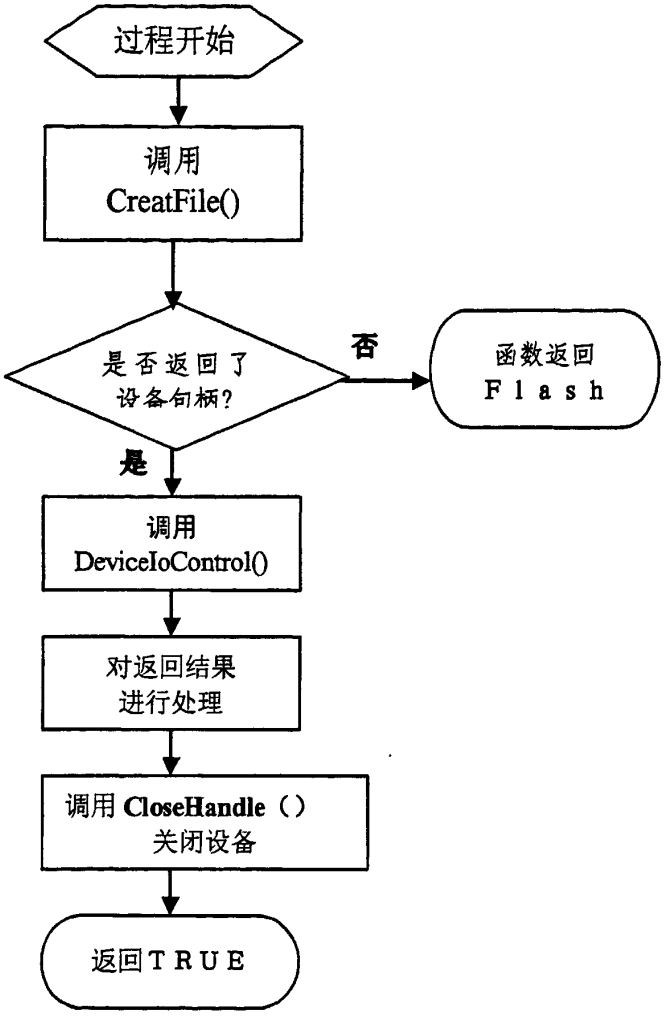


图4.3 固件架构流程图
Fig.4.3 Flow diagram of firmware framework

(3) 判断 USB 是否已经设置了闲置标志。如果已经设置了，则调用 TD_Suspend() 函数。执行 TD_Suspend()函数后，如果返回为真，则挂起微控制器。当微控制器挂起后，任务分配器就只做一件事情，即检测是否有恢复事件发生。当任务分配器检测到有一个恢复事件发生，便会调用恢复函数 TD_Resume()，使微处理退出挂起状态。

整个构架构成了固件程序的主函数。系统中 CY7C68013 进行图像数据的读取、数据的传送、I2C 接口的数据通信、USB 中断的处理都是通过固件架构来完成的。

4.5 主程序

主程序是整个固件程序的入口，是固件构架功能的具体实现。主程序开始执行后，首先是初始化变量，然后是调用初始化函数 TD_Init()来初始化设备，在 TD_Init()函数中包含了控制 I2C 接口的代码，这部分代码的作用是控制 OV7620 的工作方式。在 TD_Init()函数返回后，程序要使能中断、重新列举，最后进入任务分配器。任务分配器是一个无限循环，称之为主循环。

CY7C68013 的 8051 内核运行固件程序后，其绝大部分时间都是在执行主循环的代码。主循环的代码如下：

```
While (TRUE)    {
          TD_Poll ();
          If (GotSUD)
       {  SetupCommand ();
         GotSUD = FALSE;          // Clear SETUP flag
       }
          If (Sleep)
       {  If (TD_Suspend ())
          { Sleep = FALSE;        // Clear the "go to sleep" flag.  Do it here to
                                     prevent any race condition between wakeup and the next sleep.
            Do
            {   EZUSB_ Susp ();  // Place processor in idle mode.
            }
    While (! Rwuen && EZUSB_EXT WAKEUP ());
    EZUSB_Resume ();   // If source is the Wakeup# pin，signal the host to Resume.
    TD_Resume ();
          }
       }
       }
 }
```

代码的功能就是图 4.3 中的虚线框所描述的功能。

如果 Sleep 为真，则进入一个循环以等待唤醒事件的发生。在主循环中，会不断地调用 TD_Poll()函数，所以将读取 FIFO 内部存储器的数据的代码放在该函数里面。

如果主循环检测到 SETUP 令牌包，它将调用 SetupCommand()函数对 Setup 包进行处理。当设备收到 SETUP 时，USB 核会自动将设置数据放入 8 字节的 SETUPDAT 缓

冲区中（起始地址为 0xE6B8）。接着，固件只要从该缓冲区中读出设置数据，进行分析就可以判断是什么样的请求，然后再根据请求编写相应的代码，就能完成对请求的响应了。例如，如果设备收到 Get_Descriptor 请求，固件程序要做的事情有：首先从 SETUPDAT 读取数据，判断出是主机发出 Get_Descriptor 请求，然后固件程序再从 SETUPDAT 中读取一个字节，再判断主机要读取的是哪一个描述符，最后将该描述符的首地址放入 SUDPTRH 和 SUDPTRL 中，固件的工作就到此为止。接下来 USB 核就会处理多个数据包的 IN 传输，固件就不用再干预这个过程了。

4.6 初始化函数

固件程序中，初始化函数 TD_Init()在固件运行开始阶段调用，它在整个固件的运行过程中只被调用一次。TD_Init()函数负责对CY7C68013进行初始化以及选择OV7620 的工作方式。它首先选择 CY7C68013 的时钟频率，然后对 CY7C6803 中的端点及其缓冲区进行配置，最后选择 OV7620 的工作方式以。

TD_Init()函数的代码如下：

```
Void TD_Init (void)              // Called once at startup
{
    REVCTL =0x03;          // set the CPU clock to 48MHz
   CPUCS =0x10;            // ((CPUCS & ~bmCLKSPD) | bmCLKSPD1) ;
   IFCONFIG |=0x43;
    EP1OUTCFG = 0x7F;   //0xa0;
   EP1INCFG =0x7F;         //0xa0;
   EP2CFG = 0xA0;          //0xE0;   DIR:   in BUFFER:   4×512
   EP4CFG = 0x7F;
   EP6CFG = 0xE0;
   EP8CFG = 0x7F;
   FIFORESET = 0x80;   // activate NAK-ALL to avoid race conditions
   FIFORESET = 0x02;   // reset,   FIFO 2
   FIFORESET = 0x04;   // reset,   FIFO 4
   FIFORESET = 0x06;   // reset,   FIFO 6
   FIFORESET = 0x08;   // reset,   FIFO 8
   FIFORESET = 0x00;            // deactivate NAK-ALL
   EP2FIFOCFG = 0x11;
```

```
    EP4FIFOCFG = 0x10;
    EP6FIFOCFG = 0x08;
    EP8FIFOCFG = 0x10;
    EP6AUTOINLENH =0x02;
    EP6AUTOINLENL =0x00;     //SET autoin byte is 512 byte,
    FIFOPINPOLAR =0x14;      // PKTEND ACTIVE LOW,  SLWR active low //
    PORTACFG = 0x02;         // pa7 as SLCS & ENABLE int1
    AUTOPTRSETUP = 0x01;
     PINFLAGSAB = 0x00;      //定义 FLAGA 为可编程标志
    PINFLAGSCD = 0x00; // FLAGC as empty flag, as pointed to by FIFOADR[1: 0]
}
```

