基于可重构处理器的视觉信息预处理子系统研究

Research of Visual Information Pre-Processing Sub-System Based on Reconfigurable Processor

学科专业：微电子学与固体电子学
研 究 生：王磊
指导教师：徐江涛 副教授

天津大学电子信息工程学院
二零一三年十二月

独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作和取得的研究成果，除了文中特别加以标注和致谢之处外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得 **天津大学** 或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

学位论文作者签名：　　签字日期：　　年　　月　　日

学位论文版权使用授权书

本学位论文作者完全了解 **天津大学** 有关保留、使用学位论文的规定。特授权 **天津大学** 可以将学位论文的全部或部分内容编入有关数据库进行检索，并采用影印、缩印或扫描等复制手段保存、汇编以供查阅和借阅。同意学校向国家有关部门或机构送交论文的复印件和磁盘。

（保密的学位论文在解密后适用本授权说明）

学位论文作者签名：　　导师签名：

签字日期：　　年　　月　　日　　签字日期：　　年　　月　　日

摘 要

随着多媒体技术和数字图像技术的发展，标清、高清甚至超高清视频迅速得到普及。在带来更清晰的画质和视觉享受的同时，视觉信息处理中的数据量及运算复杂度也急剧上升。可重构处理器相对于传统处理器能够达到更高的计算功率效率，从而有效解决以上问题，成为视觉信息处理领域中的一个新的研究热点。

本文介绍了基于可重构处理器的视觉信息处理系统的整体架构，研究分析了预处理子系统各模块的功能以及系统的数据流。然后设计了预处理子系统的各个模块电路：通过 SCCB 总线对图像传感器内寄存器进行配置，使传感器输出指定大小、格式的视频数据；设计了中值滤波电路，减小数字图像在传输过程中产生的噪声，降低图像噪声对角点检测、图像拼接等高级算法的影响；设计了 DDR 读写控制电路，使图像数据在 DDR 中实现帧缓存；为实现视觉信息预处理子系统与可重构处理器之间的通信，定义了两者之间的通信协议，并设计了预处理子系统与可重构处理器的接口电路；利用 Microblaze 处理器核作为主控制器，通过 PLB 总线完成系统初始化，制式校验，功能选择，数据调度等任务。此外，本文提出了一种基于图像边缘提取的开关加权矢量滤波算法，该算法在彩色图像的降噪处理过程中，能够有效的保护图像细节。

本文设计的各模块电路在 Modelsim 下进行的功能仿真，并在 Xilinx FPGA 开发平台 Virtex-5 XUPV5 LX110T 上进行了验证。采用 OmmiVision 公司的图像传感器芯片 OV2640 作为预处理子系统的图像数据源。实验结果证明预处理子系统能够正确接收图像数据源输出的数据，预处理子系统中各模块以及整个预处理系统均能正常工作。对于改进的矢量中值滤波算法，通过 Matlab 仿真，计算 MAE、MSE、PSNR 等评价指标。对大量测试图像进行实验，统计结果显示本文提出的算法相对于标准矢量中值滤波算法， PSNR 值提高了 7.2%。

关键词：视觉信息处理，图像降噪，中值滤波，可重构处理器

ABSTRACT

With the fast development of multimedia technology and digital image technology, standard definition (SD), high definition (HD) and even ultrahigh definition video quickly gained popularity. Bringing clearer picture quality and visual enjoyment, the amount of data and computation complexity in visual information processing has also increased sharply at the same time. Compared with traditional processor, reconfigurable processor can achieve the higher calculation efficiency, so as to effectively solve the above problem, and it has become a new research focus in the field of visual information processing.

This thesis introduces the architecture of the visual information system based on reconfigurable processor and analyses the data flow in pre-processing sub-system. Then modules of the pre-processing sub-system are designed as follows: the image sensor was configured by SCCB to make the image sensor output specific image format and image size. A median filter circuit was designed to eliminate impulse noise in the image and to reduce the impact of noise on corner detection, image mosaic and other advanced image processing algorithm. A DDR controller was designed to realize image frame caching in the DDR memory. To realize communication between preprocessing system and reconfigurable processor, this thesis defined a communication protocol between them. The whole system was controlled by Microblaze, a processor core supported by Xilinx to complete system initialization, function change, format verification and data flow control. This thesis also proposed a weighted vector median filter based on edge detection to protect image details when filtering the color image.

