大连理工大学

硕士学位论文

基于FPGA的嵌入式图像采集卡的研究

姓名：彭平良

申请学位级别：硕士

专业：信号与信息处理

指导教师：李木国

20071220

摘　要

图像采集和处理技术在机器视觉和图像分析等诸多领域应用十分广泛，大部分情况下，采集卡只需将前端相机捕获的图像信息正确地传回计算机即可。但是在要求较高的应用场合需要采集卡能准确控制外部光源和相机，完成图像采集，预处理，数据传输。只有这样，用户才可以根据不同的兴趣和需求对特定的某些图像进行采集、传输以及处理，以达到某种分析目的。

本文根据国家 985 二期项目“三维粒子图像测速系统”的图像采集与处理需要，设计开发了一款以 FPGA 为核心控制芯片的嵌入式图像采集卡。采集卡以 FPGA 为逻辑和算法实现的核心器件，不仅实现了传统意义上的图像采集，而且实现了 CCD 相机控制和激光器同步曝光功能，打破了以往单纯靠增加硬件设备实现同步控制的方法，简化了系统硬件结构并节约系统成本。此外，在系统中嵌入了图像增强算法和采用 PCI 接口与计算机连接满足了高速采集的要求。同时，采用市场上广泛应用的 Camera Link 作为采集卡的图像输入接口，提高了系统的通用性、传输速率和抗干扰能力，简化图像获取设备和模拟摄像头之间需要视频解码等连接。具有嵌入式处理功能，光源同步和相机控制的采集卡将使机器视觉系统，图像测速等诸多领域的图像采集应用变得更为便捷。

论文首先对图像采集卡系统的组成、整体方案和可行性进行了论证。然后给出了图像采集卡的硬件设计。在此部分结合整体设计方案，讨论芯片的选型问题。根据所选芯片的本身特点，分模块地对图像采集卡的硬件设计原理进行了详细的阐述。接下来是图像采集卡的软件设计部分。用 VHDL 和原理图结合的方法对 FPGA 进行编程，实现了图像采集系统的各个功能模块。根据图像采集系统的要求用 DriverWorks 软件设计了图像采集卡的 WDM 底层驱动程序和上层应用程序。最后是用 FPGA 实现了带修改参数的硬件嵌入式图像处理算法——图像增强。论文中使用 QUARTUS 软件嵌入的逻辑分析仪 SignalTap 对 FPGA 设计的模块进行了硬件调试，给出了调试的时序图和调试结果，经测试分析该采集卡满足“三维粒子图像测速系统”的要求，达到了预期目标。

关键词：图像采集卡；图像增强；FPGA；PCI；驱动程序

Study on FPGA-Based Embeded Frame Grabber

Abstract

The application of image acquisition and processing technology in many fields such as machine vision and image analysis is very wide. In most circumstances, frame grabber is only responsible for transmitting the image data captured by the front-end cameras to the computer.In other special circumstances, frame grabber must accurately control cameras and synchronize external light source, in order to implement image capturing, pre-processing and image data transmitting. In this way, users can, depending on the various needs, deal with and analyze the sent-back data, so as to achieve certain purposes.

According to the image processing needs of the three-dimensional particle image velocimetry system, whichi is a 2nd project of 985 SCHEME, an embedded image frame grabber, the heart of which is FPGA, is designed and developed. The logic and algorithms are realized by programming the FPGA chip, in order to achieve both the image acquisition in traditional sense and the control of CCD camera and LASER synchronization exposure function. This breaks the traditional synchronized control methods which is simply relied on increased hardware devices, correspondingly simplify the structure of the hardware system and save the total cost. In addition, the image enhancement algorithm is embedded in the system ,the requirements of high-speed acquisition is met using PCI interface, the widely used Camera-Link is adopted as the image input interface, all these methods enhances the versatility, the transmission rate and anti-jamming capabilities of the system, also simplify the connectivity between the image acquisition equipment and the analog camera. The frame grabber, which is capable of embedded processing, camera control and light source synchronization, will make the image acquisition in many areas such as machine vision systems, image velocimetry become more convenient.

First the paper not only proves the makeup and the scheme of the image acquisition system but also the feasibility of the system. Then the hardware design is given. The problem of selecting chip is referred when masterminding the architecture of the system. According to the characteristic of the chip, the card's hardware design theory is expatiated in models. The following is the part of the software of the image acquisition card. The every model of the image acquisition system is implemented by programming the FPGA in VHDL and schematic. Based on the requirements of the image acquisition system, the WDM driver is designed by DriverWorks and application program is also given. The IP core is implemented in FPGA at last. And the function of the IP care is hardware embedded graphic arithmetic--image enhance,

in which the parameter can be changed. By using the QUARTUS' logic analyzer (SignalTap), the modules for FPGA design is debugged in hardware, and the timing map and the debug results is given.After tested and analyzed, the acquisition card meet the requirements of "three-dimensional particle image velocimetry system" and the planned targets is achieved.

Key Words: Frame Grabber; Image Enhancement; Field Programable Gate Array; Peripheral Component Interconnect; Driver

独创性说明

作者郑重声明：本硕士学位论文是我个人在导师指导下进行的研究工作及取得研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写的研究成果，也不包含为获得大连理工大学或者其他单位的学位或证书所使用过的材料。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。

作者签名：彭子豪　日期：2008.1.8

大连理工大学学位论文版权使用授权书

本学位论文作者及指导教师完全了解“大连理工大学硕士、博士学位论文版权使用规定”，同意大连理工大学保留并向国家有关部门或机构送交学位论文的复印件和电子版，允许论文被查阅和借阅。本人授权大连理工大学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，也可采用影印、缩印或扫描等复制手段保存和汇编学位论文。

作者签名：

导师签名：

2008 年 1 月 8 日

1 绪论

1.1 研究背景及意义

随着计算机、多媒体和数据通信技术的高速发展，图像信号的采集与处理在科学研究、工农业生产、医疗卫生等部门得到越来越广泛的应用。同时，具有图像功能的嵌入式应用也随之增多，从数码相机、可视电话、多功能移动电话等消费产品到门禁、数字视频监视等工业控制及安防产品。图像采集需要进行同步信号的处理，比通常意义上的数据采集过程复杂，电路的设计也较为困难。传统 PC 上的图像采集卡都是在 Philips、Brooktree 等半导体公司提供的接口芯片基础上，由专业公司开发生产。近几年来，随着图像视频传输协议的不断开放，开发研究人员针对特殊应用领域的定制开发图像采集卡已变得可行。

本论文研究是以国家 985 二期项目“三维粒子图像测速系统”为科研背景，结合 FPGA 的可编程技术，实现了图像采集、激光器同步控制和图像增强于一体的嵌入式图像采集处理系统。

1.2 国内外研究现状

按相机输出信号格式划分，图像采集卡可分为模拟图像采集卡和数字图像采集卡两大类。模拟图像采集卡需要经过 A/D 转换模块把模拟信号转换为数字量然后进行传输，在一定程度上会影响图像质量。而数字图像采集卡只是把数字相机采集好的图像数据进行传输处理，对图像不会造成什么影响。模拟采集卡和模拟摄像机一般用于电视摄像和监控领域，具有通用性好、成本低的特点，但一般分辨率较低、采集速度慢，而且在图像传输中容易受到噪声干扰，导致图像质量下降，只用于对图像质量要求不高的视觉系统。为满足不同的传输距离和传输速率，先后出现了 IEEE1394，USB2.0，DCOM3，RS-644 LVDS，Camera Link 和千兆网等数字图像传输协议。其中，Camera Link 接口协议以抗干扰、传输速率、传输距离和开放性好等诸多优点，正在逐步的取代原有的一些接口成为市场的主流。

一些研发厂商使图像采集卡适用于更广的应用领域，在原有采集功能的基础上不断增加图像采集控制功能。比如 Matrox 原来的 MeteorII 采集卡，就没有外触发控制功能，使用起来非常不方便。带外触发采图控制的图像采集卡，一方面可以准确地触发相机曝光，突出兴趣图像、提高准确性，另一方面也能减少计算机资源的占用。

可以说，图像采集卡在机器视觉系统中起了一个桥梁性的作用。虽然有些数字摄像机如 USB2.0 的摄像机已经不需要特殊的图像采集卡了，但大多少专业的机器视觉系统

还是要使用专门的采集卡。国内进行图像采集卡开发的主要有大恒、嘉恒中自和微视新纪元等三家公司，国外有 matrox、EPIX 、Euresys、Coreco，I2S 等多家公司。主要产品有图像采集卡和视频压缩卡等。因此，干扰成为影像数字图像采集卡的采集图像质量的主要因素。比如资源占有的不合理产生的数据破坏，响应速率因素造成的丢帧，分辨率造成的图像产生锯齿等问题已成为图像采集系统的关键技术。

1.3 本论文主要的工作

本论文是主要是研究一个基于 FPGA 的图像采集和处理嵌入式系统。主要从以下几个方面展开研究：

(1) 系统方案的认证。在设计开发之前，对采集卡设计的各方面参数进行了研究认证，确定正确的方案实现图像采集。

(2) 硬件板卡的开发。开发设计了一款基于 FPGA 的 PCI 嵌入式插槽卡，实现了基于 CAMREA LINK 接口的数字图像采集和处理。(第三章)

(3) 数字 CCD 的图像采集控制时序的实现。结合 Camera Link 标准的特点，使用可编程器件 FPGA 设计了图像采集系统、激光器同步逻辑和 CCD 软件外触发控制逻辑。(第四章)

(4) 串口协议及相机模式的研究。针对 IMPERX 公司的 IPX-2M30-L 相机，可选择使用 FPGA 实现的 CCD 相机模式自动设置，或用 VC6.0 编程实现通过 PC 的 RS232 串口通信设置。(第四章)

(5) PCI 嵌入式系统的驱动程序的开发。采用 DriverWorks 开发实现了 PCI9054 的驱动程序，并把一些运算移置到驱动程序中实现。(第四章)

(6) 图像增强理论的研究。用 VHDL 和原理图结合的方式对 FPGA 编程，实现了一个片上集成系统。图像增强的所有参数，通过 PC 机实现实时修改。(第五章)

2 基于 FPGA 的嵌入式图像采集处理卡的总体方案设计

2.1 图像采集系统的工作原理

本采集卡的典型应用系统——二维流场图像测速[1](PIV)的示意图[2]如图 2.1 所示。系统利用外部光源(激光器)、光学系统、相机和图像处理单元(或图像采集卡)获取被测物体(流体)的图像，最后传输至计算机进行存储和分析处理。采集卡在系统中起一个桥梁的作用，负责将被测物体的可视化图像和内在特征转换成能被计算机处理的数据。为保证系统的精度和可靠性，要求采集卡有足够采集速度和良好的图像质量。

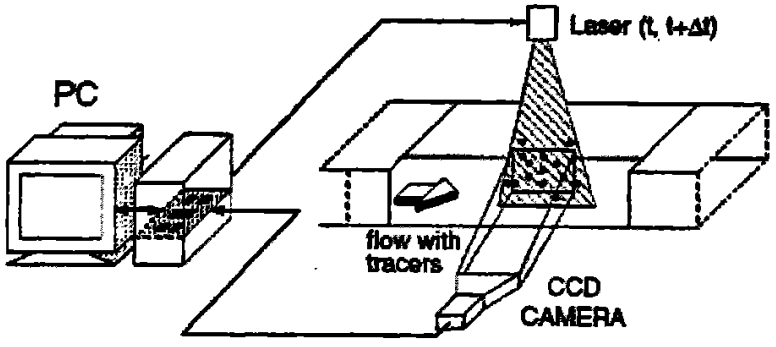

图 2.1 图像采集处理系统

Fig. 2.1 The System of Image Grabbing and Processing

2.2 图像采集卡实现方案

2.2.1 图像采集卡与计算机的接口设计方案

在设计图像采集处理卡，遇到的第一个难题就是采集处理卡与计算机的接口选择。所选接口必须满足图像数据吞吐率的要求，常用的主要有 USB、RS232、并口和 PCI 等。对于数据量吞吐大的图像卡来说，利用串行 RS-232 或标准并行口(SPP)进行数据传输显然不行，传输速度不能达到图像数据采集的要求。而 USB2.0 理论最高传输速率能达 60MBps，但实际上传输速率很难超过 10MBps。33MHz 的 32 位的 PCI 总线理论最高传输速率能达 132MBps。PCI 设备做为主控设备时，实际传输速率也能达 70MBps。能够满足图像采集应用要求，本文选择采用了高传输速率的 PCI 接口作为图像采集卡与计算机之间的通信连接接口。

2.2.2 图像采集卡输入接口的设计方案

图像采集卡需要通过图像输入接口从外部的图像输入设备中获取图像信息。通常，图像的输入接口有数字和模拟两种接口。在模拟输入时，需要先外增加 A/D 转换模块，把输入的模拟信号变成数字信号才能进行后续处理，增加了电路的复杂程度，并且在一定程度上会影响图像的质量。数字接口输入时，相机直接输出数字图像数据，这样可以有效地避免图像传输线路上的干扰。从常用的数字接口 IEEE1394，USB2.0，DCOM3，RS-644 LVDS，Camera Link 和千兆网中，本文选取 Camera Link 接口作为图像输入接口。选择时主要做了两方面的考虑，一方面综合传输距离和传输速率考虑，另一方面考虑目标 CCD 相机的支持问题。在后面的章节将陆续探讨 Camera Link 接口协议和实现方法。

2.2.3 图像数据存储的设计方案

图像信息的数据量大、实时性强，同步传输要占用大量的计算机资源，导致整个系统性能的降低和资源的极大浪费，因此图像数据的传输要采取必要缓冲措施，先采用大容量的数据缓冲器将数据缓存，之后采用高效的方式进传输，这样可以提高数据传输的效率和整个系统的性能。通常可在 FIFO 和 RAM 之间进行选择设计，由于 FIFO 没有地址线设计，在进行处理数据时，给随机读写模块存储区时带来一些麻烦。本论文采用 FPGA 片内设计两块“乒乓操作”的片内 FIFO 和片外 SDRAM 结合的方法实现图像数据缓冲。数据先经过片内 FIFO，后使高速图像数据连续存储至片外 SDRAM，避免数据的丢失。

2.2.4 核心控制器的设计方案

在图像采集时，对数据的实时性要求非常高，需要抓取图像的行场同步信号和进行图像消隐，从图像的数据流中提取图像的灰度信息。另外，需要采集卡对外部的采集元件(相机)和激光器进行曝光和触发控制，对时序要求非常严格。综上考虑，采用 FPGA 作为整个控制逻辑的核心器件。

2.3 基于 FPGA 的嵌入式图像采集处理卡的工作原理

基于 FPGA 的图像采集处理卡的嵌入式系统采用传输控制字的方式对相机进行模式设置，CCD 相机触发，激光器同步触发，图像数据的预处理和图像数据传输。整个操作过程不需主控计算机进行监控。采集卡由 PCI 总线与 PC 机进行连接，实现数据交换。通过 Camera Link 接口与相机相连，实现与相机的数据交换。本论文中的图像采集卡主要完成 CCD 相机模式设置、图像采集控制、图像采集和图像预处理等四大功能。

(1) CCD 相机模式设置：本论文基于 FPGA 的数字图像采集处理卡设计了两个 CCD 相机模式设置方案，一种是 Visual C++6.0 编程实现，通过 PC 机的 RS232 接口输出。另一种是由 FPGA 设置一个 UART 控制器，PC 只需向板卡发送相机配置信息，配置工作由板卡自动完成。

(2) 图像采集控制功能：由计算机发出图像采集命令，来控制相机的快门，同时启动外部光源（激光器），实现图像采集控制。

(3) 图像采集功能：采集卡接到采集命令，使 CCD 相机曝光成像，后通过 CAMERAL-LINK 数字接口把图像数据和同步信号送入采集卡，采集卡通过缓存至扩展 SDRAM 中。

(4) 图像预处理：图像数据采集完成后，FPGA 取出 SDRAM 中的图像数据，进行图像增强的预处理并存回 SDRAM。处理完后，采集卡上的 PCI 作为主设备传输向 PC 机请求中断，用 DMA 传输。PC 存储接收到图像数据后，可以方便地继续进行后处理。

2.4 系统可行性论证

针对图像采集系统数据量大和实时性高的特点，本文从数据传输带宽和系统的响应速度两方面评估系统的可行性。

2.4.1 数据传输带宽

本设计系统采用 IPERX 公司生产的 IPX-2M30-L 型的数字相机，分辨率为 1600×1200，帧率为 30 帧/秒的 8bits，10bits 和 12bits 可选的灰度图像。选择 8 位位宽时大概需要传输带宽为 57MBps。为了保证系统实时性和数据的完整性，在硬件系统中加了一个 4×1M×32bit 的 SDRAM 进行数据缓冲和中间数据的暂存。系统所选的 SDRAM 最大工作频率 133MHz 时，峰值传输速率达 400MBps 以上。另外，PCI9054 的传输速率可达到 132MBps。所以 PCI 总线的数据传输带宽上完全没有问题，可以满足设计要求。

2.4.2 系统响应速度

IPX-2M30-L 型的数字 CCD 相机在 40MHz 时钟上升沿同步输出 LVDS 数据。采集卡采用了独立外部时钟源 40MHz 设计和引入相机时钟设计。系统核心芯片 FPGA 在设计图像采集模块时，可以采用 CCD 输出时钟作为模块输入时钟，循环判断行场同步对应的 I/O 口，以此来保证时钟的同步性和模块的响应速度。此外，ALTERA 的 EP1C6Q 的 FPGA 支持两个 PLL，采用倍频技术可以进一步确保整个系统的响应速度。

3 基于 FPGA 的嵌入式图像采集处理卡的硬件设计

本采集卡控制系统框图如图 3.1 所示。

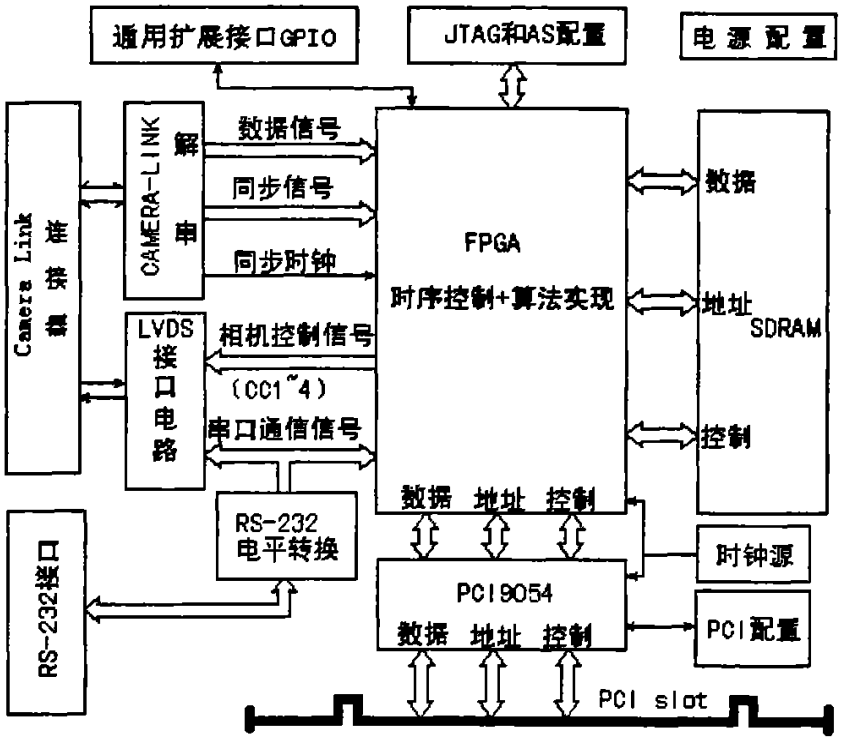

图 3.1 系统框图

Fig. 3.1 System Block Diagram

从图 3.1 可以看出，本采集卡系统是基于 PCI 的嵌入式系统，主控计算机通过 PCI 访问采集卡的资源和获取图像数据。整个采集卡以 FPGA 为核心控制芯片，通过对 FPGA 的硬件编程实现系统的时序控制和算法。

本部分主要阐述采集卡的硬件设计，文中对接口原理、芯片选择和设计思路进行了详细的分析。本采集卡主要由 FPGA 最小系统[3]，PCI 接口，CAMERA-LINK 接口，LVDS 接口，RS232 接口和 SDRAM 接口等组成。下面对每个模块一一进行阐述和分析。

3.1 FPGA 最小系统

本设计采用的是 Cyclone 系列的 FPGA，具体型号为 EP1C6Q240C8。EP1C6Q240C8 有 5980 逻辑单元(Les)，两个锁相环(PLL)，最大 186 个用户 I/O，92160 RAM 位，20 个 M4K 块。下面简单介绍一下此系列 FPGA 的一些特性。

Cyclone 系列器件[4]是基于一种全新的低成本架构。拥有嵌入式存储资源，利用 M4K 块可以实现软乘法器，以满足图像处理、音频处理和消费类电子系统的需要。软乘法器的位宽、系数位宽可以定制并且根据需要选择精度。拥有专用的外部存储器接口电路。

(1) 支持 3.3V PCI 局部总线规范 2.2 版本，支持高达 66MHz 的 32 位 PCI 总线。Cyclone 系列器件中的 I/O 单元经过专门设计，可以匹配严格的 PCI 标准所需求的建立和保持时间。为了提供最大的灵活性，每个输入信号都可以通过两个独立的延时路径输入到不同的芯片区域。

(2) 支持 SDRAM 接口，可以通过内建的专用接口与单数据率和双数据率 SDRAM 连接。

(3) 支持 10/100 及千兆以太网。用 Cyclone 系列器件实现的一台网多媒体存取控制器与物理层器件的接口速率可以达到 10Mbps、100Mbps 或 1Gbps 的最大带宽。如果结合针对 Cyclone 器件优化的 IP 核，用户可以很容易地在 Cylone 芯片上实现以太网的 MAC 功能。

(4) 支持一系列的串口总线接口，比如 I^2C，SPI，IEEE 1394，USB 2.0 等。

(5) 支持 Nios II 嵌入式处理器，而且只占用不到 600 个逻辑单元(LE)。可集成多个 NIOSII 处理器。

3.1.1 电源的设计

在高速的电路板中，由于电路的高频特性，开关的电磁辐射和线路噪声都会干扰到电路器件的电压，即器件的实际工作电压(一般要求电压偏差不超过 5%)。为了供电电压的精准和保证有足够的功率余量，减少由于电源功率不够而引起的不稳定因素，可采用线性电源模块(LDO)的方式给系统供电。其优点是电路简单，但存在一定散热问题，适合的芯片为 LT1085/6，保证最大输出电流大于 2A 即可。

整个板卡需要提供 1.5V 和 3.3V 两个电压，1.5V 给 FPGA 的内核供电，3.3V 给 FPGA 的 I/O 和 PCI9054 供电。如果采用固定输出电源芯片，只有 LT1085 等个别芯片提供 1.5V 的输出，大部分电源芯片的固定输出是 1.8V，2.5V 和 3.3V，但是都具有可调电压的型号。可调的线性电源电路连接如图 3.2 所示：

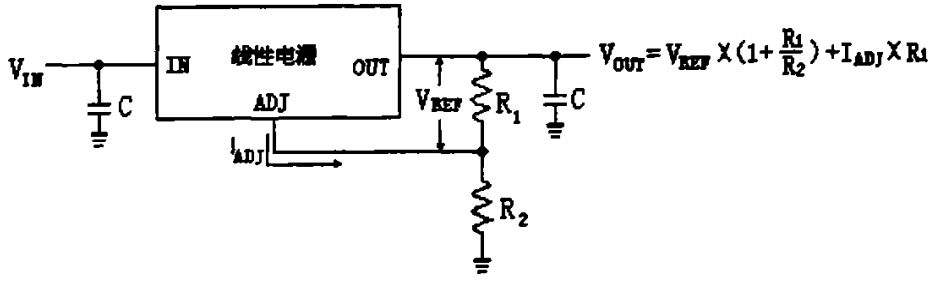

图 3.2　可调线性电源

Fig. 3.2　Adjustable Linear Power

V_{REF} 一般是 1.25V，$I_{ADJ}\times R_1$ 可以忽略。整板电源输入 V_{in} 为 5V，V_{out} 为 1.5V，那么 $R_1/R_2=1/5$，而一般要求 $R_1\in(100\Omega,150\Omega)$，那么可以选 $R_1=100\Omega$，$R_2=150\Omega$。如果采用了固定电平输出的芯片，只需要把 R_2 焊 0 欧，R_1 不焊即可。 这样简化了电源电路的设计和占地面积。基于此考虑，板上的电源部分采用了这种方式，电源设计方案[5]如图 3.3 所示。为了方便调试，在采集卡设计了两种电源的供电方案。一种是采用适配器直接供电，这种方式情况下 PCI 总线不工作，适合调试板卡时使用。另一种是采取 PCI 插槽供电方式，这种方式供电适合在系统联调时使用。但是两种方式不能同时使用。