4.7 端点配置及数据传输

CY7C68013 芯片共提供了 7 个端点，分别是：EP0、EP1OUT、EP1IN、EP2、EP4、EP6 以及 EP8[17]。

其中端点 0 为默认的控制端点，是唯一的一个双向端点，其只有一个缓冲区，最大包长度为 64 字节，用于 USB 设备本身的配置。端点 1 只能被配置为中断或块传输端点，且只支持最大长度为 64 字节的数据包，也只有一个缓冲区。端点 2、4、6、8 用于进行大数据量的传输，它们既可以配置为 IN 端点也可以配置 OUT 端点，并且支持全部四种传输方式。端点 4、8 具有 512 字节的双缓冲区。端点 2、6 的缓冲区长度可以被配置为 512 字节，也可以被配置为 1024 字节，缓冲区的深度可以是 2 级、3 级或 4 级。

因为本系统用于进行数据采集，数据的主要流向是从设备到主机，且数据非常大，所以根据这种情况，我们选择了端点 0、端点 6 这两个端点来进行数据传输。

端点 0 默认端点，无需进行配置，它被默认配置为为双向、控制端点。端点 6 被配置为 IN 块传输端点，为其分配 2048 字节的四重缓冲区。

为了最大限度的增大 USB2.0 的带宽，端点 6 的 EP6FIFOCFG 的 AUTOIN 位设置为 1（自动传输模式）。因为保证计算机在接收到数据后每一行数据能够对齐，所以要对 EP6AUTOINLEN 寄存器进行设置：将 EP6AUTOINLEN 的值设置为 640。这样 EP6 的缓冲区每将入 640 字节的数据后，就会自动将数据打包，在接收到 IN 请求后自动将包发送到计算机。

由于 FIFO 内数据是按字存放的，即每个数据为 16 位，所以 USB 设备每发送四个数据包就可组成一行图像数据。

4.8 I2C接口的控制

CY7C68013的I2C接口控制器使用了SCL和SDA两个引脚[17],可以完成两种功能：一是作为通用接口与I2C外设通信；二是在启动时从外部串行EEPROM下载固件程序。对于本系统，我们将I2C接口作为通用接口来与OV7620相连。CY7C68013的有三个寄存器来控制I2C接口的工作，这三个寄存器是I2CS、I2DAT、I2CTL。其中，I2CS为I2C总线控制及状态寄存器；I2DAT为I2C总线数据寄存器；I2CTL为I2C总线工作模式寄存器。在固件设计中专门编写了函数I2Cwrite(addr，data)来负责对I2C接口进行控制。函数对OV7620的寄存器的配置操作步骤如下：

(1) 设置I2CS的START位为1。

(2) 将外设地址和传送方向写到I2DAT中。

(3) 等待I2CS的DONE位置1。如果BERR=1或ACK=0，则转至第9步。4．将OV7620的寄存器子地址放入到I2DAT中。

(4) 等待I2CS的DONE位置1。如果BERR=1或ACK=0，则转至第9步。

(5) 将要写到OV7620寄存器的数据放入到I2DAT中。

(6) 等待I2CS的DONE位置1。如果BERR=1或ACK=0，则转至第9步。

(7) 重复2~7的操作，直到将需要传送的配置全部送出。

(8) 将STOP位置1，停止I2C接口工作。

OV7620寄存器设置如下：

```
BYTE xdata COMB [] = {0x13，  0x31}；  // 8bitBT656，  single frame
BYTE xdata RST []  = {0x12，  0x80}；
BYTE xdata COMC [] = {0x14，  0x34}；
BYTE xdata COMH [] = {0x28，  0x20}；  //progressive output
BYTE xdata COMA [] = {0x12，  0x24}；  // color bar output
```

5 USB 驱动程序及应用程序设计

5.1 WDM 概述

WDM(Windows Driver Mode，Windows 驱动程序模型)是 Microsoft 公司推出的一种驱动程序模型。WDM 减少了所需驱动程序的数量，降低了驱动程序开发的复杂性。它属于操作系统内核模式，其驱动程序由运行于内核模式的系统级代码组成。WDM 采用灵活的分层结构进行组织，用户应用程序不是直接与物理设备进行通信，而是要经过几个不同层次的驱动程序才能与物理设备进行通信。每一层次对应于一个驱动程序，在不同层次的驱动程序之间可以相互调用。WDM 引入设备对象的概念来描述一个设备。WDM 驱动程序直接操作的不是硬件本身，而是设备对象。设备对象可以分为三类：物理设备对象(PDO：Physical Device Object)、功能设备对象(FDO：Function Device Object)和过滤设备对象（filter DO：Filter Device Object）。一个物理设备可能包含多个设备对象，其中 PDO 和 FDO 有且只有一个，filter DO 可以有多个。当应用程序发与请求时，系统会为每一个请求打包并形成一个 IRP 结构，将其发送至驱动程序，并通过识别 IRP 中设备对象来区分该请求是发送给哪一个设备的。由于 WDM 动程序是操作系统内核模式的驱动程序，它可以执行任何有效的 CPU 指令，Windows 没有办法来防止 WDM 驱动程序对系统中其它进程的破坏，一不小心就会造成系统的崩溃，因此其开发难度和复杂程度比 WIN32 应用程序要大。进行 WDM 驱动程序的开发可以选用的工具软件很多，常用的有两种：一种是选用 Microsoft 提供的 DDK(Driver Develop Kit)来进行，另一种是使用 Numega 公司推出的 DriverStudio 开发工具包来开发。本系统在进行驱动程序开发时使用了 Microsoft 公司提供的 DDK 来进行开发，因为使用 DDK 进行开发，开发过程虽然相对复杂，但是开发过程更加直接。使用其它软件开发时，会对 DDK 进行封装，使用 DDK 直接进行开发可以省去对封装了解，只要对 DDK 进行了解就可以了。在编译软件的选择上，我选择了 Microsoft 公司的 Visual C++。VC++的具有完整的集成开发环境和强大的功能，可以方便程序的开发和调试，并且 VC＋＋对 DDK 提供很好了的支持。

5.2 WDM 驱动程序的分层

WDM 驱动程序可以分为 3 类，即总线驱动程序、功能驱动程序和过滤驱动程序。WDM 驱动程序的结构也被称为 WDM 驱动程序栈[1]。图 5.1 为一种可能的驱动程序栈，它表明了各类驱动程序之间的层次关系。

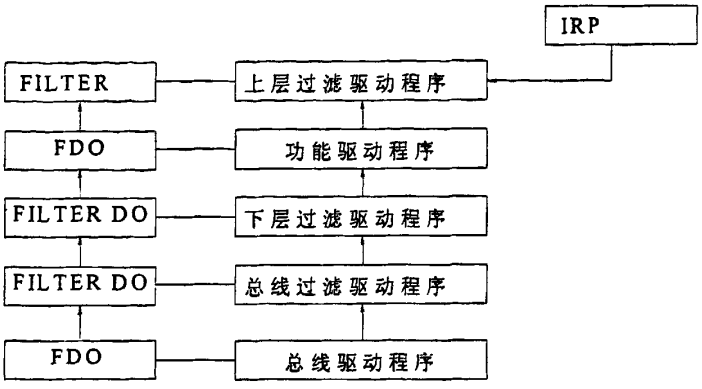


图5.1 一种可能的WDM 驱动程序栈
Fig.5.1 A possible WDM driver stack

总线驱动程序负责对 I/O 总线进行驱动，处理总线上的所有事务。它是一个必不可少的驱动程序。总线驱动程序一般由 Microsoft 提供，在 Windows 安装过程中已经将通用总线的总线驱动程序一起安装到计算机上了。功能驱动程序用于驱动一个单独的设备，设备的主要功能是由它来实现的。功能驱动程序位于总线驱动程序之上，是设备必不可少的主要驱动程序。我们所说的设备驱动程序一般都是指的功能驱动程序，一般情况下功能驱动程序由设备开发人员自行开发。过滤驱动程序是可选的，一个设备可以没有过滤驱动程序，也可以有多个过滤驱动程序。其用于过滤发向设备、设备类或总线的 I/O 请求。它又可分为 3 类：总线过滤驱动程序、下层过滤驱动程序和上层过滤驱动程序。