All the circuits designed in this thesis was simulated on Modelsim and verified on the platform of LX110T. The OV2640 chip of OmniVision was used as the HD image data source. Experimental results show that the system can correctly receive image data and the whole system can work in a right way. Simulation results on Matlab show that the proposed VMF algorithm outperforms all algorithms examined in this thesis in terms of MAE, MSE and PSNR values. Statistical results show that the PSNR values increased by 7.2% compared with standard VMF algorithm.

KEY WORDS： Visual Information Processing, Image Denoising, Median Filter, Reconfigurable Processor

目　　录

第一章 绪 论

1.1 可重构处理器概述

随着多媒体技术的不断进步，需要处理的数据量越来越大，对数据处理电路的性能、功耗和灵活性要求也越来越高。传统的解决方案通常采用两种不同的方法来实现：一种是专用集成电路（Application Specific Integrated Circuits，ASIC），一种是通用微处理器[1]。对于复杂的视觉信息处理算法，ASIC 设计具有很高的执行效率和运算精度且具有很低的功耗，但是灵活性不足，很难同时满足多种不同算法的需要。通用处理器虽然能够通过软件指令实现不同的算法，但复杂的算法通常需要很多个时钟周期才能完成，执行效率很低。

可重构计算在 20 世纪 60 年代提出[2]，并在最近 10 年有较大的发展。可重构处理器可以根据不同的用户需求动态地配置电路的实现形式，兼具了 ASIC 高性能以及通用处理器高灵活性的优点。根据可重构功能单元的复杂程度，可重构处理器可分为粗粒度可重构、细粒度可重构处理器。现场可编程门阵列（FPGA）是典型的细粒度可重构结构，以查找表作为基本的可重构功能单元，重构灵活度高，能够实现位级操作。粗粒度的可重构结构通常具有字节操作级的基本可重构处理单元。典型的粗粒度可重构处理器由一个或多个通用微处理器以及可重构单元阵列组成，如图 1-1 所示。粗粒度可重构一般具有通用算术逻辑单元（Arithmetic Logical Unit, ALU），因此与细粒度 FPGA 相比，运算性能更优，所需要的配置数据和配置时间也更少。可重构功能单元阵列的配置信息由通用处理器决定，因此用户可通过高级语言编程，修改设计可重构结构的功能。

近年来，国内外对于视觉信息处理的可重构处理器的研究，根据可重构的用途可分为两大类：一种是针对某一种或者几种视觉信息处理算法的专用可重构处理器，用于某些特定场合；一种是面向整体视觉信息处理的通用可重构处理器。

关于专用可重构处理器，国内外各研究机构的研究热点主要是针对视频编解码，DCT/IDCT，视频监控，运动估计，特征提取等特定算法所设计。如台湾国立成功大学，将基于上下文的自适应二进制算数编码（CABAC）和基于上下文的自适应可变长度编码（CAVLC）两种方法相结合，设计了一种支持高清视频H.264编码的可重构处理电路[3]。英国布里斯托大学于2011年，设计了支持H.264, VC-1 and AVS三种视频编码标准的可重构处理器，同时还设计了开发编译工具，

支持C语言风格编程[4]。台湾国立中兴大学的Yeong-Kang Lai ，Yu-Fan Lai等人[5]设计了一个可重构的IDCT电路，能够实现8×8, 8×4, 4×8和4×4的IDCT。主要优势在于该结构中采用新的分布式数学算法，使得电路中不需要乘法器和ROM，而只包括加法器和移位器，有效减小了电路面积，采用90nm工艺，能够实现720P和1080P高清电视信号的实时处理。

