分类号 _____ 学号 M200970287

学校代码 10487 密级 _____

华中科技大学

硕士学位论文

实时图像旋转系统的研究与 FPGA 实现

学位申请人：王金辉

学科专业： 机械电子工程

指导教师： 陈 冰 讲师

陈幼平 教授

答辩日期： 2012 年 01 月 05 日

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering

Study and Implementation of the Real-time Image Rotation System Based on FPGA

Candidate : Wang Jinhui

Major : Mechatronic Engineering

Supervisor : Lecturer.Chen Bing

Prof. Chen Youping

Huazhong University of Science & Technology

Wuhan 430074, P.R.China

January, 2012

独创性声明

本人声明所呈交的学位论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除文中已经标明引用的内容外，本论文不包含任何其它个人或集体已经发表或撰写过的研究成果。对本文的研究做出贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

学位论文作者签名：

日期：　　年　　月　　日

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，即：学校有权保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权华中科技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，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。

本论文属于　　保密□，在_____年解密后适用本授权书。

不保密□。

（请在以上方框内打“√”）

学位论文作者签名：　　指导教师签名：

日期：　　年　　月　　日　　日期：　　年　　月　　日

摘 要

目前，图像处理面临最主要的问题是处理速度的高速化。图像旋转作为基本的图像几何变换，需要处理大量数据，而且涉及非顺序存储访问，是图像处理系统实时性的瓶颈所在。与计算机、DSP 相比，FPGA 有可编程、可重配置、并行运算的优点,非常适合高速数字信号处理。因此，本文对基于 FPGA 的实时图像旋转系统展开研究。

为了实现实时旋转，需要提高系统的处理能力，有以下三种办法：1）并行运算，由多个处理器共同完成处理任务；2）优化算法，用更简单、快捷的方法实现，从而提高效率；3）提高系统主频，选择更优的存储器访问方案。针对上述三种办法，本文设计的实时图像旋转系统采用并行运算的可编程器件 FPGA 作为视频流处理器；经过分析对比选用适合 FPGA 的直接法完成坐标变换；而且采用了 SDRAM、SSRAM 两种类型的存储器芯片以适应顺序、非顺序的访问情况，提高访问效率。设计的过程中，充分考虑到硬件实现图像处理的特点，采用乒乓操作、分时复用、流水线等高速数字电路设计技术，利用 FPGA 丰富的资源提高图像旋转处理的实时性。

整个系统设计在 Quartus Ⅱ平台下，结合层次化、模块化思想，用 VerilogHDL 实现。同时，使用 TimeQuest 工具完成时序约束工作，提高系统的运算能力和稳定性。最终该系统在通用图像处理平台上得到验证，处理能力强，能够满足实时性要求。

关键词：实时图像处理 图像旋转 FPGA 流水线

Abstract

At present, the major problem in image processing is the high requirements on processing speed. As the basic transformation, image rotation becomes to be the bottleneck of the real-time features of the whole image processing system, because it needs to handle large amounts of data, and involves the non-sequential memory access. Comparing with the computer and DSP, FPGA is programmable, reconfigurable and has the feature of parallel computing, which makes it suitable for high-speed digital signal processing. Therefore, this paper studies on the real-time image rotation system based on FPGA.

To realize the real-time image rotation, processing capacity should be improved. There are three methods. Firstly, parallel computing, which means that two or more processor are used; secondly, optimizing the algorithm to get higher efficiency; thirdly, improves the system frequency and chooses the better memory access scheme. In this paper, the following measures are adopted to make the methods to be practice: using FPGA as the video stream processor; choosing the direct method which is suitable for FPGA to realize the coordinate transformation after analysis and comparison. And furthermore, in order to meet the requirements of sequential and non-sequential access, both SDRAM and SSRAM are used. During the design, the design methods of high-speed digital circuit like ping-pong operation, time-sharing reuse, pipeline operation are took to improve the processing speed of image rotation with the full consideration of the characteristics of image processing based-on FPGA.

The system is designed in Quartus Ⅱ platform, combined with hierarchical and modular thoughts, and programmed with VerilogHDL. Meanwhile, the timing closure is achieved using TimeQuset tool, thus computing power and stability of the system is improved. Finally, the system is verified on the common image processing platform, and its processing capability can meet the processing speed requirements for the real-time image processing system.

Keywords: Real-time Image processing, Image rotation, FPGA, Pipeline

目 录

1 绪论

1.1 课题来源与研究背景

1.1.1 课题的来源

本课题得到以下科研项目的资助：

湖北省自然科学基金：快速鲁棒多目机器视觉质量检测方法与技术研究（资助号：2005ABA269）。

1.1.2 课题的研究背景

随着计算机、微电子技术的飞速发展，数字图像处理技术已经在科学研究、工业生产、医疗卫生、教育、娱乐、管理、和通信等方面都得到了广泛的运用。它对推动社会发展，改善人们生活水平有很大帮助[1]。由于这些运用对实时性以及图像分辨率都有较高的要求，近年来大数据量的实时图像处理一直是一个非常活跃的研究领域。图像旋转作为数字图像处理的一个重要组成，在工业自动化、机器视觉、卫星遥感、医疗诊断等领域有很好的运用[2]。例如自动化工业现场工件识别与定位、电子装配线的元件自动定位以及管脚数目检测、IC 芯片上的字符识别、航空领域高分辨率数字图像显示处理、医疗方面实现三维图像重建等等[3]。此外，研究实时图像旋转对解决电子消旋有很大的帮助。例如在机载电视激光摄像与瞄准过程中，电视激光摄像框架的横滚运动引起光学系统核成像器件相对载机的运动，使得图像旋转，影响飞行员的观察一级操作，需要对目标图像进行实时反旋转变换，以便于对目标图像进行自动识别以及定位[4]。实时图像旋转过程中，需要处理的数据量大、实时性强，是提高整个图像处理系统实时性的关键所在，因此选择合适的图像处理系统平台，提高图像旋转的实时性变得尤为重要。

传统的图像处理系统平台选择包括：计算机、专用图像处理芯片以及 DSP 芯片。近几年，随着半导体加工工艺以及现场可编程门阵列（FPGA,Field Programmable Gate Array）技术的发展，越来越多的图像处理设备提供商以及科研机构选择 FPGA 作为新的图像处理系统的平台[5] 。

FPGA 是在 PAL、GAL、EPLD、CPLD 等可编程器件的基础上进一步发展的产物。

它作为半定制电路出现，即解决了定制电路的不足，又克服了原可编程器件资源有限的缺点[6]。目前,FPGA 已经从 2004 年的 90nmFPGA 发展到现在的 28nmFPGA，这意味这更高的密度、更好的性能以及更低的功耗。再通过内嵌块存储器、DSP 运算单元甚至 ARM 内核等方式，极大地方便了基于 FPGA 的数字信号处理系统的开发。基于 FPGA 的数字图像处理系统具有高性能、灵活性、方便更新以及低成本的优点。FPGA 平台的数字图像处理系统的设计方法与其它平台的有很大的不同。大规模的逻辑单元使得可以在一片 FPGA 上实现多样的视频采集、显示接口以及复杂的处理要求；可编程特性让体系结构可以自由更新，以满足不断发展的要求；运算的并行性更容易满足实时图像处理大数据量以及实时性的特性。

鉴于此，本文对基于 FPGA 的实时图像处理系统展开研究，设计 FPGA 平台的图像处理系统，并实现实时图像旋转的功能。

1.2 国内外研究现状

科学研究和统计表明，人类从外界获得的信息约有 75％来自于视觉系统[1]。早期的数字图像处理技术出现的主要目的是为了改善图像采集设备的成像质量，以方便人类获取信息。20 世纪 60 年代初期，美国喷气推进实验室首次成功应用数字图像处理技术对太空船返回月球图片信息进行处理。随着计算机技术、多媒体技术以及人工智能技术的不断向前发展，数字图像处理技术受到了前所未有的广泛重视，并获得了尤为突出的进步。数字图像处理技术已经走出了实验室，走入到更为广阔的应用领域中，包括工业领域、航空航天领域、生物医疗领域、安防领域，甚至走入了我们的日常生活[7]。对实时图像旋转系统的研究，重点在于图像旋转算法以及系统体系结构。

国外对图像旋转算法的研究可追溯至 20 世纪 80 年代。1980 年 Catmull 和 Smith 在硬件上用两步法实现了图像旋转[8]，但是第一步变换中，丢失图像中的高频部分，图像质量下降明显。Paeth 和 Tanaka 在 1986 年提出了三步实现的办法，只需通过三次平移就可以完成图像旋转处理，便于硬件实现，但是图像质量未得到很好的改善。1996 年，Kiern 提出用快速傅里叶变换实现图像旋转的方法[9]，不同于直接法、两步法或三步法在完成坐标变换后需要进行图像插值，该方法将坐标变换与图像插值合二为一，但是计算量庞大，不利于硬件。

近几年，国内对在不同的系统体系结构中满足图像旋转算法的实时性要求以及提高目标图像质量上展开了研究，并取得了众多成果。

2007 年，浙江工业大学信息学院研究了基于 GPU 的图像快速旋转算法的实现[10]。

基于计算机的图像处理平台，因为 CPU 任务繁重，通常需要设计人员对算法或者代码进行优化，即“软加速”。即便如此，通常在面对实时图像或者复杂处理时，CPU 的处理能力仍然显得力不从心，通常作为非实时图像处理平台和图像管理工作站平台。后来为了提高计算机的图像处理能力，在计算机中引入 GPU 协同 CPU 工作。CPU 运算非常复杂的序列代码，而 GPU 则运行大规模并行处理程序，实时图像处理能力有了很大提高，但如何让 CPU 与 GPU 能够更好地配合以最大地提高工作效率仍然是处理器厂商考虑的问题。

西北工业大学和中国航空无线电电子研究所联合对基于 DSP 的像旋转算法数据调度策略进行深入的研究[11]。在实时图像处理中，DSP 仍有它的瓶颈。实时图像要求每秒输出数十张的画面，通常为 24~30。实时图像处理的数据量非常大，因此它既要求 DSP 有高速的数据处理能力，也需要能快速的传输数据。DSP 芯片片内的存储空间相当有限，必须通过外扩存储器的办法来缓存待处理的数据，这样必然会增加 DSP 读取数据的时间，同时也削弱了 DSP 高速运算能力。而且，DSP 只有一个 CPU 进行运算，如果需要对图像进行模板操作，将会非常的耗费资源并且难于保证实时性。如果需要对图像进行模板操作，将会非常的耗费资源并且难于保证实时性。当然，有设计人员提出了采用 DSP 阵列的方法来实现并行算法，提高运算能力、改善实时性。国外学者 ohit S．WhtVe 等人研究了采用多 DSP 作为处理单元，对旋转算法和多种插值算法进行了对比研究。

在超大规模集成电路（VLSI）的今天，支持多种接受标准和编解码格式的专业图像处理芯片层出不穷，占据着很大的市场份额。虽然它们芯片尺寸小、功能强、功耗低，但其设计复杂，有批量要求，其中两步平移、三步平移方法适合硬件实现，已经有相应的 VLSI 结构提出。2009 年，华中科技大学图像所设计了三步平移双三次插值算法的流水线图像旋转 ASIC 架构，但是该架构只能实现-45 度到 45 度间的旋转，而且最大只支持 690X512 大小图像。

随着 FPGA 技术的发展，FPGA 已经对专业图像处理芯片构成很大的威胁。过去，FPGA 通常被用做协处理器。随着生产规模的提高，产品应用成本的下降，FPGA 常被用于系统级的设计之中。基于 FPGA 的图像处理系统可以充分利用 FPGA 高速、并行计算的特点，通过设计流水线的方式提高数字图像的处理速度，以提高系统性能，满足大数据量、实时性的处理要求。文献[12~14]都对 FPGA 平台的体系结构展开研究。虽然当前用 FPGA 进行视频图像处理已经取得了很大的进展，但是仍有很多待解决的问题，比如高实时性要求、大分辨率图像处理、复杂图像处理算法的硬件实现等。

1.3 本文研究内容及章节安排

本文的主要研究内容有：

1) 研究图像旋转算法，对比各种旋转以及插值算法，提出能够满足图像旋转处理的实时性以及便于硬件实现的处理算法。
2) 研究从算法设计到 FPGA 硬件实现的方法，讨论时分复用、流水线设计以及乒乓操作等方案，以期提高系统性能，能够满足图像实时性的处理要求。
3) 建立基于 FPGA 的实时图像处理系统，完成总体方案设计以及各功能模块的代码实现及验证工作。

本文主要对基于 FPGA 平台的实时图像旋转系统进行了研究。全文共 6 章，各章节主要内容如下：

第一章首先讨论了课题的来源，分析课题的研究背景，并在此基础上提出了课题研究的意义。然后回顾了近几年国内外在图像旋转算法以及图像处理系统体系结构上做的研究工作。最后介绍了论文的研究内容以及组织结构。

第二章介绍了传统的图像坐标变换旋转算法；然后分析旋转后目标图像出现锯齿效应，影像图像质量的原因；最后提出用双线性内插值算法提高图像质量。

第三章设计了基于 FPGA 的实时图像处理系统，对系统的各个子功能模块，包括视频采集与显示、显存、FPGA、人机界面等，进行详细说明。

第四章研究如何在基于 FPGA 的实时图像处理系统上实现实时图像的旋转。首先介绍了通过时分复用方法实现多端口存储器的设计；然后重点介绍了从旋转算法设计到 FPGA 硬件实现的过程，其中包括流水线设计、乒乓操作等设计方法；最后使用时序约束工具 Timequest 对设计进行分析、优化，让设计能够时序收敛，
从而达到更高的性能。

第五章利用实验平台对设计进行性能分析，最后给出系统消耗资源情况以及实际运用结果。

第六章对全文进行了总结，同时展望下一步的工作。

2 图像旋转算法分析

在大多数运用中，被测物与相机存在相对运动，不能保证总在图像中处于同样的位置和方向。图像旋转是调整物体到检测所需位姿常用手段。一般，图像旋转由两个基本操作组成：空间坐标变换和灰度级插值[15]。传统的空间坐标变换都是直接由旋转坐标公式得到。研究人员为了提高运算效率[16]，在其基础上进行了改进，使用坐标变换的方法，又提出了二步法、三步法、相对位置法。其中三步法适用最为广泛，又称错切法。完成坐标变换后，目标图像的质量比较粗糙，通常有比较明显的锯齿效应，需要通过双线性插值方法提高图像质量。

2.1 空间坐标变换

每帧实时图像都可以看作二维像素点阵列，我们可以通过建立合适的坐标系，给每个像素点确定一个坐标。图像旋转可以通过坐标变换得到。实现坐标变换有两种方式：1）向前映射，遍历原始图像所有像素点，并计算其在目标图像中相对应点的位置；2）向后映射，遍历目标图像中所有像素点，并计算其在原始图像中相对应点得位置。为了确保能对目标图像中所有像素点进行设定，我们采用向后映射的方式。

2.1.1 直接法

不妨设旋转中心在图像中央，原始图像为 f(x,y),目标图像为个 $g(x',y')$。则映射关系可用式 2-1 表达：

$$g(x',y')=f(x,y)=f[m(x',y'),n(x',y')] \tag{2-1}$$

其中，m(x,y)、n(x,y)函数表达了像素坐标之间的映射关系。需要指出的是，f(x,y)只在整数坐标点有定义，而通常由 m(x,y)、n(x,y)函数映射得到的坐标点为非整数坐标，这是旋转后目标图像质量下降的原因。

在图 2.1 中，点 $r(x,y)$ 旋转 α 角度后变成 $r(x',y')$。那么 (x,y)和(x',y') 之间存在如下关系：

$$\begin{cases} x'=x*\cos\alpha+y*\sin\alpha \\ y'=-x*\sin\alpha+y*\cos\alpha \end{cases} \tag{2-2}$$

对式（2-2）做变换，我们就可以得到前面所说，向后映射的变换公式。

$$\begin{cases} x = x' * \cos\alpha - y' * \sin\alpha \\ y = x' * \sin\alpha + y' * \cos\alpha \end{cases} \tag{2-3}$$