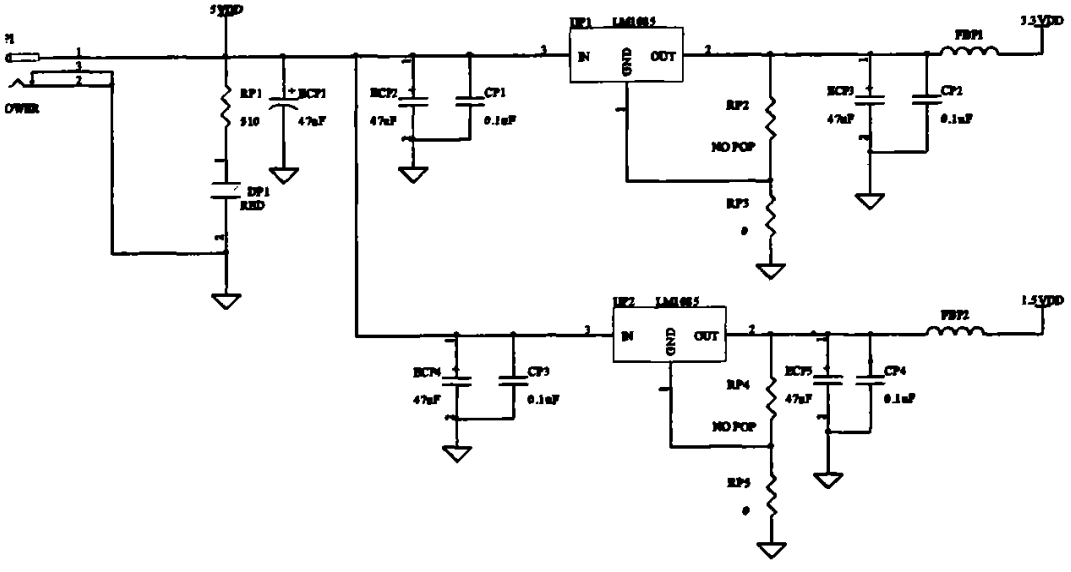

图 3.3　电源电路

Fig. 3.3　Power Circuit

3.1.2 时钟的设计

时钟可以说是一个系统的心脏，它是否稳定直接影响到系统的成败。因此，好的系统必须有一个稳定的时钟源。对于时钟源通常有无源晶体和有源晶振两种设计方法，本系统中采用外部有源晶振设计，优点是电路简单，占地小，频率范围宽和驱动能力强。时钟原理电路如图 3.4 所示。

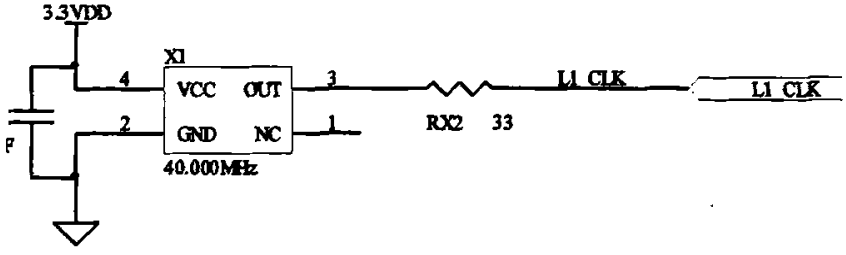

图 3.4 时钟电路

Fig. 3.4 Clock circuit

本采集卡设计了两个时钟源，一个是 40MHz,另一个是 50MHz。分别通过 PLL 资源。

3.1.3 PLL 电源设计

Altera FPGA 中的锁相环是模拟电路实现的，其对电源噪声十分敏感，所以在设计 PCB 的时候，对给 PLL 的供电部分要做一些特殊的处理。即使在设计中没有用到的 PLL，也必须对其供电。

锁相环的电源和地分别是 VCCA_PLL 和 VCCB_PLL。在给 VCCA_PLL 供电的时候，不要将其直接连到数字电源上，由于数字电源的噪声比较大，需要将 VCCA 和数字电源隔离开，防止数字电源上的噪声串入模拟电源 VCCA 而影像 PLL 稳定的工作。

要隔离开 VCCA 有几种方法，最好的方法是给模拟电源一个单独的电源平面，把所有 VCCA 管脚接到该电源平面上。不过，增加 PCB 层数会增加其成本。另一种方法是采用电源岛[6]的方式给 VCCA 供电。所谓的电源岛就是在某一个 PCB 层上单独挖出的一块模拟电源，通过磁珠(Ferrite Bead)、大电容和数字电源平面相连的方法，VCCA 管脚直接连到该模拟电源岛上。在本设计中，就采用了电源岛的设计方法。其设计电路如图 3.5 所示。

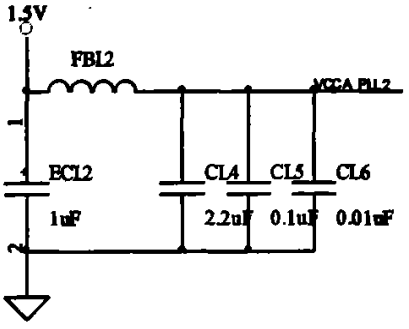

图 3.5 PLL 供电电源电路

Fig. 3.5 Power Supply Circuit of Phase Locked Logic

3.1.4 复位电路设计

为了保证系统上电能正常复位，考虑到系统电路的简单和可靠性。本系统采用 PCI 复位 RST#，同时把 RST#映射至 FPGA 的 228 号管脚。PCI 上电时，系统会产生一个复位输出至 FPGA，从而使系统复位。另外，在本设计中 PCI9054 桥芯片带有手动产生 PCI 复位信号的功能，这样在只要对 PCI9054 的寄存器进行写操作即可产生复位，给二次设计提供了很大的方便。

3.1.5 FPGA 配置接口

FPGA 的配置[7]方式可分为两大类：主动方式(active)与被动方式(passive),。为了调试方便和最终程序脱离 PC 机的运行，在该系统中采用的 AS 模式和 JTAG 口的组合方式。通过 JTAG 口可以在线配置 FPGA，和使用 SiganlTap 嵌入式逻辑分析仪调试 FPGA。通过 AS 口可以把调试完成的程序通过下载线下载到专用的配置芯片 EPCS4 中，实现 EPCS4 对 FPGA 的配置。

JTAG 接口可以用来调试 FPGA，下载速度比较快，而且支持 SignalTap。但是不能用来编程 EPCS 芯片。调试阶段采用 JTAG 模式。电缆采用 ByteBlaster(MV) 也可以用用 ByteBlaster II[8]。JTAG 配置 FPGA 的接口电路如图 3.6 所示。

AS 接口主要是用来编程 EPCS 芯片，同时也可以用来调试。具体过程是首先编程 EPCS，然后通过 EPCS 配置 FPGA，运行程序。需要考虑的是 EPCS 的编程次数是有限制的，虽然比 EPC 系列要多，但是太频繁的擦除和写入对芯片还是有一定影响的。所以，在调试结束后，程序固化的时候才使用 AS 方式。 如果采用这种方式，必须采用 ByteBlasterII 电缆才行。AS 配置 FPGA 的接口电路如图 3.7 所示。

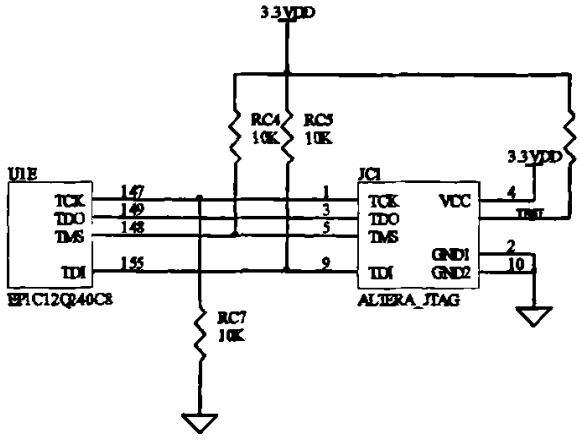

图 3.6 JTAG 配置电路

Fig. 3.6 JTAG Configuration Circuit

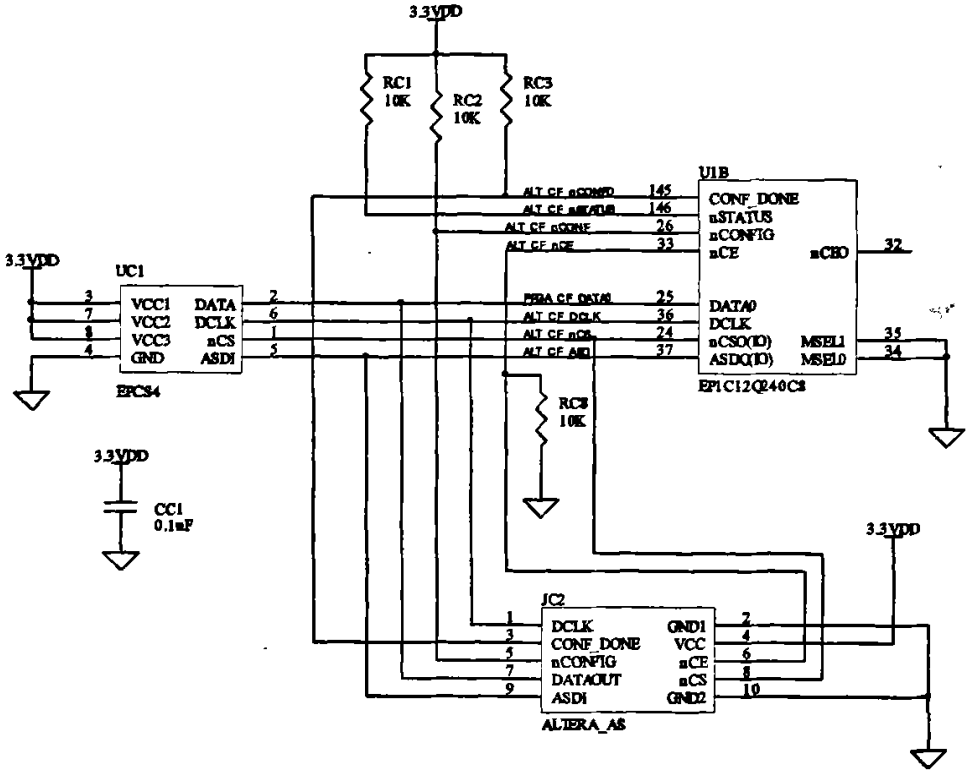

图 3.7 AS 置电路

Fig. 3.7 AS Configuration Circuit

3.2 PCI 接口

3.2.1 PCI 接口简介

1991 年下半年，Intel 公司首先提出了 PCI[9] 的概念，并联合 IBM、Compaq、AST、HP、DEC 等 100 多家公司成立了 PCI 集团，其英文全称为：Peripheral Component Interconnect Special Interest Group(外围部件互连专业组)，简称 PCISIG[10]数据宽度上看，PCI 总线有 32 位和 64 位之分，32 位 PCI 有 124 引脚，64 位有 188 引脚。目前常用的是 32 位 PCI。从总线速度上分，有 33MHz 和 66MHz 两种。目前流行的是 33MHz 的 32bit 总线，速度能达到 132MB/s，而 64bit 系统正在普及中。改良的 PCI 系统 PCI-X，最高可以达到 133MHz 的 64bit 总线，这样就可以得到超过 1GB/s 的数据传输速率。

对 PCI 设备综合考虑，32bit PCI 系统的信号按功能分类说明如表 3.1 所示。

表 3.1 PCI 主要信号说明

Tab. 3.1 Illumination of PCI main signal

管脚功能	信号名	信号作用(说明)
系统控制	CLK	PCI 时钟
	RST#	复位信号
传输控制	FRAME#	标志传输开始与结束
	IRDY#	Master 可以传输数据的标志
	DEVSEL#	当 Slave 发现自己被寻址时置低应答
	TRDY#	Slave 可以转输数据的标志
	STOP#	Slave 主动结束传输数据的信号
	IDSEL	在即插即用系统启动时用于选中板卡的信号
地址与数据总线	AD[31::0]	地址/数据分时复用总线
	C/BE#[3::0]	命令/字节使能信号
	PAR	奇偶校验信号
仲裁信号	REQ#	Master 用来请求总线使用权的信号
	GNT#	Arbiter 允许 Master 得到总线使用权的信号
错误报告	PERR#	数据奇偶校验错
	SERR#	系统奇偶校验错

在一个 PCI 应用系统中，如果某设备取得了总线控制权，就称其为“主设备”，而被主设备选中以进行通信的设备称为“从设备”或“目标设备”。对于相应的接口信号

线，通常分为必备的和可选的两大类。如果只作为目标设备，至少需要 47 条接口信号线，若作为主设备，则需要 49 条。利用这些信号线便可以处理数据、地址，实现接口控制、仲裁及系统功能。下面对主设备与目标设备综合考虑，并按功能分组将这些信号表示于图 3.8 所示[11]。

PCI 总线不同于传统的 ISA 总线，它的地址总线与数据总线是分时复用的。这样做的好处是，一方面可以节省接插件的管脚数，另一方面便于实现突发数据传输。

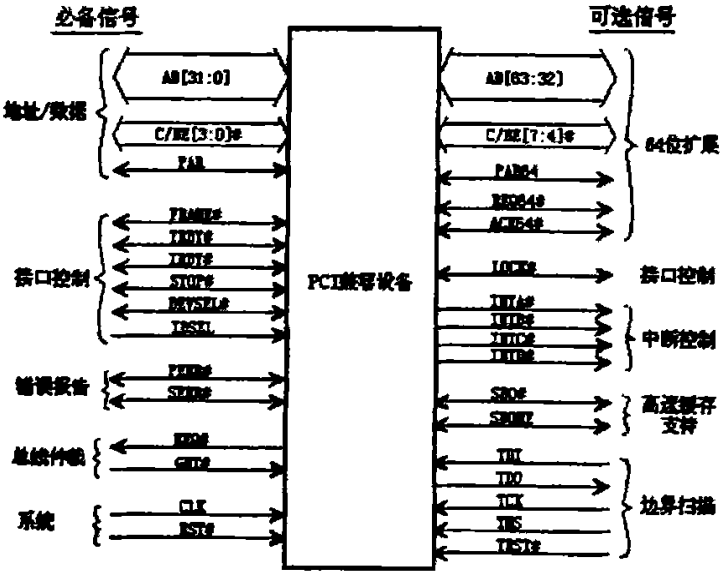

图 3.8　PCI 局部总线信号

Fig. 3.8　PCI Local Bus Signal

在做数据传输时，由一个 PCI 设备做发起者(主控，Initiator 或 Master)，而另一个 PCI 设备做目标(从设备，Target 或 Slave)。总线上的所有时序的产生与控制，都由 Master 来发起。PCI 总线在同一时刻只能供一对设备完成传输，这就要求有一个仲裁机构(Arbiter)，来决定在谁有权力拿到总线的主控权。当 PCI 总线进行操作时，发起者(Master)先置 REQ#，当得到仲裁器(Arbiter)的许可时(GNT#)，会将 FRAME#置低，并在 AD 总线上放置 Slave 地址，同时 C/BE#放置命令信号，说明接下来的传输类型。所有 PCI 总线上设备都需对此地址译码，被选中的设备要置 DEVSEL#以声明自己被选中。然后当 IRDY#与 TRDY#都置低时，可以传输数据。当 Master 数据传输结束前，将 FRAME#置高以标明只剩最后一组数据要传输，并在传完数据后放开 IRDY#以释放总线控制权。这里我们可以看出，PCI 总线的传输是很高效的，发出一组地址后，理想状态下可以连

续发数据，峰值速率为 132MB/s。实际上，目前流行的 33M 的 32bit 北桥芯片一般可以做到 100MB/s 的连续传输。

3.2.2 PCI 接口芯片的选型

PCI总线协议比较复杂，用户可以根据具体实际需求选择相应的开发方式，一般，PCI总线开发采用两种方式[12]，下面我们一一介绍一下。

一种方法是可以使用专用 PCI 接口芯片，如 PLX 公司的 PCI9052、PCI9054 等和 AMCC 公司的 S5933、S5920、S5930 等。

另一种方法是使用 CPLD 或 FPGA，通过购买 PCI 的 IP 宏来实现。不过这种方法一次性开销很大，一个 IP 一般要几千美金，并且其辅助软件工具的费用也相当昂贵。这比较适合于大批量的应用，若数量不大则最好不要采用此种方法。

基于以上考虑，我们在设计时使用专门的PCI接口芯片实现，选定为PLX公司的 PCI9054，符合设计要求，且缩短了开发周期。

3.2.3 PCI9054 介绍

PCI 9054 是PLX公司推出的一种32位33MHz的PCI 总线主控I/ O 加速器。它采用多种先进技术,可以将复杂的PCI 接口应用设计变得非常简单。利用PCI 9054 灵活的局域总线可以方便地连接多种存储器、I/ O 外围设备和CPU, 其中包括与Motorola 公司的 MPC 860、Intel 公司的960、IBM公司的PPC 401等处理器之间的直接连接。PCI9054 可广泛用于Motorola 公司的MPC 860 适配器设计、Compact PCI热交换适配器、PCI 总线主控适配器和嵌入式主机等设计系统中。PCI9054[13-14]的主要功能特点如下：

(1) 符合PCI V2. 2 规范, 是一种新型的32位33MHz总线主控接口控制器。

(2) 具有132 兆字节/ 秒的PCI 突发传输速度。

(3) 采用通用总线主控接口, 并配备了先进的数据流水线架构, 其中包含两个DMA引擎, 可编程目标和起始器、数据传输模式和PCI 信息传输功能。

(4) 与PCI V2. 2 电源管理规范兼容。

(5) 支持PCI 双地址周期(DAC)。

(6) 内含可编程中断生成器, 能进行可编程突发管理。

(7) 支持Type 0 和Type 1 配置周期。

(8) 支持与MPC 850/ 860 PowerQUICC、Intel i960和IBM PPC 401 CPU 以及类似的总线协议设备进行局域总线的直接连接。

(9) 可用3. 3V 和5V 容错的PCI 信号支持通用PCI 适配器设计。

(10) 有32 位多路复用或非多路复用局域总线,可支持8 位、16 位以及32 位外围设备和存储设备,其局域总线操作速度高达50MHz。

(11) 支持CompactPCI 热交换功能。

3.2.4 PCI9054 接口电路设计

PLX公司的PCI9054为用户的嵌入是开发提供了丰富的接口，并且3.3V/5V兼容，为开发者提供了方便。这些接口包括电源和地接口，串行EEPROM接口，PCI系统总线接口，局部总线模式和独立处理器接口，C、M和J模式局部总线接口。

(1) 电源和地接口：所有的PCI连接器都需要四条电源线：＋5V、+3.3V、+12V和－12V。其中，3.3V由需要该电压的宽展板提供，＋5V和±12V必须由系统提供。这四种电源其加电和关电顺序没有特殊的规定，可以按任意顺序进行。每当加电时，或者5V(3.3V)电源不符合标准要求时，系统会发出RST＃信号，以保障系统安全。为了得到干净的电源，在每个电源管脚应该加上一个0.1uF的退偶电容。并且采用了独立电源层和独立的地层设计，保证了整板的等电势。此外，TEST管脚,采用跳线设计，TEST为低时芯片工作在正常模式。

(2) PCI系统总线接口：PCI9054符合PCI2.2协议，提供PCI系统总线所有信号。PCI9054与PCI插槽的连接如图3.9所示。

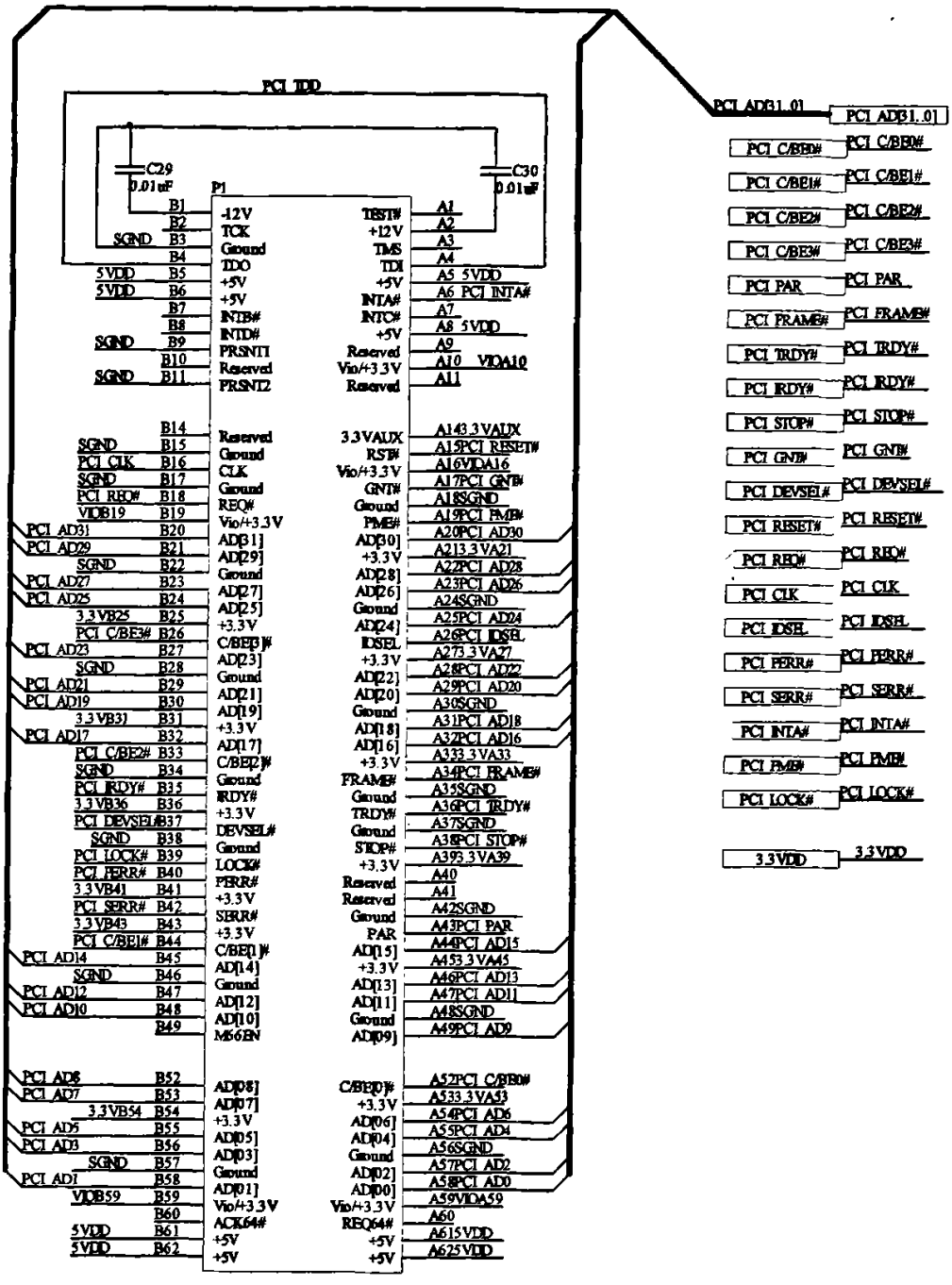

图 3.9 PCI 插槽接口电路

Fig. 3.9 Interface Circuit of PCI Slot

地址/数据复用总线：AD[31..0]。与 PCI 系统总线的 AD[31..0]直连。

系统引脚：PCLK、RST#。与 PCI 系统总线的 CLK 和 RST#直连。

控制总线：C/BE[3:0]#、PAR、FRAME#、IRDY#、TRDY#、STOP#、LOCK#、IDSEL、DEVSEL#，ENUM#。分别与 PCI 系统总线上的同名信号直连。