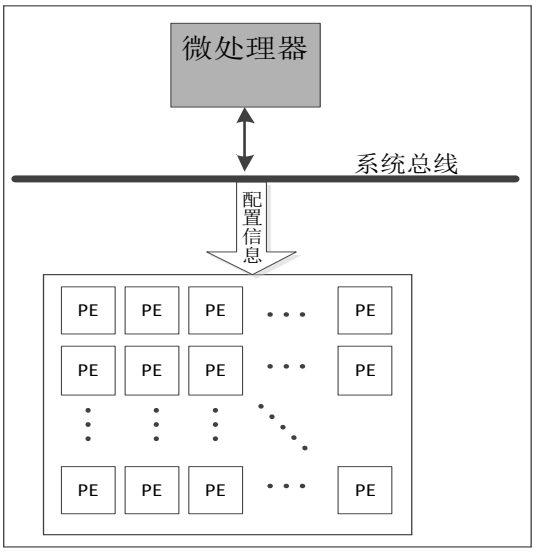


图 1-1 典型可重构处理器结构

相比专用可重构处理器，通用可重构处理器系统更加复杂，设计更加困难，但相应的功能也更加全面，应用更为广泛，研究价值也更高。意大利博洛尼亚大学[6]将细粒度（1-bit）的 FPGA，中粒度（4-bit）的可配置处理器以及粗粒度(16-bit)的可重构阵列结合在一起，通过灵活高效的 64 位片上网络（network on chip，NOC）方式进行通信，设计成功一款 SOC。该款 SOC 以 ARM 核作为主处理器，ARM 处理器搭载操作系统，控制整个系统的通信、同步以及可重构机制。通过 ARM 处理器编程可实现高级的数据同步，全局数据信号处理等操作，而大部分的运算则分配给各个适当的可重构单元。芯片基于 90nm 工艺，峰值功耗为 2.5W。执行视频监控运动检测算法时，该 SOC 能够传送 120GOPS，功耗为 1.45W。以往的设计总是基于大量并行处理单元来提高运算速度，满足视频处理的需求，但这些设计仅仅关注特征提取部分，却没有专门的硬件电路来支持各种高级机器学习算法。台湾大学设计了一款 Semantic Analysis SOC（SASOC）[7]，能够同时进行视频处理和机器学习。这款 SASOC 有以下几个特点：（1）集成了两个子系统：一个图像流处理系统，支持像素级特征提取操作；一个特征流处理系统，支持向

量级机器学习算法。（2）层次化的存储结构和数据流网络式设计。（3）通过层次化的 3 级向量处理器结构，支持高级机器学习算法，同时支持很大的吞吐量。（4）多时钟域的动态频率技术，通过动态地平衡负载，使得功耗减小 50%。东京大学 Takashi Komuro 等人 2010 年设计了一种 1000f/s 的视觉信息嵌入式处理系统[8]，通过硬件可重构以及简单的算法执行，可以满足所有的性能要求。该系统包括一个嵌入式微处理器和两块 FPGA。每块 FPGA 在系统中作为协处理器，包含相应的存储器，DMA 控制器，以及图像处理单元。主控制单元通过指令寄存器阵列向各个协处理器发送指令，使得图像处理单元能够执行任务级并行和像素级并行处理。2012 年，韩国科学技术院设计了一款多媒体处理器[9]，主要用于图像处理，机器视觉，3D 图形以及 augmented reality（AR）。该处理器嵌入了用于外部存储器接口的可重构输出驱动，根据运行的处理器与存储器的通道失配情况，配置不同的驱动能力，能够实现 8 倍于以前的处理器的存储带宽。处理器主要由 3 个可编程的部分组成：模式可配置的向量处理单元（MCVPUs）；统一的滤波单元（UFU）；统一的掩模。MCVPUs 包含 32 个 16bit 处理器核，能够支持图像级与图形级处理两种可配置的工作模式。可配置的模式使得在 AR 应用中能够实现流水线的帧处理，与串行 AR 处理相比，能够实现 1.7 倍加速。UFU 支持 16 种滤波操作，提高多媒体处理器的速度和性能。2011 年，香港 Polytechnic 大学的 Wing-Yee Lo 等人设计了一款基于 SIMD 架构的高性能视频处理器[10]。该处理器中采用一种新的 SIMD 架构，有两个新的特点：1）并行存储结构（具有可变块大小和长度的支持），允许程序员有很大的灵活性来执行不同大小的块和不同长度的字的数据存取。2）可配置 SIMD 结构，通过一个交叉开关允许几乎“随机”的数据访问 SIMD 寄存器。美国德克萨斯大学奥斯汀分校 2012 年设计了一款可以对单像素进行操作的动态可重构处理器[11]，基于多目标优化策略，建立了功耗-性能-精度空间（Power-Performance-Accuracy space，PPA），PPA 的值存在 DDR 里面，通过读取 DDR 的值，动态管理与控制处理器的功耗、性能、精度。