将式（2-3）用矩阵形式表示，如：

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} \tag{2-4}$$

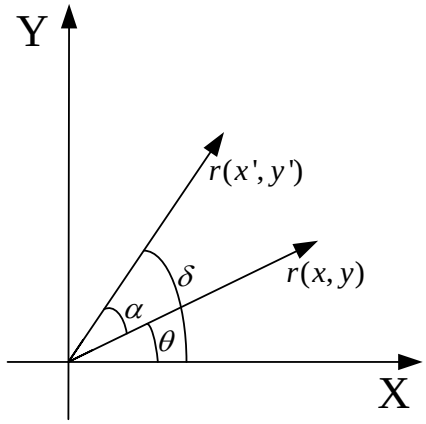

图 2.1 旋转坐标变换

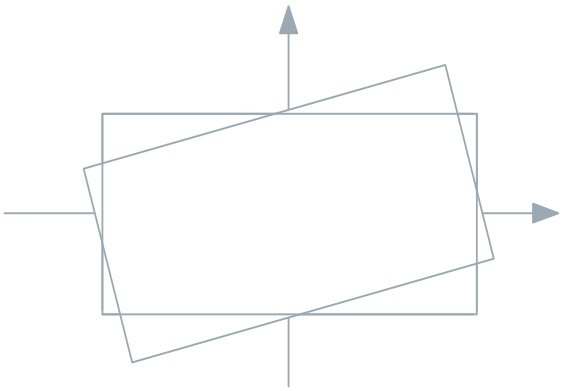

图 2.2 旋转矩阵超出

使用式（2-4）完成图像旋转，就是通常所说的直接法。它通过遍历目标图像中的每一个像素点，在原始图像中寻找对应点。如图 2.2 所示，旋转后，部分目标图像会超出原始图像范围，超出的部分按灰度级最低处理。直接法对每个像素点都需要进行一次矩阵变换，计算量大，因此研究人员对它进行了改进。

2.1.2 错切法

式（2-4）可以等效为式（2-5）：

$$\begin{bmatrix} x \\ y \end{bmatrix} == \begin{bmatrix} 1 & 0 \\ \tan(\alpha/2) & 1 \end{bmatrix} \begin{bmatrix} 1 & -\sin\alpha \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \tan(\alpha/2) & 1 \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} \tag{2-5}$$

通过对传统的坐标旋转公式的分解，二维空间中的旋转运算分解被为三次一维空间中的平移运算。这种方法被称为错切法，也是当今广为采用的方法。错切法很大程度上减少了运算量，为硬件实现实时图像旋转提供可能。但是由于执行三步运算，需要缓存 3 次图像，占用存储空间，而且得到的图像质量较低。

2.1.3 相对位置法

如图 2.3 所示，图片在旋转后，点 $A_{00}A_{01}A_{10}A_{11}$ 保持相对位置不变。因此，只需 $A_{00}A_{01}A_{10}$ 的位置，我们就可以知道 A_{11} 的位置，如下式：

$$\begin{cases} x'_{11} = x'_{01} + x'_{10} - x'_{00} \\ y'_{11} = y'_{01} + y'_{10} - y'_{00} \end{cases} \tag{2-6}$$

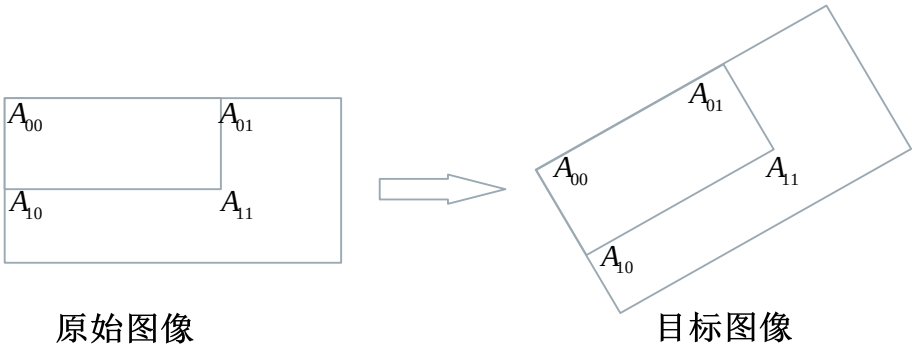

图 2.3 相对位置法

通过相对位置法，我们只需对第一行和第一列的像素点依直接法做矩阵运算，其余像素点用式（2-6）做加减法即可，避免了大量的矩阵运算。

上述三种方法都非常适合硬件实现。直接法一步到位，而且避免了中间运算引入误差，损失精度。因此，我们选择在 FPGA 中直接法完成坐标变换。

2.2 锯齿效应

当前，终端显示器都由点阵的像素点构成，在表现非水平或者竖直的直线时，必然会出现直线的歪曲，即锯齿效应。位图经过缩放、旋转等变换，都会使锯齿效应增强。如图 2.4 所示，因为显示分辨率过低，在表现非水平或竖直的直线以及明显边缘时，离散的像素点间，灰度值缺少过渡，因此有很明显的锯齿[17]。对于这种情况，我们需要对图像边缘进行柔化处理，使图像边缘看起来更平滑，更接近现实[18]。目前，GPU 采用了最新的全屏抗锯齿（FullScreenAnti-Aliasing）技术就是将图像边缘及其两侧的像素颜色进行混合，然后用新生成的具有混合特性的点来替换原来位置上的点以达到柔化物体外形、消除锯齿的效果。

依空间坐标变换方法，计算得到的对应点坐标通常不是整数。如果简单地使用取整或者最邻近插值的方法得到目标图像，目标图像会有比较明显的锯齿效应。对于这个问

题，有三种方法：1）出现锯齿效应是因为显示器的分辨率过低。最简单的办法就是增加采样点，即减小显示器的像素尺寸，由于显示器通常已经固定了，此方法不太可行；2）对像素矩阵进行滤波，因为走样通常由于图形突变而发生，利用低通滤波器削除高频成分，可以减少图形的失真，这也是早期的 GPU 抗锯齿方法；3）因为方法 2）会造成边缘损失，为了降低边缘损失，将边缘及平坦部分区别对待，选择不同的图像差值方法[19]。

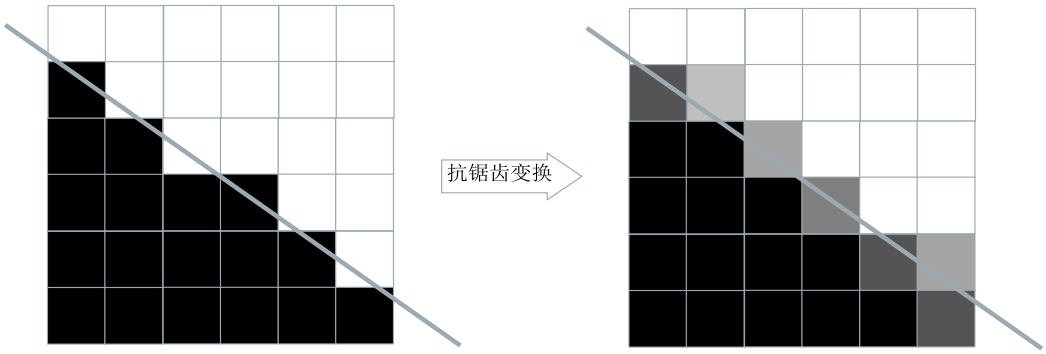

图 2.4 锯齿效应

比较上述三种方法，其中方法 3 得到的图像效果最好，但是其计算复杂度也大，不利于硬件实现以及保证实时性。综合考虑图形质量以及设计复杂度，我们选用方法 2 完成图形的抗锯齿的效果，常用的就是双线性插值算法。

2.3 双线性插值

传统的图像插值算法包括：最邻近插值、双线性插值以及三次样条插值。综合比较，最邻近插值计算简单，但是目标图像质量下降明显；三次样条插值考虑邻近像素点的灰度值及其变化率，因此精度最高，但其计算较为复杂；采用双线性插值算法，它能够很好地避免锯齿效应，得到的图像质量好，而且易于硬件实现[20~22]。因此，我们选用双线性插值算法完成目标图像的插值。

根本上来说，双线性插值算法是一种加权算法，如图 2.5 所示。它以到最近四个像素点的距离为参考权值，经两次线性插值，综合得到当前点的灰度值。计算公式如下：

$$\begin{cases} f(dx,0) = f(0,0)*(1-dx) + f(0,1)*dx \\ f(dx,1) = f(1,0)*(1-dx) + f(1,1)*dx \\ f(dx,dy) = f(dx,0)*(1-dy) + f(dx,1)*dy \end{cases} \quad (2\text{-}7)$$

从公式中我们可以看出，目标点的灰度值由周围四个像素点依权值取平均得到，有

低通滤波的效果，故而能够起到抗锯齿的效果。而且，公式形式非常整齐，在 FPGA 中很容易用乘加器（MAC）实现。如图 2.6 所示，第一个乘加器实现 $f(dx,0)$ 的计算，第二个乘加器实现 $f(dx,1)$ 的计算。然后再加计算结果输入给第三个乘加器，得到最终的插值结果 $f(dx,dy)$。因此，只需设计好时序，在不同的时钟周期将像素值以及对应的权值输入给乘加器即可。而且，乘加器设计为流水线结构，可以提高整个系统的数据吞吐量。

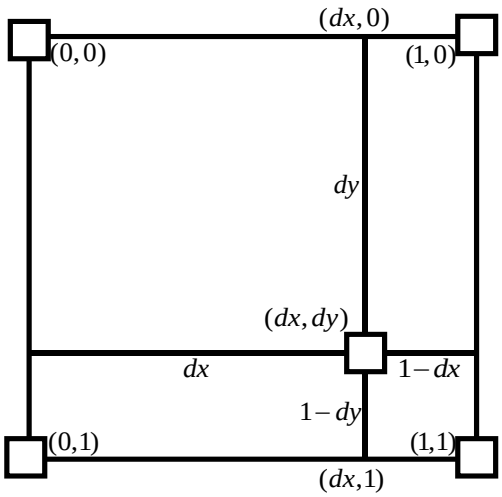

图 2.5 双线性插值算法

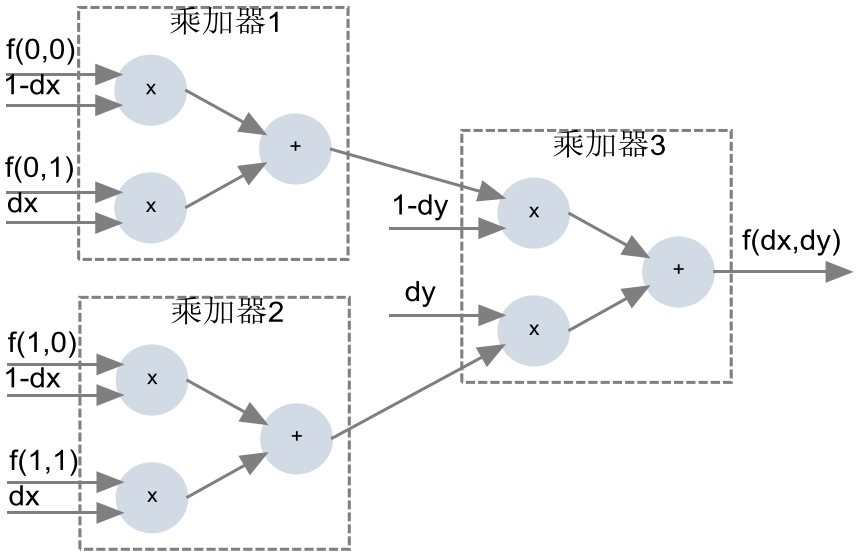

图 2.6 乘加器实现双线性插值

2.4 本章小结

本章首先对几种空间坐标变换方法进行分析对比，结合 FPGA 特点，选择直接法完成图像旋转实现简单，而且可以避免中间步骤的图像损失；然后对图像旋转或者缩放都会产生的锯齿效应进行了分析；最后综合考虑图像质量以及实现难易程度，得出双线性插值算法更适用于图像抗锯齿的结论。

3 实时图像旋转系统的总体设计

计算机之父冯诺依曼在 1945 年起草 EDVAC（电子离散变量自动计算机）设计报告时，将它设计为由 5 个基本部分组成：1）运算器 CA；2）控制器 CC；3）存储器 M；4)输入装置 I；5）输出装置 O。这种组织结构一直延续至今，为绝大部分的数字信号处理系统采用，包括本文设计的基于 FPGA 的实时图像处理系统。

3.1 系统总体框架

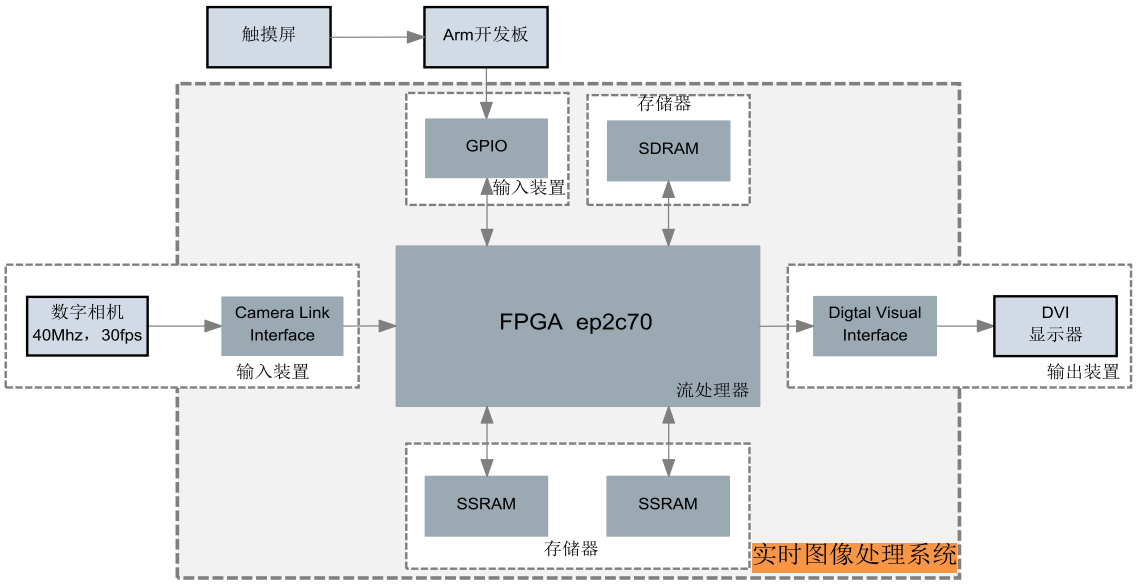

图 3.1 系统总体框架

如图 3.1 所示，整个系统由数字图像输入装置、输出装置、存储器以及流处理器 4 个部分组成。

- 输入装置：系统输入包括实时图像的输入以及人机界面控制信号的输入。图像采集使用 Camera Link 接口的百万像素 CCD 相机，，视频格式为 1000*1000@30Hz。人机界面由触摸屏和 ARM 控制板构成。
- 输出装置：在 DVI 监视器上，实时显示旋转后的目标图像。
- 存储器：包括 SDR SDRAM 以及 SSRAM 两类存储器，实现图像缓存。
- 流处理器：负责实现运算器 CA 以及控制器 CC 的功能。选用 Altera 公司 Cyclone2

系列的 FPGA。

3.2 输入装置

图像处理系统的输入包括实时图像以及人机界面。

3.2.1 视频采集

为了得到被测物图像，我们需要摄像机。摄像机能够将通过镜头聚焦于像平面的光线生成图像，使用的传感器可分为 CCD（charge-coupled device）和 CMOS（complementary metal-oxide semiconductor）两类。两者主要区别在于经光电转换后，数据读取的方式不同。受工艺的限制，CCD 以行为单位进行传输，最后经传感器边缘的放大器进行放大输出；CMOS 则每个像素都会邻接放大器及 ADC，可用类似内存电路的方式读取。在灵敏度、分辨率、噪声控制方面 CCD 要优于 CMOS；而 CMOS 也具有低成本、低功耗、高整合度的优点。

选择摄相机时，除了传感器类型以外，还应该考虑合适的摄相机接口。目前，市场上的采集卡常用的接口类型有：模拟接口、Camera Link、IEEE 1394、GigE。GigE 以其传输速度高、距离远且不需专用接插件及线缆的优点，在市场上很有吸引力，预计在不远的将来会成为市场的主流。但是，它的协议复杂，独立实现的困难较大，因此我们选用基于 LVDS（低电压差分信号技术）的 Camera Link 接口。Camera Link 已经成为高速数字图像采集的标准，它的最高传输速率可达 640Mb/s。FPGA 在处理该接口时，设计非常方便。可直接使用串并转换后的时钟信号、28 个数据信号以及 4 个有效信号（帧有效、行有效、数据有效以及一个备用信号）。

综合比较，我们选择了 IMPERX 公司 Camera Link 接口的 CCD 黑白相机 1M48。帧频为每秒 30 帧图像，图像分辨率为 1000X1000，像素时钟为 40Mhz。