应用程序请求被打包成一个 IRP 结构从 WDM 驱动程序栈的顶部进入，由处于上层的驱动程序先处理这些请求，再由上层的驱动程序逐层向下传递。

WDM 驱动程序分层形成驱动程序栈的同时，每个驱动程序所建立的设备对象也就自然地形成了一定的层次关系，即形成了一个设备对象栈。总线驱动程序对应一个 PDO，每一个过滤驱动程序对应一个 filter DO，功能驱动程序对应一 FDO。对于设备对象栈来说，应用发出的 I/O 请求也是由设备对象栈的顶部进入，再逐渐向下层的设备对象传递。

5.3 I/O 请求包(IRP)

Windows2000/XP 操作系统中 I/O 请求一般是包驱动的，I/O 请求按照一种特殊的数据结构被打包，并在驱动程序栈中进行传递和跟踪，这种特殊的数据结构形成的包被称为 I/O 请求包（IRP）。IRP 是 WDM 内核模式对象的一种，它具有一组负责进行操作的 I/O 管理例程。

当 I/O 管理器接收到 I/O 请求时，首先对这个 I/O 请求进行打包，形成一个 IRP，并对这个 IRP 进行初始化，接着就把这个 IRP 传递到驱动程序栈的顶层，再由顶层的驱动程序将这个 IRP 交给这个设备的各层驱动程序进行处理。

图 5.2 为 IRP 的结构，它由两个部分组成。其中，一个部分为大小固定的“首部”，另一个部分是大小可变的“堆栈”。首部包含了 IRP 的内部属性（即 I/O 请求的通用管理信息），而堆栈则含有与该 I/O 请求相关的参数，由一个或多个 I/O 栈单元组成。

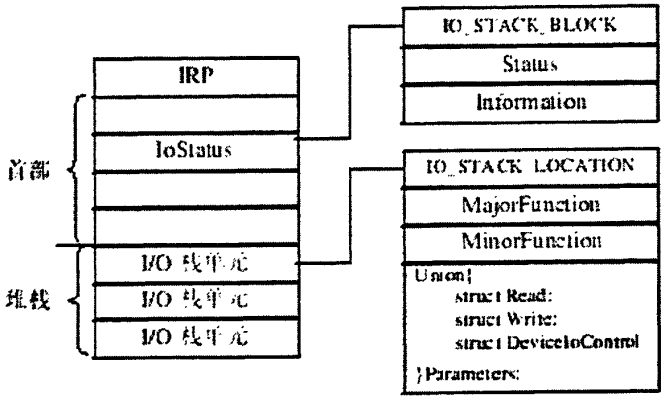


图5. 2 IRP的结构
Fig.5.2 The structure of IRP

IRP 结构的首部有 IoStatus 字段，这个字段中保存了 I/O 请求经过某个驱动程序处理后的状态。IoStatus 字段采用了 IO_STACK_BLOCK 结构，其内部有两个成员变量：一个是 Status 成员变量，另一个成员变量是 Information。IRP 结构的堆栈中栈单元都采用了 IO_STACK_LOCATION 结构。其中，MajorFunction 为主功能代码，每个 I/O 请求都必须有一个主功能代码；MinorFunction 为次功能代码，它是可选的。

5.4 WDM 驱动程序的组成

WDM 驱动程序是由一些例程组成的。一个 WDM 驱动程序由 5 个基本的例程组成，这 5 个例程为：驱动程序入口例程、即插即用例程、分发例程、电源管理例程、卸载例程。

驱动程序入口例程是整个驱动程序的入口点，负责驱动程序的初始化并向系统指明其它各个例程的指针，所有的驱动程序都必须包含入口例程。

驱动程序入口例程的代码中指出了其它例程的名字。其中，MyUSBAddDevice 和 MyUSBPnpIrp 为即插即用例程；分发例程包括 MyUSBCreate、MyUSBClose、

MyUSBIOCTL这三个例程；MyUSBPowerIrp 为电源管理例程；MyUSBDrvUnload为卸载例程。这些例程必须在驱动程序的其他位置写出其实现函数。

5.5 USB设备驱动程序编程接口

为了让USB设备驱动程序能够方便地将IRP发送到下层驱动程序从而实现与硬件的通信，Windows 操作系统提供了 USB 总线驱动程序的接口 USBDI（USBDriver Interface）。图5.3说明了USBDI与WDM驱动程序栈的关系。

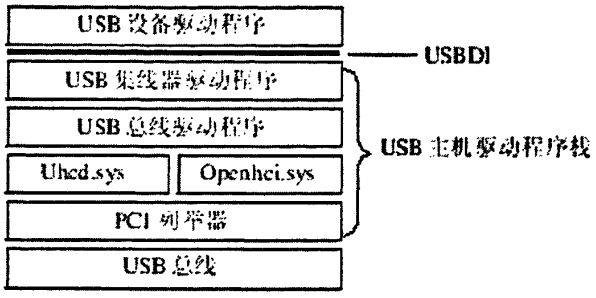


图5.3 USBDI与WDM驱动程序栈
Fig.5.3 USBDI and WDM driver stack

由图7.3可以看出USBDI位于USB设备驱动程序与USB主机驱动程序栈之间，起到了一个通信桥梁的作用。USBDI为驱动程序开发人员提供了一系列的实现USB设备驱动程序和USB主机驱动程序栈之间通信的接口函数。USB设备驱动程序通过调用这些接口函数来向主机驱动程序栈发出IRP，以启动它与USB设备之间的通信。

5.6 应用程序设计

5.6.1 USB设备驱动程序与应用程序之间的通信

在Windows中，应用程序实现与WDM通信是通过调用Win32 API函数来实现的。其实现过程为：首先应用程序调用 CreateFile 函数来打开设备，设备成功打开后再调用DeviceIoControl 函数和WDM进行通信，包括从WDM中读取数据和写数据到WDM两种情况，也可用 ReadFile 函数从WDM中读数据或用 WriteFile 函数写数据给 WDM。当应用程序完成读写时，用 CloseHandle 函数关闭设备。