1.2 视觉信息处理系统简介

基于可重构处理器的视觉信息处理应用系统设计是国家“863”计划项目—面向通用可重构处理器的关键技术的研究的子课题，主要用于本课题中所使用的通用可重构处理器对于视觉信息处理计算需求的示范应用和演示，围绕可重构处理器的应用开发和功能验证进行设计。

该系统有三个主要模块组成：视觉信息预处理子系统、可重构处理器子系统、

视觉信息后处理子系统。其中视觉信息预处理子系统用于实现多路高清视觉信息源数据的采集、格式转换，以及数字图像的降噪、白平衡，并且提供与可重构处理器及后处理子系统的接口和必要的数据缓存等。可重构处理器负责典型视觉信息算法的实现，如：归一化互相关 (NCC)、三次曲线插值、Harris 角点检测、加权中值滤波以及用于特征点匹配的 RANSAC 匹配、离散余弦变换等。视觉信息后处理子系统负责视频增强、图像缩放等功能，并将处理后的结果进行标准制式显示。本论文主要进行视觉信息预处理子系统的研究与设计。

通用处理器在进行图像的边缘检测、特征提取、模式识别、拼接融合等复杂的视觉算法处理之前，为了便于处理器对视觉信息进行分析、理解和识别，必须对从视觉信息源采集到的原始数据进行一定的预处理。视觉信息预处理技术主要用来突出有用视觉信息，抑制无关视觉信息同时改善图像的视觉效果。常见的视觉信息预处理算法包括图像中噪声的抑制、图像对比度的加强、图像边缘特征的增强、图像白平衡处理、伪色彩处理等。本论文主要对图像降噪技术进行了研究与设计。

图像降噪的目的在于根据容许的性能来改善图像的质量[12]。图像降噪算法按照其处理的图像域的不同可以分为两大类：像素域图像降噪算法与转换域降噪算法。

像素域降噪算法的实现方式是直接对图像的像素值执行滤波处运算。像素域滤波一般可分为平滑线性滤波和非线性滤波。均值滤波器和 Wiener 滤波器是典型的空间线性滤波器。均值滤波器通过计算当前像素一定邻域内像素的平均值来取代当前像素的值，是一种最简单的滤波降噪方法。均值滤波模糊了图像，并且降噪性能也比较差。Wiener 滤波器[13]利用图像的均值、方差等局部统计特性，进行图像的先验局部统计估计，并依据最小均方差准则进行降噪处理。统计排序（非线性）滤波器首先将滤波窗口所包含的图像区域中的像素以某种方式进行排序，然后将统计排序结果所决定的值来代替当前像素。中值滤波器是最知名的统计排序滤波器，其对噪声的抑制能力比均值滤波更优秀，同时还能有效地保护图像的细节。标量中值滤波更适用于灰度图像的降噪处理，针对多通道的彩色图像，人们提出了矢量中值滤波[14]的算法，解决了标量中值滤波处理彩色噪声图像时产生人造色彩的问题。为了改善滤波效果，解决传统中值滤波算法降噪能力与保护图像细节之间的矛盾，人们提出了多种改进的滤波算法[15,16,17]，主要包括加权中值滤波及开关中值滤波两大类。Yaroslavsky 邻域滤波是另一种常见的非线性滤波器，其基本工作原理是通过计算窗口中心像素点的灰度值与邻域点的相似性来确定滤波窗口的各个系数。对于视频图像而言，除了在二维空间域以外，图像内

容在一维时间域也存在一定的相关性。因此对视频图像的降噪处理还可以利用像素在视频流中相邻帧图像的邻域空间内的所有像素的相关性，基于运动补偿（Motion Compensation，MC）的方法，形成一个空时域的三维滤波器。有限冲击响应滤波器（FIR）和递归滤波器（IIR）是三维滤波器的两种主要实现形式。