3.2.2 人机界面

人机界面既可以显示图片信息，也可以用于输入用户数据。用户通过触摸屏设置好旋转角度，ARM 板利用处理器的计算能力，计算出坐标变换所需用到的正余弦值，最后通过预留的 GPIO 口传递数据。ARM 板功能非常丰富，除配置 SDRAM、FLASH 外，还带有键盘、触摸屏、串口硬盘等接口。利用这个人机界面，不仅避免了在 FPGA 上实现不擅长的正余弦值计算，更合理的分配计算资源，而且为以后系统升级、丰富功能预

留了很好的接口。比如：文件的查找、删减、分类等管理工作，实时性要求不高但计算复杂的数学运算。

3.3 输出装置

选择合适的图像输出装置，应该综合考虑分辨率、接口类型、成本等多方面因素。

目前高清电视以及数字影院等创新技术进展非常迅速，人们越来越关注高分辨率图形显示技术。表 3.1 列出了目前不同终端设备上能够达到的最高分辨率。

表 3.1 不同终端设备的最高分辨率

终端设备	分辨率（像素）
HDTV	1920X1080
数字影院	4096X1714
视频会议	1280X720
医疗成像	3000X3000
工业监控	1280X720
军事监控	4000X4000

分辨率越高，意味着需要处理的数据量越大。FPGA 以其灵活的结构、丰富的资源，在应对大数据量上有很大的优势。为了配合前文视频采集中选择的摄像头，本文选用 1280X1024 显示分辨率、DVI 接口的 LCD 显示器。其输出标准如表 3.2 所示。

表 3.2 图像输出标准

分辨率	刷新频率	行频率	像素时钟频率
1280X1024	60HZ	64.0kHz	108.0MHz

该模块的设计方法如下：首先利用 FPGA 内部的 PLL 单元生成 108.0MHz 的像素时钟；然后根据视频输出标准，生成行场同步信号，如图 3.2 所示。并在数据有效期间，从 FIFO 中读取像素数据，将 FIFO 设置为即取即得模式。因为是黑白相机，因此给 R、G、B 三个寄存器赋同样的值。

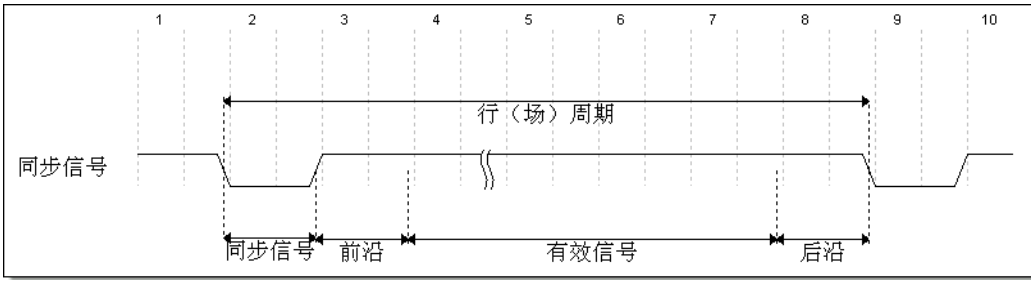

图 3.2 行场同步信号

需要注意的是，此时显示分辨率为 1280X1024，比摄相机的分辨率 1000X1000 要大，我们可以选择在剩余的部分的背景颜色或者叠加背景图片。而且，显示器的帧率比摄像机的要高，这都需要存储器能够缓存图片。

3.4 存储器

图像处理过程离不开存储器。边缘检测中常见的 3X3 模板操作，需要处理系统至少能够缓存两行的图像数据；帧间滤波或者运动估计，需要对多帧图像进行处理，需要处理系统能够缓存多帧的数据；同样地，图像旋转也要求系统能够至少缓存一帧的数据，以方便目标图像寻求对应点。无论是 GPU 还是 FPGA，内部存储资源使用都非常灵活，但数量有限。要完成帧缓存，只能通过添加外部存储器的方法实现。为了使系统更加灵活，我们用到了 SDR SDRAM（Single Date Rate Synchronous Dynamic Random Access Memory）以及 SSRAM（Synchronous Static Random Access Memory）两种类型的存储芯片。

3.4.1 SDRAM

随着系统时钟频率的提高，传统的异步接口 DRAM 已经不能满足设计的需要，成为整个系统性能的瓶颈[23]。同步 DRAM 即 SDRAM 在异步 DRAM 的外围添加一个同步的接口逻辑，从而实现更高的频率以及更高的带宽。IS42S16160B 型号的 SDRAM 共有 256Mb 的存储空间，最高工作频率可达 166MHz。它支持长度 1、2、4、8 以及全页的突发读写、突发中断；支持自动刷新；CAS 延时可配置为 2 或 3 个时钟，对应不同的最高工作频率。它支持的命令包括：NOP(空操作)、BST（突发停止）、READ（读）、WRITE（写）、READ with auto precharge（自动预充电读）、WRITE with auto precharge（自动预充电写）、ACT（行有效）、PRE（预充电）、SELF（自刷新）以及 MRS（设置模式寄存器）。其状态流程图见图 3.3。

上电之后，首先对 SDRAM 进行初始化，整个电路进入确定的状态；然后设置模式寄存器，包括读写模式、突发类型、突发读写长度等等；完成寄存器设置，开始进入正常的工作状态。SDRAM 存储 1bit 数据只需要 1 个晶体管，可以拥有更大的存储空间，为了提高生产成本以及降低功耗，不得不采用行列地址线复用技术以减小芯片体积。这使得 SDRAM 的读写操作变得更为复杂：读写操作时应该分别确定行列地址，以访问相应的存储空间。SDRAM 采用 ACT 命令打开某一行，与此同时还需发送 bank 地址。列地址由 READ 或者 WRITE 命令在 ACT 命令 tRCD（RAS to CAS Delay）时间后发出。

读写完成之后，SDRAM 回到行有效的状态，此时如果想要访问其它行，需要先发送 PRE 命令以关闭该行，回到 IDLE 状态，然后才能开始一次新的读写过程。

总的来说，SDRAM 的驱动器设计比较复杂，体现在其各个命令之间满足时序关系而不互相冲突。通过 $\overline{RAS}$ 、$\overline{CAS}$ 、$\overline{WE}$ 、$\overline{CS}$ 以及地址线 A[10:0]、BA[1:0]的组合，共同构成 SDRAM 的控制接口，如图 3.4 所示。

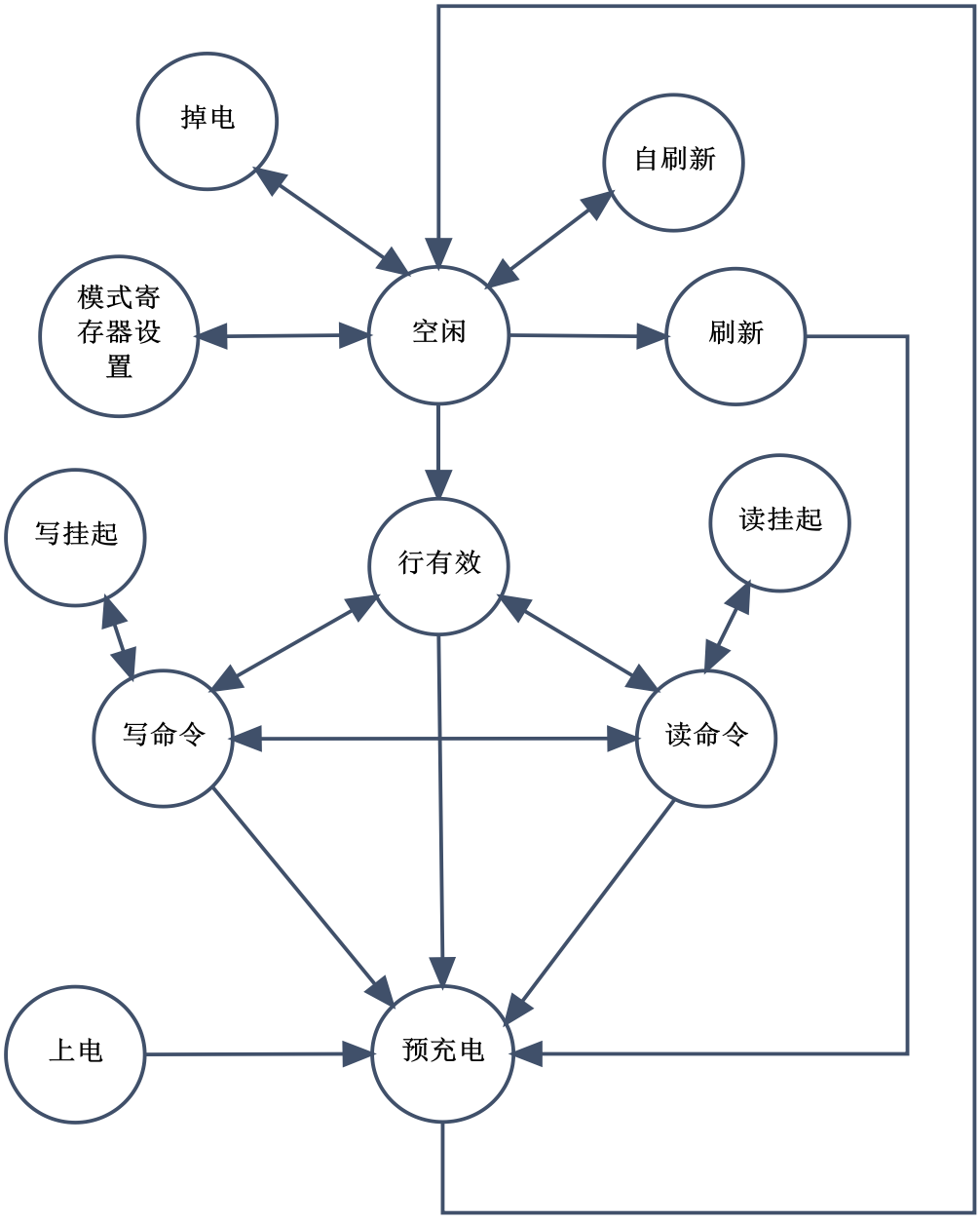

图 3.3 SDRAM 工作流程图

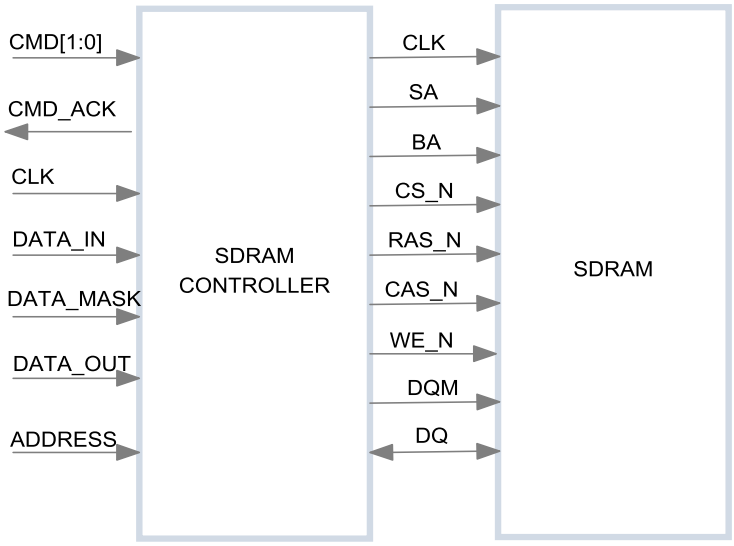

图 3.4 SDRAM 控制器接口图

根据模块化、层次化的设计思想，将 SDRAM 控制器分为 3 个模块分别设计：control_interface、command 以及 data_path。

- Control_interface

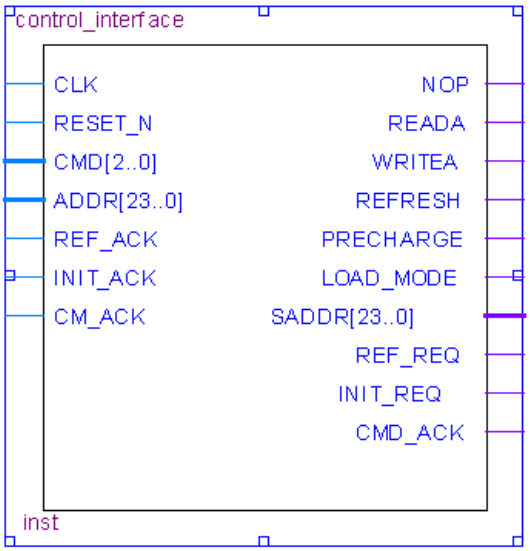

图 3.5 control_interface 模块

该模块负责接受用户发送的指令 CMD[2:0]，并将该编码信号转为单独的读写等控制命令。ADDR[23:0]用于接受读写地址；CMD_ACK 用于应答用户。

- Command

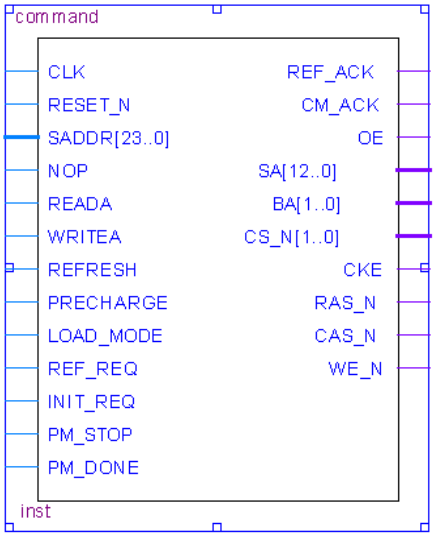

图 3.6 command 模块

该模块负责将之前生成的单独的读写等控制命令转换为RAS_N、CAS_N、WE_N等信号组合。重点在于根据 SDRAM 时序，控制各个指令的执行次序以及时间间隔。

- Data_path

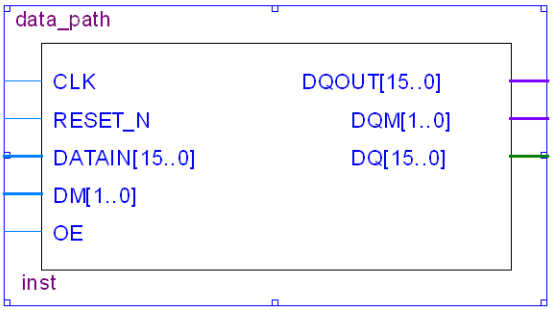

图 3.7 data_path 模块

该模块接收用户输入数据 DATAIN、以及来自 SDRAM 的数据 DQOUT。通过 OE 信号来控制 SDRAM 数据输入或者输出。DQM 信号起到数据掩膜作用，能够屏蔽不需要的数据。

SDRAM 运行速度高、存储空间大，非常适合作为图像处理系统的帧缓存存储器。但是从上面的状态图以及描述可知，SDRAM 读写操作需要准备工作，包括刷新、行列地址选通、固定延迟等等。假设采用单一地址访问模式，SDRAM 的工作效率很低，读写一个数据需要额外消耗更多的周期数。为了提高 SDRAM 的数据吞吐量，配置 SDRAM 为全页读写模式，在同样消耗的情况下，能够访问更多的存储空间。采用页模式能够很大程度上提高读写效率，避免 SDRAM 频繁启动读写命令所需的延迟（包括 tRCD、CAS Latency），但是这对于乱序的存储空间访问的情况并无太大改善。因此，系统设计中，

我们选用了第二类型的存储器——SSRAM。

3.4.2 SSRAM

SRAM 是最基本的，也是最常用到的挥发性存储器。几乎所有的计算机系统都有用到它。同 DRAM 相似，更高性能的 SRAM 也采用同步逻辑来提升性能。IS61LPS102418A 是高速低功耗的流水线型同步静态随机存储器，有 1048576 个存储单元，位宽为 18。与 SDRAM 不同，SSRAM 的存储单元由 4~6 个晶体管构成，因此不需要及时刷新以保证数据不会丢失。SSRAM 的存储空间比 SDRAM 的要小，不需采用地址线复位技术，因此其访问操作更为简单。

SSRAM 为了能够适应各种使用要求，它支持全字节写入、分字节写入；支持突发模式访问；支持电源管理等等。不同于 SDRAM 通过设置模式寄存器的方式，SSRAM 由各个信号的组合完成设置。它通过 ADSP#（地址状态处理器）或 ADSC#（地址状态控制器）来启动突发读写，并通过 ADV#（突发地址使能）来控制突发读写长度。参考数据手册，读写时序可简化为图 3.8 所示。