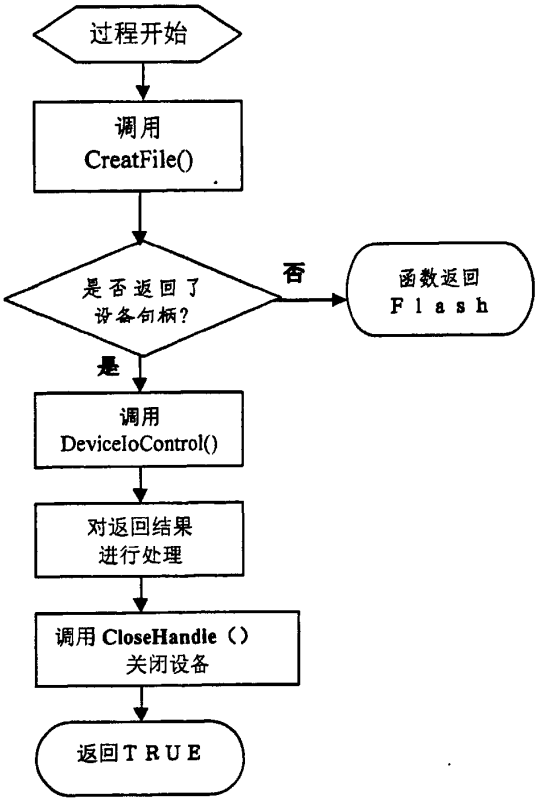


图5.4 应用程序与WDM 进行数据传输的函数流程图
Fig.5.4 Flow chart of data transfer between application and WDM

由于本系统的驱动程序在入口例程（DriverEntry）中未定义主功能代码IRP_MJ_READ 和 IRP_MJ_WRITE 的实现例程，所以不能使用 ReadFile 和 WriteFile 来读写数据，只能使用 DeviceIoControl 来与 WDM 进行通信。系统中专门给应用程序与WDM 进行数据传输编写了函数。图 5.4 为该函数的流程图。

5.6.2 应用程序

在进行应用程序开发时，选用了 VC++来进行开发。应用程序在下载 USB 固件程序、读取 USB 设备的描述符、读写数据时，首先都会调用 CreateFile 函数。如果 CreateFile 函数打开设备，则会返回设备的指针。应用程序中 CreateFile 函数的调用代码为：

```
*hDeviceHandle=CreateFile(DeviceName,        //设备名的指针
                          GENERIC WRITE,     //访问模式
                          FILE SHARE WRITE,  //共享模式
```

```
                    NULL,          //安全属性指针
                OPEN EXISTING,          //如何创建
                0, NULL);
```

在 CreateFile 函数成功返回后，应用程序就可以调用 DeviceIoControl 函数与 WDM 进行通信了。DeviceIoControl 函数的原型为：

```
BOOL DeviceIoControl(HANDLE hDevice,      // CreateFile 返回的设各句柄
    DWORD dwIoControlCode,      //调用驱动程序的控制命令
    LPVOID lpInBuffer,          //输出数据的数据缓冲区地址
    DWORD nInBufferSize,        //输出数据的数据字节数
    LPVOID lpOutBuffer,         //返回数据的缓冲区地址
    DWORD nOutBufferSize,       //返回数据的缓冲区字节数
    LPDWORD lpBytesReturned,    //驱动程序实际返回的数据字节数的变量地址
    LPOVERLAPPED lpOverlapped//指向 OVERLAPPED 结构的变量地址
```

DeviceIoControl 函数的调用相对 CreateFile 函数要复杂，从设备读取或向设备写入的内容不同其调用的代码也不同。从设备读取还是向设备写入，具体的数据内容又是什么，这些都由 DeviceIoControl 函数的第二个参数 dwIoControlCode 来区分。系统的应用程序中定义了 6 个 IOCTL 代码，来完成应用程序所需要的功能。这 6 个 IOCTL 代码及功能为：IOCTL_GET_DEVICE_DSCPTR 、IOCTL_GET_CFG_DSCPTR 、IOCTL_GET_STRING_DSCPTR 用于读取 USB 设备的设备描述符、配置信息、字符串描述符；IOCTL_BULK_READ 用于主机从 USB 设备读取块传输数据、IOCTL_BULK_WRITE 用于主机向 USB 设备发送块传输数据；IOCTL_GET_CURRENT_FRAME_NUMBER 用于读取当前的帧号。

当 DeviceIoControl 函数返回后，就可以调用 CloseHandle 函数来关闭设备了。CloseHandle 函数的调用非常简单，只有一个参数，即设备的句柄。其调用代码为：CloseHanle(hDeviceHandle)。

另外，为了保证应用程序与 USB 设备之间数据传输的稳定性，应用程序专门建立了两个线程来进行数据传输。采集系统的应用程序的界面如图

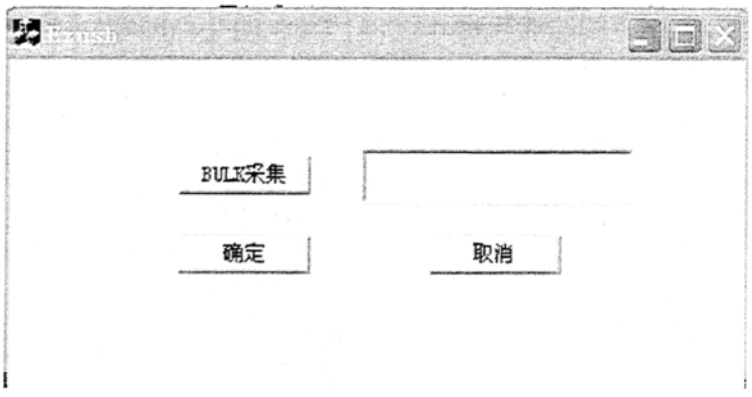


图 5.5 图象采集窗口

Fig.5.5 Image acquisition window

图 5.5 所示。在用户点击了“BULK 采集”按钮后，应用程序首先判断 USB 设备是否正在读取数据，如果正在读取数据则不启动线程而是报告“正在读取数据”。如果 USB 没有执行数据读取任务，则启动 TestThread 线程。在 TestThread 线程中，首先对线程中用到的变量进行定义和初始化，然后打开 USB 设备。如果 USB 设备不能成功打开，则提示“打开 USB 设备失败”，并结束线程；如果打开 USB 设备成功，则设置用于 USB 传输的管道号并建立事件对象，接着就启动 TestThread 的另一个线程 TransferThread。