图像噪声一般集中在高频段，因此可以通过低通滤波的方式进行降噪处理。通过傅里叶变换，将图像信号转换到频域，然后选择某种低通滤波器（Low Pass Filter，LPF），例如巴特沃斯低通滤波器、高斯低通滤波器等，将高频信号滤除，实现降噪的目的。另外，小波理论由于具有良好的时频局域化特性，近年来在图像降噪、分割、压缩领域得到了广泛的应用。

本论文设计的预处理子系统还担当着整个原型演示系统的主控制器的角色。预处理子系统中各个子模块以及可重构处理器和后处理系统在预处理子系统中CPU 的控制之下，完成初始化，制式校验，功能选择，数据调度等任务。

传统处理器设计领域国内与国外存在着巨大的差距，而在可重构处理器领域，国内外起点相近。对可重构处理器的研究有助于缩小中国在处理器方面与国际的差距。基于可重构处理器的视觉信息处理系统的研究能够为可重构处理器提供良好的验证平台，推动可重构处理器的研究发展。

1.3 本文的主要研究内容及论文组织结构

本文主要研究设计了一套基于可重构处理器的视觉信息预处理子系统，由CMOS 图像传感器输出的图像数据经过格式转换、滤波降噪等处理，进入 DDR 帧缓存，然后发送给可重构处理器及后处理系统；提出了一种消除冗余计算的矢量中值滤波电路；提出了一种基于图像边缘检测的开关加权矢量中值滤波算法；对系统的各个功能模块进行软件仿真及 FPGA 验证；对提出的改进矢量中值滤波算法进行软件仿真。具体章节安排如下：

第一章介绍了可重构处理器和视觉信息预处理技术的研究背景、研究意义以及国内外研究现状，本论文的主要研究内容和章节安排。

第二章介绍了基于可重构处理器的视觉信息处理系统的总体架构，对预处理子系统中的数据流进行了分析。

第三章详细介绍了视觉信息预处理子系统中各个功能模块的设计过程，包括图像传感器配置，视频数据接收，图像制式转化，中值滤波降噪处理，图像帧缓存控制以及预处理子系统与可重构处理器通信接口等。

第四章提出了一种消除冗余计算的矢量中值滤波电路和一种基于图像边缘

缘检测的开关加权矢量中值滤波算法。

第五章对系统的各个功能模块进行了软件仿真和 FPGA 功能验证，对实验结果进行分析，并对提出的改进算法进行 Matlab 仿真。

第六章对本论文的研究内容进行了总结，分析了目前工作中需要解决的问题，并对接下来的研究工作做了展望。

第二章 预处理子系统总体架构

2.1 视觉信息处理系统整体架构

本设计开发的视觉信息处理系统主要用于为可重构处理器提供高清图像数据源、多种外设接口和图像数据的输出结果，能够将多种视觉信息处理算法的运算结果显示给用户，同时为用户提供该可重构处理器的开发环境，在对可重构处理器进行应用开发、功能验证的同时，向用户提供可重构处理器在各项应用中的性能表现的量化表征和评测结果，以反馈给设计者对可重构处理器进行优化和改进。

如图 2-1 所示，完整的视觉信息处理系统包括视觉信息数据源及云台控制系统，视觉信息预处理子系统、后处理子系统以及可重构处理器子系统。其中视觉信息数据源由 CMOS 图像传感器提供，为整个系统提供高清视频数据。图像传感器固定在云台上，单片机通过控制云台上的步进电机可以对图像传感器的方位和姿态进行调整，以适应视觉信息处理算法的要求。预处理子系统将采集到的视觉信息和数据，一方面根据具体的功能要求进行基本的预处理，如图像降噪、制式转换、图像缓存等，另一方面需要实现与可重构处理器以及后处理子系统的数据通信，并可以根据可重构处理器反馈回的运算结果向云台控制系统发送相关的控制信息，以调整图像传感器的角度。可重构处理器子系统包括通用微处理器和可重构单元阵列，同时拥有其独立的 DDR 存储器和相关控制电路。图像数据经预处理子系统处理后发送给可重构处理器，可重构处理器根据外部指令进行归一化互相关 (NCC)、三次曲线插值、Harris 角点检测、加权中值滤波以及用于特征点匹配的 RANSAC 匹配、离散余弦变换、图像拼接融合等处理，并将处理后的运算结果反馈给预处理子系统。后处理子系统从预处理子系统接收图像数据，并获得图像拼接融合所需要的仿射矩阵，完成双路视频的融合过程，同时由于融合后的图像为不规则图像，后处理系统还要实现对图像的合理裁剪以便于缩放至标准视频图像格式。裁剪后的图像经过水平方向双三次插值，垂直方向双线性插值进行缩放处理。为改善视频图像的显示效果，还要对视频图像进行直方图均衡化的视频增强处理过程，完成图像的对比度增强。最后将视频图像转换到标准格式并利用 CH7301 驱动芯片，通过 DVI 接口将图像数据在显示器上输出显示。显示格式可支持 800 ×600，1024 ×768，1280 ×720，1366 ×768，1366 ×1024，