仲裁引脚(仅主控)：REQ#、GNT#。分别与 PCI 系统总线的 REQ#和 GNT#直连。

错误状态引脚：PERR#、SERR#。与 PCI 系统总线中的 PERR#、SERR#直连。

中断：INTA#。PCI9054 只支持 INTA#，INTB#、INTC#和 INTD#直接悬空设计即可。INTA#与 PCI 系统总线的 INTA#直连。

电源管理事件信号：PME#。与 PCI 系统总线的 PME#直连。

(3) 局部总线模式和独立处理器接口：PCI9054 可以工作在 C,M 和 J 三中模式下。C 模式下地址和数据是非复用的，开发难度较小，在本设计中采用 C 模式。通过模式选择控制引脚 MODE[1:0]进行控制。通过 BIGEDN#管脚可以设置局部总线设备读取数据字节序(Endian)的方式为 Big/Little Endian。本系统 PCI9054 模式设置如表 3.2 所示。

表 3.2 PCI9054 模式设置

Tab. 3.2 PCI9054 Mode Set

信号名称	设置值	含义
Mode[1:0]	00	C模式
	01	J模式
	10	保留
	11	M模式
BIGEND#	0	Big Endian
	1	Little Endian

C模式与大部分的高性能的处理器无缝联接，而且本地总线时序简单。在本采集卡系统使PCI9054工作在C模式。为此，设计中使MODE[1:0]管脚下拉至数字地，即使MODE[1:0]= “00”采用C模式。BIGEDN#管脚串接上拉到VCC, 即使BIGEDN# = “1”，采用Little Endian(默认)。

C模式局部总线接口：C模式下局部总线的接口信号全部与FPGA直连。PCI9054信号在FPGA中的进行信号映射。

3.2.4 PCI9054 的配置寄存器

(1) PCI9054配置接口电路。

每个PCI设备必须对每个PCI功能实现一个配置空间，每个配置空间由一组配置寄存器组成，用来识别设备、控制PCI功能等。PCI9054可以工作在默认配置模式和串行EEPROM配置模式。为了方便访问和修改配置寄存器中的参数，本文采取外接串行EEPROM的方法配置PCI9054。但是值得注意的是EEPROM型号必须被PCI9054支持。本文选择holtek公司生产的HT93LC56，配置接口电路如图3.10所示。图中EES为片选信号，EEDI/EEDO为EEPROM输入/输出信号串行数据，EESK为时钟信号。

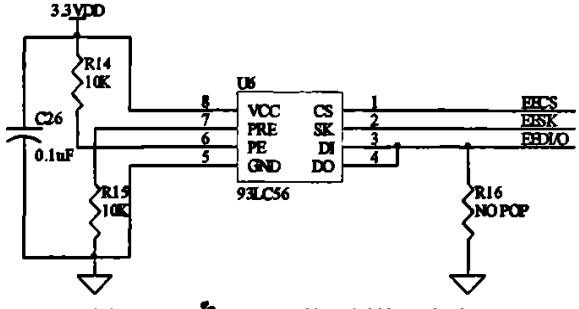

图 3. 10 PCI9054 的配置接口电路

Fig. 3.10 Configuration Interface Circuit of PCI9054

如果图3.10中R16如果焊接上，PCI9054可以按默认配置进入系统。否则必须采取外接EEPROM配置进入系统。

(2) EEPROM的烧写

烧录EEPROM有两种方式，一是比较传统的方法，即采用烧录机进行烧录。采用这种方式时，在调试过程中EEPROM需采用插件式封装的芯片(DIP)，因为每烧录一次，就需要拔出芯片到烧录机上进行烧写。这样频繁插拔，很明显会对芯片造成一定损害。当然对于最终的产品来讲，也可以采用贴片封装的EEPROM，因为在调试过程中已经把EEPROM的值调好了，把贴片的EEPROM在烧录机上烧好值再焊接在板子上。用威龙VLP系列通用编程器烧写EEPROM时界面如图3.11所示。

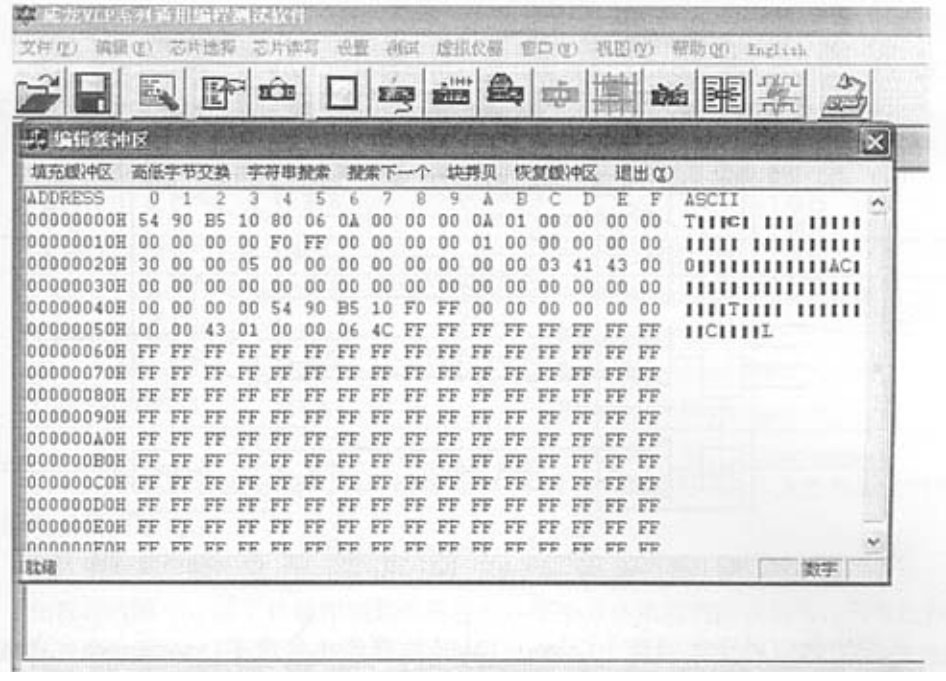

图3. 11 编程器烧写EEPROM界面

Fig. 3.11 Graphical User Interface of Programmer Writing EEPROM

二就是采用PLX公司的工具Plxmon[15-16]进行在线烧录，采用这种方式时，直接采用贴片封装的EEPROM即可。可以预先把空白的EEPROM芯片焊接在板子上，然后把PCI卡插在计算机的PCI插槽上，开机进入windows系统，运行PLXMON进行在线烧写。

3.3 Camera Link 接口

3.3.1 Camera Link 接口标准的介绍

Camera Link[17]是为视觉应用开发的通讯接口，由数家工业相机及采集卡厂商共同制定的新型接口标准。标准的本身是基于如图3.12所示Channel Link技术[18]。Channel Link是最先进的用LVDS传输数字信号的方法，弥补了传统的RS-422、RS-644等标准在传输速率和传输电缆要求等方面的不足。Camera Link接口以其高性能、低成本以及其连接的便利性等特性，迅速被多数摄像头及图像采集卡的生产商所支持，成了连接摄像头和图像采集卡的传输新标准。

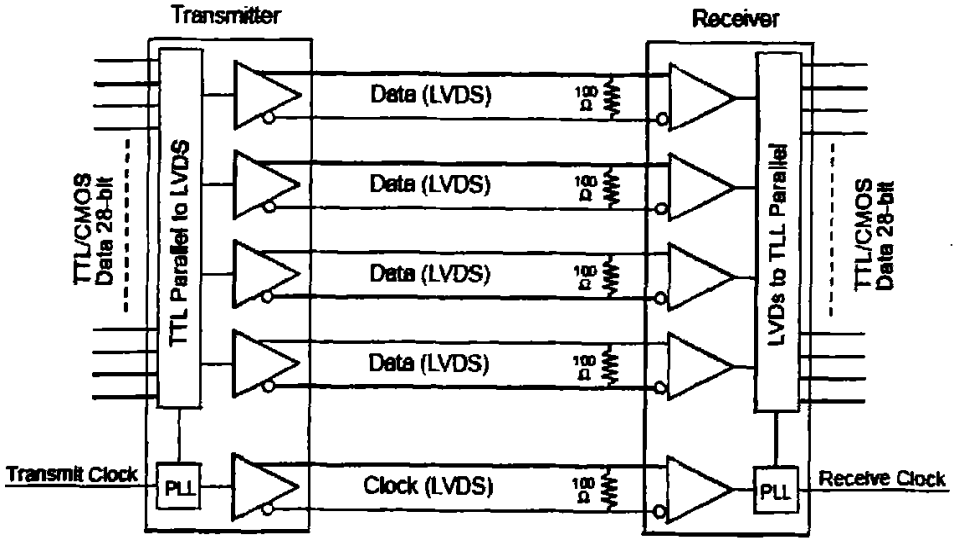

图 3.12 Channel Link 框图

Fig. 3.12 Channel Link Block Diagram

从图中我们可以看到整个Camera link传输通信由发送器(transmitter)、接收器(receiver)和LVDS传输线路共同完成。首先，发送器实现并-串转换功能，把28位CMOS/TTL 数据转化成4路LVDS数据流。经由LVDS传输线路传输至接收器。接收器实现串-并转换功能，把4路LVDS数据流转化为28位CMOS/TTL数据。传输速率高达2.38Gbps。28位数据在时钟上升沿被传输和采样。Channel Link技术建立在LVDS技术之上，采用低摆动差分电流模式驱动，能有效地减少电磁干扰(EMI)的影响。

Camera Link接口标准为不同相机厂家和用户提供了丰富的可选择的接口配置。有基本配置、中等配置和完整配置等3种配置。基本配置只有一对发送/接收器，一次能够传输和接受28位视频数据。中等配置配有两对发送/接收器，一次传输和接受56位视频数据。完整配置拥有3对发送/接收器，一次能够完成84位视频数据。

在该系统中使用基本方式，在发送端，Camera Link接口相机的发送器将前端CCD器件的28位单端数字信号和一个时钟信号，按7:1比例串行化为四个数据流和一个时钟的5对LVDS信号。在接收端，采集卡的LVDS接收器将收到的5对LVDS信号转换为28位TTL电平信号和一位时钟信号。数据在时钟上升沿有效。

Camera Link接口为相机定义了相机控制信号、视频同步信号、电源、串口通信等接口信号，供相机厂家扩展和用户二次设计。

(1) 相机控制信号

在所有配置中定义了4个LVDS的通用目的相机接口：CC1(Camera Control)，CC2，CC3和CC4作为相机的输入信号和采集卡的输出信号。它们的定义如表3.3所述。

表 3.3 相机控制信号

Tab. 3.3 Camera Control Signal

信号名称	缩写	描述
Camera Control 1	CC1	EXSYNC (external synchronization signal)
Camera Control 2	CC2	Reserved for future use
Camera Control 3	CC3	Reserved for future use
Camera Control 4	CC4	Reserved for future use

从表3.3中可以看出，只有CC1可被使用，在本设计中则采用CC1作为外部触发信号，来触发相机的曝光。

(2) 视频同步信号

在视频传输中，除了传输视频数据信号外，还必须传输视频同步信号，只有这样，在接收视频数据的时候才不会发生错误。Camera-Link 协议定义如表3.4所述的像素限制信号作为相机和外部的同步信号。在两个有效行之间，LVAL会跳变几个无效的像素点。跳过的像素点的数量由相机的型号和预触发设定。对于有效的像素这个信号被定义为高。这部分将在后面的相机部分会做详细的探讨。

表 3.4 像素限止信号

Tab. 3.4 Pixel Qualifier Signal

信号名称	缩写	信号描述
帧同步信号	FVAL	当FVAL为高时表示相机正输出一帧有效数据
行同步信号	LVAL	当FvAL为高时，LVAL为高表示相机正输出一有效的象素行
数据有效信号	DVAL	当FvAL为高并且LVAL为高时，DVAL为高表示相机正输出
预留	Spare	这个信号被保留作为将来的使用

(3) 配置电源和串口通信

Camera-link相机电源单独由一独立适配器提供。由两个RS-644 LVDS对SerTFG和SerTC实现相机和采集卡之间异步串口通信。其中SerTFG是从相机到采集卡的LVDS对，SerTC是从采集卡到相机的LVDS对。串口通信的数据格式必须为1个起始位，1个停止位，无奇偶检验，无握手。

3.3.2 LVDS技术的介绍

(1) LVDS技术简介

LVDS(Low Voltage Differential Signal 即低电压差分信号) 接口[19]又称RS644总线接口，是20世纪90年代才出现的一种数据传输和接口技术. 是一种低摆幅的差分信号技术，它使得信号能在差分PCB线对或平衡电缆上以几百Mbps的速率传输，其低压幅和低电流驱动输出实现了低噪声和低功耗。IEEE在两个标准中对LVDS信号进行了定义，ANSI-644，TIA-644和EIA-644推荐最大速率为655$Mbps$，理论极限速率为1.923$Mbps$。LVDS接收器具有很高的输入阻抗，在设计时候，往往将驱动器的大部分输出电流流过100 Ω的匹配电阻，使在接收器的输入端产生大约350 mV的电压。当驱动器翻转时，它改变流经电阻的电流方向，因此产生有效的逻辑"1"和逻辑"0"状态。当驱动状态反转时，流经电阻的电流方向改变，于是在接收端产生一个有效"0"或"1"的逻辑状态。LVDS驱动器由一个恒定电流源(通常为$3.5mA$)驱动一对差分信号线组成，LVDS接收器有很高的DC阻抗，几乎不会消耗电流，与传输线阻抗匹配的终端电阻(约为100欧)跨接在两条差分信号线上，并尽可能靠近接收器输入端，绝大部分的驱动电流将流经100欧的终端电阻，并在接收器输入端产生大约350mV的压降。

(2) LVDS信号电平和抗噪特性

LVDS物理接口使用1.2V偏置电压作为基准，提供大约400mV摆幅。LVDS驱动器由一个驱动差分线对的电流源组成(通常电流为3.5mA)，LVDS接收器具有很高的输入阻抗，因此驱动器输出的电流大部分都流过100欧姆的匹配电阻，并在接收器的输入端产生大约350mV的电压。

电流源为恒流特性，终端电阻在100至120欧姆之间，则电压摆动幅度为：$3.5mA\times100\Omega=350mV$；$3.5mA\times120\Omega=420mV$。

由于 LVDS信号物理电平变化在$0.85-1.55V$之间，其由逻辑"0"电平到逻辑"1"电平变化的时间比TTL电平要快得多，所以LVDS更适合用来传输高速变化信号。因为其低压特点，所以其功耗也低。

从差分信号传输线路上可以看出，若是理想状况，线路没有干扰时，在发送侧，可以形象理解为：

$$IN=IN^{+}-IN^{-} \tag{3.1}$$

在接收侧，可以理解为：

$$IN^{+}-IN^{-}=OUT \tag{3.2}$$

所以：

$$IN = OUT \tag{3.3}$$

在实际线路传输中，线路存在干扰，并且同时出现在差分线对上，在发送侧，仍然是：

$$IN = IN^{+} - IN^{-} \tag{3.4}$$

线路传输干扰同时存在于差分对上，假设干扰为 q，则接收则：

$$(IN^{+} + q) - (IN^{-} + q) = IN^{+} - IN^{-} = OUT \tag{3.5}$$

可见仍然有(3.3)式成立。所以噪声被抑制掉。上述可以形象理解差分方式抑制噪声的能力。在实际芯片中，是在噪声容限内，采用“比较”及“量化”来处理的。

3.3.3 Camera Link 接口芯片的选型

Camera Link 标准指定视频数据传输由美国国家仪器有限公司(National instruments)的 28 路 Channel Chip。与 Camrea Link 兼容的 Channel Chip 接收芯片有 DS90CR282，DS90CR284，DS90CR286，DS90CR286A，DS90CR288，DS90CR288A。Channel Chip 的驱动芯片集成在相机内，在设计时，只需考虑接收芯片的选择。根据其供电电源和工作频率，选择 DS90CR282 作为本采集卡系统的接收芯片，选择 DS90LV047 和 DS90LV048 作为控制信号的电平转换芯片。

3.3.4 Camera Link 接口电路设计

整个 Camera Link 接口电路负责是接收 IPX-2M30-L 相机输出的 Camera Link LVDS 信号，并将其转化为 FPGA 芯片所能处理的 TTL 电平的控制信号或并行的图像数据流。相机采用标准 MCR-26 连接器输出。整个接口电路由两个部分组成，一部分是用 Channel Chip 芯片 DS90CR282 对串行的 LVDS 信号进行解串操作，另一部分就是使用 DS90LV47(8)进行电平转换。

(1) 具体接口电路。

Camera Link 接口电路如图 3.13 所示。应相机厂商要求，Inner Shield(屏蔽信号)通过一个零值电阻与采集卡的系统地相连。所有输入 LVDS 信号对的正负端串接一个 100 欧姆的电阻。用三个容值为 0.1uF，0.01uF 和 0.001uF 电容并联对 DS90CR282 芯片[20]的输入电源端进行退偶。整个接口设计服从 Camera Link 接口协议，相机输出信号由 4 对 LVDS 通道输出(RxIN0,RxIN1,RxIN2,RxIN3)，1 对同步时钟，4 对相机控制信号(CC1,CC2,CC3,CC4)和负责串行通信的 SerTFG 和 SerTC 共同组成。

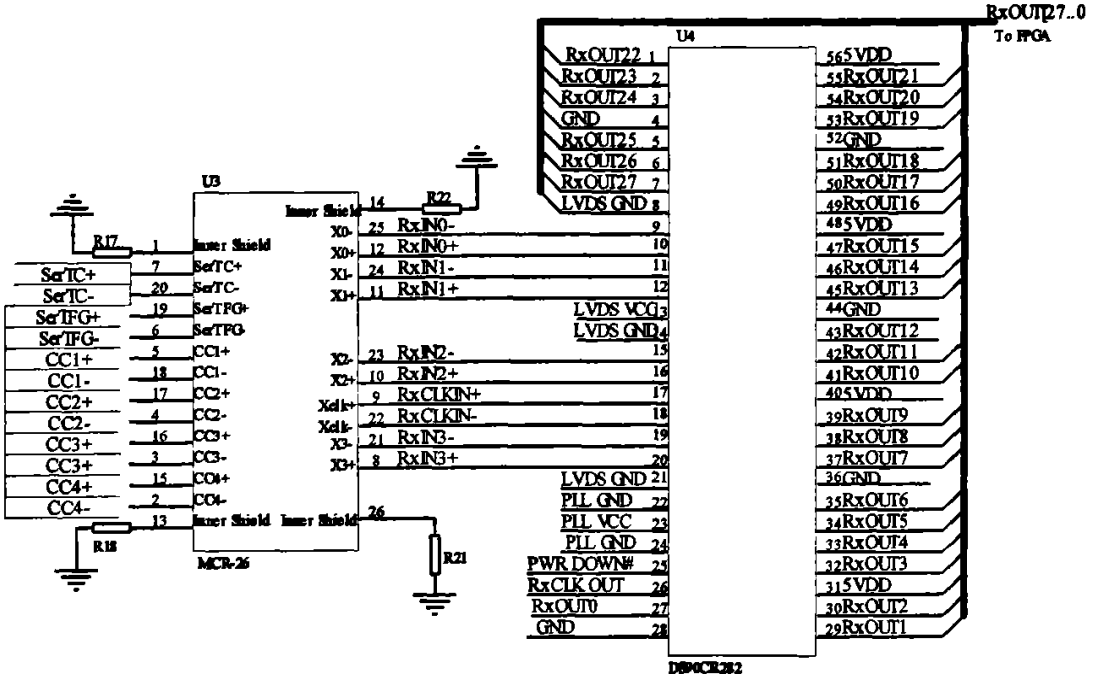

图 3.13 Camera Link 接口电路

Fig. 3.13 Camera Link Interface Circuit

(2) 接口电路设计思想

芯片 DS90CR282 接收四路 LVDS 通道信号和时钟信号 Rxclk，将其转化为 RxOUT[27..0]的 TTL 图像信号和一路时钟输出，作为 FPGA 的 I/O 信号。FPGA 通过时钟信号和输入的 I/O 信号，进行帧头和图像数据的分析，具体的过程在后面的软件部分进行详细阐述。具体的 28 位数据的分配表如表 3.5 所示，包括 24 位数据，3 个同步信号(LVAL, FVAL and DVAL)和一个 NC 信号[21]。芯片 DS90LV047 和 DS90LV048 将信号 CC1，SerTC 和 SerTFG 做电平转换，实现相机的外部触发控制和串口通信。在本设计中，采用 SerTC 和 SerTFG，增加了 RS232 接口与相机直接进行通信的设计，具体实现在后面章节中的串口电路部分和 UART 控制器部分将继续阐述。

表 3.5 基本配置的位分配表

Tab. 3.5 The Bit Assignment Corresponding to the Base Configuration

Bit	Port	8 Bit	10 Bit	12 Bit
Bit[0]	Port A0	A0	A0	A0
Bit[1]	Port A1	A1	A1	A1
Bit[2]	Port A2	A2	A2	A2
Bit[3]	Port A3	A3	A3	A3
Bit[4]	Port A4	A4	A4	A4
Bit[5]	Port A5	A5	A5	A5
Bit[6]	Port A6	A6	A6	A6
Bit[7]	Port A7	A7	A7	A7
Bit[8]	Port B0	B0	A8	A8
Bit[9]	Port B1	B1	A9	A9
Bit[10]	Port B2	B2	N/C	A10
Bit[11]	Port B3	B3	N/C	A11
Bit[12]	Port B4	B4	B8	B8
Bit[13]	Port B5	B5	B9	B9
Bit[14]	Port B6	B6	N/C	B10
Bit[15]	Port B7	B7	N/C	B11
Bit[16]	Port C0	N/C	B0	B0
Bit[17]	Port C1	N/C	B1	B1
Bit[18]	Port C2	N/C	B2	B2
Bit[19]	Port C3	N/C	B3	B3
Bit[20]	Port C4	N/C	B4	B4
Bit[21]	Port C5	N/C	B5	B5
Bit[22]	Port C6	N/C	B6	B6
Bit[23]	Port C7	N/C	B7	B7
Bit[24]	LVAL	LVAL	LVAL	LVAL
Bit[25]	FVAL	FVAL	FVAL	FVAL
Bit[26]	DVAL	DVAL	DVAL	DVAL
Bit[27]	SPARE (N/C)	N/C	N/C	N/C

3.4 RS232 串口

3.4.1 RS232 串口的介绍

RS-232-C[22]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。

串口通信的概念非常简单，串口按位(bit)发送和接收字节。尽管比按字节(byte)的并行通信慢，但是串口可以在使用一根线发送数据的同时用另一根线接收数据。它很简单并且能够实现远距离通信。比如 IEEE488 定义并行通行状态时，规定设备线总常不得超过 20 米，并且任意两个设备间的长度不得超过 2 米；而对于串口而言，长度可达 1200 米。典型地，串口用于 ASCII 码字符的传输。通信使用 3 根线完成：地线，发送，接收。由于串口通信是异步的，端口能够在一根线上发送数据同时在另一根线上接收数据。其他线用于握手，但是不是必须的。串口通信最重要的参数是波特率、数据位、停止位和奇偶校验。

(1) 波特率：这是一个衡量通信速度的参数。它表示每秒钟传送的位数(bit)。例如 300 波特表示每秒钟发送 300 位(bit)。通常意义的时钟周期就是指波特率，例如如果协议需要 4800 波特，那么时钟是 4800Hz。这意味着串口通信在数据线上的采样率为 4800Hz。通常电话线的波特率为 14400，28800 和 36600。波特率可以远远大于这些值，但是波特率和距离成反比。高波特率常常用于放置的很近的仪器间的通信，典型的例子就是 GPIB 设备的通信。