另外我们也可以使用 Cypress 公司提供的“EZ-USB Control Panel”软件从 USB 接口读到的数据进行了数据比较。

EZ-USB Control Panel 如图 5.6 所示：

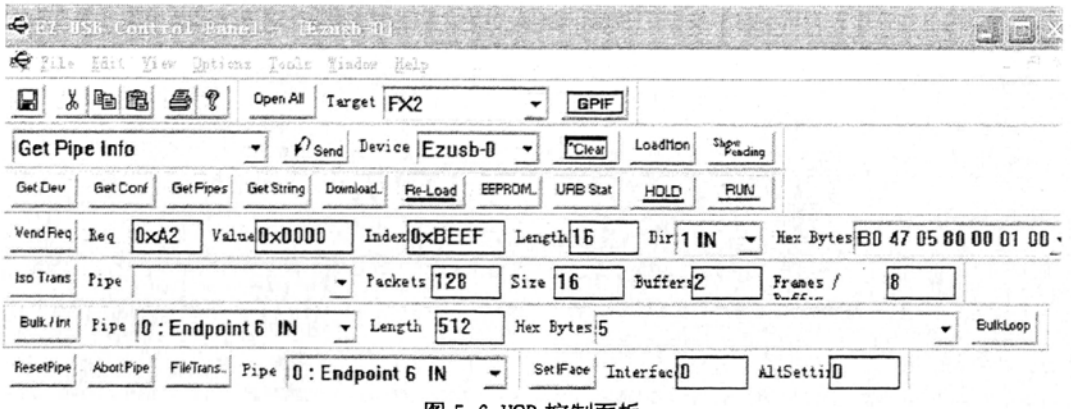


图 5.6 USB 控制面板

Fig.5.6 EZ-USB Control Panel

按 Bluk/Int 按钮，采集到的数据如下：

ResetPipe | AbortPipe | FileTrans.. | Pipe | 0 : Endpoint 6 IN | SetIFace | Int

```
0040 BB E0 24 FE 60 27 14 60 38 24 FD 60 11 14 60 27
0050 24 06 70 50 E5 23 90 E6 B3 F0 E5 24 80 3C 12 07
0060 F5 50 3E E5 2B 90 E6 B3 F0 E5 2C 80 2D E5 25 90
0070 E6 B3 F0 E5 26 80 23 E5 27 90 E6 B3 F0 E5 28 80
0080 19 90 E6 BA E0 FF 12 0B 08 AA 06 A9 07 7B 01 EA
0090 49 4B 60 0D EE 90 E6 B3 F0 EF 90 E6 B4 F0 02 02
00A0 AF 02 02 9E 02 02 9E 12 0C 15 02 02 AF 12 0C 39
00B0 02 02 AF 12 0C 31 02 02 AF 12 0C 03 02 02 AF 12
00C0 0C 53 40 03 02 02 AF 90 E6 B8 E0 24 7F 60 15 14
00D0 60 19 24 02 70 63 A2 01 E4 33 25 E0 FF A2 04 E4
00E0 33 4F 80 41 E4 90 E7 40 F0 80 3F 90 E6 BC E0 54
00F0 7E FF 7E 00 E0 D3 94 80 7C 00 40 04 7D 01 80 02
0100 7D 00 EC 4E FE ED 4F 24 27 F5 82 74 0C 3E F5 83
0110 E4 93 FF 33 95 E0 FE EF 24 A1 FF EE 34 E6 8F 82
0120 F5 83 E0 54 01 90 E7 40 F0 E4 A3 F0 90 E6 BA F0
0130 90 E6 BB 74 02 F0 02 02 AF 02 02 9E 12 0C 55 40
0140 03 02 02 AF 90 E6 B8 E0 24 FE 60 16 24 02 60 03
0150 02 02 AF 90 E6 BA E0 B4 01 05 C2 01 02 02 AF 02
0160 02 9E 90 E6 BA E0 70 55 90 E6 BC E0 54 7E FF 7E
0170 00 E0 D3 94 80 7C 00 40 04 7D 01 80 02 7D 00 EC
0180 4E FE ED 4F 24 27 F5 82 74 0C 3E F5 83 E4 93 FF
0190 33 95 E0 FE EF 24 A1 FF EE 34 E6 8F 82 F5 83 E0
01A0 54 FE F0 90 E6 BC E0 54 80 13 13 13 54 1F FF E0
01B0 54 0F 2F 90 E6 83 F0 E0 44 20 F0 80 72 80 5F 12
01C0 0C 57 50 6B 90 E6 B8 E0 24 FE 60 19 24 02 70 4E
01D0 90 E6 BA E0 B4 01 04 D2 01 80 54 90 E6 BA E0 64
01E0 02 60 4C 80 39 90 E6 BC E0 54 7E FF 7E 00 E0 D3
01F0 94 80 7C 00 40 04 7D 01 80 02 7D 00 EC 4E FE ED
0200 4F 24 27 F5 82 74 0C 3E F5 83 E4 93 FF 33 95 E0
0210 FE EF 24 A1 FF EE 34 E6 8F 82 F5 83 80 0D 90 E6
022D A0 80 08 12 09 18 50 07 90 E6 A0 E0 44 01 F0 90
```

图 5.7 USB 控制面板采集到的数据

Fig.5.7 acquired data of Control Panel

从图 5.7 可以看出 USB 控制面板能够从 USB 接口中读取管道信息和数据，说明 USB 接口工作正常。

通过运行应用程序，产生的数据可以通过 USB 接口送到计算机。

运行应用程序后显示的图像

图 5.8 上位机上显示的图像

Fig.5.8 display images in PC

结　论

本文“基于USB2.0和FPGA的图像采集、存储系统研究”，就是要使用USB接口来构成一个高速的USB图像采集、存储系统，论文的主要内容包括：

(1) 硬件电路的设计。在掌握USB 2.0协议的基础上，设计了一块图像采集卡来实现对图像的采集。设计工作包括芯片的选择，采集系统电路原理设计、FPGA内部逻辑设计、PCB设计。

(2) USB固件程序设计。在USB固件程序设计部分，对USB固件的构架作了研究，掌握了USB控制器CY7C68013的固件程序设计语言和设计方法，并编写了USB固件程序。

(3) 驱动程序和应用程序设计。本文对WDM驱动程序设计作了研究，内容包括：WDM驱动程序的分层、内核模式对象、驱动程序的例程、驱动程序编程接口等。在应用程序方面，掌握了在VC++开发环境下的应用开发技术，并编写了利用USB接口进行数据采集的应用程序。

(4) 系统的调试。系统的调试工作包括硬件的调试和软件的调试，由于系统使用的芯片较多，所以调试难度也相对比较大。在调试过程使用的方法是先将系统分成几个大的部分分别调试，在确定各个部分能正常工作后，再将各个部分结合起来进行调试。

由于个人的时间和能力限制，课题工作没有全部完成。最主要的问题不能正确的控制FPGA对采集到的数据完全正确FLASH进行存取。在经过十个月时间的设计和调试后，发现现在设计的系统存在着一些需要改进的地方，主要有：

(1) FPGA与FLASH之间应该使用缓存更大FIFO来连接。而在系统调试过程中发现：若采集640×480的图像，帧频为30fps，则FPGA将一行数据写入到FIFO之后，行与行之间的时间间隔仅为18μs，FIFO在这个时间里面不可能将数据全部读到外部存储器FLASH中，所以会造成FIFO在采集数据时出现问题。这样只能通过降低帧频或减小输出窗口的大小等办法来解决这个问题。

(2) PCB板应该设计为四层板或六层板。本系统在PCB设计中，由于只采用了两层板来设计，所以造成电路板面积很大，过孔太多、走线过长、线间距小等问题。这些问题又可能会导致电路板线间的串扰、阻抗不匹配等问题。

(3) 应该选择延迟时间更小，内部缓存更大的FPGA，直接使用内部缓存。

由于EP1C6T144C8工作在5V，而除图像传感器以外的其它器件都工作在3.3V，所以在设计时会因电压的不匹配而带来很多不便。而且EP1C6T144C8较高的工作电压

以及内部的TTL门电路会造成USB设备的功耗太大。总之，系统中存在着很多不足的地方，还需要进一步的努力来完善。

参 考 文 献

[1] 王成儒，李英伟编著.USB2.0 原理与工程开发.国防工业出版社.2004 年

[2] 边海龙，贾少华编著.USB2.0 设备的设计与开发.人民邮电出版社.2004 年

[3] 刘青丽.基于 USB2.0 接口的数据采集系统的设计与实现.西南交通大学.硕士学位论文.2001 年

[4] USB 应用开发技术大全。人民邮电出版社.2007 年

[5] 许永和编著.EZ-USB FX 系列单片机 USB 外围设备设计与应用. 北京航空航天大学出版社.2002(11)