1400 ×1050，1600 ×1200，1920 ×1080 等格式。

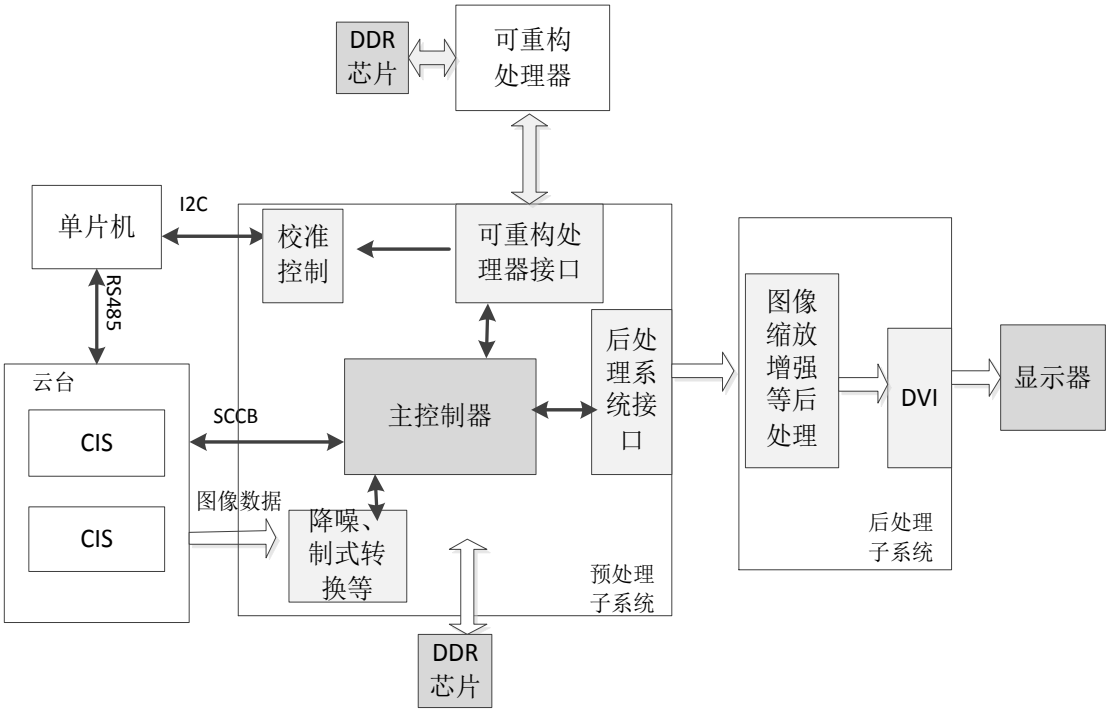


图 2-1 基于可重构处理器的视觉信息处理系统总框图

2.2 高清视频数据源及输出时序

2.2.1 高清视频数据源

OV2640芯片是OmniVision公司生产的一款CMOS图像传感器芯片。该芯片输出的图像信号即为数字图像信号，不用再进行A/D转换。该款图像传感器芯片的像素阵列为1600×1200，因此可以支持最高达1600×1200的分辨率，像素尺寸为2.2um×2.2um。在最大分辨率下可实现帧频为15fps，分辨率为SVGA时输出帧频为30fps，当分辨率缩小到CIF格式时，输出帧频可达60fps。通过SCCB总线（Serial Camera Control Bus）对该芯片进行配置，可实现UXGA、SXGA、SVGA以及从SXGA到40×30间任意大小的图像，同时可实现YUV（422/420）、原始RCB格式以及RGB565/555等不同格式的图像输出。该款图像传感器芯片输出稳定图像的温度范围为0 °C 至 50 °C 。传感器的动态范围为50dB，灵敏度为0.6V/Lux-sec。