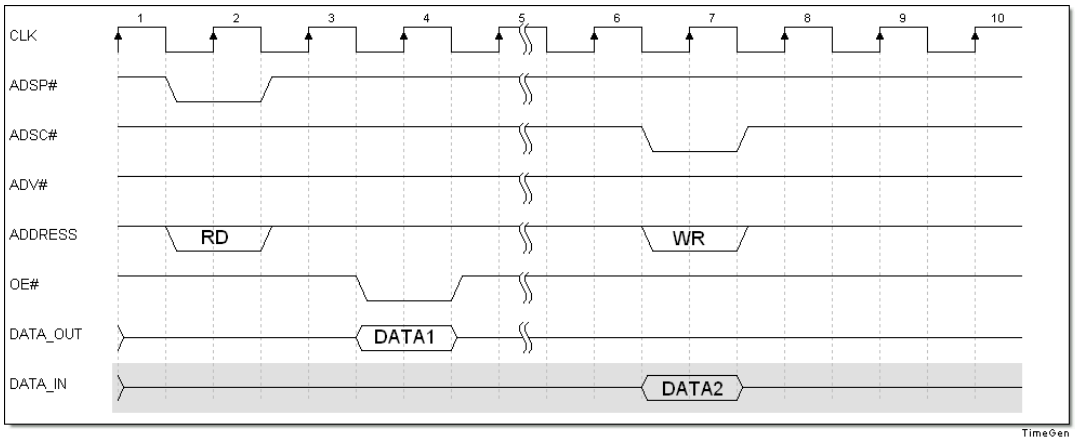

图 3.8 SSRAM 读写时序

读数据的时候，在第一个周期发送读命令、将 ADSP#信号拉低，与此同时输入读地址；紧接着在第二周期将 ADSP#信号拉高；第三个周期可以将 OE#信号拉低，此时在数据总线上可以得到 SSRAM 输出数据。此方法与突发读模式比，看似效率偏低，两个周期才能读出一个数据，但实际上可以通过背靠背操作来实现高效率的乱序地址访问。背靠背操作时，将第二的读操作的发送读地址隐藏于第一个读操作的数据传输过程中，从而实现数据总线上没有空闲周期，始终都在发送数据。

写数据的时候，在第一个周期同时发送写命令，将 ADSC#信号拉低，于此同时，发送写地址以及写数据。从时序图可以看出，写操作能在一个周期内完成，非常便于实现

乱序地址的访问。

对比 SDRAM 和 SSRAM，SSRAM 存储芯片更适合用于快速随机地址访问存储的应用；SDRAM 存储芯片则更适合于长突发顺序地址访问的应用。这两种存储芯片在实时图像图像处理系统中可以发挥各自的优势，适用不同的运用场合。

3.5 流处理器

依冯诺依曼设计的计算机组织结构，由运算器 CA、控制器 CC 共同实现数据运算。传统的处理器执行过程：取指令——译指令——执行，将运算器和控制器各自的功能做了很好的划分。控制器负责运算数据的接收、存储、发送；运算器负责数据的算术运算、逻辑判断等等。传统的设计都是在 CPU 架构体系上的，设计人员在硬件 CPU 的基础上实现软件编程以及算法实现。，这必然会受到 CPU 处理器指令集的限制。FPGA 代表的是硬件编程，我们能够进行自由的硬件架构以及软件开发。基于 FPGA 的实时图像旋转系统中，我们设计流处理器来代替紧密结合的运算器与控制器，完成图像变换、插值、滤波等操作。更为重要的是，FPGA 可重配置的特性，让系统可以实现其它图像处理功能，成为一个通用的图像处理平台。

本图像旋转系统选用的 FPGA 是 CycloneⅡ系列 FPGA 中的 EP2C70F896。CycloneⅡ系列 FPGA 是继低成本的 Cyclone 系列 FPGA 在市场上获得成功之后，Altera 公司推出的更低成本的芯片。它将低成本的 FPGA 密度扩展到 68416 个逻辑单元（LEs），从而能够实现更加复杂的数字系统。CycloneⅡ采用 90nm 工艺，与市场上竞争对手采用的 90nm 工艺的 FPGA 相比，其性能要高出 60%而功耗降低近一半，尤其是其价格几乎可以与 ASIC 产品竞争。这些都使得该系列 FPGA 能够被广泛运用于消费电子、汽车电子、音视频处理、通讯以及测试测量等终端产品市场。该系列 FPGA 的主要性能比较见表 3.3。

表 3.3 CycloneⅡ系列 FPGA 性能比较

	EP2C5	EP2C8	EP2C20	EP2C35	EP2C50	EP2C70
LE 数量	4608	8256	18752	33216	50528	68416
M4K RAM 块数量	26	36	52	105	129	250
RAM 总量/位	119808	165888	239616	483840	594432	1152000
嵌入式乘法器	13	18	26	35	86	150
锁相环	2	2	4	4	4	4
最大可用 I/O 数	158	182	315	475	450	622

设计所选的 EP2C70F896 在该系列 FPGA 中占有最丰富的资源，大大加强了可编程

设计的能力[24]。其特性有：

(1) 逻辑阵列快（LAB）

LAB 以行列形式在 FPGA 器件中排列，每个 LAB 包含 16 个 LE、控制信号（1 个同步清除、2 个异步清除、1 个同步加载、2 个时钟、2 个时钟使能等）、LE 进位链、寄存器进位链以及 LAB 本地互联。利用 VerilogHDL 语言，可以轻松实现比较判断、加法移位以及复杂的状态机。实现的时候，Quartus Ⅱ编译器会用一个 LAB 或相邻的多个 LAB 实现逻辑设计，从而提高设计的效率。

(2) 嵌入式存储器（RAM）

Cyclone Ⅱ系列 FPGA 内嵌由多列 M4K RAM 组成的存储器。M4K RAM 工作模式非常灵活，可以配置成真双口 RAM、简单双口 RAM、单口 RAM、ROM 以及 FIFO，最高工作速率可达 250MHz。设计中常用双口 RAM 或者 FIFO 实现数据的缓存以及跨时钟处理，1152000 的存储位空间能够满足我们的使用需求。

(3) 嵌入式乘法器

嵌入式乘法器为 Cyclone Ⅱ系列的 FPGA 提供了数字信号处理的能力，在实现快速快速傅里叶变换（FFT）、离散预先变换（DCT）以及有线脉冲响应滤波器（FIR）等都能发挥重要作用。该嵌入式乘法器可以配置成 9X9 或 18X18 的乘法器进行工作，如果使用输入/输出寄存器，最高性能可达到 250MHz。而且，在 Quartus Ⅱ中乘法器的使用非常简单。通过 GUI 界面设置数据类型、位数、流水线级数等等就可以例化一个所需乘法器。坐标旋转变换中需要用到乘加运算，可以直接调用嵌入式乘法器实现。

(4) 锁相环（PLL）

EP2C70F896 含有 4 个锁相环，分别位于 FPGA 的四个角落。PLL 主要完成 VCO（压控振荡器）的输出频率 f_{VCO} 与输入参考信号频率同相位的同步。它具有以下特性：输入时钟倍频及分频；时钟相移；可编程的占空比； 3 个时钟输出，驱动全局时钟网络；锁定指示输出等。目前 FPGA 设计中，多采用同步电路设计。在 FPGA 上内嵌 PLL，用户可以根据设计需要自由设定系统工作始终频率以及时钟偏移量。这在设计片外存储器驱动程序的时候，显得尤为重要。比如前面所提 SDRAM 控制器，为了实现理想的存储访问工作，SDRAM 控制器时钟同 SDRAM 工作时钟应该保留一定的相位偏差。

3.6 本章小结

本章介绍了实时图像旋转系统的总体设计情况。在冯诺依曼计算机组织结构的指导下，根据图像处理系统的需要，完成了输入装置、输出装置、存储器、流处理器的设计，能够实现图像采集、存储、处理以及显示的工作。该系统采用 SDRAM 和 SSRAM 两种类型的存储器，能够高效地实现顺序、乱序的存储器访问，提升系统性能。使用 FPGA 作为流处理器，其可编程结构以及丰富的资源能够满足高性能的运用。

4 实时图像旋转系统的 FPGA 设计与实现

数字电路设计方法有自下而上（Down-Top）以及自上而下（Top-Down）两种方法。自下而上的设计方法，设计者根据自己的经验试探性的将规模大、功能复杂的数字系统按逻辑功能划分为若干个子模块，一直到这些子模块可以用经典的方法以及标准的逻辑功能模块实现为止，最后组装成为所设计的数字系统。自上而下的设计方法又称为方案设计法，利用层次化模块化的思想，将数字系统划分为若干个易于实现的模块，编制相应的模型，通过系统仿真验证后，再由设计师完成具体模块的设计。这种方法，可以由多名设计师共同完成硬件系统的不同模块，每个设计师负责自己承担的一部分，既可以提高设计速度以缩小产品的生产周期，又可以保证设计质量，降低设计错误的可能性。自顶向下的设计图如图 4.1 所示。

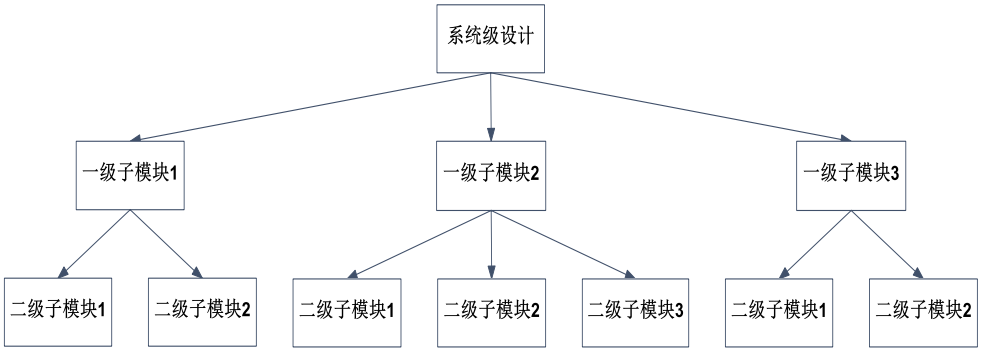

图 4.1 自顶向下设计方法示意图

对比两种设计方法，我们采用自顶向下设计方法来进行实时图像旋转系统的设计。根据总体方案设计，将 FPGA 设计为视频流处理器，其功能框图如图 4.2 所示。流处理器的概念来自于 NVIDIA 于 2006 年推出新一代显卡时的技术参数表，其作用是处理由 CPU 传输过来的数据，处理完后转换为显示器可以辨识的数字信号。传统的流处理器是针对某一特定的视频处理应用而设计的不可编程的流处理器。目前，为了提高显卡性能以及设计的灵活性，系统中常会引入一个通用的可编程器件。基于 FPGA 的流处理器，视频流依次流经各个流处理单元以及流存储器，最后输出给显示器，具有完全的设计灵活性。而且 FPGA 可重配置的特点，流处理器可以根据需要完成不同的处理要求。

如图 4.2 所示，实时图像旋转系统由多个处理单元模块以及存储器控制器模块组成。图像采集模块为视频流的输入。然后视频流依次经过图像滤波、图像增强、实时旋转以及 SDRAM/SSRAM 构成的流存储器，最后被图像显示模块处理为显示器可以辨识的数

字信号。GPIO 负责与 ARM 板的通讯，接收图像预处理参数以及实时图像旋转处理所需要的运算参数等。主用功能模块的作用以及实现方法将在各小节中作详细说明。

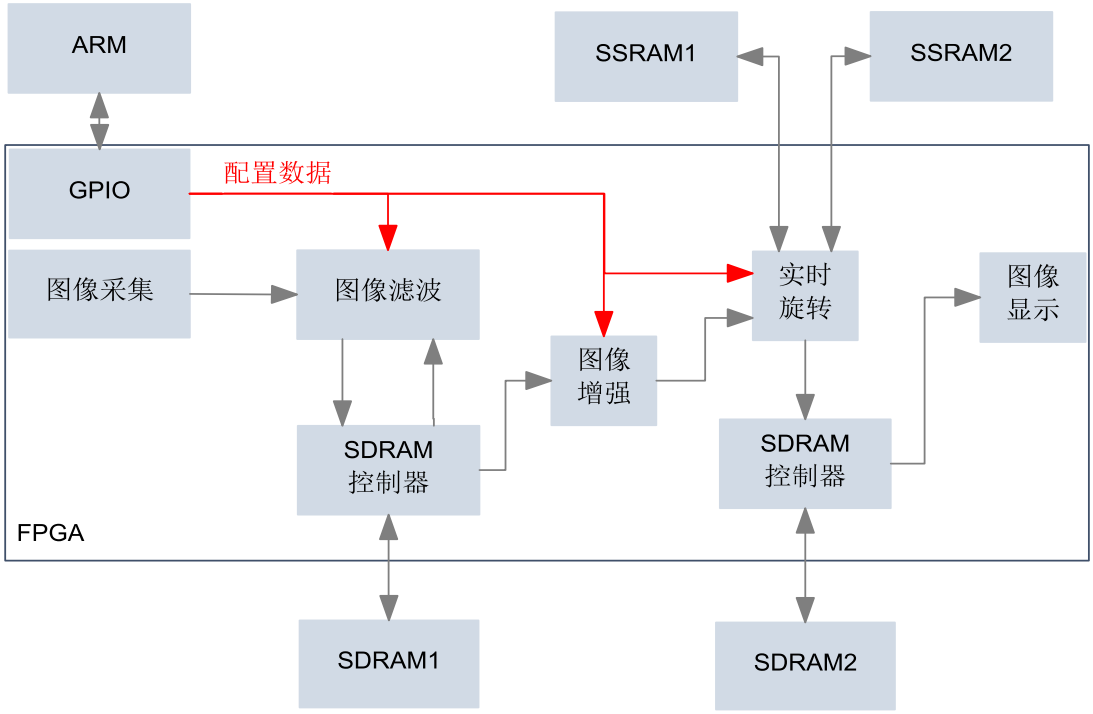

图 4.2 基于 FPGA 的视频流处理器功能框图

4.1 图像预处理

图像预处理是对图像进行前期处理，具有数据量大、抽象程度低的特点。其主要目的是消除图像中无关的信息，增强有关信息的可检测性。预处理过程一般由数字化、直方图均衡化、滤波、平滑、增强等。摄像机采集图像时，感光元件将光学信号转变成电信号，再经 A/D 变换成 FPGA 可处理的数字信号，不可避免的会引入噪声[25~28]。在图像上，表现为椒盐噪声或者高斯噪声。因此，需要对图像进行滤波处理，以降低噪声幅度。常用的平滑滤波器利用模板进行卷积计算，对噪声有明显的消除效果，不过同时得到的图像会变得模糊。一般，运算用的平滑模板尺寸越大，图像可视细节越少，模糊越严重。为了避免这种结果，我们不再简单地进行像素点间取平均，而是利用相邻两帧直接的相关性，采用递归滤波的方式。

美国哥伦比亚广播公司实验室在 1971 年提出时序递归滤波，用于降低动态图像随机噪声，改善图像的质量，提高性噪比，并具有良好的效果[29]。与静态图像不同，数字摄像机采集的动态数字图像是按帧周期性重复的，除部分高速切换的图像之外，相邻两帧

之间的相关系数能够达到 0.8，即它们间有 80%的共同信息。时域递归滤波的原理可用图 4.3 表示。

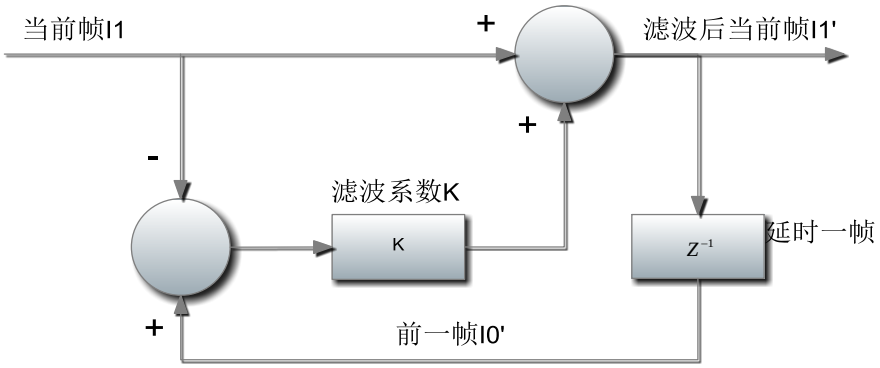

图 4.3 递归滤波原理图

从图 4.3 可看出，该滤波算法实际上就是对相邻两帧做差以得到动态图像的变化量，然后乘以一个滤波系数 K（取值在 0~1 之间），再叠加到当前帧中，从而大大降低随机噪声，而对图像几乎没有影响，图像边缘信息得到很好的保留。为了方便 FPGA 实现，滤波系数 K 取值 2 的整数次方，从而整个算法只需要简单的加减移位运算就可完成。使用 VerilogHDL 语言描述如下：

```
case(isw)
  RECURSIVE_16:
        Frame_Stone_out<=FrameNew-FrameNew[15:4]+Frame_Stone_in[15:4];
  RECURSIVE_8:
        Frame_Stone_out<=FrameNew-FrameNew[15:3]+Frame_Stone_in[15:3];
  RECURSIVE_4:
        Frame_Stone_out<=FrameNew-FrameNew[15:2]+Frame_Stone_in[15:2];
  default:
        Frame_Stone_out<=FrameNew;
endcase
```