[6] 萧世文编著.USB2.0 硬件设计.清华大学出版社.2002 年

[7] 巫锴，袁祥辉等.基于 USB 接口的高速图像采集系统设计.半导体光电.2005.Vol(26).增刊.pp122-144

[8] 清源科技.TMS320C54x FIFO 硬件开发教程.机械工业出版社.2003 年

[9] 清源科技.TMS320C54x FIFO 应用程序设计教程.机械工业出版社.2004 年

[10] 陈金鹰主编.FIFO 技术及应用.机械工业出版社.2004 年

[11] 代少升.红外焦平面阵列实时非均匀性校正研究.重庆大学.博士学位论文.2004 年

[12] TI.EP1C6T144C8 Fixed-Point Digital Signal Processor Data Manual.2003

[13] Cypress.CY7C68013 EZ USB FX2 USB Microcontroller , High Speed USB PeripheralController.2002

[14] 刘青丽.基于USB2.0 接口的数据采集系统的设计与实现.西南交通大学.硕士学位论文.2001 年

[15] 袁祥辉著.固体图像传感器及其应用（修订版）.重庆大学出版社.1996 年

[16] OmniVision.OV7620 Product Specifications -Rev. 1.3.2000

[17] Cypress.EZ-USB FX2 Technical Reference Manual v2.1.2001

[18] Keil Software.Cx51 Complier， Optimizing C Compiler and Library Reference for Classic andExtended 8051 Microcontroller User Guide.2001

[19] 武安河，邰铭等编著 WDM 设备驱动程序开发.电子工业出版社.2003 年

[20] 张剑波，颜钢锋.基于 GPIF 的 USB-ATA 解决方案.电子技术应用.2002.Vol.28 .No.7.pp14~17

[21] 曹跃胜.高速 PCB 设计技术.计算机工程与科学.1998.Vol.20.No.4.pp70~75

[22]郑超,岳育明.EZ-USB FX2 与 TMS320C6201 的接口设计.电讯技术 .2003.Vol.43.No.6.pp111~115

[23] 邓思豪，曾春年.TMS320C54x 共享数据的新方案.武汉理工大学学报（信息与管理工程版）.2003.Vol.25.No.1.pp38~41

[24] 何立民编著.MCS-51 系列单片机应用系统设计.北京航空航天大学出版社.1990 年

[25] 尹勇，欧光军，关荣锋编著 FIFO 集成开发环境 CCS 开发指南.北京航空航天大学出版社.2003 年

[26] 余国华，冯启明.基于 CMOS 图像传感器的视频采集系统设计.武汉理工大学学报（交通科学与工程版）.2004.Vol.28.No.1.pp145~147

[27] 胡汉才编著.单片机原理及接口技术.清华大学出版社.1996 年

[28] 马忠梅，籍顺心等编著.单片机的 C 语言应用程序设计.北京航空航天大学出版社（修订版）.1999年

[29] Andrzej Podgorski， Rafal Nedwidek， Michal Pochmara.Implementation of the USB Interface inthe Instrumentation for Sound and Vibration Measurements.Data Acquisition and AdvancedComputing Systems: Technology and Applications.2003.9.pp159~163

[30] Wright Nick， Judd Bob.Using USB as a data acquisition interface.Evaluation Engineering.2004.Vol. 43.No. 6.pp20~26

[31] Wei Mingzhi， Stover R.J..A USB 2.0 computer interface for the UCO/Lick CCD cameras.TheInternational Society for Optical Engineering.2004.Vol. 5499.pp476~480

[32] Wu Xia..High speed data acquisition system using USB interface.Journal of ElectronicMeasurement and Instrument.2004.Vol. 2.pp1151-1155

[33] Don Anderson， Dave Dzatko 著.USB系统体系（第二版）.中国电力出版社.2003年

[34] David McCombs 著PC数据采集——使用C++测量物理量.中国电力出版社.2004年

[35] 钱能主编.C++程序设计教程.清华大学出版社.1999年

[36] Charles Petzold 著Windows程序设计.北京大学出版社.1999年

[37] 李哲英，骆丽，刘元盛编著FIFO基础理论与应用技术.北京航空航天大学出版社.2002年

[38] TI.TMS320VC54x FIFO Reference Set.2001

[39] TI.TMS320VC54x FIFO Enhanced Peripherals Ref Ser.2001[42] TI.Code Composer Studio User' s Guide.2001

[40] 王道宪主编.VHDL电路设计技术.国防工业出版社.2004年

[41] 宋万杰，罗丰，吴顺君著.FPGA 技术及其应用.西安电子科技大学出版社.1999年[45] 齐洪喜，陆颖编著.VHDL 电路设计实用教程.清华大学出版社.2004年

攻读硕士学位期间发表学术论文情况

题目：《微电脑控制客车空调延时启动系统研究》

作者：张 通[1]，徐正藻[2]，孙培岩[3]

作者单位：1.3 大连理工大学能源与动力学院

2. 大连理工大学汽车工程学院

发表时间：2008 年 12 月

文章页码：尚未出版

期刊介绍：《测试技术学报》主办单位为中国兵工学会，主要刊登有关测试技术的重点科研项目、基金项目的研究成果以及兵器及民用测试的应用技术，现已被美国 EI 光盘版及俄罗斯 PЖ收录。是《中国科学引文数据库》源刊，即中国科技核心期刊。

期刊号：ISSN1671-7449

CN14-1301/TP

致　谢

本论文是在大连理工大学动力系内燃机专业孙培岩副教授的悉心指导下，论文的选题由孙副教授确定，在具体的开发实践过程中给予了我许多指导和帮助。工作中孙老师渊博的专业知识、勤奋严谨的治学精神和丰富的实际经验都使我为之钦佩。在开发本图像采集系统期间，我不仅学到了许多本专业的最新知识和行业动态，也从各位师长身上学到了许多为人处事的方法。在此，向在该系统开发过程中给予我帮助的师长表示衷心的感谢！

在此期间，同教研室的其他老师和同学们也给予了我学习上和生活上的莫大帮助。满长忠老师、唐运榜老师，薛冬新老师提供了许多有关资料和建议，在系统开发中起到了很大的作用。同时，感谢张家驰，王东镇，王晓峰，丁春雨同学的帮助，他们在编写程序和硬件设计方面以及后期的论文的编写等方面提供了许多创意性的建议！同时，感谢实验室的刘鹏、越峰等各位师弟师妹，感谢他们在这几年中给予我的鼓励、支持和帮助！衷心感谢百忙之中评阅论文和参加答辩的各位专家和教授。

在近三年的研究生学习阶段，我从身边的老师和同学中学习了很多知识，使我受益匪浅。在这次实际工程项目的整个研发过程中，使我对系统研发、调试的整个过程有了深刻的认识，外协的过程中也增强了我的社会实践经验，最后感谢我的家人在我进行硕士课题过程中对我的关怀。

谨以此论文献给所有关心和帮助我的师长、同学和家人们！

基于USB2.0和FPGA的图像采集、存储系统研究

作者： 张通
学位授予单位： 大连理工大学

相似文献(10条)

1.期刊论文 赵杨.高升久.刘桂芬.ZHAO Yang.GAO Sheng-jiu.LIU Gui-fen 基于FPGA的多功能红外图像源系统设计 -激光与红外2008,38(11)

提出了一种基于现场可编程门阵列(FPGA)的多功能红外图像源系统的设计方案.信号源系统在降低图像系统的研制成本、缩短研制周期和提高研制质量方面具有重要的作用.以Altera公司的EP2S60-FPGA作为硬件架构,设计了一种红外图像源系统.该系统通过数据压缩、数据扩展及插值、制式转换等方法实现红外图像数字视频及模拟视频的双通路输出,用于二次开发和图像显示.所设计的红外图像源系统具有参数易调与小尺寸低功耗等特点,可方便下载现场红外图像信号和实时输入CCD图像信号,并按照红外图像格式输出,从而为红外图像信号处理器提供输入激励.