(2) 数据位：这是衡量通信中实际数据位的参数。当计算机发送一个信息包，实际的数据不会是 8 位的，标准的值是 5、7 和 8 位。如何设置取决于你想传送的信息。比如，标准的 ASCII 码是 0～127(7 位)。扩展的 ASCII 码是 0～255(8 位)。如果数据使用简单的文本(标准 ASCII 码)，那么每个数据包使用 7 位数据。每个包是指一个字节，包括开始/停止位，数据位和奇偶校验位。由于实际数据位取决于通信协议的选取，术语“包”指任何通信的情况。

(3) 停止位：用于表示单个包的最后一位。典型的值为 1，1.5 和 2 位。由于数据是在传输线上定时的，并且每一个设备有其自己的时钟，很可能在通信中两台设备间出现了小小的不同步。因此停止位不仅仅是表示传输的结束，并且提供计算机校正时钟同步

的机会。适用于停止位的位数越多，不同时钟同步的容忍程度越大，但是数据传输率同时也越慢。

(4) 奇偶校验位：在串口通信中一种简单的检错方式。有四种检错方式：偶、奇、高和低。当然没有校验位也是可以的。对于偶和奇校验的情况，串口会设置校验位(数据位后面的一位)，用一个值确保传输的数据有偶个或者奇个逻辑高位。例如，如果数据是 011，那么对于偶校验，校验位为 0，保证逻辑高的位数是偶数个。如果是奇校验，校验位位 1，这样就有 3 个逻辑高位。高位和低位不真正的检查数据，简单置位逻辑高或者逻辑低校验。这样使得接收设备能够知道一个位的状态，有机会判断是否有噪声干扰了通信或者是否传输和接收数据是否不同步。

3.4.2 RS232 串口的电平转换芯片的选型

RS-232 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，否则将使 TTL 电路烧坏，实际应用时必须注意。对于两个进行通行的端口，这些参数必须匹配 FPGA 的所有接口是 TTL(LVDS)，所以需要将 RS232 电平转化为 TTL 电平。本文采用应用较为广泛的电平转换芯片 MAX232。

3.4.3 RS232 串口通信接口电路设计

IPX-2M30-L 相机支持 Camera Link 标准中的串口通信 LVDS 信号对 SerTC 和 SerTFC。设计考虑，由通过计算机的 RS232 串口与相机进行通信，实现相机模式的设置。为此，我们先把 LVDS 信号转化为 TTL 信号，作为 FPGA 的输入或者再由 TTL 电电平信号转化为 RS232 电平，与计算机的串口相连。

选择 National Semiconductor 公司的 DS90LV048/47[24-25]作为 LVDS 与 TTL 的电平转换芯片。选择德州仪器公司的 MAX232 芯片[25]作为 RS232 和 TTL 电平之间的转换芯片。串口通信接口电路如图 3.14 所示。

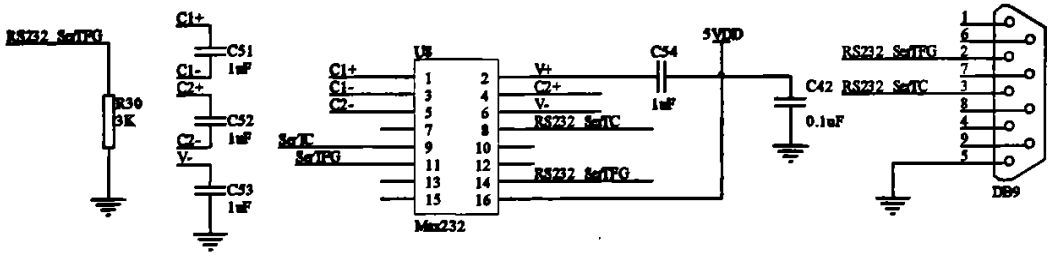

图 3.14　串口通信接口电路

Fig. 3.14　Interface Circuit of Serial Port

3.5　扩展存储器设计

在本系统中，由于 Camera Link 相机采集回来的数据量非常大，如果光靠 FPGA 的片内存储是不能满足要求的，为了使系统及时地接收和处理数据，采用外部扩展存储器设计。同步动态随机存储器(SDRM)具有高速、大容量、价格低廉等优点，因而成为缓冲存储器的首选。

3.5.1　寄存器的选型

在本系统中选用了HYLIX公司的HY57V283220T的SDRAM，存储容量为4BANK×1Mbit×32。

HY57V283220T型SDRAM具有如下特点[26]：

(1) 3.3V供电，所有的数据输入输出均在输入时钟的上升沿采样。

(2) 支持自动刷新和自刷新功能(Auto refresh and self refresh)。

(3) 刷新周期要求4096 refresh cycles / 64ms。

(4) 可编程的突发长度和突发类型，或者全页模式读写。

(5) 可编程的CAS延迟(2或3个时钟周期)。

(6) 支持突发读—突发/非突发写操作。

3.5.2　SDRAM 接口电路设计

SDRAM 原理图如 3.15 所示

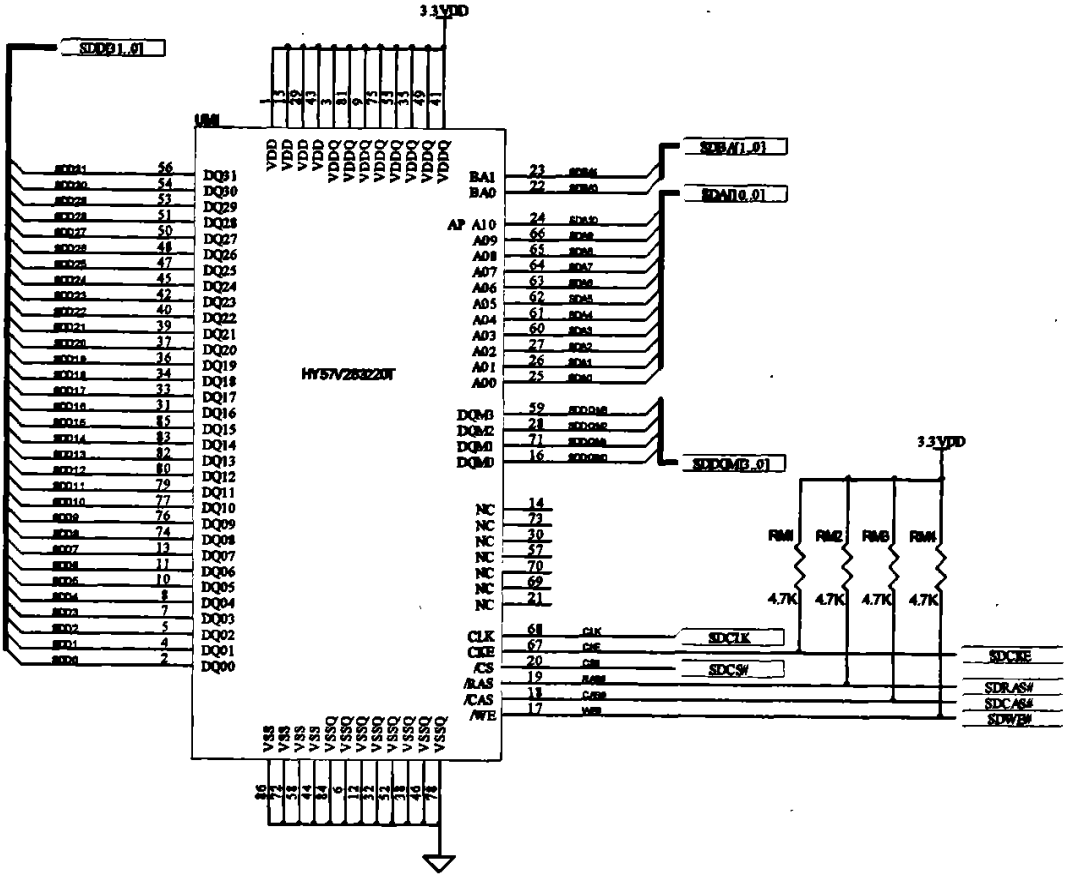

图3.15 SDRAM原理图

Fig. 3.15 SDRAM Schematic Diagrams

SDRAM 的所有控制信号和数据地址信号全部映射到 FPGA 管脚上。为了保证在 FPGA 不访问 SDRAM 时，SDRAM 的状态稳定，需对 SDRAM 的状态信号 WR#，CAS# 和 RAS#进行上拉处理。

3.6 电路板的设计和实现

基于 FPGA 的数字图像采集处理卡是一个 PCI 插槽卡，系统时钟高达 40MHz，属于高速板。速度问题成为整板设计成败的关键因素。在本部分主要针对高速板设计和 PCI 局部总线规范进行分析。

3.6.1 PCI 板连接器的设计

PCI 局部总线定义了两种扩展板连接器，一种是 5V 信号环境的，另一种是 3.3V 信号环境。同时还定义了三种电气类型的扩展板，分别是 5V 板，3.3V 板和通用板(即可用于 5V 环境又可用于 3.3V 环境)。在 PCI 连接器和板上的金手指都设计有相应的键位以防止将一个板插入不适当的位置。PCI 局部总线的电气规范中提供了 5V 和 3.3V 两种信号环境，两者不能混合使用，但是，通过设计可使 5V 的元件工作于 3.3V 的信号环境，反之亦然。也就是说，对于一个给定的 PCI 局部总线，所有的元件必须使用同一个信号规则，元件可以混合使用，但信号环境必须是 5V 或者 3.3V 中的一个。通用扩展板能够检测出当前所用的信号环境属于哪一种，并能够自适应该环境。基于信号协议兼容性的考虑，本系统采用通用扩展板设计。值得注意的是，各类型板子之间的区别在于它们所采用的信号协议，而不是电源线的连接。

3.6.2 电路板的技术指标

母板上 PCI 局部总线的控制信号都要求有上拉电阻，这是为了保证他们在没有设备驱动总线的情况下仍具有稳定的值。此类信号有 FRAME#、TRDY#、IRDY#、DEVSEL#、STOP#、SERR#、PERR#、LOCK#、INTA#、REQ64#和 ACK64#。点到点和 32 位共享信号不需要上拉电阻，总线停靠即可保证它们的稳定。最大上拉电阻和最小上拉电阻的公式如下[27]：

$$R_{max}=\frac{V_{cc}(\min)-V_x}{\text{负载数}\times I_{il}} \tag{3.6}$$

$$R_{min}=\frac{V_{cc}(\max)-V_{ol}}{I_{ol}+16\times I_{il}} \tag{3.7}$$

其中，16 为最大负载数；V_x 在 5V 信号环境下为 2.7V,而在 3.3V 信号环境下为 $0.7V_{cc}$；V_{ol}，I_{ol}，I_{il} 为 5V/3.3V 信号环境下的直流或交流指标。计算出在 3.3V 信号环境下，上拉电阻的最小值 $2.42k\Omega$，典型值 $8.2k\Omega$。在本设计中选择上拉电阻 $4.7k\Omega/5k\Omega$。

3.6.2 PCB 的设计

整个设计在 PROTEL 99SE 环境[28-29]下完成。一般电路板设计可分为以下几个部分：

(1) 设计原理图。本论文采用模块分层设计，首先设计顶层原理图文件，而后逐个实现。

(2) 生成网格表。有顶层原理图生成与顶层原路图名相同的网格表文件。在 PCB 板设计工程中，选择导入设计的网格表文件即可。

(3) 电路板布局。由于本采集卡是 PCI 插槽卡，所以 PCI 的金手指之间的间距和机械固定件之间的间距要求非常严格，设计不合理将导致整板的失败。设计此类插槽卡可以通过 PROTEL 99SE 的印刷电路板生成向导(printed circuit board wizard)中的 IBM&APPLE PCI bus format 向导选择 PCI short card universal /32bit 生成四层(2 个平面层)PCI 插槽卡。当然完全可以用手工代替印刷板向导的工作，只不过设计起来比较繁琐而已。由于整板电源网络有 5V，3.3V 和 1.5V，可以用“分裂的电源层”技术来实现。接下来的工作就是往电路板上放置元件了。一般以美观和布线方便为准则进行放置。

(4) 电路板走线。本设计的为四层板，采用全手工布线方式。采用优先级布线原则。先对要求高的信号线进行走线，而后布要求较低的，最后次之。PCI 卡的 PCB 布局布线要求非常严格，所有 32 位接口信号的最大走线长度为 38.1mm，时钟走线长度有严格的要求，长度为 $63.5mm \pm 2.5mm$,而且只能连到一个负载上[30]。

(5) DRC 设计校验。为了保证设计的正确无误，都要进行这一步骤。

(6) 制作电路板。

(7) 电路板调试。一定要自己先定个调试计划，以至于每个工序都按步骤来完成。一般先应该对生产出的目标板进行整体的检测。看其是否全按自己预期的要求实现，之后再去焊接元器件。完整焊接完毕后，电源是否与地之间短接，对地电阻多大，时钟源等等一项项的检测。最后上电检测，这应该是一个反复循环的过程。

4 基于 FPGA 的嵌入式图像采集处理卡的软件设计

4.1 基于 FPGA 的嵌入式图像采集处理卡的驱动程序设计

在 Windows 操作系统中，为了保证系统的安全性和可移植性，对应用程序对硬件的操作进行了限制，尤其 Windows 2000 和 Windows XP，不支持直接对系统的硬件资源的操作。在设计开发 PCI 设备时，需要开发人员开发相应的驱动程序来实现对 PCI 设备的操作，用户应用程序通过驱动程序来访问 PCI 设备。在 Windows XP 和 Windows 2000 中，设备的驱动程序必须根据 Windows 驱动程序模型(WDM)设计。

4.1.1 WDM 简介

Windows 驱动程序模型[31](windows driver model)是 Microsoft 公司力推的全新驱动程序模式，与以往的驱动程序模型比较，它支持即插即用(Plug and Play,PnP)、电源管理和 Windows 管理诊断(Windows Management Instrument,WMI)等技术。在 Windows 操作平台上，WDM 已成为主流的驱动模型。驱动程序采用的是如图 4.1 所示的分层结构模型。

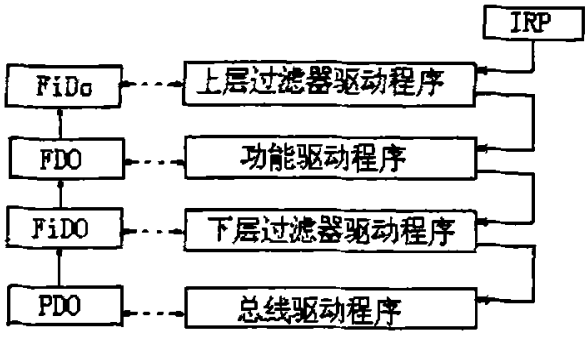

图 4.1 WDM 驱动程序层次结构

Fig. 4.1 WDM Driver Hiberarchy

图中左边是一个设备对象堆栈，设备对象是操作系统为帮助软件管理硬件而创建的数据结构。处于堆栈最底层的设备对象称为物理设备对象(Physical Device Object)，或简称为 PDO。在设备对象中间有一个对象称为功能设备对象(Function Device Object)，或简称为 FDO。在 FDO 的上面或者下面可能存在一些过滤器设备对象(Filter Device Object)。

操作系统的 PnP 管理器按照设备驱动程序的要求构造设备对象堆栈。总线驱动程序的一项任务就是枚举总线上的设备，并为每个设备创建一个 PDO。一旦总线驱动程序检查到新硬件的存在，PnP 管理器就创建以个 PDO，之后便开始描绘如图 4.1 所示的结构。在 WDM(Windows Driver Model)中，每个硬件设备至少有两个驱动程序。其中一个驱动程序被称为功能(function)驱动程，就是通常意义上的硬件设备驱动程序。它了解硬件工作的所有细节，负责初始化 I/O 操作，有责任处理 I/O 操作完成时所带来的中断事件，有责任为用户提供一种设备适合的控制方式。

另一个驱动程序称为总线(Bus)驱动程序。它负责管理硬件与计算机的连接。例如 PCI 总线插入到 PCI 槽上的设备并确定设备的资源使用情况。它还能控制设备所在的 PCI 槽的电流开关。总线(Bus)驱动程序是指 PCI 总线的 PDO 驱动，由 WINDOWS 提供，不需要我们去设计。在工作过程中，总线驱动将扫描 PCI 插槽，并枚举它们，负责管理 PCI 总线的物理信号。所以，我们所指的驱动程序设计，实质上是功能驱动程序(function driver)。一个完整的驱动程序包含许多例程，当操作遇到一个 IRP 时，它就调用驱动程序中的例程来执行该 IRP 的各种操作。有些例程则是必须的，比如 DriverEntry 和 AddDevice,还有 DispatchPnP,DispatchPower 和 DispatchWMI 派遣函数。

4.1.2 驱动程序开发工具的选择

开发 WDM 驱动程序的常用开发工具主要有：微软提供的驱动开发包 DDK(Driver development kit),Jungo 公司的 Windriver，NuMega 公司的 DriverStudio。

(1) DDK

Device Development Kit(DDK)[32-33]编写的驱动程序时，直接使用 DDK 开发设备驱动程序，开发难度非常大，开发周期比较长，有很多繁琐的工作需要做。

(2) Windriver

WinDriver[34-35] 是 Jungo 公司开发的用于编写设备驱动程序的开发工具。在 Windows 环境下,因为 WinDriver 不需要设计人员了解操作系统的专业知识，而且整个开发过程的编码和调试都可以在用户模式下进行。使用 Windriver 开发驱动程序，开发十分简单。采用 WinDriver 工具开发完的驱动程序是通过 WinDriver 内核模块(Windrvr.SYS)来实现对硬件的具体操作，使得驱动的开发变成了一个通用驱动程序的定制和调用的过程。在执行的效率和移置不是很方便，工作效率不是很高。

(3) DriverStudio

DriverStudio 是一套为简化 Widows 设备驱动程序和应用程序而开发的开发、编译和调试的软件工具包。其中的 DriverWorks 是建立 DDK 的基础之上，以面向对象(OOP)

的方式，将 WDM 和 NT 下的驱动程序编写所需的于内核访问及对硬件的访问的 DDK 的函数封装成了类，因而不会影响整个驱动程序的执行速度和效率。DriverWorks 最大的优点是用户可以通过驱动生成向导——Driver Wizards 生成驱动框架，大大简化了驱动程序的开发难度，减少了工作量。用户只需在生成的框架下做相应的更改和功能添加，来满足自己需求即可。

(4) 驱动程序开发工具的选择

经上述阐述比较，本论文中驱动程序开发采用 DriverStudio 的软件中的 DriverWorks 来开发采集处理卡的 WDM 驱动程序[36]。

4.1.3 驱动程序的设计与实现

开发 PCI 设备功能驱动程序 WDM,主要有三个方面的问题：硬件访问、中断处理和 DMA 传输。

硬件访问：X86 处理器有两种独立的地址空间，分别是 I/O 地址和内存地址。其中 I/O 地址只有 64KB，而内存空间现在达到了 4GB。对于微机处理卡，其中的 I/O 和存储器芯片可以定位于这两种独立的地址空间中。

中断处理：使用类 KInterrupt 进行操作。分为中断的初始化，将中断服务例程连接到一个中断、中断延迟处理和解除其连接等过程。

DMA 传输：PCI9054 提供了两个独立的可编程的 DMA 通道，通道 0 和通道 1。在驱动程序开发时，DriverWorks 提供了三个类：KDmaAdapter，KDmaTransfer，KCommonDmaBuffer 类，用来实现 DMA 操作。KDmaAdapter 类用于建立一个 DMA 适配器，它说明 DMA 通道的特性。KDmaTransfer 类用于 DMA 传输控制。KCommonDmaBuffe 类用于申请系统提供的公共缓冲区。

(1) 驱动程序框架的生成

利用 DriverStudio 软件中的 VC 驱动生成向导 Driver Wizards 为 PCI9054 的生成 WDM 驱动程序框架按顺序主要有如下几个步骤：

① 创建一个 visual studio 支持的驱动程序工程。

② 选择创建驱动程序的模型，本设计选择 WDM Driver。

③ 选择驱动程序类型，选择 WDM 功能驱动(WDM function Driver)。

④ 选择设备总线类型和填写设备标识。本选择为 PCI，PCI vendor ID＝10B5，PCI device ID＝9054。界面如图 4.2 所示。

⑤ 设置驱动程序文件类名和文件名，通常不需更改，取默认值即可。

⑥ 选择驱动程序处理请求类型，本设计中做如图 4.3 所示的选择。处理应用程序读/写、I/O 控制和 cleanup 请求。

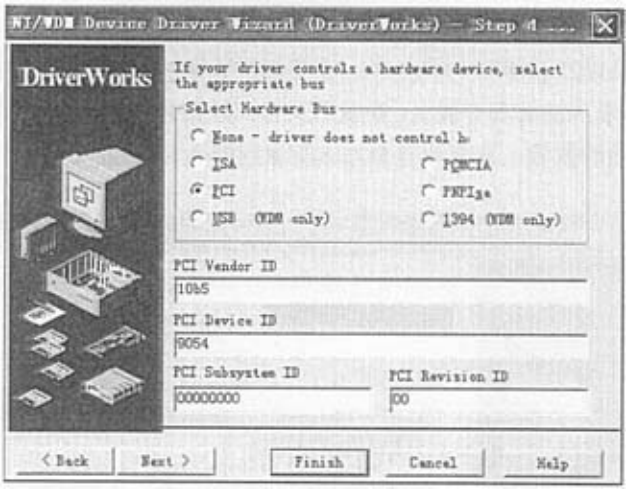

图 4.2 总线类型选择

Fig. 4.2 Selection of the Appropriate Bus in Drivers Model

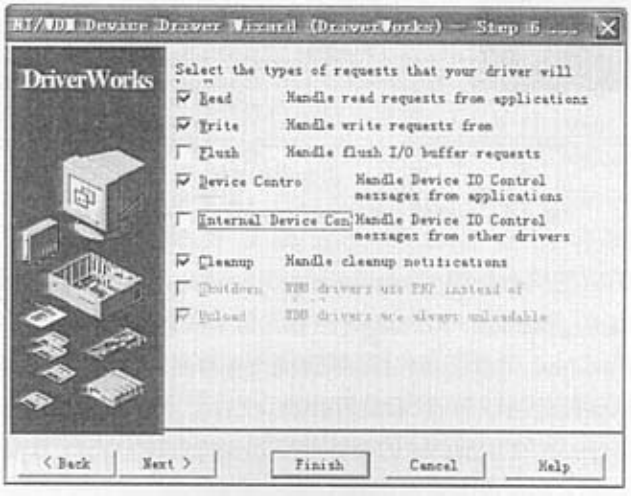

图 4.3 请求类型选择

Fig. 4.3 Selection of the Type of Request in Drivers Model

⑦ 选择 IRP 队列的串行处理机制，可以有不排序、由系统管理和由驱动程序管理三种方式。对于实际存在的 PCI9054 设备在某个时刻只能有一个操作在进行。要求在一个 IRP 未处理完之前，绝不允许处理另一个 IRP，所以必须对串行的 IRP 进行排序。选择系统管理方式和串行所有的读写请求。