其中，FrameNew 为当前帧 I1，Frame_Stone_in 与 Frame_Stone_out 分别表示前一帧 I0’以及滤波后当前帧 I1’。isw 参数来自人机界面接口，用户可以根据最终显示效果调整滤波参数 K。

需要注意的是，该算法需要缓存一帧的图像作为参考图像。FPGA 内部的 RAM 资源有限，我们需要用到片外的 SDRAM 来实现图像缓存。

4.2 多端口 SDRAM 控制器

大多数存储器器件都是通过单一的接口来传输数据。然而，在视频处理、实时显示等场合存在多个微处理器或控制逻辑需要访问共享的存储器。一个真正的多端口存储器允许多个外部器件同时访问存储器，而且不需要外接控制逻辑，每个端口都可以根据需要配置成只读、只写或者两者都可以。由于多端口存储器构造复杂，成本随端口增加而显著增加。因此我们用一个多端口控制器和一个单一接口的存储器芯片构造多端口存储器，由多端口控制器接受外部读写请求，并仲裁访问权[30]。

设计多端口 SDRAM 控制器需要注意两个设计原则：

(1) 各端口带宽之和小于 SDRAM 实际的带宽。在第三章设计存储器时介绍了选用的 SDRAM 最高工作时钟为 166MHz，但因为在其工作周期内需要刷新、预充电、寻址等必要的操作，不能总处于数据访问的状态，因此其实际带宽只有理论带宽的 80%左右，甚至更低。各个端口的带宽之和必须小于 SDRAM 的实际带宽，否则，数据访问将会出现异常。设计时，需要计算各个端口的带宽，如果其总和超过 SDRAM 的实际带宽，应该考虑使用多片的 SDRAM。该原则可用式 4-1 描述。

$$\sum_{i=1}^{i=n} BW_i < BW_{sdram} \qquad (4\text{-}1)$$

(2) 各端口不要求同时对 SDRAM 进行访问。实时图像旋转系统对存储器的访问不同于 CPU 对存储器的访问，前者的功能比较单一，而且是顺序存储的；而后者则需要满足功能繁多、使用复杂的要求。因此在 FPGA 中采用时分复用的方法具有很高的效率。多端口 SDRAM 控制器实际上与 SDRAM 只有单一的接口，它通过时分复用的思想[31~33]，分配各个端口的访问权限。端口优先级仲裁算法通常有两种：固定优先级算法以及循环优先级算法。应该根据需求，选择合适的仲裁算法。

下面以 4.1 中实现递归滤波算法为例，讨论多端口 SDRAM 控制器的设计，图 4.4 为其示意图。

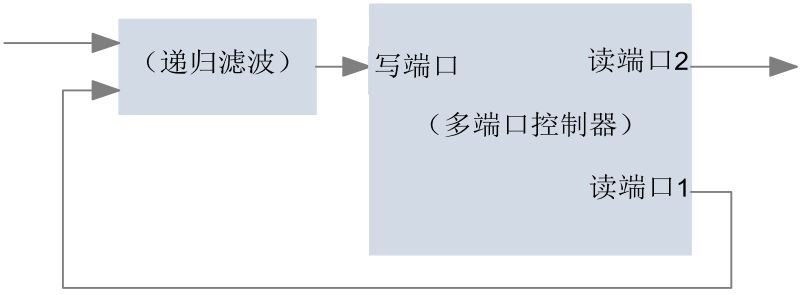

图 4.4 多端口 SDRAM 控制器运用

递归滤波算法需要缓存一帧图像，因此需要一个写端口，两个读端口。写端口将滤波完成的视频流，顺序写入 SDRAM 中。读端口 1 负责读取 SDRAM 中的数据并传输给递归滤波模块作为参考帧；读端口 2 负责将滤波完成的视频流取出给后续的处理模块。第三章设计输入装置时，选用的相机帧频为 30 帧每秒，分辨率为 1000X1000。因此，3 个端口的带宽总和为：3*30*1000*1000=90MHz。综合考虑 SDRAM 控制器的效率,将 SDRAM 设计为工作在 160MHz 时钟下，即可满足带宽设计原则。端口优先级仲裁算法选择固定优先级算法，降低仲裁逻辑的复杂性，其中读端口 2 拥有最高的优先级，读端口 1 其次，写端口最低。

除此之外，多端口 SDRAM 控制器仍需解决端口接口以及控制读写发生的问题。因为各个端口极有可能工作于与 SDRAM 不同的时钟域，因此接口可以采用 FIFO(先进先出缓冲器)来设计。FIFO 能够起到缓冲数据流以及隔离不同时钟域的作用。FPGA 中实现 FIFO 可以直接调用现成的 IP 核，减少设计的工作量。读写发生则由 FIFO 状态决定。首先判断读端口 2 的 FIFO 里面数据是否小于 SDRAM 每页的长度，若是则由读端口 2 获得访问权限，否则判断读端口 1，以此类推。多端口 SDRAM 控制器需要根据读写动作，自动生成读写地址，因为是顺序访问，而且工作于页模式，因此自动生成地址可通过自增 SDRAM 每页长度来实现。当然，如果 FPGA 上存储资源比较紧凑，可以考虑将突发读写模式改为每次读写半页或四分之一页长度，然后突发中断的方式，比如实现递归滤波算法时设计的多端口控制器，每次读写长度为 128，是四分之一页长度，此时 FIFO 只需设置为 256 深度即可。

4.3 图像旋转控制器

图像旋转控制器是实时图像旋转系统的核心单元，图 4.5 所示为其功能框图。它由 FIFO 接收预处理之后的视频流，然后将每帧图像旋转人机界面输入的角度，再由 FIFO 缓冲输出。因为系统采用双线性插值的方式，目标图像每个像素点的灰度值由原始图像 4 个对应像素点的灰度值确定，处理的数据量增长为原来的 4 倍。通过设计流水线，并行执行的方式，提高了系统的带宽，保证了图像旋转的实时性。

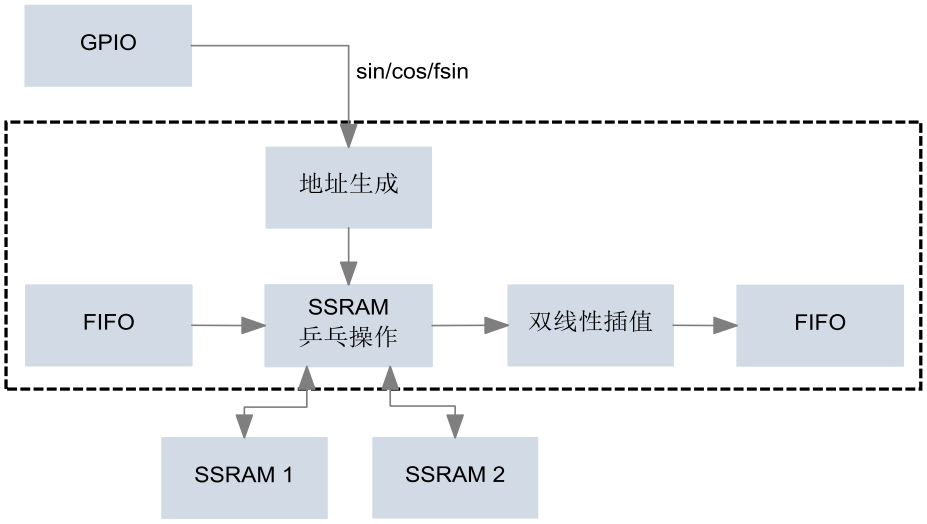

图 4.5 图像旋转控制器功能框图

4.3.1 SSRAM 乒乓操作

因为直接法实现空间坐标变换涉及到乱序访问，因此采用 SSRAM 作为帧缓存器。SSRAM 的存储空间为 18Mb，每帧图像大小为 12Mb。旋转变换需要将原始图像缓存下来，为了使流处理器能提高带宽，满足实时性要求，我们采用两片 SSRAM 进行乒乓操作。乒乓操作是 FPGA 设计中常用的设计方法[34~36]，它将数据依节拍输入到不同的数据处理单元中，也能够依节拍从不同的数据处理单元中获得处理后的数据。其本质上还是面积换速度的思想。对于两片 SSRAM 的乒乓操作，可用图 4.6 表示。对 SSRAM1 写入原始图像时，利用 SSRAM2 中保存的图像完成图像插值工作，旋转完成之后进行交换，使视频流能无间断的流经各个处理单元。

图 4.6 SSRAM 乒乓操作

写入 SSRAM 时，每帧图像的第一个像素点对应 SSRAM 的第一个存储空间，然后顺序依次写入，SSRAM 只需工作于 40MHz 时钟即可，与像素时钟同频。读出时，双线性插值算法要求每次读出 4 个像素点的灰度值，因此 SSRAM 需要工作于 4 倍像素时钟，

即 160MHz。读地址由坐标变换模块计算得到，在旋转非零角度时，通常为乱序的地址。为了同时满足读写请求，设计 SSRAM 工作于 160MHz 时钟，该时钟由像素时钟经 PLL 四倍频得到。图 4.7 为 Quartus II 自带工具 SignalTap 抓取到的数据。从中我们可以看出，此时 SSRAM2 正在进行数据写入工作，oSRAM2_A 顺序递增；SSRAM1 则正在接受地址生成模块产生的对应像素点地址，并将对应点的数据读出。根据 SSRAM 手册可知，数据滞后于读信号两个时钟出现，对比知道数据无误，正是写入的数据。

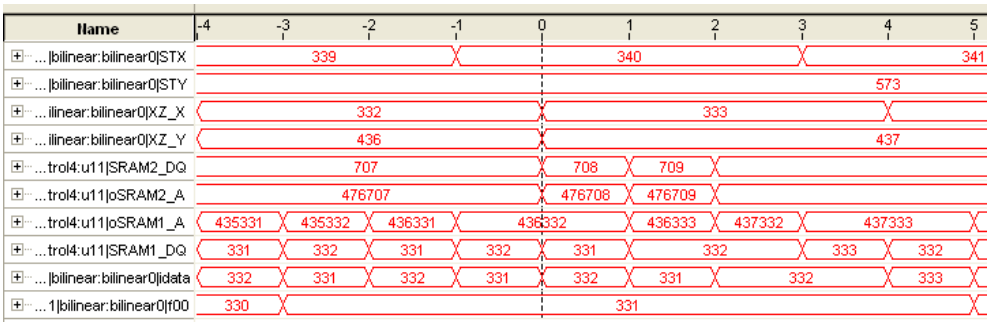

图 4.7 Sgnal_Tap——乒乓操作

4.3.2 坐标变换

图像旋转算法分析一章中介绍了空间坐标变换的几种实现方法。直接法虽然计算量大，每个像素点都需要进行矩阵运算以得到对应坐标，但是它公式整洁，运算简单，在 FPGA 中实现非常方便。

首先以目标图像的中心为坐标原点建立坐标系，图像尺寸为 1000X1000，每个像素点的坐标可用[X,Y]表示。其中 X,Y 的取值范围为-499.5~499.5,在 FPGA 中用 11 位的寄存器描述。观察式（2-4）可知，坐标变换仍需计算正余弦值。为了计算方便，正余弦值由 ARM 计算得到，然后用 GPIO 传递给 FPGA，分别用 Sin、Cos、Fsin 表示正弦、余弦、负的正弦值，在 FPGA 中用 15 位的寄存器描述。数据格式见表 4.1。

表 4.1 坐标变换的数据格式

数据	位数	数据格式		
X/Y	13bit	1bit 符号位	9bit 整数位	1bit 小数位
Sin/Cos/Fsin	15bit	1bit 符号位	1bit 整数位	13bit 小数位
Result_X/Y	28bit	1bit 符号位	13bit 整数位	14bit 小数位

然后，在 FPGA 中例化两个乘加器完成式（2-4）的计算，分别用于计算变换后的 Result_X 与 Result_Y。以 Result_X 为例介绍乘加器的作用。图 4.8 所示为乘加器的设置界面，将乘加器设置为 3 个乘法器，第一个乘法器计算 X*Cos 项，第二个乘法器计算

Y*Fsin 项，第三个乘法器用于将旋转坐标变换为图像坐标。图 4.8 右侧部分可以设置乘加器的数据格式，依表 4-1 设置即可。同理，在另一个乘加器中可以完成 Result_Y 的计算。

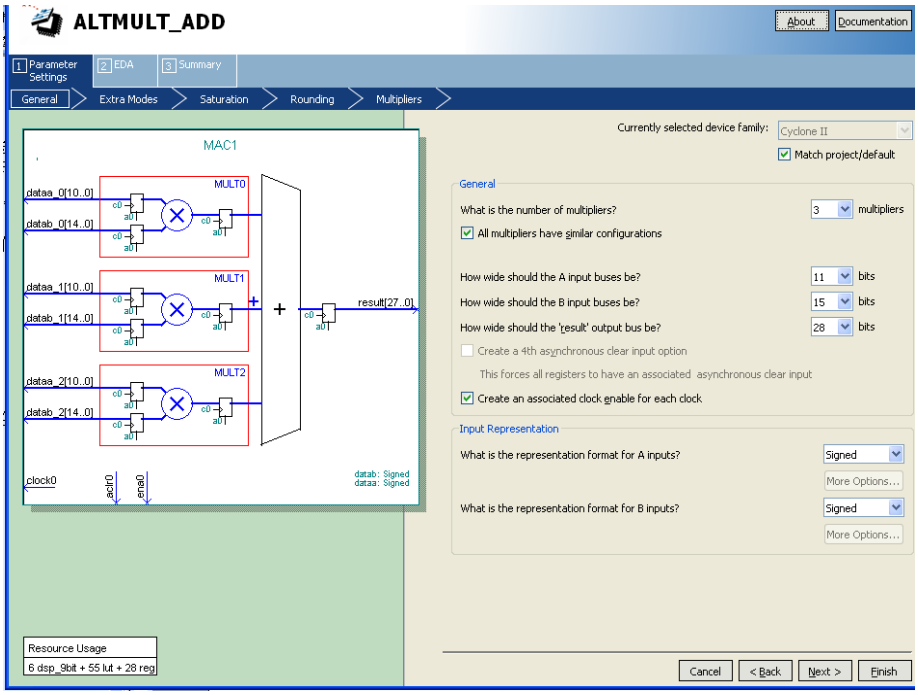

图 4.8 FPGA 中的乘加器

该乘加器为流水线结构，内部有三级寄存器，因此，当乘法器工作于 160MHz 时钟时，只需 2 个时钟周期即可完成计算。即它在一个像素时钟周期内就可完成坐标变换，可以保证视频流不会中断。

图 4.9 所示为该运算的状态机。初始时，状态停留在 3’b000 等待状态，捕捉人机界面的旋转参数，旋转开始后离开 3’b000，不再更新 Sin、Cos、Fsin 值，避免图像出错。3’b001 状态遍历目标图像的每个像素点，获得其坐标并传递给乘加器完成坐标变换。因为乘加器需要 2 个周期才能完成工作，3’b010 与 3’b011 进行等待；3’b100 时候计算完成，此时将结果 Result_X、Result_Y 保存并进入 3’b001 状态开始扫描下一个像素点。直至完成一帧图片的坐标变换，状态回到 3’b00,等待下一次旋转动作的发生。