2.期刊论文 朱晓丽 关于机器视觉及其应用问题的探讨 -中国教育技术装备2010,""(9)

1 机器视觉的概念

简而言之,机器视觉就是用机器代替人眼来做测量和判断.机器视觉系统是指通过机器视觉产品(即图像摄取装置,分CMOS和CCD两种)将被摄取目标转换成图像信号,传送给专用的图像处理系统,根据像素分布和亮度、颜色等信息,转变成数字化信号;图像系统对这些信号进行各种运算来抽取目标的特征,进而根据判别的结果来控制现场的设备动作.

3.期刊论文 牛一帆 机器视觉在印刷质量检测中的应用 -印刷质量与标准化2009,""(9)

机器视觉(Hachine Vision),顾名思义就是将自动控制机器的能力和视觉传感结合起来.形象地说,人通过眼睛感知外部世界的变化,然后通过大脑进行分析处理、作出判断,最后由手和脚完成动作.机器视觉就是将这一过程通过具有运算能力的自动化控制设备来完成的.具体来说,机器视觉是用摄像机代替人眼,用计算机代替人的大脑来进行测量与判断的.机器视觉系统(Machine Vision System)是指通过机器视觉产品(即图像摄取装置,分CHOS和CCD两种)将被摄取目标转换成图像信号,传送给专用的图像处理系统,根据像素分布和亮度、颜色等信息,转变成数字化信号;图像系统对这些信号进行各种运算来抽取目标的特征,如;面积、长度、数量、位置等;再根据预设的容许度和其他条件输出结果,如:尺寸、角度、偏移量、个数、合格/不合格、有/无等,最后根据判别的结果来控制现场的设备动作.

4.学位论文 冀曙光 基于分形理论的VDR雷达图像编码方法的研究 2007

航行数据记录仪，简称VDR(Voyage Data Recorder)，是一种专门用于记录、保存和分析船舶航行过程中重要信息参数的智能化记录设备。在海上交通事故发生后，VDR所记录的数据对于分析事故原因，进行海事责任判定具有不可替代的作用。VDR系统中的雷达图像系统用于采集和存储雷达的图像信号。经过压缩后再将数据传输到存储器存储。图像画面分辨率可高达1280×1024像素，采集周期通常为每15秒一帧。

分形理论是现代数学与非线性科学研究中十分活跃的一个分支，最近十年它在计算机图像处理和分析中显示出越来越重要的作用。图像压缩、自然图像的模拟、图像纹理分析、模式识别等方面都可见大量的用分形理论来分析和研究的报道。

图像压缩编码技术日新月异，尤其是分形编码技术突破了原有编码技术的界限，对于某些图像而言，有着其它传统算法无可比拟的压缩比，引起了人们的极大关注。分形编码是一个前沿性的课题，虽然在理论上它具有很大的优越性，但是就实践而言，人们对它的研究才处于萌芽阶段。

本文回顾了VDR与分形的发展历程，并简要介绍了有关分形编码的相关知识。给出了基于分形理论的VDR雷达图像编码方法的数学基础，论并比较了不同分形图像压缩方法的优缺点。在将分形理论应用于VDR雷达图像编码中，取得了有价值的实验结果。本文所涉及的主要工作如下：

第一，对船载数据记录仪进行了全面的分析与介绍；

第二，对分形理论的相关知识进行了系统的分析研究；

第三，将分形理论应用于图像编码算法当中，并获得了相应的实验结果。

5.期刊论文 王红彪 浅谈机器视觉 -自动化博览2008,25(1)

简单的讲机器视觉就是用机器代替人眼来做测量和判断.通过成像器件(即图像摄取装置,分CMOS和CCD两种)将被摄取目标转换成图像信号,传送给专用的图像处理系统,根据像素分布和亮度、颜色等信息,转变成数字化信号;图像系统对这些信号进行各种运算来抽取目标的特征,进而根据判别的结果来控制现场的设备动作.

6.学位论文 杨振 基于子带的SAR图像压缩编码 2005

作为微波遥感的代表,合成孔径雷达(Synthetic Aperture Radar,SAR)因具有全天候,全天时对地球表面进行观察的能力以及空间分辨率高的特点,使得SAR在民用和军事方面发挥着越来越大的作用.在SAR图像系统中,从图像的获取,到图像的发送、传输及接收,如何实现对SAR图像有效的数据压缩,也是图像处理的基本要求.本文对SAR图像的数据压缩进行了研究,提出了一种基于子带的块截断编码算法,并通过仿真验证了算法的有效性.子带编码在语音编码中应用的非常普遍,而将它用于图像编码是对子带编码这一概念的扩充.它的基本思想是:在发送端利用数字滤波器组将图像信号分解成高、低不同的频带,然后根据各频带信号的统计等特性,选择与之相适应的编码方式进行编码.块截断编码是一种具有良好实时性的快速压缩算法,自它出现之后,便在图像压缩领域得到了广泛应用,并不断发展.本文在分析了基于完全重构子带滤波器组的图像分解与合成的基础上,针对图像子带分解后,不同子带所含信息重要性的不同,提出了一种新颖的基于子带分解的预处理块截断编码算法,并将其应用于SAR图像的压缩.仿真结果表明,该方法比未采用预处理的压缩方法具有更好的实时性和重建效果.

7.期刊论文 兰海军.文友先 机器视觉技术的发展和应用 -湖北农机化2007,""(5)

1机器视觉系统的概述

机器视觉(又称计算机视觉)是指用计算机来实现人的视觉功能,也就是用计算机来实现对客观的三维世界的识别.简言之,机器视觉就是用机器代替人眼来做测量和判断.机器视觉系统是指通过机器视觉产品(即图像摄取装置,分CMOS和CCD两种)将被摄取目标转换成图像信号,传送给专用的图像处理系统,根据像素分布和亮度、颜色等信息,转变成数字信号;图像系统对这些信号进行各种运算来抽取目标的特征,进而根据判别的结果来控制现场的设备动作.

8.学位论文 闽亚 海洋表面油膜的ERS-SAR图像提取及浮油面积估算 2007

近年来，通过卫星遥感对海洋表面油膜进行检测技术已经得到普及，许多学者对此给予很大的关注。合成孔径雷达(SAR)使用微波辐射来探测描述地球表面信息，克服了由传统视频遥感成像带来的相关技术问题。SAR对海表面敏感的原因是因为短波的存在，而风是波浪产生的最重要的原因。当风吹在海表面时，产生了波长少于5毫米毛细波。当风继续激发毛细波，它们把能量转移到波长较长的波直到平衡，此时中间波和短波就形成了。另外，重力波也将能量转移到重力毛细波。由于短波的产生以及由于浮油粘性而使波的传播停止等原因，在清洁的海洋表面就形成了浮油覆盖的油膜层。从SAR传感器的角度，浮油的特点就是在雷达图像上海面后向散射能量的减少并通过与周围区域比较形成一个黑色区域而重新得到复原。SAR与其他遥感系统相比较，其优势在于它能在不同的天气情况下起作用。由于被油覆盖的海水表面可以在雷达图象上形成黑色斑点，因此合成孔径雷达(SAR)为人们检测浮油提供了一种很好的手段。