⑧ 添加续存储在注册表中的内容。设置好了参数后，当驱动程序开始运行时，驱动程序就会将信息从注册表中读出，卸载后又存入注册表。本设置中，不添加任何内容。

⑨ 添加 PCI 的资源。本设计中添加的资源如图 4.4 所示。

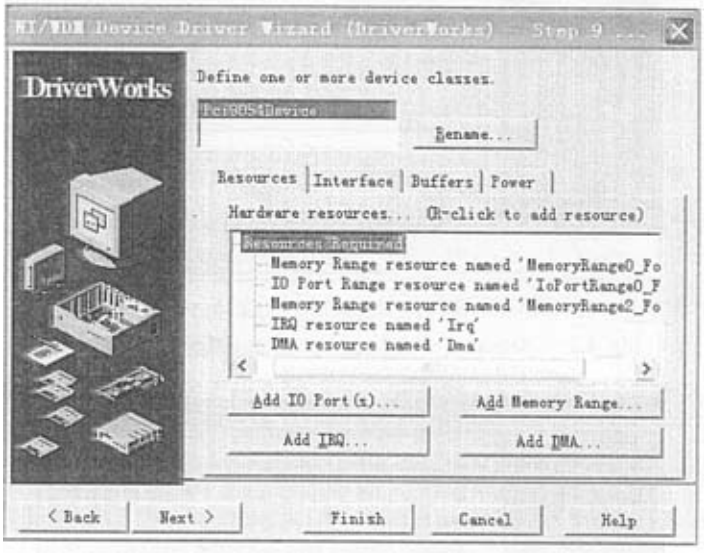

图 4.4 添加硬件资源

Fig. 4.4 Adding Hardware Resources in Driver Model

内存或 I/O 方式的映射资源 MemoryRange0_ForReg:用于访问 PCI 寄存器内存映射空间，IoPortRange0_ForReg:用于访问 PCI 寄存器的 I/O 端口的映射空间，MemoryRange0_ForLocal:用内存方式访问局部总线资源(本设计中为 FPGA)，中断资源 Irq, Dma。这步中还需要设置打开设备使用的接口方式，可以由符号链接名(symbolic link)方式和 GUID 接口方式(只有 WDM 的驱动程序才支持此种方式)。本设计选择符号链接名方式打开设备。由于本采集卡需要传输大量的图像数据，所以在选择采用直接(direct)读/写访问的方式。支持必要的电源管理。

⑩ 添加自定义的 IOCTL。添加用于应用程序启动和传 DMA 地址的 I/O 控制功能 PCI9054_IOCTL_800_READDMA 和用于单个写访问数据至局部总线 FPGA 的 I/O 控制功能 PCI9054_IOCTL_801_WRITELOCAL。最后，生成 win32 平台程序。

至此，DriverWorks 就会为 PCI9054 生成一个 WDM 的驱动程序框架。这时生成的驱动程序不完成任何处理。通过下一节的阐述来完成 PCI9054 的 WDM 驱动程序。

(2) 驱动程序功能的实现[31,37-39]

采用 DriverWork 为 PCI9054 创建 WDM 驱动程序的基本结构。结构中包括一些常用的例程。驱动程序收到系统发出的或者应用程序发出的 IRP 时，作出反映调用运行相应的例程，具体的对应关系详见下面的论述。

本采集卡的 PCI 设备驱动程序主要由初始化设备、创建和删除设备、对 PCI9054 寄存器的访问、对 PCI9054 局部总线设备(即 FPGA)内部资源的访问、中断处理、DMA 和处理电源管理请求等模块组成。下面将从这几个部分进行阐述。

① 初始化设备

初始化模块是每个实际物理设备驱动程序中必不可少的。其中框架中的 DriverEntry 例程即为驱动程序的初始化例程。在 DriverEntry 例程中包含各个 IRP 的分发例程入口指针。当装载 PCI9054 驱动程序时，PnP 管理器为驱动程序调用一次 DriverEntry 例程。主要有三个功能：

设置 AddDevice、Unload、Dispatch 和其他例程的入口指针。

可以从注册表中读取一些需要的信息以初始化驱动程序，本设计没有导入。

初始化其他的在驱动程序范围的数据结构和资源。

在 PCI9054 设备被首次枚举的时候，系统初始化时调用 AddDevice 例程。AddDevice 例程负责创建一个 PCI9054 设备对象，并把它连接到以 PDO 为底层的设备堆栈中。当系统的 PNP 管理器在取得设备的资源后会自动向驱动程序发出 IRP_MN_START_DEVICE 的 IRP，在该 IRP 中包含了设备的资源信息。IRP_MN_START_DEVICE 对应的分发例程是 OnStartDevice 例程，主要功能是初始化物理设备，更具体的说就是初始化设备的物理资源。通过调用 I.AllocatedResources()和 I.TranslatedResources()来获取系统为 PCI9054 分配的资源列表(raw resources)。列表中包括 PCI9054 的寄存器 I/O 和 Memory 映射资源，局部总线的 Memory 映射资源。构建和初始化一个 DMA 适配器对象，获得 DMA 资源。OnStartDevice 例程节略如下：

```
NTSTATUS PCI9054Device::OnStartDevice(KIrp I)
{
......
```

```
//获得资源列表
PCM_RESOURCE_LIST pResListRaw = I.AllocatedResources();
PCM_RESOURCE_LIST pResListTranslated = I.TranslatedResources();
#define MAX_DMA_LENGTH 0x100000    // 0x100000 is 1 MB
//构建一个 DMA ADAPTER 对象
DEVICE_DESCRIPTION dd;
.....
m_Dma.Initialize(&dd, m_Lower.TopOfStack()); //初始化 dma
m_Buffer.Initialize(&m_Dma,2048);          //申请 2k 的公共缓冲区内存
KPciConfiguration PciConfig(m_Lower.TopOfStack());
status = m_MemoryRange0_ForReg.Initialize(
//获得内存方式 PCI 寄存器资源映射
pResListTranslated,
pResListRaw,
PciConfig.BaseAddressIndexToOrdinal(0)
);
status = m_MemoryRange1_ForLocal.Initialize(
//获得内存方式局部总线资源映射
pResListTranslated,
pResListRaw,
PciConfig.BaseAddressIndexToOrdinal(2)
                                              );
}
status = m_IoPortRange0_ForReg.Initialize(
pResListTranslated,
pResListRaw,
PciConfig.BaseAddressIndexToOrdinal(1)
                                          );
status = m_Irq.InitializeAndConnect(          //初始化并且连接中断
  pResListTranslated,
  LinkTo(Isr_Irq),
  This
                                        );
m_DpcFor_Irq.Setup(LinkTo(DpcFor_Irq), this);
```

```
                                        //设置中断延迟响应例程 DpcFor_Irq
  m_MemoryRange0_ForReg.outd(INTCSR,0x40100);
                                        //允许 PCI 中断和 DMA 通道 0 中断
  return status;    //返回 NTSTATUS。成功，返回 STATUS_SUCCESS
}
```

② 创建和删除设备

应用程序对板卡操作时，首先需要通过函数 CreateFile()，发送 IRP_MJ_CREATE 给驱动程序，收到 IRP 后调用例程 Create()进行处理创建设备句柄，打开设备。打开设备的方式有同步方式和异步方式，本设计采取的时同步方式打开设备，即正常模式。系统动态配置 PCI 资源，所以在设备删除的时候必须释放所有的内存、I/O 和中断资源。并且需要把中断例程与 PCI9054 设备的断开。

③ PCI9054 寄存器的访问

在驱动程序中，我们需要通过访问 PCI9054 的内部寄存器来获取或设置 PCI 内部资源的工作状态。比如在后面叙述中的 DMA 和中断状态位等信息都是通过访问 PCI9054 内部寄存器的状态位来获得。是在 OnStartDevice ()线程成功初始化后，通过 m_MemoryRange0_ForReg.outd(offset，data)向 PCI 偏移量为 offset 写入数据 data，data=m_MemoryRange0_ForReg.ind(offset) 从 PCI 偏移量为 offset 读出数据送至 data。

④ 局部总线设备(即 FPGA)内部资源的访问

驱动程序为局部总线资源映射的内存空间 m_MemoryRange1_ForLocal 的输入输出控制，即可实现数据的传输。在本设计中应用程序需要写一些 FPGA 内部寄存器的值。

函数 data=m_MemoryRange1_ForLocal.ind(offset)实现读取局部总线上地址为 offset 的值操作。

函数 m_MemoryRange1_ForLocal.ind(offset，data)实现单个数据的输出。

在本设计中，图像数据的读出是通过 DMA 实现的，来满足数据的传输率。具体的 DMA 传输过程在下一节的应用程序和驱动程序通讯做详细的阐述。做一个详细的分析。SerialRead 例程：在开发 PCI9054 的驱动程序时，考虑到读取数据量大和效率的因素，采取 DMA 的方式。SerialRead 例程的主要工作为初始化 DMA 和启动 DMA。

⑤ 中断处理

中断处理需要中断服务例程和延迟调用例程，这两个例程在头文件需要对其进行声明。声明如下：

```
class PCI9054Device : public KPnpDevice
{
```

```
……
    MEMBER_ISR(PCI9054Device, Isr_Irq);//声明 Isr_Irq 是中断服务例程
    MEMBER_DPC(PCI9054Device, DpcFor_Irq);
    //声明 DpcFor_Irq 是中断延迟调用例程
    ……
    KInterrupt          m_Irq;
    KDeferredCall       m_DpcFor_Irq;
……
};
```

声明后，在 OnStartDevice()例程中所示的初始化和连接中断服务线程 Isr_Irq。

```
status = m_Irq.InitializeAndConnect(          //初始化并且连接中断
    pResListTranslated,
        LinkTo(Isr_Irq),
        This
        );
```

只要 PCI9054 对应的中断号，有中断产生，运行例程 Isr_Irq。由于计算机中断资源是复用性，所以我们设计 PCI9054 中断线程的时候，要在在例程 Isr_Irq 判断中断是否为采集卡的 PCI9054 发出的，如果不是，返回 FALSE；若是，进行必要的处理，请求一个延迟调用线程 DPC，然后返回 TRUE。在设计时，本系统采用如 4.5 所示的中断服务例程的流程框图[30]进行实施。

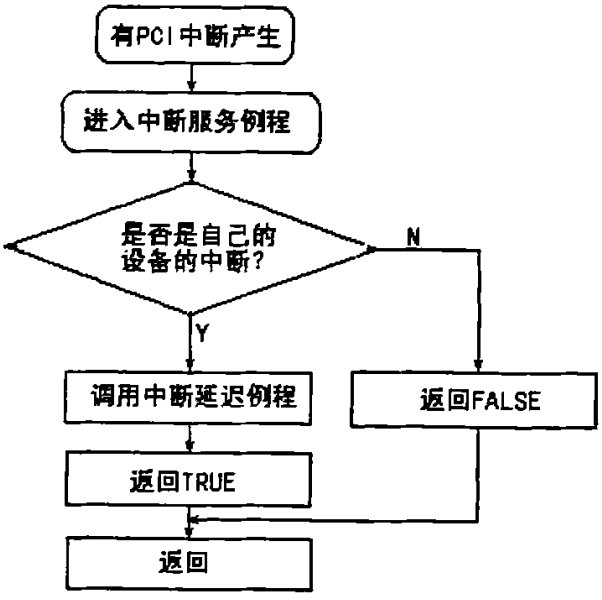

图 4.5 中断服务例程的流程框图

Fig. 4.5 Flow Chart of Interrupt Serve Routine

判断中断是否是自己的设备产生的方法是，程序通过读取 PCI9054 的中断寄存器 INTCSR 的中断标志位来判断。如果中断标志位有效，说明是自己设备发出，如果判断是自己设备发出时，一定要先屏蔽中断，后在清除中断，以避免中断重复处理和占用中断源。

在中断设计时需要注意的是，中断服务例程的处理时间应尽量的短，由于中断服务例程在 DIRQL 级别上运行，很多函数不能调用。所以在中断服务例程中在 DISPATCH_LEVEL 上调用一个中断延迟处理。本设计的任务是调用 OnDMAReady 线程，准备下一次 DMA。

⑥ DMA[40]

启动 DMA 传输，需要对 PCI9054 的如下寄存器进行设置：

DMA 模式寄存器 DMAMODE0，设置 DMA 通道 0 的 DMA 模式。

PCI 地址寄存器 DMAPADR0，设置 PCI 的起始传输的物理地址。

PCI 局部总线设备地址寄存器 DMALADR0，设置局部总线设备起始传输地址，由于本设计只有一个局部总线设备，所以地址直接设置为零即可。

DMA 传输的字节数寄存器 DMASIZE0，设置一次 DMA 的传输字节数。DMA 传输字节数寄存器 DMASIZ0：本设计基于传输速度的考虑，设置为 1024，即一次 DMA 传输 1M 字节数据。

DMA 传输方向寄存器 DMADPR0，设置 DMA 传输的方向。有 PCI 至 LOCAL 和 LOCAL 至 PCI。本设计中是读 DMA，所以为 LOCAL 至 PCI。

DMA 状态寄存器 DMACSR0，设置启动 DMA。通过查询状态寄存器还可以知道一次 DMA 传输是否完成，本论中启动 DMA 传输的程序。

```
VOID PCI9054Device::StartDMA(ULONG PAddress,ULONG NBytes)
{
m_IoPortRange0_ForReg.outd(DMAMODE0,0x20C00);//DMA 的模式
m_IoPortRange0_ForReg.outd(DMAPADR0,PAddress);//PCI 的物理地址
m_IoPortRange0_ForReg.outd(DMALADR0,0x0);//FPGA 的地址
m_IoPortRange0_ForReg.outd(DMASIZ0,NBytes);;//传输字节数
m_IoPortRange0_ForReg.outd(DMADPR0,0x8); //传输方向：局部总线设备至 PCI
m_IoPortRange0_ForReg.outb(DMACSR0,0x3); //启动 DMA
}
```

其他功能：在 PCI9054 驱动程序开发时， PnP 操作和 Power 电源操作采用框架生成的默认操作，在本级不作处理，而直接传入下一级进行处理。

4.1.4 驱动程序安装

通过上一节的论述，基于 FPGA 的数字图像采集处理卡的驱动程序的功能基本完成。工程编译后，便可生成一个驱动程序可执行文件 PCI9054.sys。而驱动程序要成功的安装，至少还必须提供一个扩展名为 INF 的驱动程序安装信息文件[41]。它是一个文本文件，含有安装一个 WDM 设备驱动程序需要的所有必备信息，包括要复制的文件列表、要创建的注册表项等。这里就不再赘述，详细请参见参考文献[41]。

4.2 基于 FPGA 的嵌入式图像采集处理卡的应用程序

4.2.1 与 WDM 驱动程序通信的上层应用程序

虽然驱动程序是为设备的硬件层编程服务的，但同样需要提供和应用程序进行通信的能力，从而最终达到应用程序控制设备的目的。

在 Windows 中，应用程序实现与 WDM 通信的过程[31]是：应用程序先用 CreateFile 函数打开设备，然后用 DeviceIoControl 和 WDM 进行通信，包括从 WDM 中读数据和

写数据给 WDM 两种情况。当应用程序退出时，用 CloseHandle 关闭设备。这将产生对应于设备对象的相应 IRP，这个 IRP 传输到驱动程序后，驱动程序根据 IRP 的信息对支持的硬件进行操作。

通信应用程序设计时，需包含 windows.h，winioctl.h，PCI9054ioctl.h 等三个头文件。PCI9054ioctl 文件中定义了开发人员在驱动向导 DriverWizard 中定义的应用程序调用驱动程序的控制命令。

4.2.2 进行相机模式设置的串口通信程序

本程序设计目的是通过计算机的 RS232 接口与相机实现串口通信，从而进行相机模式的设置。本着设计简单和方便的原则，本程序的实现直接采用 Visual C++6.0 的 MSComm 控件实现。具体的实现方法[42]如下：

(1) 在建立好的工程中添加 MSComm 控件。控件全称为 Microsoft Communications Control, version 6.0。

(2) 初始化串口：设置 MSComm 控件的属性。在文件中添加控件变量，对控件变量进行初始化。

```
m_MySerial.SetCommPort(1);                    ///选择串口 1
m_MySerial.SetInputMode(1);                   ///设置输入数据模式
m_MySerial.SetInBufferSize(1024);             ///设置输入缓冲区大小
m_MySerial.SetOutBufferSize(512);             ///输出缓冲区大小
m_MySerial.SetSettings("9600,O,8,1");
//设置通信参数,8 位数据位+1 位停止位+无奇偶校验
if(!m_MySerial.GetPortOpen())
    m_MySerial.SetPortOpen(TRUE);             //打开串口
m_MySerial.SetRThreshold(1);
m_MySerial.SetInputLen(0);
m_MySerial.GetInput();
```

(3) 添加串口事件消息处理函数 OnComm()。其主要任务是从串口接收数据。通过函数 m_MySerial.GetInput()和对接收的数据进行变量转换来实现。

(4) 发送数据。由函数 m_MySerial.SetOutput(COleVariant(send_data))实现。

根据 IMPERX 公司相机的具体配置参数，设计出了相机模式设置程序。相机模式设置界面如图 4.6 所示。

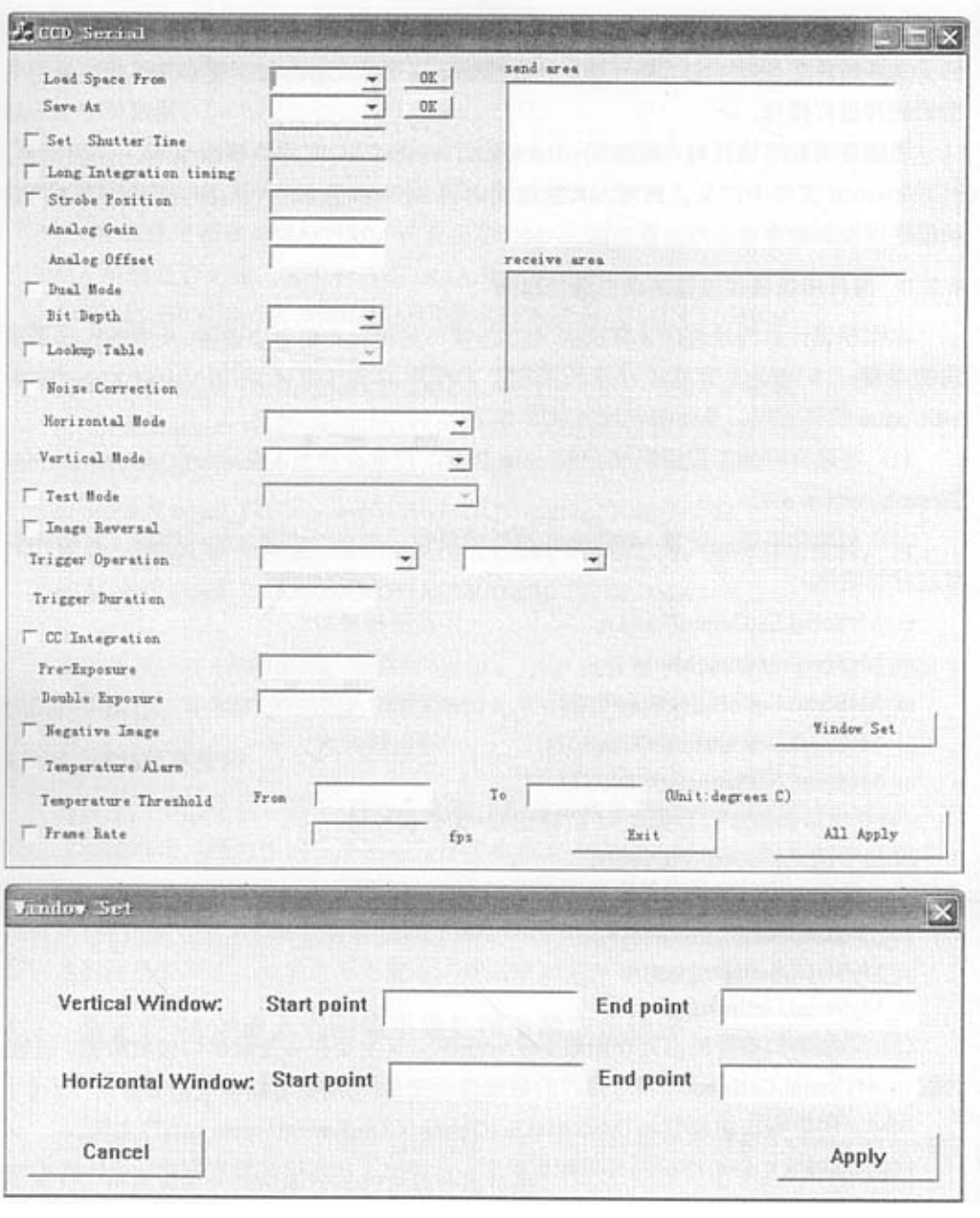

图 4.6　相机模式设置界面

Fig. 4.6　GUI of Camera Model Configuration

4.3 FPGA 内部逻辑结构

采集卡核心控制芯片是一片 FPGA，需要实现逻辑时序控制和保证数据的完整性。该系统选用 ALTERA 公司的 Cyclone 系列的 EPIC6Q240C8 芯片，共有 5980 个可用的逻辑单元(Les)，185 个 I/O 口，2 个 PLL 等，其丰富的资源确保了系统的使用。

基于 FPGA 的嵌入式图像采集卡的 FPGA 内部逻辑结构如图 4.7 所示。

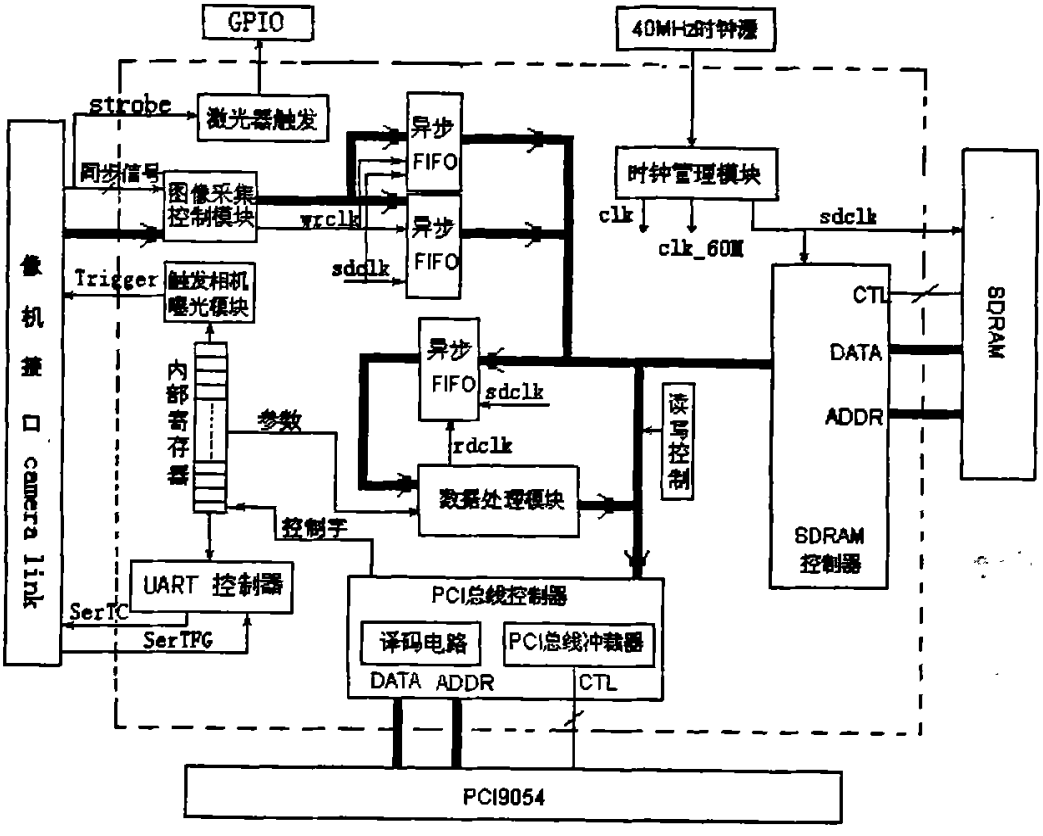

图 4.7 FPGA 内部体系结构

Fig. 4.7 Interior Logic Architecture of FPGA

图中虚线框内即为 FPGA 的内部模块图，不同的模块对应不同的功能。所有采集工作由参数的配置，相机模式设置，图像数据采集，预处理，图像数据传输等五个步骤组成。

(1) 参数配置

参数配置由 PCI 总线控制器实现。主控计算机通过 PCI 接口，把需要相机配置参数和相机触发等控制字输入到采集卡，通过译码器对输入参数进行译码后，送到内部寄存器中。同时总线控制器负责 PCI 局部总线仲裁。