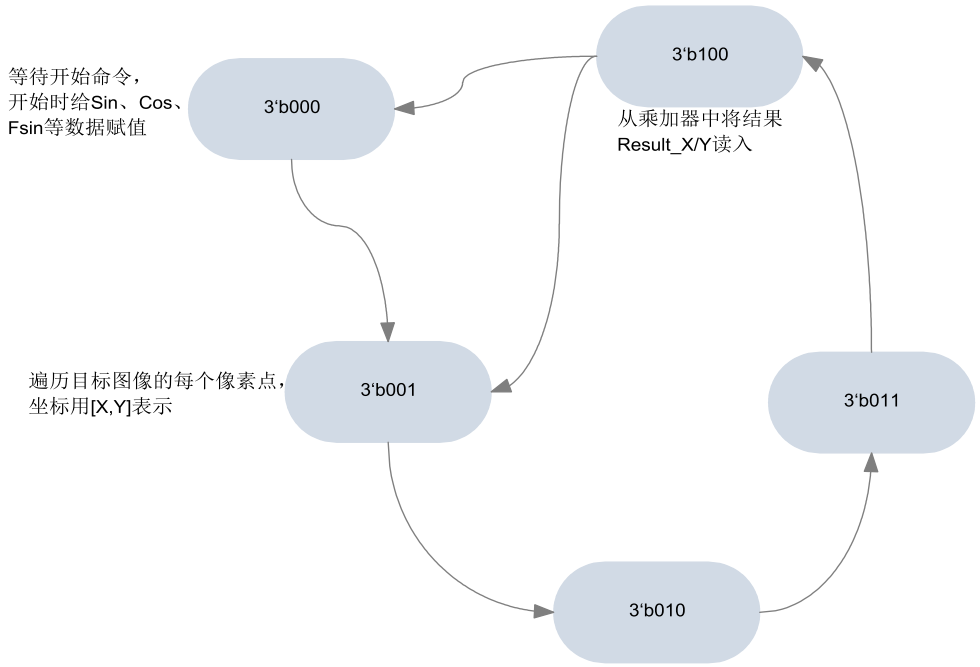

图 4.9 地址生成状态机

4.3.3 双线性插值

流水线处理是高速数字电路设计中常用的一种手段。早在 1968 年，流水线就被用于高速乘法器设计中，随后又出现在 FFT、DCT 等更复杂的数字信号处理单元中。它在不过分增加资源消耗的情况下，能有效提高数字电路的数据吞吐量[37~39]。根据所定义的方式，流水线可分成三类：同步流水线、异步流水线、行波流水线。FPGA 设计因为部分综合器不能支持异步时序逻辑综合，而且异步时序逻辑很难控制组合逻辑和延时所产生的冒险和竞争，因此更适合设计同步流水线。与处理器中流水线设计不同，FPGA 的同步流水线通常是专用于某个运算任务，设计时不必考虑流水线冒险的因素，因此能够获得更高的执行效率。前面已经介绍，双线性插值需要处理的数据量是其他处理模块的四倍。如果此单元设计不合理，将成为系统带宽的瓶颈，限制图像旋转系统的实时性。采用同步流水线设计，能够提高系统的数据吞吐量。其实现方法就是将组合逻辑分解为若干级，中间用有记忆功能的寄存器连接。

首先我们利用坐标变换过程中得到的 Result_X 与 Result_Y 计算对应点在 SSRAM 中的存储地址 addr。其中包括保留整数部分（MID_X,MID_Y）以及判断是否超出原始图像边界（XZ_X,XZ_Y）两个步骤。初始地址 addr=XZ_Y*1000+XZ_X。地址 addr、addr+1、addr+1000、addr+1001 分别对应最邻近的 4 个像素点。然后依次将其作为 SSRAM 地址

给 SSRAM 控制器，得到连续输出的数据流 data1、data2、data3、data4。如图 4.10 所示，将数据流寄存一级，此时 data1 与 data2 成为一组，对应公式（2-7）中的 $f(0,0)$ 与 $f(0,1)$；data3 与 data4 成为一组，对应公式（2-7）中的 $f(1,0)$ 与 $f(1,1)$。因此，例化乘加器实现公式（2-7）时，只需一个乘加器即可计算得到公式（2-7）中的前面两项结果 $f(dx,0)$，$f(dx,1)$。同理，将 $f(dx,0)$，$f(dx,1)$ 数据流寄存两级，然后可以发现 $f(dx,0)$ 与 $f(dx,1)$ 组成一组，将他们作为乘加器的输入，即可得到最终的插值结果。当然，设计时需要注意各数据之间的延时关系，保证计算能够忠于算法。乘加器的设置前面已经提到，这里此处不再详述。从最终结果来看，每四个时钟周期就可完成一次双线性插值工作，即可以在一个像素时钟内完成插值工作，因此，整个系统的实时性得到了保证。

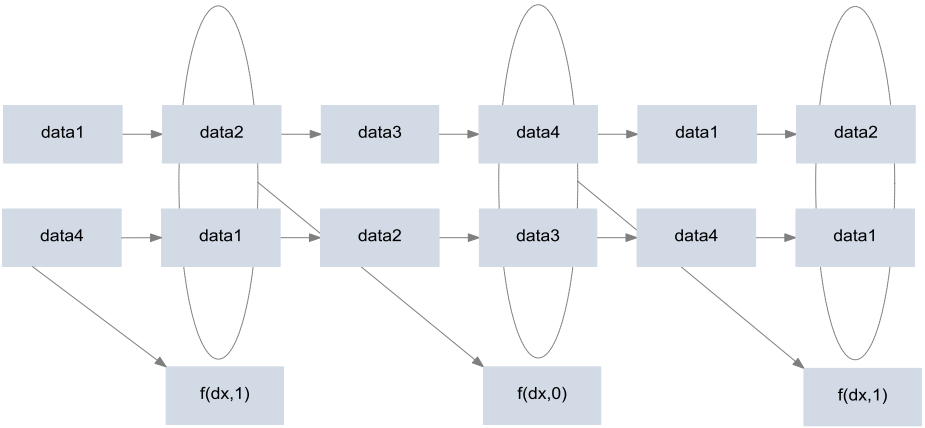

图 4.10 双线性插值算法实现

图 4.11 所示为旋转 45 度时采集到的数据。数据连续地从 SSRAM 中被读出，将数据寄存一级后分别作为乘加器的 f00、f01 输入，两个时钟后计算得到 fx0 与 fx1，再送给下一个乘加器得到最终结果。每 4 个时钟周期可以完成一次计算。

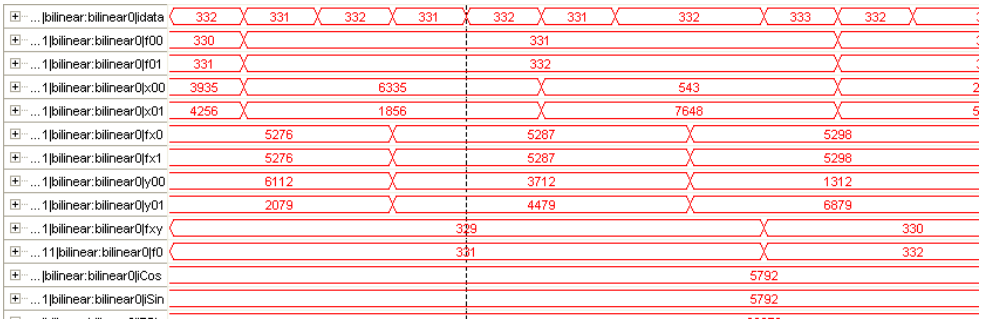

图 4.11 Sgnal_Tap——双线性插值

从图像旋转控制器的主要模块的实现过程来看，视频流能够无停顿地被各个处理单元处理，虽然中间会引入一些延时，但实时图像旋转系统能够不间断地对视频流完成旋转。

4.4 时序约束

电子工程师在绘制 PCB 时，需要合适的约束条件。与此类似，FPGA 设计离不开时序约束。时序约束的目的是：规范设计的时序行为、表达设计者所期望要满足的时序条件，指导综合和布局布线阶段的优化算法，并提供正确的时序报告。在 FPGA 设计中添加时序约束，可以提高系统设计的 fmax，并能通过阅读时序报告了解设计的关键路径。高速数字电路的设计，离不开时序约束[40]。

本小节将对时序约束的原理进行分析，并利用 Quartus 提供的完全集成的时序约束分析工具 TimeQuset 对实时图像旋转系统进行时序约束，并达到时序收敛的目的。

4.4.1 时序约束原理

要完成时序约束，首先要理解同步时序电路的一些基本概念。其中最重要的两个概念就是建立时间 Tsu 和保持时间 Th，如图 4.12 所示。

- 建立时间是指在时钟到来之前，数据保持稳定不变的时间，如果建立时间不够，数据将不能在这个时钟上升沿被打入触发器。
- 保持时间是指在时钟到来之后，数据保持稳定不变的时间， 如果保持时间不够，数据同样不能被打入触发器。

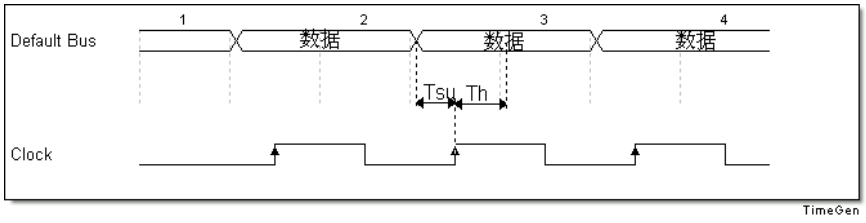

图 4.12 建立时间与保持时间的关系

完整的时序分析包括寄存器到寄存器分析、I/O 和异步复位通路分析等。根据数据传输路径的不同，下面将分为 3 种情况进行讨论。

(1) 情况 1：寄存器到寄存器

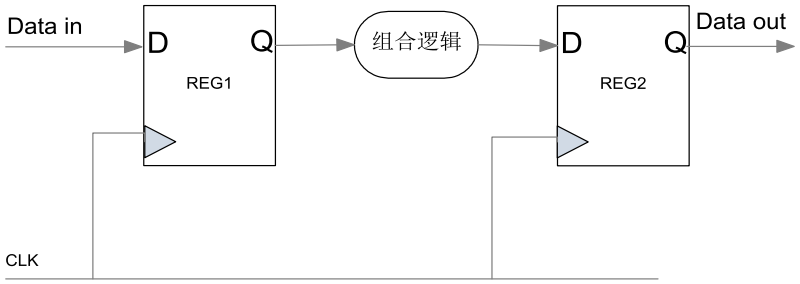

图 4.13 寄存器到寄存器情况

如图 4.13 所示，数据流入第一个寄存器 REG1 后经组合逻辑再进入第二个寄存器 REG2。假设时钟周期为 T，寄存器访问时间为 Treg，组合逻辑占用时间 Tlogic，uTsu 为 FPGA 内部寄存器的建立时间，uTh 为 FPGA 内部寄存器的保持时间。

那么数据从 REG1 的输入端传输到 REG2 的输入端所花时间为：

$$Tarrive = T_{reg} + T\log ic \tag{4-2}$$

为了满足建立时间约束，必须有：

$$T - uTsu > Tarrive \tag{4-3}$$

为了满足保持时间约束，必须有：

$$uTh < Tarrive \tag{4-4}$$

观察式（4-3）以及式（4-4），发现建立时间的约束和时钟周期有关，当系统在高频时钟下无法正常工作时，可以通过降低时钟频率的方式使系统完成工作。保持时间约束则与系统时钟无关，如果设计不合理，使得布局布线工具无法布出高质量的时钟树，那么无论如何调整时钟频率也无法达到要求。只能通过对设计进行重新的修改设计，才可能正常工作，大大降低了工作效率。在 FPGA 设计中，时钟树采用平衡树结构，保证时钟到达每个寄存器的时钟偏斜很小或几乎没有，因此保持时间约束通常是满足的，建立时间的约束才是重点，因为它与系统最高频率 fmax 紧密相关。结合式（4-2）与式（4-3），可得时钟周期 T 与建立时间 uTsu 之间的关系式（4-5）。在 FPGA 器件中，因为系统资源是固定的，uTsu 以及 Treg 通常都比较固定，因此想要提高系统最高频率 fmax，必须降低 Tlogic。这可以通过划分组合逻辑，中间插入多级寄存器的方式实现。

$$T > uTsu + Treg + T\log ic \tag{4-5}$$

(2) 情况 2：外部芯片到输入寄存器

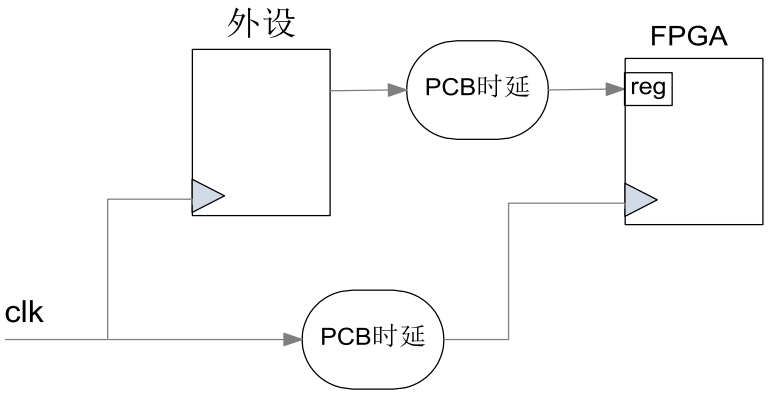

图 4.14 寄存器输入情况

如图 4.14 所示，在同时钟源的情况下，外部芯片数据经过 PCB 板上的延时，进入 FPGA 中。用 Tlaunch 以及 Tlatch 表示数据的发生沿以及接收沿，Tco 为外设的访问时间。

那么，数据到达 FPGA 的输入寄存器所花的时间为：

$$Tarrive = Tlaunch + Tco + Tpcb + Tpin2reg \quad (4\text{-}6)$$

为了满足系统的建立时间，必须有：

$$Tarrive < Tlatch + Tclk\_pcb - uTsu \quad (4\text{-}7)$$

为了满足系统的保持时间，必须有：

$$Tarrive > Tlatch + Tclk\_pcb + uTh \quad (4\text{-}8)$$

对于这种情况，不能通过划分组合逻辑的方式实现时序收敛。观察式(4-6)~式(4-8)，真正可以调整的变量为 Tlatch 与 Tlaunch，其它部分在完成电路图设计后，都相对固定。

(3) 情况 3：输出寄存器到外部芯片

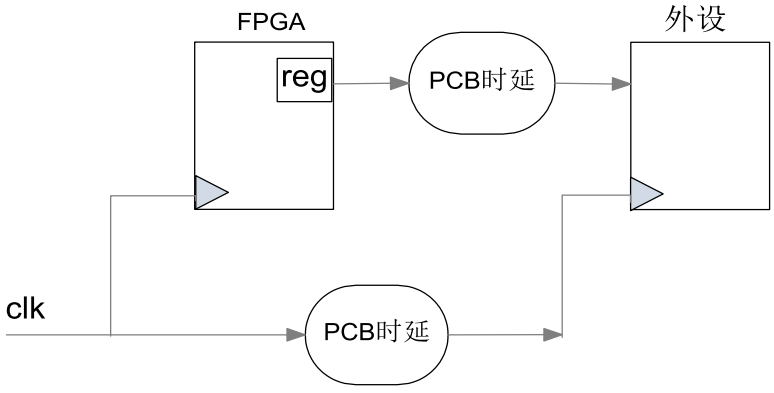

图 4.15 寄存器输出情况

如图 4.15 所示，FPGA 输出数据给外设，数据在传输工程中有 PCB 走线延时。其中 Tsu、Th 表示外设的输入寄存器要求的建立时间与保持时间。

那么数据离开 FPGA 寄存器到达外设的寄存器输入端所花时间为：

$$Tarrive = Tlaunch + uTco + Treg2pin + Tpcb \quad (4\text{-}9)$$

为了满足系统的建立时间，必须有：

$$Tarrive < Tlatch + Tclk\_pcb - Tsu \quad (4\text{-}10)$$

为了满足系统的保持世家，必须有：

$$Tarrive > Tlatch + Tclk\_pcb + Th \quad (4\text{-}11)$$

与情况（2）类似，这种情况也可以通过调整 Tlaunch 与 Tlatch 之间的偏斜来满足系统的建立时间与保持时间。

从上述三种不同传输路径的情况来看，我们设计所需满足的约束条件都是以满足建立时间约束和保持时间约束为基础的。

4.4.2 时序约束工具——TimeQuest

前面小节讨论了不同路径的时序分析方法。本节将介绍如何用 TimeQuest 工具完成时序约束。TimeQuset 是 Altera 公司为自己的 FPGA 设计的一款时序约束分析工具，它支持 GUI 以及命令行的方式完成约束输入以及查看报告[41]。TimeQuest 分析器使用数据请求时间，数据到达时间来验证电路性能，检测潜在的时序违规。TimeQuset 分析器确定设计正常工作必须符合的时序关系，针对请求时间检查到达时间，已验证时序。