油膜图像信息提取的两个关键问题是SAR图像本身受到斑点噪声及雷达图像上类油信号的干扰，从而使雷达图像模糊不可辨。连续的微波信号产生的斑点噪声破坏了雷达图像的清晰性。基本的纹理一般是由于斑点噪声影响的。而且，斑点的出现把图像的显示率损坏还有它影响人的解释和景物分析任务。因此，在SAR图像上仔细地减少斑点噪声而使边缘和纹理信息得以保留是非常重要的，因此，减少斑点噪声是浮油检测、目标跟踪和物标识别的一个必不可少的程序。在图像系统中撤除斑点噪声时而没有把图象边缘弄脏(变模糊)是非常重要的。一般来看，噪声在图像信号上体现为高空间频率。基于傅立叶变换减少斑点噪声影响的方法通常采用抑制高频率组分该方法同样也影响了边缘图像的清晰性和分辨率。小波变换能提供在空间域和频谱域都较

好的定位。小波基的非线性阈值技术在降噪效果上比线性降噪技术明显具有优势。在这篇学术论文文章里，我们提出了一种应用小波包变换的斑点噪声抑制的新方法。

在所提出的方法中，本文使用小波包变换方法，它不仅分解近似子空间，而且是细节子空间，这在图像压缩中是常用的方法。因为所有细节成分被分解，被分解的子空间次数量大于其在小波变换中分解的数量。以这样很大数量有效的扩展，基于变换的小波包能大大提高对分段图像频谱的控制。引入这种控制的代价是增大了计算的复杂性(计算量从O(M)增大到O(Mlog M))。可能的分解数量经常是很大，所以寻找每一个分解的阈值然后撤除斑点噪声往往成为不切实际的空想。因此，人们迫切需要寻找一种使用某种特殊评判标准而寻找最优分解的高效率的算法。在本文的方法里，使用叠加性价值函数分解标准。这个函数可以确定二维函数的熵或称为负平均信息量。熵接近于零表明这函数几乎不带任何信息的。换句话说高信息含量价值是表示那函数以许多非零熵值。所以，从降噪的角度，使熵值非零的数量最大的价值函数是选择最佳分解的合理判据。该方法的原理在于，在压缩中，用一个价值函数找到更多数量的包含近似于零价值的成分然后把它们删除或取阈值到零。由这个想法启发，我们通过叠加性价值函数提出了基于能适应的小波包方法来解决消除斑点噪声。叶节点阈值是通过每一叶节点以及整个图像的噪声的估计方差变化进行计算。因此，阈值操作可以用更多接近零系数处理在更加狭窄的更高的频带信号，使降噪效果得到大大体高。问题的关键是如何为每个分段信号的小波系统选择合理的阈值。首先，把整个图像的估计斑点噪声分差估计。然后，叶节点阈值是通过每一叶节点以及整个图像的噪声的估计方差变化进行计算。最后，除了近似子空间以外每个叶节点子空间以自适应的方式进行处理。文章最后对本文提出的方法的降噪结果与目前常用的人工斑点噪声滤波方法以及基于小波的软性和硬性阈值处理法在图像视觉效果和信躁比(即峰值信号对噪声信号的比值)(PSNR)方面进行比较。

为了解决第二个问题，本文采用基于模糊聚类的自动识别技术来区别浮油与其他类似于东西。所谓其他类似于浮油的东西包括自然薄膜，油膏冰，门限风速区域，风庇护陆路，雨滴形成区域，水流形成区域，内部波浪和上升流等。在它们之中，自然薄膜形成的混淆是最大的问题。在本文里，油的自动识别是使用Mamdani模糊模型来解决的。在这种方法，模糊分类器对分割图象的结果进行识别，而分割是通过图像平滑和阈值运算是来完成的。由于图像平滑不仅消除噪声也消除细节，总值模糊的照片只能在某种程度上代表原始图象。因此，它在识别过程中可能将影响接下来的处理，比如说运算浮油面积。在本论文的工作中，应用小波包变换的斑点噪声抑制方法来进行图像的平滑处理，而对非常小的物标则采用区域滤波的方来去除。采用灰度阈值算法将图像分为两类，一类具有用户定义值一下的像素图像，而另一类是具有定义值以上的像素。这是一个为提取图象的黑地区的重要操作。对应于数字式数的所有像素点低于阈值的记录为一个黑色区域的像素点。阈值计算的结果将最初的海面图像部分分割为黑暗和明亮两个不同部分。对可能物标像素组的提取信息被送到系统分类器中。

通过上面提出的方法进行的分割结果与图像平滑算法的结果比较。本文上述所提出的方法能给出与原始图像更加近似的图象，并且模糊分类器可能做出更加精确的决定。最后，识别出的浮油覆盖面区域通过区域标记的方法被计算出来。

9.期刊论文 孔祥梅.王宝军 实时图像采集和去噪系统 -中国科技信息2007,""(24)

随着图像处理技术和多媒体技术的广泛应用,为了不影响图像采集系统的速度,对总线传输速率的要求也越来越高.去除噪声同时还要保留图像信号的高频信息,系统采用并行算法设计思想设计改造现已有中值滤波方法,提出层叠中值滤波结构模块进行滤波,并基于PCI总线在数据传输的优点,利用PCI总线完成实时图像系统采集后数据传输,给出整个硬件构造方法,利用VHDL硬件开发语言在ALTERA的可编程整列(FPGA)上实现,说明系统的可行性.

10.学位论文 巫锴 基于USB接口的图像采集系统设计 2005

图像信号的采集和处理在科学研究、工农业生产、医疗卫生、公共安全等领域得到了越来越广泛的应用，而这些工作都需要一套高速的图像系统来完成。同时图像采集也是进行图像处理、图像压缩、图像识别的基础，所以对图像采集系统的研制有着重要的现实意义和价值。要对图像进行采集就需要一种高速的，能进行长时间、大吞吐量数据传送的计算机接口。USB 2.0接口就是一种符合图像采集要求的计算机接口。同时USB接口还具有支持热插拔、占用系统资源少、易于扩展、使用方便等优点。当前，计算机的许多外围设备都采用了USB接口来实现与计算机间的数据通信。

本课题“基于USB接口的图像采集系统设计”，就是要设计一套高速的图像数据采集卡，并通过该采集卡上的USB2.0接口来实现高速的数据传输，将数据送到计算机去进行存储和显示。在课题研究中，设计了图像采集卡的硬件电路，制作了PCB板，并对图像采集卡的软件部分进行了开发。该图像采集卡由五大部分组成：CMOS图像传感器电路、CPLD控制电路、SRAM存储器、数字信号处理器DSP和基于USB 2.0接口的数据传输部分。

论文将USB接口与其它计算机接口进行了比较，总结出USB接口的优点，并深入地研究了USB 2.0协议。

在硬件设计方面，论文提出了图像数据采集系统总体方案，设计了图像采集卡的工作原理，介绍了采集系统中用到的主要芯片以及系统硬件电路中主要模块的设计、CPLD内部逻辑设计和PCB设计等内容。

在软件设计方面，论文讲述了采集系统的DSP程序的设计开发过程；还在USB2.0协议的基础上提出了采集系统的CY7C68013固件程序的设计思路。在完成采集系统固件程序开发的基础上，论文对固件程序中主程序、初始化函数、端点配置以及GPIF接口设计等内容进行了论述。WDM驱动程序是实现访问和控制硬件的软件接口，应用程序为用户提供界面并最终将USB数据传输的结果提供给用户，论文叙述了采集系统的USB接口WDM驱动程序和应用程序的开发。论文实现了从DSP到计算机的数据传输，并给出数据传输的结果。