(2) 相机模式设置

FPGA 设计的相机模式配置模块实现相机模式自动配置，相机模式配置模块的工作原理是：

首先，PC 机通过驱动程序把相机设置参数传输给由图像采集卡。其次，UART 控制器在内部寄存器中读取预设置的相机参数，完成相机进行模式设置。

(3) 图像数据采集

图像数据采集由图像采集控制模块，相机曝光模块、激光器触发模块实现图像数据流的接收和激光器与相机之间的同步。

采集卡接收到开始采集的命令控制字后，图像采集卡用相机控制信号 CC1 发出相机曝光的触发信号 Trigger。通过 CCD 相机返回的选通脉冲 strobe，同步外部光源。相机正确曝光输出数据，采集卡接收图像数据并且抓取同步头，对有效图像数据进行锁存至片内 FIFO，实现乒乓操作。后转存片外至 SDRAM。完成图像数据采集工作。

(4) 预处理和传输

图像预处理由数据处理模块实现，图像传输由 PCI 总线仲裁器实现。

图像采集完成后，间隔一个时间(1000 个系统时钟周期)开始取 SDRAM 的数据进行图像增强预处理，处理完成后，存回 SDRAM。FPGA 发 PCI 中断，请求数据传输，送回主控计算机。

4.3.1 时钟管理设计

时钟可以比喻成数字逻辑中的血液，在本论文中几乎所有的信号都需要依靠时钟上升沿传递。因此，时钟管理的重要性不言而喻。在本论文设计中采用 ALTERA 的时钟资源和 PLL 来有效地管理时钟，来解决设计中的时序问题。

在设计应用中，本文采用调用 MegaWiard 中的 ALTPLL 来生成所需要的锁相环。在过程中，只需设置倍频和分频系数，以及相移的具体的度数或者延时的大小，MegaWiard 会自动设置 PLL 内部居体的参数，同时也会检查用设置的合法性。这样，我们在设计时，可以非常方便地产生需要的 PLL 类型和参数，而无需关心其内部复杂的结构。

在本设计中进行参数设置后生 PLL 模块如图 4.8 所示。晶体振荡器输入 40MHz 的时钟源 clk_40MHz，经过 PLL 模块输出 40M 的 clk, 60M 的 clk_60MHz 和 100Md 的

clk_100MHz。其中 clk 是系统的工作时钟，clk_60MHz 是给 UART 模块的输入时钟，clk_100MHz 是 SDRAM 的工作时钟 sdclk。

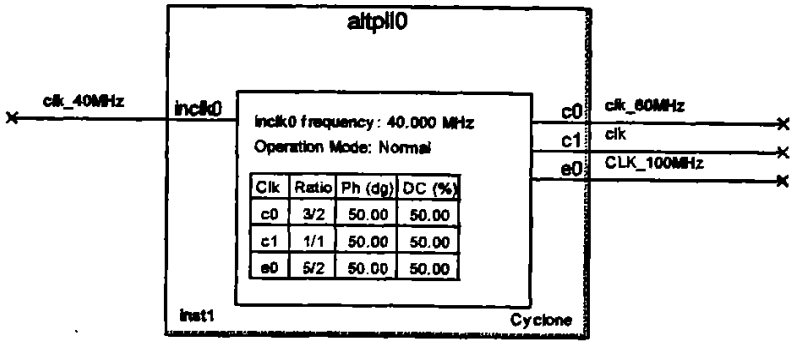

图 4.8 锁相环

Fig. 4.8 Phase Locked Logic

4.3.2 PCI 总线仲裁器

(1) 局部(LOCAL)总线仲裁逻辑

在总线控制逻辑中，总线仲裁逻辑是最关键、最核心的部分，直接影响计算机运行的稳定性。PCI 在使用局部总线时，必须先发出总线申请 hold 信号，在局部总线设备允许(holda=1)后，才能进行数据交换。所以，如果总线仲裁逻辑设计不合理，在计算机访问 PCI 局部总线的硬件资源进行时，计算机就会死机或出现不稳定[43]。因此 PCI9054 本地总线的所有控制逻辑必须服从如图 4.9 所示的 PCI9054 的局部总线仲裁逻辑[15]。

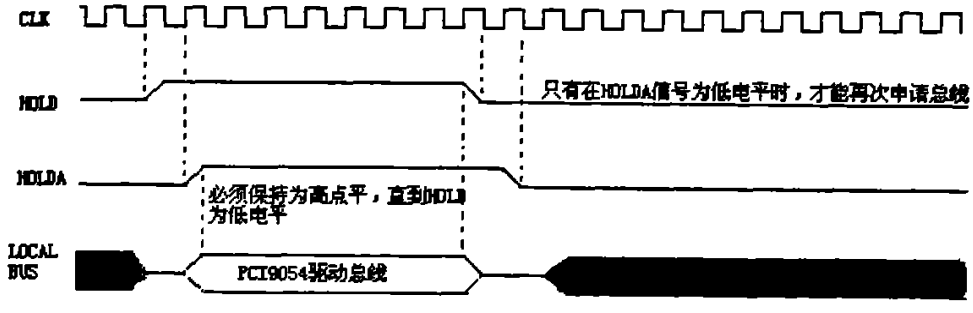

图 4.9 PCI9054 局部总线仲裁

Fig. 4.9 PCI9054 Local Bus Arbitration

经由分析，本设计采用 D 触发器(D_TRIGGER)设计，由全局时钟 clk 上升沿来控制 holda 的输出状态。VHDL 语言描述的代码如下：

```
process(clk)
begin
  if clk'event and clk='1' then
     holda_reg<=hold;
  end if;
  holda<=holda_reg;
end process;
```

(2) ready 信号的控制

在 PCI 获得总线掌管权(holda 为高)时，发数据前，PCI9054 首先发送 ads 信号，局部总线设备 FPGA 使 ready 信号有效来响应数据传输请求，ready 信号有效时建立数据传输，数据传输完成后，PCI9054 发 blast 信号结束数据传输。在本设计中，采用状态机的设计方法，其状态转换图如图 4.10 所示。

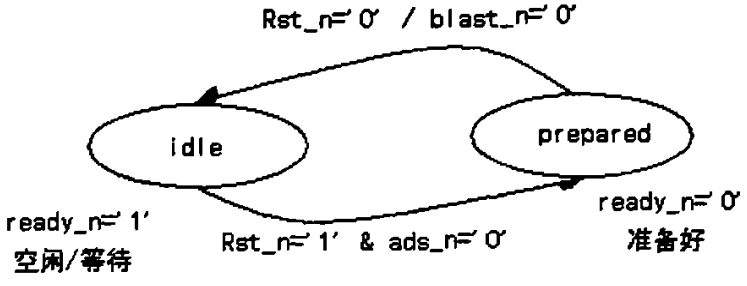

图 4.10 Ready 控制的状态转换图

Fig. 4.10 State Machine of Ready-controlling

在 ready 信号输入使能时，c 模式下 ready 信号表示总线准备好，可以进行数据传输。设计时，采用 ads 和 blast 组合产生 ready 的控制时序。ads 和 blast 信号有效宽度均为一个时钟周期，所以 ready 信号的控制采取的是可以同步时序控制方法，只需检测 ads 和 blast 信号的电平即可产生符合控制逻辑的时序。

图 4.8 中，idle 为闲置等待状态，prepared 为准备好状态。在系统复位信号 Rst_n 有效或者 blast_n 有效时，状态机无条件地进入空闲等待状态(idle)。在复位完成和 ads_n 信号有效时，状态机进入准备好状态(prepared)。

(3) 总线译码电路

在整个系统中设计了多个内部寄存器。那么对应不同的地址线时，总线上的数据也表示不同的意义。由于电路设计比较简单，只需在地址总线数据有效时，对其采用 CASE 和 WHEN 进行判断即可实现。

4.3.3 SDRAM 控制器

在 SDRAM 控制器[44-45]的 FPGA 实现方案中，本论文中参考了 ALTERA 公司的 SDR SDRAM 控制器的 IP 核设计。使用 FPGA 的自顶向下[46]的模块化设计思想，把 SDR SDRAM 控制器分为接口控制模块，命令控制模块和数据流控制模块等三个模块。具体的实现过程是：首先分析顶层模块的功能，再将其功能分类细化，分配到不同的子模块去实现，然后自底向上的先逐步完成各个子模块的设计，最后将子模块相互连接生成顶层模块。SDRAM 控制器的系统级框图如 4.11 所示。

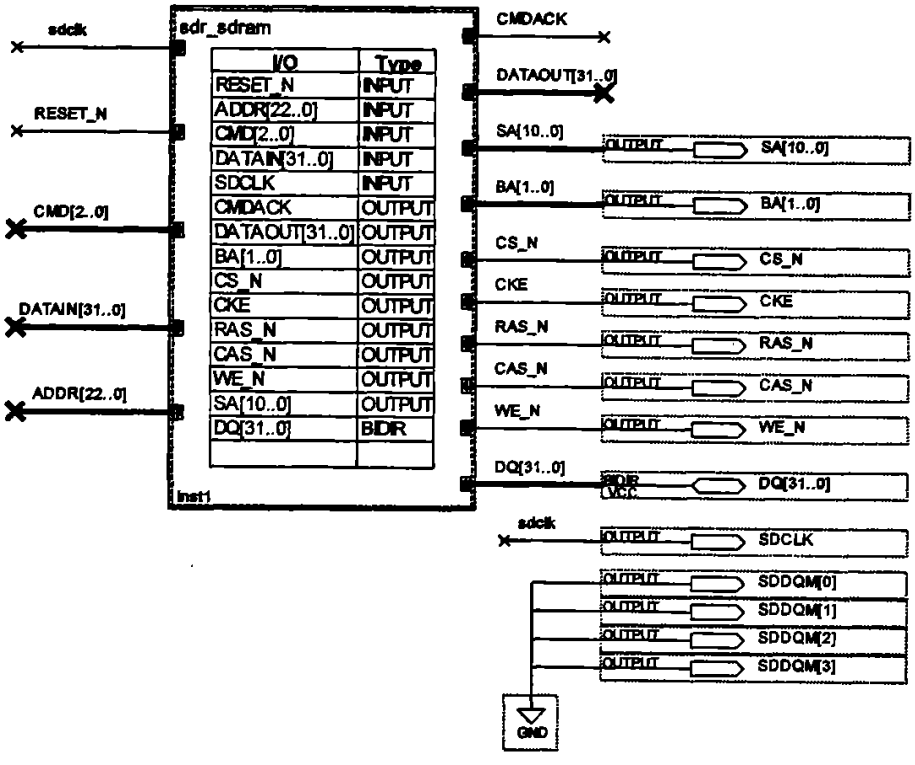

图 4.11 SDR SDRAM 控制器系统级框图

Fig. 4.11 SDR SDRAM Controller System-Level Block Diagram

下面简单介绍一下图 4.10 中的 SDRAM 控制器的主要信号：

片选信号 CS_N，时钟使能信号 CKE，输入输出使能信号 SDDQM_N[3..0]，列地址选通信号 CAS_N，行地址选通信号 RAS_N，写使能信号 WE_N，地址 SA[10..0]及数据信号 DQ[31..0]。本设计考虑到 SDRAM 的使用情况让 CS_N，CKE 和 SDDQM_N[3..0]上电后一直有效。所有信号均与 SDRAM 的时钟信号 SDCLK 同步。

SDRAM 的写入数据信号 DATAIN[31..0]和读出数据信号 DATAOUT[31..0]，接口地址信号 ADDR[20..0]，系统复位信号 RESET_N，接口命令字 CMD[2..0]和命令握手信号 CMDACK。CMD[2..0]为不同值代表 SDRAM 不同的操作。可以由自己自行定义，在本设计中，接口命令定义如下表 4.1 所示。

表 4.1 接口命令

Tab. 4.1 Interface Commands

CMD[2..0]	命令的操作	描述
000	空操作	没有任何操作
001/010	读/写操作	SDRAM 执行带自冲电的读/写操作
100	刷新操作	SDRAM 自动刷新
101	加载模式寄存器操作	SDRAM 加载模式寄存器
110	加载寄存器 1	配置 SDRAM 控制器的延时
111	加载寄存器 2	配置 SDRAM 控制器的自刷新周期

(1) 接口控制模块

接口控制模块负责接收 SDRAM 控制器的输入命令 CMD[2..0]，并对输入的命令作出应答 CMDACK，将待执行的命令传给下一层命令控制模块，提供定时刷新功能（自动刷新周期在初始化过程中由 ADDR[22..0]设定）。

(2) 命令控制模块

把接口模块接收的待执行命令(读、写、刷新和冲电等)翻译成 CAS,RAS 和 CS 信号与 SDRAM 连接的信号，命令和信号的对应关系如表 4.2 所示。

表4.2 SDRAM的基本操作和主要控制信号

Tab. 4.2 Basal Operation and Primary Controlling Signal of SDRAM

命令名称	CS_N	RAS_N	CAS_N	WE_N
命令禁止（NOP:command inhibit）	H	X	X	X
空操作(NOP:No Operate)	L	H	H	H
激活操作（ACT:Select bank and active row）	L	L	H	H
读操作（RD:Select bank and column, and start READ blast	L	H	L	H
写操作（WR:Select bank and column, and start WRITE	L	H	L	L
突发停止（BT:Blast terminate）	L	H	H	L
预充电（PCH:Deative row in bank or banks）	L	L	H	L
自动刷新或自我刷新（REF:auto refresh or self refresh）	L	L	L	H
加载模式寄存器（LMR:Load mode register ）	L	L	L	L

(3) 数据流控制模块

为 SDRAM 的读写时的数据传输提供三态门控制路径。在 SDRAM 的时钟信号 SDCLK 上升沿进行数据的读写。

接口电路对 SDRAM 的操作过程主要有初始化，读/写操作和刷新操作。

① 初始化操作

SDRAM 首先要进行初始化操作。在上电后等待 100ns， 至少执行 1 条空操作(NOP)，然后对所有页执行预充电操作(PCH)，接着向各页发出两条刷新操作指令(REF)，最后执行 SDRAM 工作模式的设定 LMR 命令用来配置 SDRAM 工作模式寄存器。SDRAM 工作寄存器可以根据具体应用的需要进行设置。SDRAM 模式寄存器的的格式[47]如图 4.12 所示。

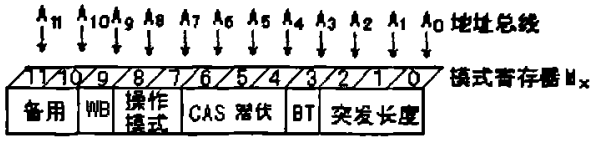

图 4.12 SDRAM 模式寄存器

Fig. 4.12 SDRAM Mode Register

图中突发长度可以为 1(M0:M2=000)、2(M0:M2=001)、4(M0:M2=010)、8(M0:M2=011)和全页模式(M0:M2=111)。初始后的 SDRAM 在得到了 RAS、CAS、WE 的值后开始执行相应的命令。

② 读写操作

读写操作主要完成 SDRAM 和 FPGA 的数据交换。在对 SDRAM 进行读、写过程中，必须要先进行页激活 ACT 操作，保证存储单元是打开的，以便从中读取地址或者写入地址，然后通过预充电 PCH 命令实现来关闭存储单元。在进行写(WR 命令)操作时，起始地址和数据都会被寄存，而进行读操作时，起始地址被寄存，第一个数据经 CAS 延迟时间(通常为 1～3 个时钟周期)后出现在总线上。最后，操作终止(BT)：当 SDRAM 顺次的进行读、写操作后，当到达到突发长度或者突发终止指令 BT 出现时，SDRAM 将终止其操作。

4.3.4 相机模式配置模块

整个相机模式控制模块由一个 UART(universal asynchronous receiver/transmitter)控制器[48]完成。控制器要做的工作就是，发送相机模式设置字符串，接收相机反馈字符串，和对反馈的字符串进行正确的判断。在本设计中要求 UART 控制器能对相机反馈回来的结果进行判断处理，选择继续发送还是重发。主要的设计思想是：PC 向采集卡发送模式设置命令和设置参数，FPGA 接收到设置命令和参数后，通过 UART 控制与相机实现串口通讯，完成相机的参数设置。

相机串口采用标准的 RS-232 接口协议，主要参数的选择：波特率 9600bit/s，1 位起始位，8 位数据位，无奇偶校验，1 位停止位。这样，设计就要求 UART 对相机反馈回来的 10 位数据中去除起始位和停止位，提取反馈字符，和把待传输的字符加上起始位和停止位变成 10 数据输出。为了层次清晰方便，UART 控制器整体上采用模块化设计，由波特率发生器，数据接收模块、数据发送模块和内核控制模块等组成。下面分别从这个四个模块进行阐述。

(1) 波特率发生器

波特率发生器的功能是产生和 RS-232 通信所采用的波特率同步的时钟，这样才能方便地按照 RS-232 串行通讯的时序要求进行数据接收或者发送。图 4.13 表示了波特率时钟和 RS-232 接受信号 RxD 之间的时序关系，波特率时钟的频率就是波特率。

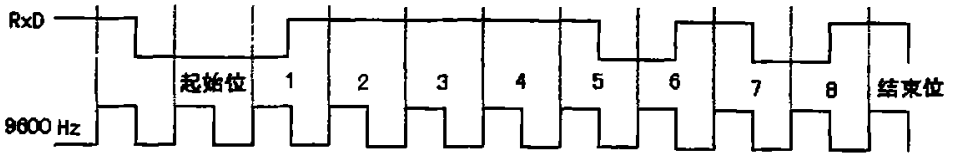

图 4.13 波特率和接收信号时序图

Fig. 4.13 Timing Diagram of Baud Rate and Receiver Signal

实现波特率时钟的设计思想就是设计一个计数器，在本设计中系统的时钟为40MHz，相机模式设置的串口通信时钟为 9600Hz，如果直接对系统时钟进行分频，分频倍数为 $40\times10^6/9600=4166.667$，其值为一个小数，设计起来比较复杂。在本设计中先将系统时钟通过如图 4.8 所示锁相环 IP 核，输出为 60MHz 的时钟。

在对输出的 60MHz 的时钟进行 6250 倍（$60\times10^6/9600$）分频。分频的 VHDL 程序节略如下：

```
  PROCESS(rst_n,CLk_60MHz)              --计数器线程
BEGIN
IF (rst_n='0')THEN
    counter<=(others=>'0');
ELSIF(rising_edge(CLk_60MHz))THEN      --在输入时钟的上升沿进行计数
    IF(counter="1100001101010" ) THEN  --计数值等于 6250 复位
        counter <= (OTHERS =>'0' );
    ELSE
    counter <= counter +1;
    END IF;
END IF ;
END PROCESS;
PROCESS(counter,band_clk)
BEGIN
    IF (counter<"0110000110101") THEN  --计数值小于 3125 时，输出低电平
        band_clk<='0';
    ELSE                               --计数值大于 3125，小于 6250 时，输出高电平
        band_clk<='1';
    END IF;
END PROCESS;
```

(2) 数据接收器

实现相机传输出的 SerTFG 信号的成功接收。并把 8 位数据传给 UART 内核进行判断。由信号检测器、移位寄存器、锁存模块和计数器共同构成实现。整个接收器的状态转换图如图 4.14 所示。

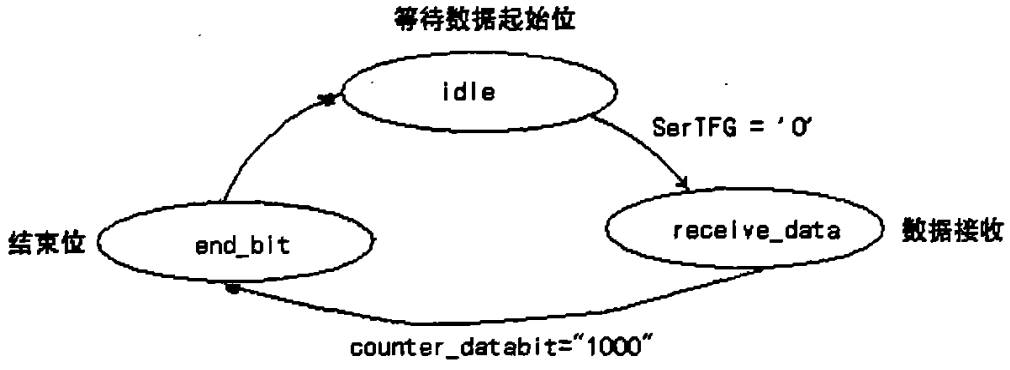

图 4.14 UART 接收器状态转化图

Fig. 4.14 State Machine of UART Receiver

信号检测器负责监测 RS－232 输入端的 SerTFG 信号，当有新的数据传输时(即 SerTFG='0'时)，数据接收模块由移位寄存器接收数据，同时计数器开始计数。计数值等于 8 时，移位寄存器即成功接收到 8 位数据时，锁存数据，传给 UART 内核控制模块判断、处理。

(3) 数据发送器

这个模块的主要工作是将待传输的 ASCⅡ加上一位起始位和一位停止位，转化成 10 位的串口数据流按波特率输出。数据发送模块首先产生一位起始位，移位寄存器将待传输的数据从高位至地位依次移出 8 位数据位至输出端 SerTC，数据移完了之后，产生一位停止位，结束此次传输。数据开始传输时，同时使能计数器，由计数器的值来控制目前发送的是什么位。

(4) 内核控制模块

内核控制模块负责整个传输过程的控制，负责接收和存储待传输的字符串，负责判断相机反馈的确认字符串。如果反馈的字符串为错误提示信息，则重新发送。否则继续发送下一个待发送的字符。

4.3.5 相机触发和激光器触发模块

为了更好的说明分析此模块，有必要对相机的工作模式进行一下了解。IMPERX 的 CCD 相机可以工作在标准和外部触发模式，外部触发模式又分软件外部触发和硬件外部触发。

(1) 标准模式。相机自由的连续输出图像，每秒 30 帧。

(2) 硬件外部触发模式。在这种模式下，触发信号是通过相机后身的一个连接器 I/O 输入 TRIGGER。

(3) 软件外部触发模式。在这种模式下，相机外部触发信号由通过标准 Camera Link 接口的 CC1 信号产生。本设计就采用软件外出发模式实现相机的外部触发。

不管是软件还是硬件外部触发都支持标准、快速抓取和双曝光，具体的工作模式可以通过对相机模式进行设置。

相机工作在标准外部触发模式的时序图[21]如图 4.15 所示。

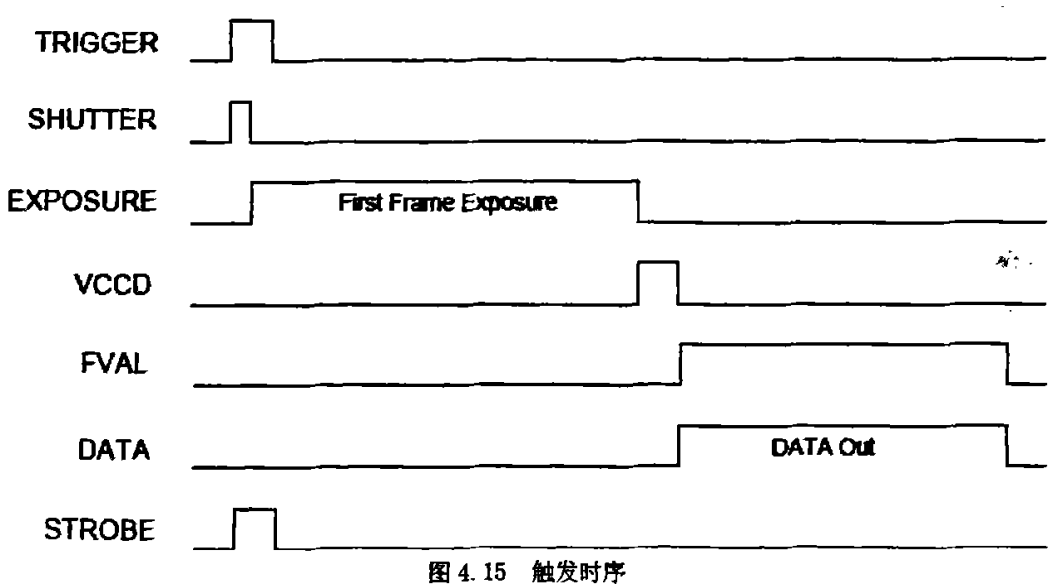

图 4.15 触发时序

Fig. 4.15 Triggering Timing Diagram

首先，相机空闲等待外部触发信号(软件触发时用 CC1)，一但收到外部触发信号(TRIGGER)，相机开始清除水平和垂直寄存器，同时发送一个有效时间为 5 微秒的快门信号(SHUTTER)清除像素，后开始积分。CC1 的有效时间决定相机的第一帧的曝光时间。在第一帧曝光结束后，相机工作在自由模式，这时帧率决定曝光时间。触发信号