图 4.16 为使用 TimeQuest 的流程图。首先对工程进行分析综合，以得到综合前的设计网表。然后运行 TimeQuest 使用 GUI 或者命令行的方式建立 SDC 文件，并将其加入到工程中。开始工程的完整编译，编译过程中，Quartus II 工具会根据约束内容自动调整布局布线算法，以达到时序收敛的目的。编译完成后，可以得到时序分析报告。查看分析报告，对系统的关键路径进行修改，或者调整时钟偏移量，重新编译工程。重复上述步骤，直到满足时序约束条件。

现在对实时图像旋转系统进行分析。本系统采用同步时序电路设计，因此含有寄存器到寄存器的路径，此路径应该添加周期约束。该约束能对该时钟所驱动的所有寄存器的建立时间与保持时间进行检查，并在布局布线的过程中调整寄存器间组合逻辑的分配，争取最好的时序收敛结果。此外，本系统还用到片外的 SDRAM 以及 SSRAM，需要 FPGA 与外设进行数据交互。数据线是双向总线，因此同时包含了外设到输入寄存器以及输出寄存器到外设的路径，这部分路径应该添加输入最大最小延时约束、输出最大最小延时约束。通过查看报告，调整启动沿与捕捉沿之间的相位偏斜，达到时序收敛。图 4.17 所示为最终达到时序收敛时的约束报告。其中编号 4,5,6 分别为 SSRAM、

SDRAM1、SDRAM2 的控制器时钟，其最大工作频率均超过目标工作频率 160MHz，因此实现了时序约束的目的。

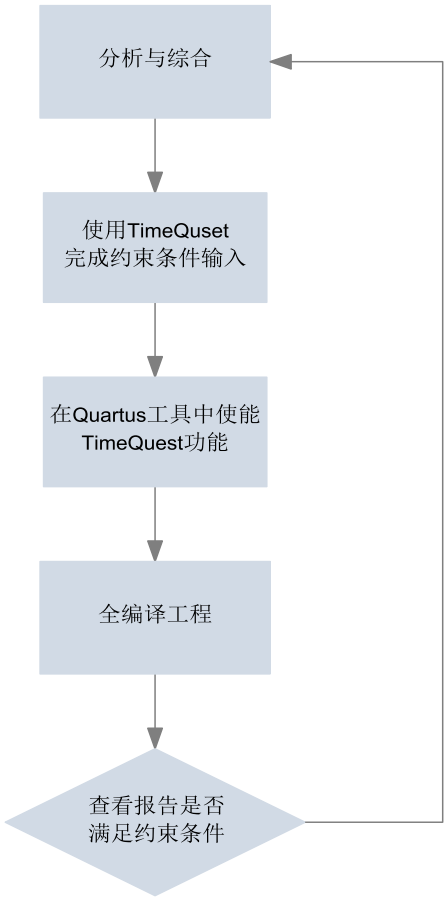

图 4.16TimeQuset 工具使用流程图

Slow Model Fmax Summary

	Fmax	Restricted Fmax	Clock Name	Note
1	69.43 MHz	69.43 MHz	rxclk	
2	131.58 MHz	131.58 MHz	pll2:pll2|altpll:altpll_component|_clk0	
3	136.99 MHz	136.99 MHz	clk50_1	
4	164.64 MHz	164.64 MHz	pll1:pll1|altpll:altpll_component|_clk1	
5	167.79 MHz	167.79 MHz	pll4:pll4|altpll:altpll_component|_clk0	
6	171.29 MHz	171.29 MHz	pll3:pll3|altpll:altpll_component|_clk0	
7	217.82 MHz	217.82 MHz		

167.79 MHz

图 4.17 时序约束报告

4.5 本章小结

本章详细介绍了实时图像旋转系统的实现过程。首先介绍了动态图像滤波常用到的递归滤波算法；然后介绍多端口 SDRAM 控制器的意义；紧接着详细介绍旋转控制器组

成及三个分模块的功能与实现方法，使用到了多种 FPGA 高速数字电路设计常用的设计技巧。最后讨论如何使用时序约束工具指导编译工具，提高系统工作时钟频率。

5 实验结果

本章首先介绍了实时图像旋转系统的 FPGA 设计流程，然后介绍了实验平台，并在其上进行测试，最后给出该系统在医疗图像处理中的运用结果。

5.1 FPGA 设计流程

图 5.1 为 FPGA 的设计流程图。

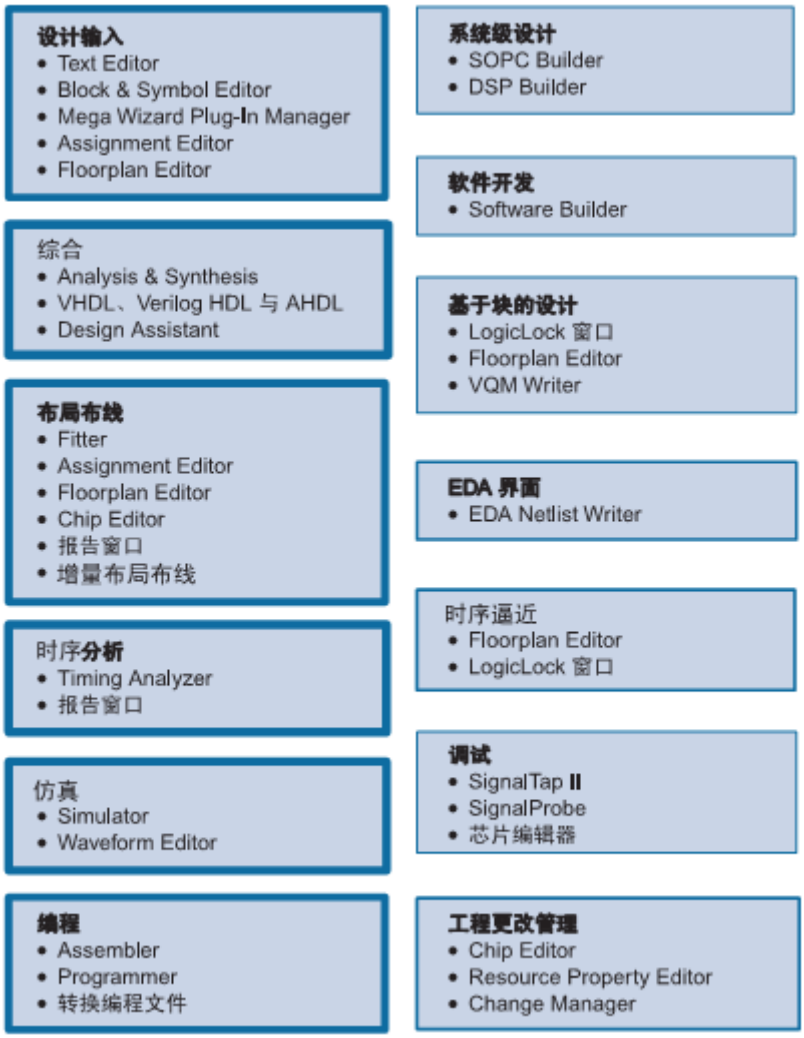

图 5.1 FPGA 设计流程

根据模块化、层次化设计思想，完成第四章中各个功能模块的代码编写及功能验证后，连接各模块，添加时序约束。然后依据图 5.1 所示流程，完成综合、布局布线、时序分析、仿真工作。最后完整编译整个工程，得到可下载文件。

图 5.2 所示为 Quartus II 编译完成后的报告。报告中显示整个实时图像旋转系统消耗了 14%的 LE 资源、24%的存储器资源以及 11%的嵌入式乘法器。因此，系统仍有相当多的资源可以用于实现其它更复杂的功能。仔细分析该系统，FPGA 被设计为流处理器，视频流可以不间断地输入，依次经过滤波、增强、旋转等模块的处理后在显示器上显示。虽然每帧图片从输入到离开系统有一定的延时，但是从流水线的结果来看，每秒钟能够处理完 30 帧的图片。即可视每帧图片的处理时间为 33.3ms，达到实时图像旋转系统的实时性要求。系统的瓶颈主要在于 SSRAM 控制器的时钟。SDRAM 虽然也被设计为工作在 160MHz 时钟，但是其带宽设计有较大的余量；而 SSRAM 的时钟 160MHz 恰好为像素时钟的四倍，带宽余量很小。而且从时序约束报告来看，该模块最高工作频率 fmax 也只能达到 164.64MHz。这使得系统在处理更高像素时钟的视频流时，变得非常困难。

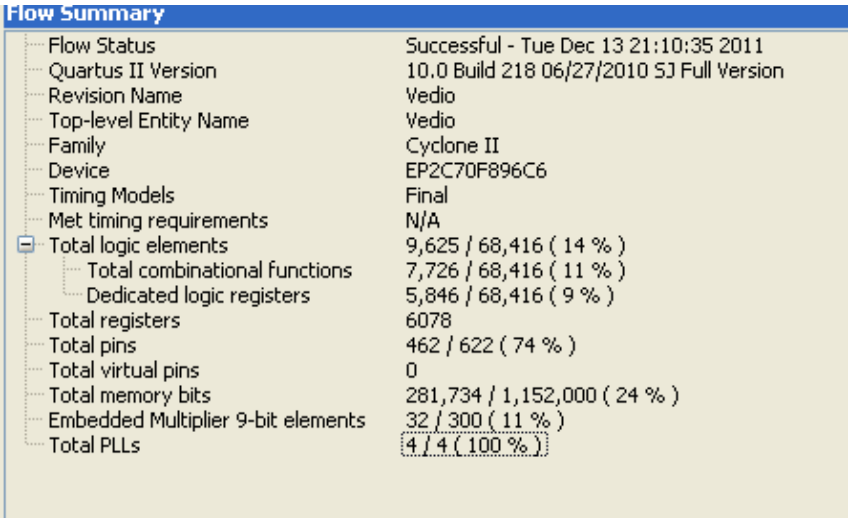

图 5.2 资源消耗情况

5.2 实验方案

本系统的实验平台为通用图像处理板，其实物图如图 5.3 所示。该图像处理板支持多路视频输入与输出，集成了高速光纤传输模块，而且存储器资源也非常丰富，包括 4 片 SDRAM 以及 2 片 SSRAM，能够提供本文设计的实时图像旋转系统所需的各种资源。我们将在该通用图像处理板上对系统进行测试。表 5.1 列出测试项目。根据系统设计选择的摄像机，系统需要能够处理 1000X1000 大小的实时图像，并能够在 0°～360°范围

内实现任意角度旋转。

下面将从理论上对系统进行分析。系统能够处理的图像尺寸最大可以是 1024X1014，主要是受 SSRAM 存储资源的限制。所选用的坐标旋转算法使得旋转角度可以是 0°~360°范围内的任意角度，而且其正余弦值用 15bit 的有符号定点小数表示，能够满足旋转角度的分辨率要求。在设计旋转变换单元时，在每帧图像变换开始前，将当前 ARM 传输过来的 Sin/Fsin/Cos 数值保存到相应的寄存器中，直至下一次变换开始才进行更新。因此，在切换旋转角度时，不会出现图像花屏的情况。

表 5.1 测试项目

测试项目	测试内容
图像大小	1000X1000
旋转角度	1.0°~360°范围内任意角度旋转 2.分辨率：1° 3.特殊情况：不旋转的情况，即 0°
实时性	连续输入帧频为 30 帧每秒，像素时钟 40Mhz 的图像
旋转角度切换	切换旋转角度，图像不出现花屏情况

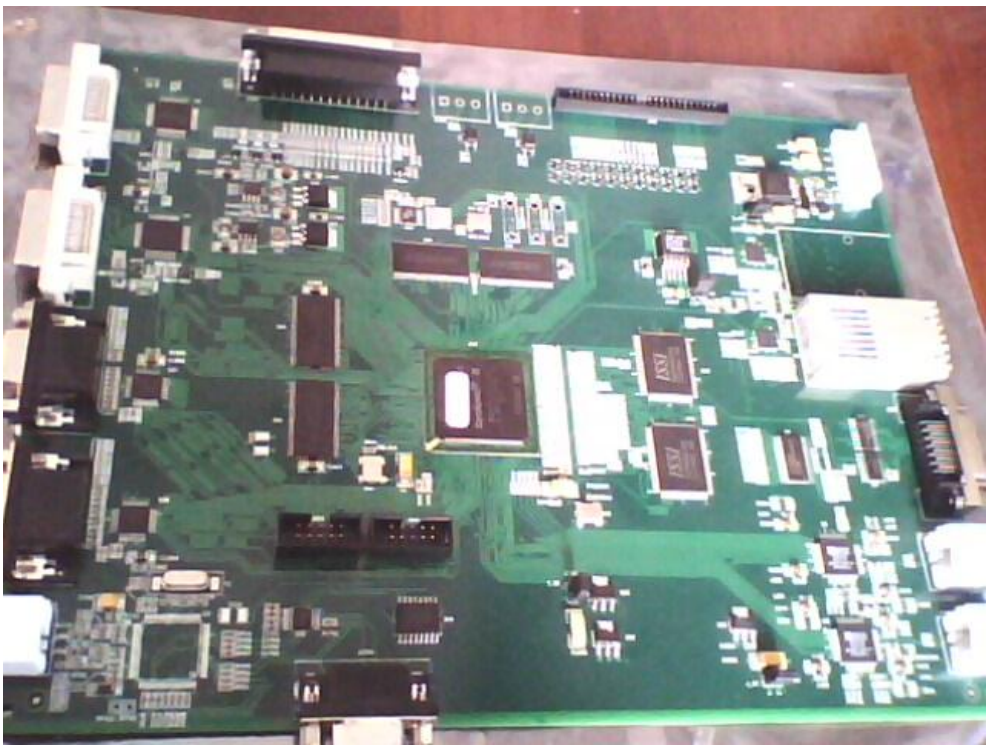

图 5.3 实验平台

上面从理论上分析了系统能够满足各个测试项目的要求，现在将用 Signal_Tap 工具对系统进行测试。Signal_Tap 工具是 Quartus II 自带的一款在线调试工具，能够随时捕捉 FPGA 中各个寄存器、模块 IO 节点的数据。我们将利用 Signal_Tap 工具对各个环节的数据进行捕捉分析。为了方便对工程进行测试，设计一幅测试图片如图 5.4 所示。该测试图片具有以下特点，每行的像素灰度值依次从 0 到 999 递增，整幅图片显示为从黑到白过渡。该测试图片以 30 帧每秒的帧频，源源不断的输入给图像滤波模块，然后依次经过各个处理单元，最后输出。其规律的灰度值变化特点，有助于判断各个处理模块的时序是否正确。