(CC1)变为高电平后，可以设置成抓取 1 到 250 帧图像或工作在自由模式。在快门信号(SHUTTER)发出的同时，相机会发出持续 200 微秒的阈门信号(STROBE)来同步外部光源。

相机触发模块设计的目的是：正确控制 CC1 触发时刻，实现相机何时触发控制；控制 CC1 的高电平时间(有效时间)实现第一帧曝光时长。在本模块的设计中，采用这种方法实现相机的曝光的可控性。如果采用连续外部触发模式，两个连续触发脉冲 Trigger 之间的间隔必须大于帧周期，以保证操作正确。为此，本论文采用命令字来控制是否需要触发 Trigger 信号输出，配合 FVAL 信号来保证两个连续触发脉冲 Trigger 之间的间隔。VHDL 实现的程序段见附录 A。

激光器触发模块设计的目的是：在自然光不能满足相机曝光要求的情况下，图像采集系统往往需要引入外部光源增强环境曝光。比如 PIV 系统就采用激光器作为外部光源，就需要外部对其触发，触发的时刻当然也必须在相机曝光的有效范围内。美国 TSI 公司设计的 PIV 系统的解决方案是通过增加硬件设备(同步器)来实现光源和相机曝光的同步，这样一方面增加系统的成本，另一方面使用起来不太方便。本文基于此目的，在采集卡上增加相机曝光触发模块，实现对外部光源的触发，使外部光源与相机时序同步，使相机在光线度不高的情况下能发挥最大功效。设计方法是采用图 4.15 中所示阈门信号(STROBE)来同步外部光源，模块检测阈门信号的上升沿来触发光源使能信号。设计中采用参数条件的方法实现可调光源时长，以满足不通的需求。VHDL 实现的程序段见附录 A

4.3.6 图像采集控制模块

本模块主要实现同步分离，根据图像同步头时钟信号、FVAL、LVAL 和 DVAL 把数据图像中的有效像素信息提取出来。IMPERX 的 IPX-2M30-L 相机的帧时序图[21]如图 4.16，行时序图如图[21]4.17 所示。

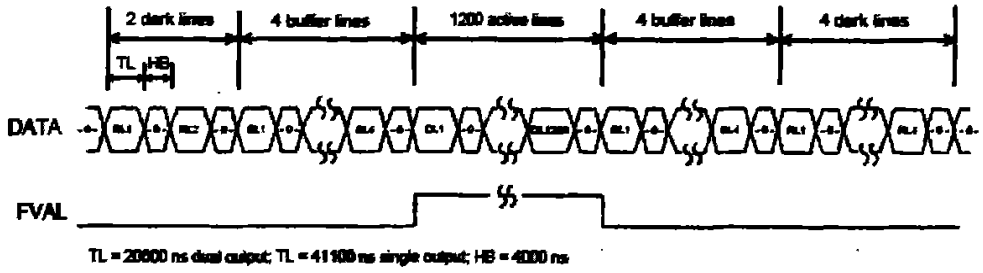

图 4.16 帧时序图

Fig. 4.16 Frame Timing Diagram

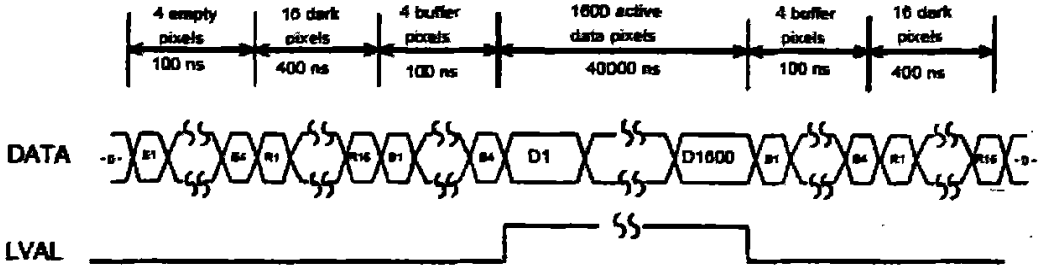

图 4.17 行时序图

Fig. 4.17 Line Timing Diagram

在单一模式下，一帧图像中由两行伪行(RL1 和 TL2)和 4 行缓冲的帧头、1200 行有效数据和四行缓冲加四行伪行的帧尾。一行图像又由 4 个空像素、16 个伪像素和 4 个缓冲像素组成的行头，1600 个有效像素，4 个缓冲像素和 16 个伪像素组成的行尾，共同组成。像素位宽可编程为 8 位、10 位和 12 位。在图像采集控制模块主要的工作就是使采集卡抓取行场同步信号，使输出数据时序按输入的节拍进行。要保证图像的完整性就必须从一场图像的第一行开始读取，对于隔行扫描输出的图像则必须从一帧的第一行开始读取。读取每行图像数据时，则从该行的第一个像素开始。因此，在读取图像数据前应先判断场和行的起始位置。图 4.18 是通过 I/O 接口方式读取图像数据的流程。读取每个像素数据前先查询数据状态，如果数据已准备好则读取数据。为了简化电路设计，用处理器直接读取同步信号，然后找出场和行的起始位置。整个同步时序由 FVAL，LVAL 和相机同步时钟产生。

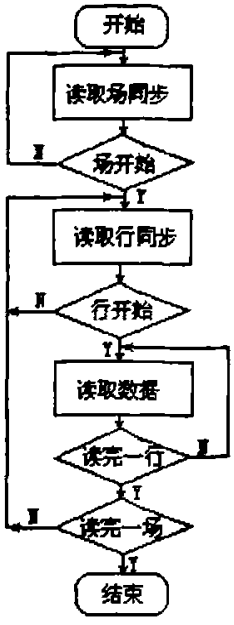

图 4.18 读取图像数据的流程图

Fig. 4.18 Flow Chart of Reading Image Data

4.3.7 图像预处理模块

硬件实现图像增强的预处理工作。本模块所涉及的内容较多，放到第 5 部分重点阐述。

4.4 FPGA 内部逻辑调试结果及分析

4.4.1 调试工具

本论文采用 ALTERA 的 EDA 工具 Quartus II 5.0 完成整个 FPGA 的设计。调试时采用嵌入式逻辑分析议 SignalTap II来完成工程的调试。下面对 SignalTap 做一下简单的介绍。

Altera 公司的 Quartus II 软件中的 SignalTap II是一种基于逻辑分析核的嵌入式逻辑分析仪，满足 FPGA 开发中硬件调试的要求，并且具有无干扰、便于升级、使用简单等特点。SignalTap II 嵌入逻辑分析仪集成到 Quartus II 设计软件中，能够捕获和显示可编程单芯片系统(SOPC)设计中实时信号的状态，这样开发者就可以在整个设计过程中以系统级的速度观察硬件和软件的交互作用。它支持多达 1024 个通道，采样深度高达

128Kb，每个分析仪均有 10 级触发输入/ 输出，从而增加了采样的精度。SignalTap II 为设计者提供了业界领先的 SOPC 设计的实时可视性，能够大大减少验证过程中所花费的时间。目前 SignalTap II 逻辑分析仪支持的器件系列包括：APEXT II,APEX20KE, APEX20KC, APEX20K,Cyclone, Excalibur, Mercury, StratixGX, Stratix。SignalTap II 将逻辑分析模块嵌入到FPGA 中，逻辑分析模块对待测节点的数据进行捕获，数据通过JTAG 接口从FPGA 传送到QuartusII 软件中显示。使用 SignalTap II 无需额外的逻辑分析设备，只需将一根 JTAG 接口的下载电缆连接到要调试的 FPGA 器件。SignalTap II 对 FPGA 的引脚和内部的连线信号进行捕获后，将数据存储在一定的 RAM 块中。因此，需要用于捕获的采样时钟信号和保存被测信号的一定点数的 RAM 块。使用 SignalTap II 的一般流程是：设计人员在完成设计并编译工程后，建立 SignalTap II (.stp) 文件并加入工程、配置 STP 文件、编译并下载设计到 FPGA、在 Quartus II 软件中显示被测信号的波形、在测试完毕后将该逻辑分析仪从项目中删除。以下描述设置 SignalTap II 文件的基本流程[49]：

(1) 设置采样时钟。采样时钟决定了显示信号波形的分辨率，它的频率要大于被测信号的最高频率，否则无法正确反映被测信号波形的变化。SignalTapII 在时钟上升沿将被测信号存储到缓存。

(2) 设置被测信号。可以使用 NodeFinder 中的 SignalTap II 滤波器查找所有预综合和布局布线后的 SignalTapII 节点，添加要观察的信号。逻辑分析器不可测试的信号包括：逻辑单元的进位信号、PLL 的时钟输出、JTAG 引脚信号、LVDS（低压差分）信号。

(3) 配置采样深度、确定 RAM 的大小。SignalTap II 所能显示的被测信号波形的时间长度为 T_x，计算公式如下：

$$T_x = N \times T_s \tag{4.1}$$

其中，N 为缓存中存储的采样点数，T_s 为采样时钟的周期。

(4) 设置buffer acquisition mode。buffer acquisition mode包括循环采样存储、连续存储两种模式。循环采样存储也就是分段存储，将整个缓存分成多个片段(segment)，每当触发条件满足时就捕获一段数据。该功能可以去掉无关的数据，使采样缓存的使用更加灵活。

(5) 触发级别。SignalTap II支持多触发级的触发方式，最多可支持10级触发。

(6) 触发条件。可以设定复杂的触发条件用来捕获相应的数据，以协助调试设计。当触发条件满足时，在signalTap时钟的上升沿采样被测信号。

完成STP 设置后，将STP 文件同原有的设计下载到FPGA 中，在QuartusII中SignalTap II窗口下查看逻辑分析仪捕获结果。SignalTap II可将数据通过多余的I/O 引脚输出，以供外设的逻辑分析器使用；或输出为csv、tbl、vcd、vwf文件格式以供第三方仿真工具使用。

本论文在下面的阐述中均采用了SignalTap II对内部时序进行分析。值得一提的是，在完成调试后，若要下载固化FPGA，需删除stp文件，以免浪费系统资源。

4.4.2 PCI 仲裁器调试结果分析

PCI 仲裁器主要由一个 PCI 总线仲裁 PCI 总线数据译码器构成。逻辑分析结果如图 4.19 所示。

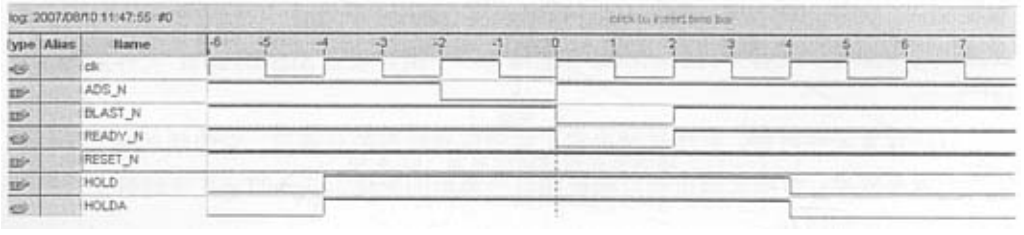

图 4.19 PCI 总线仲裁器调试结果

Fig. 4.19 Debug Result of PCI Bus Arbitration

从仿真结果中，可以看出在 PCI 发出 HOLD 信号后，总线仲裁器对其作出了正确的响应 HOLDA。并且对 ADS_N 和 BLAST_N 的逻辑产生了正确的 READY_N 逻辑，设计合理。

4.4.3 SDRAM 控制器调试结果分析

SDRAM 是动态随机存储器，在进行读写的时候需要对其刷新(预充电)，才能锁存住数据。这样在数据存储前和存储后均需要对页进行预充电操作。在本设计中由于图像采集的数据量比较大，采用全页写和全页读的方式对 SDRAM 进行访问。

SDRAM 全页写的时候调试的时序图如图 4.20 所示，时序控制电路向 SDRAM 控制器发出写命令(CMD=“010”)和写地址后，控制器反馈 cmdack 握手的同时把待写入 SDRAM 的数据送到数据总线 DATAIN 上。

SDRAM 全页读的时候调试的时序图如图 4.21 所示，时序控制电路向 SDRAM 控制器发出读命令(CMD=“001”)和读起始地址后，控制器反馈 cmdack 握手，之后 CL－1

+7个时钟周期后，读出的数据出现总线DATAOUT上。整个调试时序结果符合SDRAM控制时序。

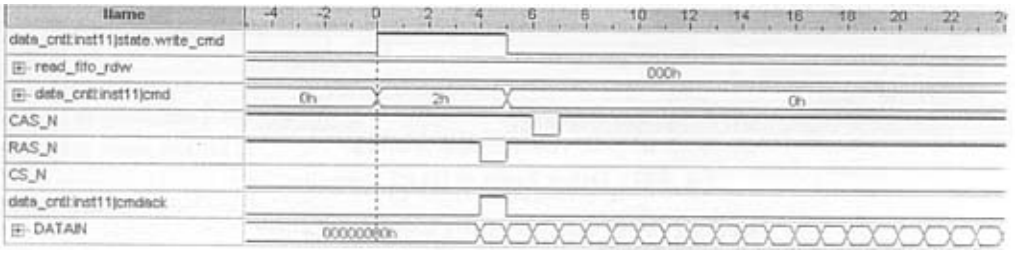

图 4.20 SDRAM 全页突发写时序图

Fig. 4.20 Full-Page Write Burst Timing Diagram

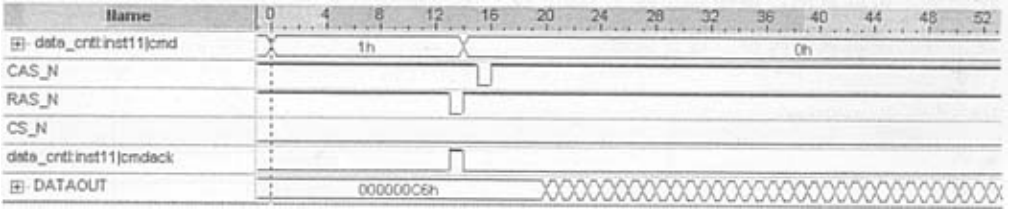

图 4.21 SDRAM 全页突发读时序图

Fig. 4.21 Full-Page Read Burst Timing Diagram

4.4.4 相机模式配置模块调试结果分析

UART 控制器输出"0F"字符串的调试结果如图 4.22 所示。

图中时间轴 0 至 10 为 I/O 口 SerTC_Signal 输出“0000011111”序列，去除 1 位开始位和 1 位结束位，即为字符串 0F 对应的 LVTTL 电平译码输出，逻辑设计正确。

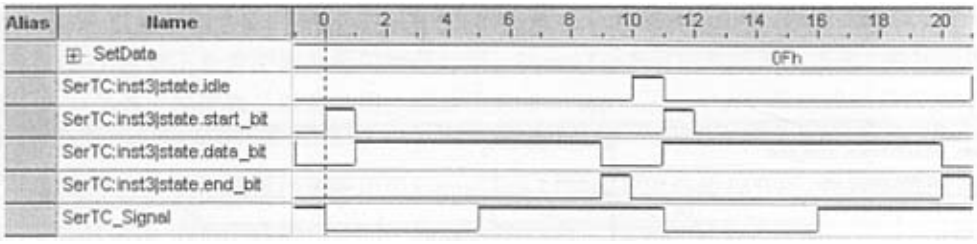

图 4.22 UART 控制器调试结果

Fig. 4.22 Debug Result of UART Controller

4.4.5 相机触发和激光器触发模块调试结果分析

本设计采用 000F 控制字来触发相机曝光，触发模块调试结果如图 4.23 所示。从时序图中，可以看出，在控制字变为 000F 时，采集卡输出一个持续为 6 个时钟宽度的触发信号。逻辑设计符合，设计合理。

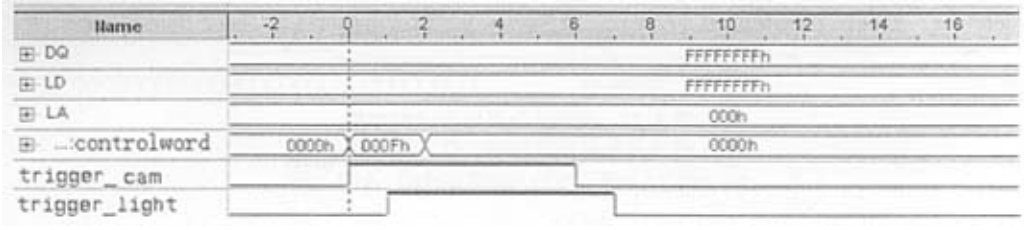

图 4.23 相机触发时序图

Fig. 4.23 Camera Triggering Timing Diagram

4.4.6 图像采集控制模块调试结果分析

图像采集模块在进行行同步时的调试结果如图 4.24 所示。时间轴 0 至 800 为行有效，表示一行中有 1600 个有效像素点。模块将有效图像数据经由 FIFO 缓冲。结果符合设计。

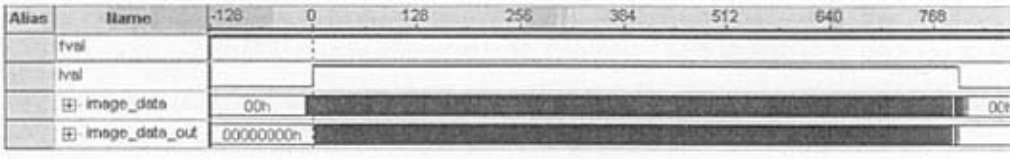

图 4.24 行同步时序图

Fig. 4.24 Horizontal Synchronization Timing Diagram

场同步调试的结果时序图如图 4.25 所示。时间轴‘0’时刻有效后，同样有个场消隐时间，这时候不传输有效像素信息。调试结果符合设计要求。

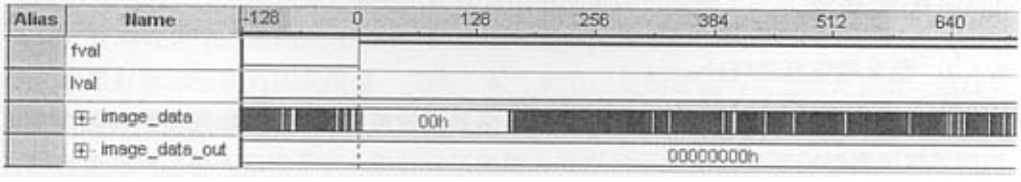

图 4. 25　场同步时序图

Fig. 4.25　Vertical Synchronization Timing Diagram

5 图像采集卡嵌入图像增强算法

5.1 图像增强

5.1.1 图像增强算法理论

图像在采集过程中不可避免的会受到传感器灵敏度、噪声干扰以及模数转化时量化问题等等因素影响而导致图像无法达到人眼的视觉效果，为了实现人眼观察或者机器自动分析的目的，对原始图像所做的改善行为，就被称作图像增强技术[50]。为此图像增强技术虽然是改善图像质量的通用方法，但是它也同样带有针对性，它必须是针对某一特定的需要而采用的特定的算法来实现图像质量的改善。

图像增强技术根据各种不同目的而产生了多种算法，根据处理空间的不同把这些算法分为基于空间域的图像增强算法和基于变换域的图像增强算法。基于空间域的图像增强算法又可以分为空域的变换增强算法、空域的滤波增强算法以及空域的彩色增强算法；基于变换域的图像增强算法可以分为频率域平滑增强算法、频率域的锐化增强算法以及频域彩色增强算法。

本论文采用的是空间域图像增强算法[51]，主要利用一定的图像灰度值映射准则来调整图像灰度的动态范围，从而实现图像的增强。把这种方法描述为：

$$g(x,y)=T[f(x,y)] \tag{5.1}$$

其中，$f(x,y)$和$g(x,y)$表示处理前和处理后的图像。T 表示映射准则。空间域增强的算法简化形式为：

$$s=T(r) \tag{5.2}$$

其中r是$f(x,y)$在任意点(x,y)的灰度级。s是$g(x,y)$在任意点(x,y)的灰度级。

为了灵活地突出兴趣空间，本论文采用分段线性变换映射准则，准则可以表示为：

$$s_i=\begin{cases} f_1(r_i) & (r_i \in R_1) \\ f_2(r_i) & (r_i \in R_2) \\ f_3(r_i) & (r_i \in R_3) \end{cases} \tag{5.3}$$

其中R_1,R_2,R_3表示不同的灰度空间。$r_i \in U_R$是原始图像的灰度值，$s_i \in U_R$为处理后的灰度值。根据需要，我们可选择不同的空间和映射准则实现。

5.1.1 本论文的图像增强算法

本论文采用基于点操作的增强方法，也称为灰度变换[52-53]。采用直接灰度变换方法实现增强原图各部分的反差。实际中往往采用增加原图里某两个灰度值间的动态范围来实现。在本设计中，采用图 5.1 所示的直接灰度变换的方法，横坐标表示原图像的灰度值，纵坐标表示变换后的灰度值，实线为本论文中讨论的增强对比度变换曲线。虚线表示不作任何变换的曲线图。可以看出把原图中的 0 至r_1和r_2至2^n-1间的动态范围明显减小，而原图中灰度值r_1至r_2之间的动态范围增加了。其中n为图像位宽。

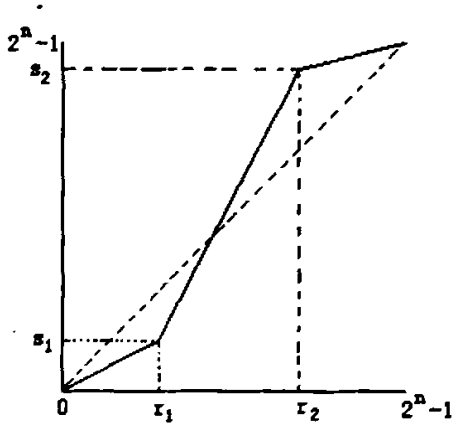

图 5.1 直接灰度变换图

Fig. 5.1 Chart of Direct Gray-level Transformation

这样，分段线性变换的数学表达式[53]如下：

$$g(x,y)=\begin{cases}\dfrac{s_1}{r_1}f(x,y) & f(x,y)\in(0,r_1)\\ \dfrac{s_2-s_1}{r_2-r_1}\left[f(x,y)-r_1\right]+s_1 & f(x,y)\in(r_1,r_2)\\ \dfrac{(2^n-1)-s_2}{(2^n-1)-r_2}\left[f(x,y)-r_2\right]+s_2 & f(x,y)\in(r_2,2^n-1)\end{cases} \tag{5.4}$$

简化表达式，化简如下：

$$g(x,y)=\begin{cases}k_1f(x,y) & f(x,y)\in(0,r_1)\\ k_2\left[f(x,y)-r_1\right]+s_1 & f(x,y)\in(r_1,r_2)\\ k_3\left[f(x,y)-r_2\right]+s_2 & f(x,y)\in(r_2,2^n-1)\end{cases} \tag{5.5}$$

式中参数k_1，k_2，k_3可能是一个小数，为了在 FPGA 里面能够进行计算，首先需要对这三个参数进行定点化处理。整个定点化的工作被嵌入到驱动程序中，用户只需输入接口数据r_1，r_2，s_1和s_2，驱动程序生成定点化结果k_1，k_2，k_3和移位的位数bit_1，bit_2，bit_3送入 FPGA。本论文采用定点化方法是，先将小数进行乘 2 操作(移位操作)，如果先到达整数或者大于 512 提前完成乘 2 操作，否则对其 32 次的乘 2 操作。实现定点化的子程序如下。

```
float_data dingdianhua(float r1)
{
int temp=0;                          //定点化结果
float temp1=0;
int bit=0;                           //乘 2 的次数(移位次数)
float_data data;                     //数据结构
while (bit<32 && r1<512)             //如果循环次数小于 32 而且数小于 512
{
    temp=r1;
    temp1=temp;
    if ((r1-temp)==0)                //如果移为整数，则跳出
        break;
    r1=r1*2;
    bit++;
}
data.float_data1=temp;               //定点化结果
data.shift_bit=bit;                  //移位的位数
return (data);                       //返回乘 2 的次数(移位的位数)和定点化后的数据
}
```