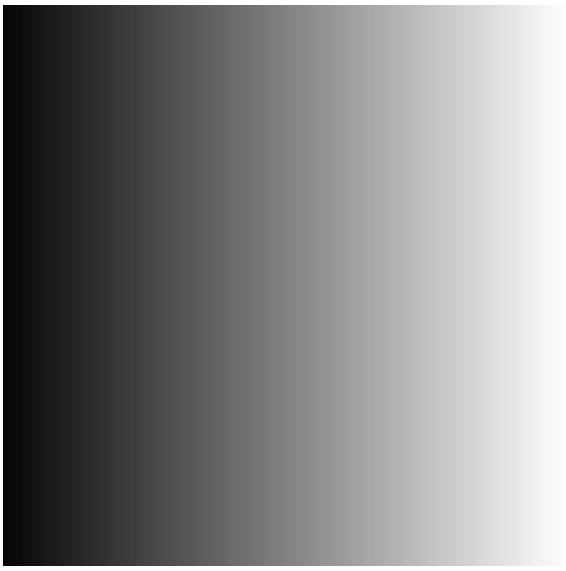

图 5.4 测试图片

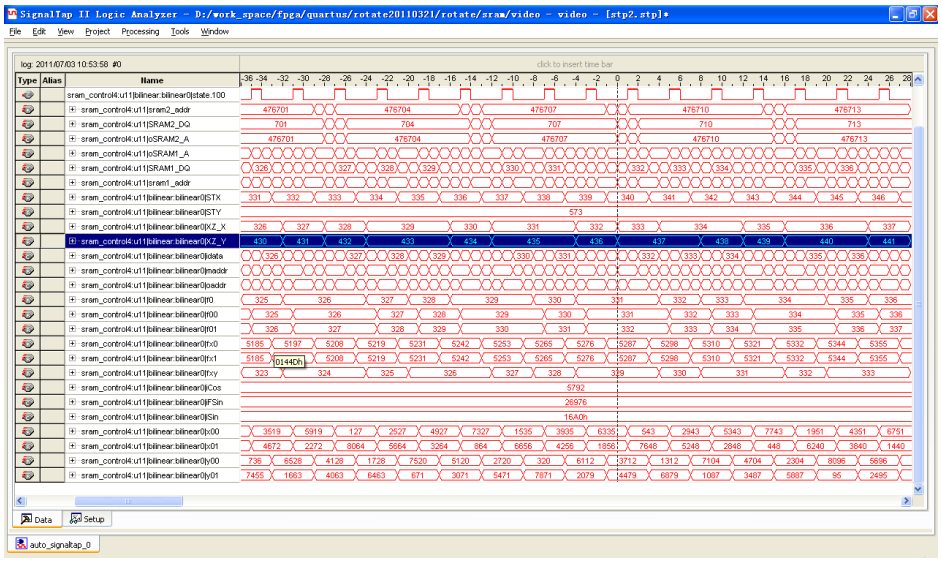

图 5.5 Signal_Tap 结果

图 5.5 为利用 Signal_Tap 工具捕捉到系统将测试图片旋转 45° 时的数据。

Sin/Cos=15’h16a0，该数据始终保持不变，直至下一帧图像变换才更新。图中，处理单元正对目标图像 573 行的数据进行处理。依次监测流水线的各个阶段，图像旋转算法得到了很好的实现。切换不同的旋转角度，再次采集数据分析。

5.3 测试结果

将本实时图像旋转系统应用于 X 光医疗影像诊断设备，图 5.6 为系统在采集 30 帧每秒的帧频，分辨率 1000X1000 大小的图像时，旋转不同角度的效果图。从结果来看，该系统很好地实现了图像旋转的功能，保证了处理的实时性，而且图像边沿部分过渡平滑，没有明显的锯齿效应，得到的图片效果好，切换角度时，不存在花屏的情况。

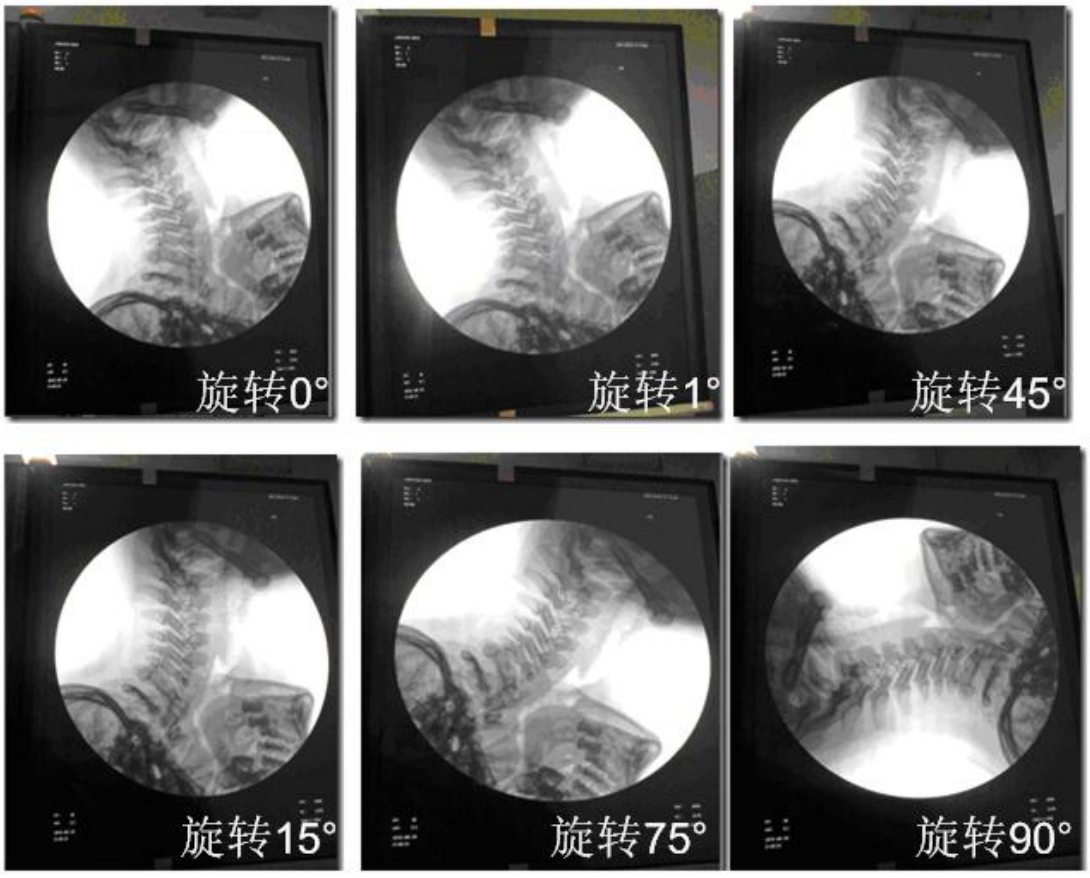

图 5.6 医疗设备运用结果

6 总结与展望

6.1 总结

随着图像处理技术在各个领域不断普及，人们对图像处理的实时性要求越来越高。图像旋转作为图像处理中的基本几何变换，处理的数据量大，容易成为实时处理系统的瓶颈，对其展开研究，提高图像旋转系统的实时性很有实际意义。目前，微电子技术迅猛发展，芯片制造工艺也不断提高，低成本、高性能的 FPGA 在嵌入式系统中扮演着越来越重要的角色。FPGA 结构可编程、可重配置、并行运算的特点，使它非常适合运用高速数字信号处理场合。本文希望利用 FPGA 的这些优点，提高图像旋转处理的实时性，完成实时图像旋转系统的设计。

本文首先对课题进行了深入的调查研究，并整理相关文献资料，总结出实时图像处理技术的特点，最终选择 FPGA 作为实时图像旋转系统的开发平台。传统的图像处理平台处理速度慢，很难保证处理的实时性，而基于 FPGA 的处理平台则在处理高分辨率、大数据量的实时图像方面有天然的优势。然后讨论各种图像旋转算法。与传统的算法优化方法不同，FPGA 是全硬件并行结构处理，适合于数据量大、运算简单的情况，根据这些特点，放弃使用优化的坐标变换算法，而是选择最基本的直接法完成坐标变换，再用双线性插值算法提高图像的质量。最后，完成了图像旋转系统的软硬件设计（软件是指 FPGA 上 VerilogHDL 代码的编写、验证）。 硬件设计主要根据冯诺依曼计算机的组织结构，将系统划分为输入装置、输出装置、存储器以及流处理器。存储器选择了 SDRAM 与 SSRAM 两种类型的存储器，使硬件结构更加灵活。软件设计就是在 FPGA 上实现流处理器，根据层次化、模块化的设计思想，设计各个运算单元，实现流水线方式处理。

本文给出了实时图像旋转系统在 X 光医疗影像诊断设备上的运用结果。实际表明该系统能够满足实时性处理要求，而且得到的图像质量好。同时，从系统的编译报告也可以看出，设计能够满足时序要求，而且剩余资源非常丰富，可用于其它功能的实现。

6.2 展望

本文完成了实时图像旋转系统的功能，能够保证帧频 30 帧每秒，像素时钟 40MHz，

分辨率 1000X1000 大小图像的处理实时性。但是，整个系统仍需在以下几个方面开展进一步研究工作：

(1) 从时序报告来看，SSRAM 控制器时钟 fmax 只能达到 164.64MHz，在处理更高像素时钟的实时图像时会有困难。可以利用时序约束工具，找到系统的关键路径，修改代码，对其进行优化，以提高系统性能。

(2) DDR2 SDRAM 拥有更大的存储空间以及更高的工作频率，可以达到 800MHz 的数据传输速率。用它取代 SDRAM，可以降低系统成本，而且方便系统升级。当然，DDR2 SDRAM 控制器的设计是一个难点。

(3) 本文的实时图像旋转系统，旋转角度由人机界面输入。可以对 FPGA 如何实现正余弦值计算展开研究，比如 CORDIC 算法，这样可以减小系统体积，也符合 SOC（system on chip）设计思想。

(4) 如果要将系统运用于在线产品检测等图像匹配场合，仍需考虑如何在 FPGA 上自动识别倾斜角度。

致 谢

时光荏苒，转眼间已经在华科度过了 6 年的时光。回顾这段求学经历，有太多的人给予我关怀和帮助。在此，我向他们表示诚挚的谢意。

首先，我要衷心地感谢我的导师陈冰讲师、陈幼平教授。感谢两位导师为我创造了和谐宽松的学习、科研环境，感谢导师在论文创作期间时时的鼓励与指导，感谢导师在生活上无微不至的关心与帮助。导师渊博的知识、广阔的胸怀、严谨扎实的工作态度、平易近人的作风使我受益匪浅，是我终生学习的榜样。学生的成长，离不开导师的辛勤栽培。

感谢谢经明副教授对我的关心与帮助，老师悉心的关心与无私的帮助，我常记心中。

感谢王建庄博士对我的指导与帮助。项目期间，博士一直给予我细心的指导和无私的帮助，让我能够更快的进入项目，掌握如何用 FPGA 实现图像处理。博士渊博的知识、敏捷的思维、平易近人的作风、踏实努力的工作态度是我学习的榜样。一起南京出差的日子，难以忘怀。

感谢实验室宋小军、董海涛、周沛、沈忱、乔昱、任向杰、李俊美、翁金飞、张绍林、彭万、田峰等诸多同门在科研生活中对我的关心与帮助。和你们一起切磋、共同长进，每天的生活都是丰富多彩。

此外，感谢我的家人长期以来对我的鼓励与支持。正因为你们，我才能够安心学习，顺利完成学业。你们是我前进的动力与幸福的源泉。

最后，谨以此文感谢所有关心、支持帮助过我的人。

王金辉

2011 年 12 月 于华中科技大学

参考文献

[1] 贾永红.数字图像处理.第 l 版.武汉:武汉大学出版社,2003.

[2] 冯升同,郭立群,欧阳荣.基于 MATLAB 仿真的图像旋转不变识别.实验室研究与探索,2011,30 卷 1 期:5~8.

[3] Christors Davatzikos. Spatial Transformation and Registration of Brain Images Using Elastically Deformable Models[J],Computer Vision and Image Understanding. V01.66,No.2,May 1991:207~222.

[4] 刘妍.基于 FPGA 的数字图像实时消像旋的方法研究:[硕士学位论文].长春理工大学,2008.

[5] SW Park and Y Seo. Real-time camera calibration for virtue studio. Real-Time Imaging,2000,6:433~448.

[6] 田耘,徐文波,张延伟等.无线通信 FPGA 设计.北京:电子工业出版社,2007.

[7] DongDu,RunshiHou and JiaxinShao. Registration of real-time X-ray image sequences for weld inspection. Testing and Evaluation,2010.

[8] Wolherg G. Digital Image Warping[M]. Los Alamitos California:IEEE Computer Society Press,1990:20l~209.

[9] Kiern L G,Michael O A and Klemm H. Fast Fourier method for the accurate rotation of sampled images,1997.

[10] 刘耀林,邱飞岳,王丽萍.基于 GPU 的图像快速旋转算法的研究及实现. 计算机工程与科学.V01.30,No.6,2008:48~51.

[11] 李筱琳,冯燕,何亦征.基于 DSP 的图像旋转算法数据调度策略. 微型电脑应用,2008,24(3):6~8.

[12] 石磊.基于FPGA的视频图像阴影校正关键算法的研究与实现:[硕士学位论文].华中科技大学,2008.

[13] 王建庄.基于 fpga 的高速图像处理算法研究及系统实现:[博士学位论文].华中科技大学,2011.

[14] 徐飞.基于 FPGA 的视频图像旋转的设计与实现:[硕士学位论文]. 江苏大学,2010.

[15] 王滨海,许正飞,陈西广等.图像旋转算法的分析与对比.光学与光电技术,2011,9 卷 2 期:46~49.

[16] 石慎,张艳宁,郗润平.基于 Bresenham 画线算法的图像快速高精度旋转算法.计算机

辅助设计与图形学学报,2007,19 卷 11 期:1387~1392.

[17] J Allebach,PW Wong.Edge-directed Interpolation[A].Proceedings of IEEE International Conference on Image Processing,1996.3:707~710.

[18] SW Lee andJK Paik.Image Interpolation using Adaptive Fast B-spline Filtering [A].Proceedings of IEEE International Conference on Acoustics,Speech,Signal Processing.1993,5:l77~l 80.

[19] 史小白.图形反走样算法及其硬件模型研究:[硕士学位论文].南京航空航天大学,2003.

[20] 帅金晓,颜永红,彭琰等.双线性插值图像放大算法优化及硬件实现.核电子学与探测技术,2009,29:55~58.

[21] Tohru Yamada and Rokuya Ishii.An algorithm to magnify images[J]. Proceedings of the IEEE international Conference on Industrial Technology, 1996:674~677.

[22] OlukayodeA Ojo and Tatiana G.An algorithm for integrated noise reduction and sharpness enhancement[J].IEEE Transactions on Consumer Electronics, 2000, 46(3):476~480.

[23] Mark Blach.完整的数字设计.北京,清华大学出版社,2006.

[24] 张志刚.FPGA 与 SOPC 设计教程——DE2 实践.西安,西安电子科技大学出版社,2007.

[25] 张璐.医学图像去噪方法分析与比较:[硕士学位论文].上海交通大学,2010.

[26] A Rosenfeld and M Thurston,Edge and curve detection for visual scene analysis. IEEE Trans.Computers,1971, 20:562~569.

[27] J Babaud,AP Witkin and M Baudin. Uniqueness of the Gaussian kernel for scalespace filtering. IEEE Trans.Pattern Analysis and Machine Intelligence, 1986,8:26~33.

[28] T Kasparis,N.S Tzannes and Q Chen. Detail-preserving adaptive conditional median filters,Electronic Imaging,1992,1:358~364.

[29] 冯鹏,魏彪,米德伶等.基于时域递归滤波的动态数字图像降噪.重庆大学学报,2005,2(28):23~28.

[30] 何云斌,张玉芬.多端口 SDRAM 控制器的设计与实现.微计算机嵌入式系统运用,2009,25:71~73.

[31] J Duprat and JM Muller. The CORDIC algorithm: New results for fast VLSI implementation, IEEE Trans. Computers, Feb 1993,vol.42:168~178,.

[32] JAvLee and T Lang. Constant-factor redundant CORDIC for angle calculation and rotation, IEEE Trans. Computers, Aug 1992, vol. 41: 1016~1025

[33] Xiong Cheng-Yi, Tian Jin-Wen and Liu Jian. Efficient high-speed/low-power line-based architectures for two-dimensional discrete wavelet transform using lifting scheme.

IEEE Trans. on Circuits and Systems for Video Technology, 2006, 16(2):309~316

[34] 方勇,吕国强,彭良清.3D 显示器视频转换系统设计及其 FPGA 实现.液晶与显示,2007,2(22):94~97

[35] 王彦刚,彭东林,易文翠等.乒乓操作方法在增量式时栅信号处理电路设计中应用.电力自动化设备,2007,1(27):84~86

[36] 王智,罗新民.基于乒乓操作的异步 FIFO 设计及 VHDL 实现.电子工程师.2005,6(31):13~16

[37] 杨丽杰,崔葛瑾.基于 FPGA 的 FIR 滤波器设计方法的研究.2006,12(32):93~96

[38] A Chandrakasan and W Bowhill.Design of High Performance Microprocessor circuits, IEEE Press, 2000

[39] GD Michdi. Synthesis and Optimization of Digital Circuits[M]. McGraw-Hill 1994

[40] CT Gray,Liu W and Cavin R.Timing constraints for Wave-pipelined systems [J].IEEE Trarmction on Ccfnputer-Aided Design,1994,13:987~1004

[41] Altera.TimeQuest Timing Analyzer Quick Start Tutorial.Dec 2009. Available: www.altera.com

附录 攻读学位期间发表学术论文目录

[1] 王金辉,陈冰,王建庄. 实时图像仿射变换系统的研究与实现.机械与电子（已录用）

实时图像旋转系统的研究与FPGA实现

作者：王金辉
学位授予单位：华中科技大学