5.2 图像增强算法的 FPGA 实现

本论文利用 FPGA(现场可编程门阵列)的并行、实时处理的特性，实现图像增强的片上集成系统(SOC)。系统将图 5.1 中的r_1，r_2，s_1和s_2设计成接口参数，用户通过主控计算机的应用程序可以反复配置参数，直到得到预期的结果为止。

5.2.1 FPGA 算法的 VHDL 实现

为了方便阐述，把整个 FPGA 实现图像增强算法，分为几个阶段。首先，PC 机通过应用程序送r_1，r_2，s_1和s_2。而后，由驱动程序中的定点化程序将系数进行定点化，后通过 PCI9054 把 5.5 式中k_1，k_2，k_3，r_1，r_2，s_1，s_2和移位参数bit_1，bit_2，bit_3送到 FPGA 的内部寄存器中。这样，FPGA 中嵌入的图像增强算法模块就能从 SDRAM 中取出原始图像数据进行增强，并把经处理后的图像数据存回 SDRAM 中。

图像增强模块首先取回数据，对取回的数据进行判断，把图像数据分为 3 个区间。并作相应的减法。结果跟定点后的系数进行定点乘法，之后将结果数据进行移位操作，然后通过累加输出结果。

常用的并行处理有两种最基本的连接模式：流水线连接和并行阵列连接。针对该算法，采用流水线连接方式进行。在流水线结构中，一个大任务被分解成复杂性大致相同的小任务，各小任务在流水线上同时执行，整个任务的速度取决于执行时间最长的子任务的执行时间。因而，具有处理速度高的特点。在本论文设计中把增强算法模块化分成判断模块，减法模块，乘法模块，移位模块和累加模块，并将其进行流水连接。算法逻辑框图如图 5.2 所示。

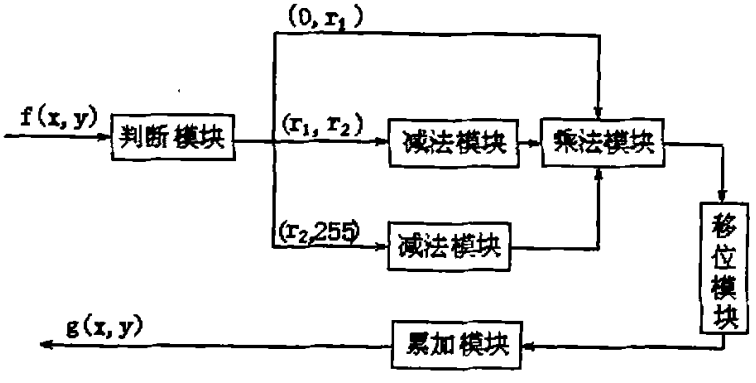

图 5.2 图像增强算法逻辑框图

Fig. 5.2 Ligic Block Diagram of Image Enhancement Arithmetic

Cyclone 器件中的 M4K 块支持软乘法器，在设计中采用 ALTERA 的 IP 实现。乘法器的 IP 核如图 5.3 所示。

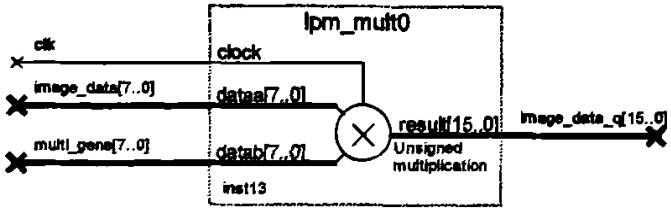

图 5.3　8 位乘 8 位的乘法器

Fig. 5.3　Parameterized Multiplier Megafunction

5.2.2　FPGA 算法调试结果分析

通过 SignalTap 抓取图像值为 0x08 的图像增强算法的调试结果如图 5.4 所示。系数 *k1* 为 1，*bit1* 为 4 时，图像经算法后的像素值 image_data_out 为 1，符合算法结果正确。

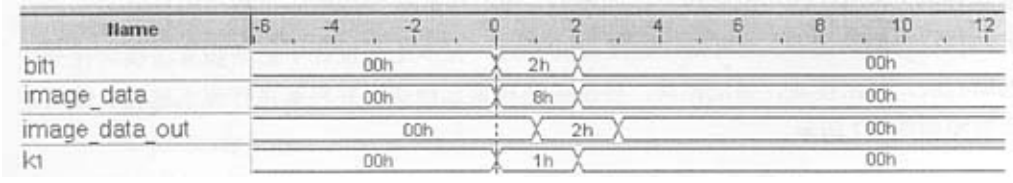

图 5.4　图像增强算法调试结果

Fig. 5.4　Debug Result of Image Enhancement Arithmetic

5.2.3　图像增强的结果分析

选择参数 $r_1=20$，$r_2=80$，$s_1=5$ 和 $s_2=200$，根据公式图 5.5 计算出 $k_1=0.25$，$k_2=3.25$，$k_3=0.314286$。定点化后结果为 $K1=1$、$bit_1=2$，$K2=13$、$bit_2=2$，$K3=321$、$bit_3=11$。图 5.5 所示为本文所设计的图像采集卡采集标定板的原始图像，图 5.7 为标定板进行采集卡进行图像增强后的图像。对比原始图像和增强后的图像，在视觉效果方面有了明显的改善。

图 5.6 所示为原始图像的直方图，图 5.8 为图像增强后的直方图。增强后的图像对灰度从 0 至 20 和从 80 至 255 的像素点进行了压缩，扩大了 20 至 80 的动态范围，使原图像中的斜横明显被减弱。

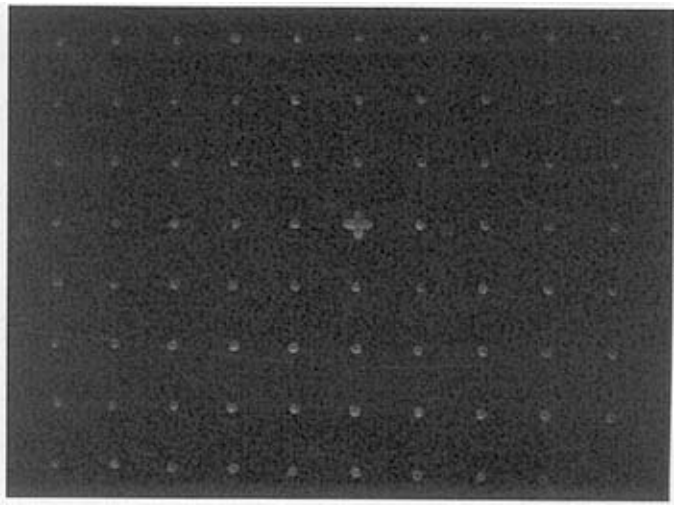

图 5.5　原始图像

Fig. 5.5　Original Image

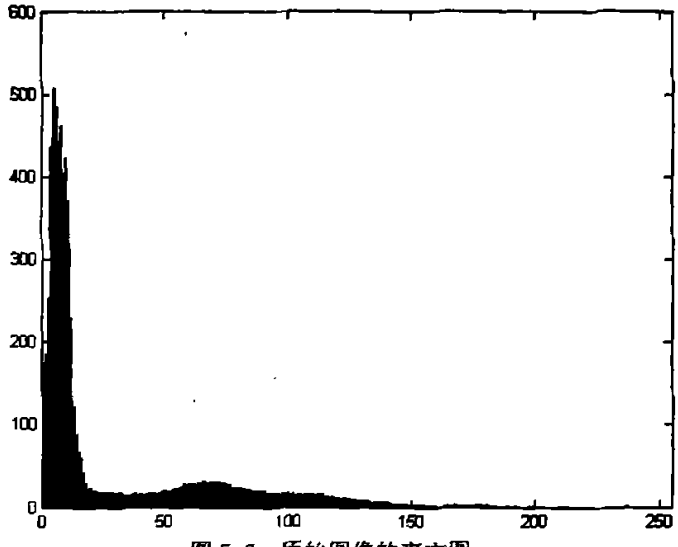

图 5.6　原始图像的直方图

Fig. 5.6　The Histogram of Original Image

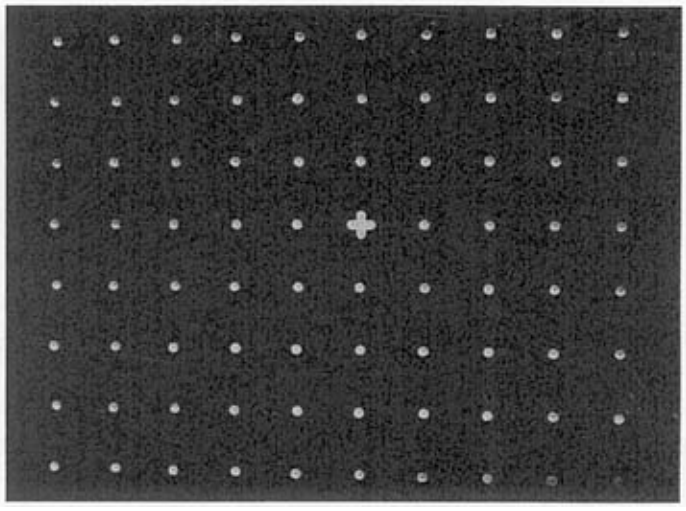

图 5.7　增强后的图像

Fig. 5.7　Enhanced Image

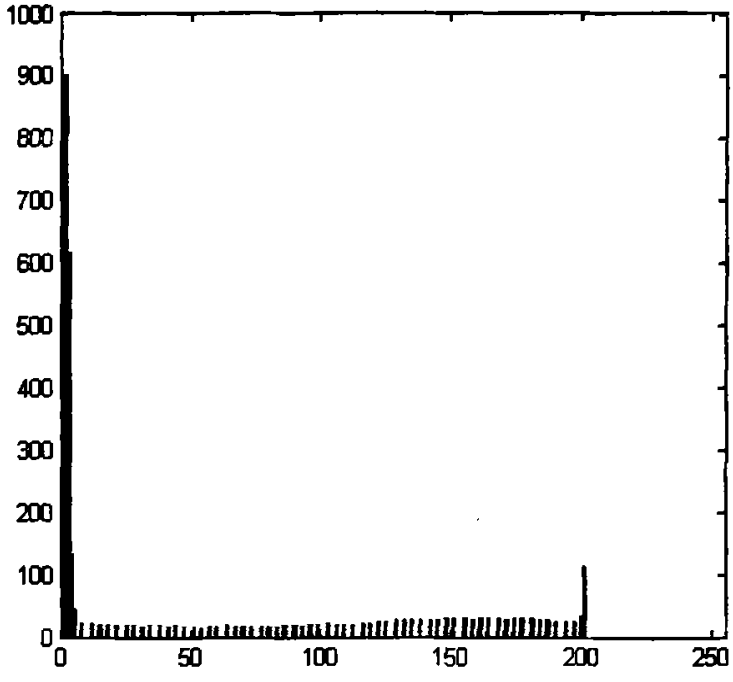

图 5.8　增强后图像的直方图

Fig. 5.8　The Histogram of Enhanced Image

结　论

本文在理解图像采集系统，数字 CCD 相机和分析二维粒子图像测速技术解决方案的基础上，设计实现了基于 FPGA 的嵌入式图像采集处理卡。采集卡控制 IMPERX 公司的 IPX-2M30-L 数字 CCD 相机和同步外部激光源进行图像的实时采集，灰度图像位宽可为 8、10 和 12 位。采集卡以 FPGA 为核心控制芯片，采用 DS90CR282 Channel Chip 把图像数据进行解串，经由 FPGA 两块片内 FIFO 进行乒乓操作，缓存至片外 SDRAM。图像数据传输完成后，FPGA 从 SDRAM 中取出图像数据，进行硬件图像增强。处理完之后，换页存储到片外 SDRAM，请求 PCI 接口将数据传输到 PC 机。本论文完成了以下几部分工作：

(1) 提出了 FPGA+PCI+Camera Link 图像采集系统解决方案，经过详细的评估和论证，认定该方案是可行的。

(2) 完成整个采集卡的硬件电路设计，在 Protel 99SE 环境中绘制出印刷电路板。成功实现了 PCI 时钟、LVDS 和 SDRAM 总线信号等对信号质量要求较高电路的布线，经上电测试整个硬件电路能够正常工作。同时论文中给出了一些板卡调试的意见。

(3) 在 QUARTUS 5.0 环境下，用 VHDL＋原理图的方法设计实现了图像采集卡的各个控制模块和图像增强算法。用 SignalTap 调试 FPGA 的设计模块，并给出调试时序图和仿真结果，对程序运行结果分析表明系统的软硬件设计很好的达到了项目的预期目标。

(4) 对比选择了 DriverStudio 的 DriverWorks 作为驱动程序开发软件，编写了 WDM 底层驱动程序和上层应用程序。程序能够在 Windows XP 和 2000 系统中稳定运行，满足设计的要求。

(5) 在 FPGA 内嵌入了图像增强集成系统，用硬件并行处理实现，经仿真可行。

该图像采集卡由于硬件 FPGA 本身的局限性，浮点运算能力有待进一步的提高。采用浮点 DSP 和 FPGA 组合，或选择性能更优 Stratix 系列的 FPGA 作为核心器件，将是一个不错的选择。本论文采用 33MHz 的 32 位 PCI 总线设计，总线速率可以进一步的提高。可以考虑提高 PCI 总线频率至 66MHz，或将总线宽度增加至 64 位。本文采用的 Camera Link 接口标准，虽然在传输距离、传输速率和兼容性方面较早期的 RS-644 或 RS-422 技术标准有了很大的提高，但仍然存在一定的弊端。千兆网数字接口标准的出现，将会在传输距离方面得到一定的改善。

参 考 文 献

[1] 崔恒.流体二维图像测速系统的研究:(硕士学位论文).大连:大连理工大学,2006.

[2] Toshihito Fujiwara,Kenji Fujimoto,Tsutomu Maruyama.A Real-Time Visualization System for PIV[J].Lecture Notes in Computer Science, 2003, Volume 2778:437 - 447.

[3] 章丽萍,周凤星.基于EP1C3T144 的最小系统开发板的设计[J].武汉科技大学学报,2007,30(3):293-295.

[4] 任爱峰,初秀琴.基于 FPGA 的嵌入式系统设计[M].西安:西安电子科技大学出版社,2005:17-26,333-354.

[5] National Semiconductor Corporation.LM1085 3A Low Dropout Positive Regulators[EB].August 2001.

[6] 吴继华,王诚.Altera FPGA/CPLD 设计[M].北京:人民邮电出版社,2005:66-69.

[7] 房磊,张焕春,经亚枝. FPGA的配置及接口电路[J].电子质量,2004,(01):2-3

[8] Altera corporation.Byterblaster parallel port down cable data sheet[EB].1988, ver.2.01:299-313

[9] 陈东,余松煜.PCI总线规范及接口[J].微型机及应用,1996-8.

[10] PCI Local Bus Specification Revision 2.1[EB].June 1995:4-6.

[11] PCI Local Bus Specification Revision 2.2[EB].December 1998:7-20.

[12] 吴业进,刘锋.PCI9052总线接口芯片及其ISA模式应用[J]. 电子工程师,2003,(04):24-26

[13] PLX Technology,Inc.PCI 9054 Data Book [EB].V2.1,January 2000.

[14] 黄身锞.PCI总线接口芯片9054及其应用[J].世界电子元器件,2006,(10):57-59.

[15] PLX Technology,Inc.PCI SDK SOFTWARE DEVELOPMENT KIT User's Manual[EB].Version 4.10, May,2003:57-69.

[16] PLX Technology, Inc .PCI SDK SOFTWARE DEVELOPMENT KIT Programmer's Reference Manual[EB].Version 4.10, May,2003.

[17] 李宁,汪骏发.基于Camera Link的高速数据采集系统[J].红外,2005,(07):31-37.

[18] Basler Vision Technologies.Camera Link Technology Brief[EB].March 2001:6-20.

[19] 张涛,舒林锋,郑冬军等. LVDS接口原理和标准及在平板显示系统中的应用[J].计算机与数字工程,2007,(09):184-186.

[20] National Semiconductor Corporation.DS90CR281/DS90CR282 28-Bit Channel Link[EB].November 1996:1-15.

[21] LYNX Inc. .LYNX User's Manual RA04[EB].August 2005.

[22] 陈炳权. 基于FPGA器件的RS232—C接口设计及其扩展[J].攀枝花学院学报,2006,23(05):106-109

[23] National Semiconductor Corporation. DS90LV047A 3V LVDS Quad CMOS Differential Line

Driver[EB], July 1999.
[24] National Semiconductor Corporation. DS90LV048A 3V LVDS Quad CMOS Differential Line Driver[EB], July 1999.
[25] Texas Instruments Incorporated. MAX232 dual EIA-232 driver/receiver datasheet, 2004.
[26] Hynix Semiconductor Inc. HY57V283220(L)T(P)/ HY5V22(L)F(P) 4 Banks x 1M x 32Bit Synchronous DRAM Rev. 0.9[EB], July 2004.
[27] 李贵山,陈金鹏.PCI 局部总线及应用[M].西安:西安电子科技大学出版社, 2003:8-16.
[28] 崔玮.Protel 99SE电路原理图与电路板设计教程[M].北京:海洋出版社,2005.
[29] 张伟,王力.Protel 99SE基础教程[M].北京:人民邮电出版社, 2006.
[30] 尹勇, 李宇.PCI总线设备开发宝典[M].北京:北京航空航天大学出版社,2005:25-26,130-138.
[31] 武安河.Windows 2000/XP WDM 设备驱动程序开发[M].北京:电子工业出版社,2005:5-7,323-352.
[32] 刘蓬,张培仁.用 DDK 开发 WDM 驱动程序[J]. 计算机应用,2003,23(S2):248-250.
[33] 游南林.用 DDK 开发 Windows NT 下的设备驱动程序[J].微型机与应用,1999,(08):8-11.
[34] 张增辉,沈澈,陈子瑜.基于 WinDriver 工具的 PCI 卡驱动程序开发[J]. 核电子学与探测技术,2006,26(3):267-369.
[35] 宋玉贵,王世凯,李海. 基于 Windriver Kernel PlugIn 的驱动程序设计[J]. 西安工业学院学报,2005,25(4):311-314.
[36] Compuware Corporation.Using DriverStudio Development Tools Release 3.1[EB].2003.
[37] 孟华.用DriverWorks开发PCI设备的WDM 驱动程序[J],电子与信息工程,2006-4(2):121-124.
[38] 陈富章,李伟光,高严松等.基于PCI总线运动控制卡WDM 驱动程序设计[J],微计算机信息,2007-23(7-1):213-215.
[39] 司玉美,申会民,耿爱辉等.基于PCI 总线数据通信卡WDM 驱动程序设计[J].计算机测量与控制,2006,14(2):259-261.
[40] 邰铭,武安河.WINDOWS下PCI接口卡WDM驱动程序的DMA编程技术[J]. 微计算机信息,2003,19(10):79-80.
[41] Chris Cant著.Writing Windows WDM Device Drivers.孙义,马莉波,国雪飞等译.Windows WDM 设备驱动程序开发指南[M].北京:机械工业出版社,2000:179-203.
[42] 龚建伟,熊光明.Visual C++/Turbo C串口通信编程实践[M].北京:电子工业出版社, 2004.4-12
[43] 童鹏,吴新建.PCI9054 芯片接口设计中若干问题的深入研究[J].电子技术应用,2005,(10):64-66.
[44] Altera Corporation.SDR SDRAM Controller White Paper ver1.1[EB],August 2002:1-16.
[45] 田丰,邓建国,李巍等.SDRAM控制的设计与 VHDL 语言实现[J].电子技术应用,2005(12):74-77.
[46] 胡振华.VHDL 与 FPGA 设计[M].北京: 中国铁道出版社,2002:157-175.
[47] Micron Technology,Inc.SYNCHRONOUS DRAM data sheet[EB].2002.
[48] 刘锐,楼兴华.FPGA 数字电子系统设计与开发实例导航[M].北京:人民邮电出版社,2005:134-178.

[49] 郭佳佳,胡晓菁. 使用 SignalTapII 逻辑分析仪调试 FPGA[J],今日电子, 2005,(5):45-47.

[50] 俞诗鲲,郑建生,曾欣. 用 CPLD 实现嵌入式平台上的实时图像增强[J]. 电子技术应用,2003,(12):10-12.

[51] 韩娟娟,邓文怡,娄小平. 基于 FPGA 的图像增强处理系统的设计与实现[J]. 微计算机信息,2007-23(9-2):229-230.

[52] 何东建. 数字图像处理[M]. 西安:西安电子科技大学出版社,2003:54-63.

[53] 章毓晋. 图象处理与分析[M]. 北京:清华大学出版社,1999:72-80.

附录 A 相关程序段

相机触发程序

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_unsigned.all;
--  Entity Declaration
ENTITY CC1_Trigger IS
  PORT
  (
   start_trigger : IN STD_LOGIC;      --触发命令字的译码信号
   fval: IN STD_LOGIC;                --帧同步
   camera_clk : IN STD_LOGIC;         --相机同步时钟
   reset_n : IN STD_LOGIC;            --系统复位信号
   trigger_out : OUT STD_LOGIC        --触发输出
  );

END CC1_Trigger;
--  Architecture Body
ARCHITECTURE CC1_Trigger_architecture OF CC1_Trigger IS

TYPE state_trigger IS
(
idle,
trigger_generate
);
SIGNAL state:state_trigger;
SIGNAL counter :STD_LOGIC_VECTOR(2 DOWNTO 0);
BEGIN
   PROCESS(state)                          --状态声明线程
   BEGIN
      CASE state IS
         WHEN idle =>
            trigger_out<='0';
```

```
        WHEN trigger_generate =>
            trigger_out<='1';
    END CASE;
  END PROCESS;
  PROCESS (camera_clk,reset_n)      —状态转移线程
  BEGIN
    IF reset_n ='0' THEN
       state<=idle;
    ELSIF rising_edge (camera_clk) THEN
       CASE state IS
          WHEN idle =>
             counter<="000";
             IF (start_trigger = '1' and FVAL = '0')   THEN
                state<=trigger_generate;
             ELSE
                state<=idle;
             END IF;
          WHEN trigger_generate =>
             IF counter ="111" THEN  —持续 8 个有效电平
                state<=idle;
             ELSE
                state<=trigger_generate;
                counter<=counter+1;
             END IF;
       END CASE;
    END IF;
  END PROCESS;
END CCl_Trigger_architecture;
```

攻读硕士学位期间发表学术论文情况

李木国, 彭平良. 基于 FPGA 的运动控制卡的设计与实现. 计算机工程与设计, 2008, 5.

（属于本论文中的第 4 章第三部分的内容）

致　谢

本论文是在李木国教授的悉心指导下完成的，李老师渊博的专业知识，严谨的治学态度，精益求精的工作作风，诲人不倦的高尚师德，严以律己、宽以待人的崇高风范，朴实无华、平易近人的人格魅力对我影响深远。不仅使我树立了远大的学术目标、掌握了基本的研究方法，还使我明白了许多待人接物与为人处世的道理。本论文从选题到完成，每一步都是在恩师的指导下完成的，倾注了恩师大量的心血。在此，谨向李木国教授表示崇高的敬意和衷心的感谢！

本论文的顺利完成，离不开教研室师兄、师姐和师弟师妹门的关心和帮助。在此感谢张群老师、王静老师、杜海师兄和王磊师兄，他们知识渊博、工作严谨、待人诚恳热情，在论文完成过程提出了许多的宝贵的见解。感谢师弟方辛、胡永军和谢香林，在我完成论文期间给予的帮助和支持。

感谢我的父母，感谢他们这么多年对我的养育和支持。感谢我的亲人，感谢他们对我的关心和帮助。

感谢我的爱人李晓宁，在我论文的修改过程给予的莫大帮助。

同时还要感谢师母几年来在生活上给予我的关心和帮助。