OV2640图像传感器芯片内有多个功能寄存器，通过SCCB总线对这些寄存器进行配置，可以改变图像传感器输出图像的分辨率、格式、帧频是否压缩等。通

过SCCB对OV2640芯片进行读寄存器操作时，其设备地址为61，进行写寄存器操作时，其设备地址为60。通过读取传感器内某些寄存器的值，可以判断图像传感器的工作模式和工作状态。当对传感器内的某一个寄存器配置完成后，需要对该寄存器进行读操作，若果读取的寄存器数值与之前配置的值相同则说明配置成功。否则说明配置失败，需要重新进行配置。

2.2.2 图像传感器数据输出时序

通过SCCB总线完成对图像传感器芯片内的寄存器的配置之后，图像传感器便会输出指定格式的图像数据。OV2640芯片的主要接口信号如下：

- 3v3：输入电源电压（3.3V）
- GND：地
- SIO_C：SCCB时钟线
- SIO_D:SCCB接口的串行数据输入输出端
- VSYNC: 帧同步信号（输出）
- HREF：行同步信号（输出）
- PCLK：像素时钟（输出）
- XCLK：时钟信号（输入）
- D0-D9：数字图像数据（输出）
- RESET:复位（低电平有效）
- PWDN：功耗模式选择（正常使用拉低）

其中数据输出为8-bit格式时，选择D2-D9 8位数据信号。

以UXGA大小的YUV422格式的图像输出为例，图像数据与PCLK同步，当HREF信号为高电平时，图像数据有效。像素输出时序如图2-2所示，可以看出，当行同步信号HREF为高电平时，在PCLK的上升沿对图像数据进行采样，可以接收到有效地图像数据。在一个行周期内，HREF信号保持高电平不变，也就是说同一行的像素输出是连续的。但是行与行之间的像素输出是不连续的，相邻行像素的有效数据输出会有若干个像素时钟的时间间隔。

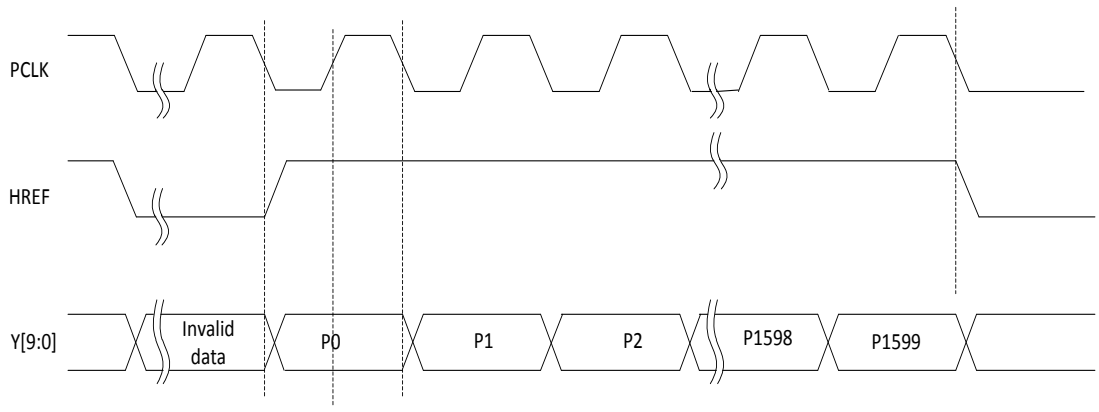


图2-2 OV2640的像素输出时序

OV2640芯片输出图像的帧格式如图2-3所示。其中t_p为像素时钟周期当帧同步信号VSYNC为高电平时表示将要产生一帧图像。VSYNC信号保持4个行时间的高电平后变为低电平，在经过一定周期后行同步信号有效，图像传感器开始输出有效图像数据。

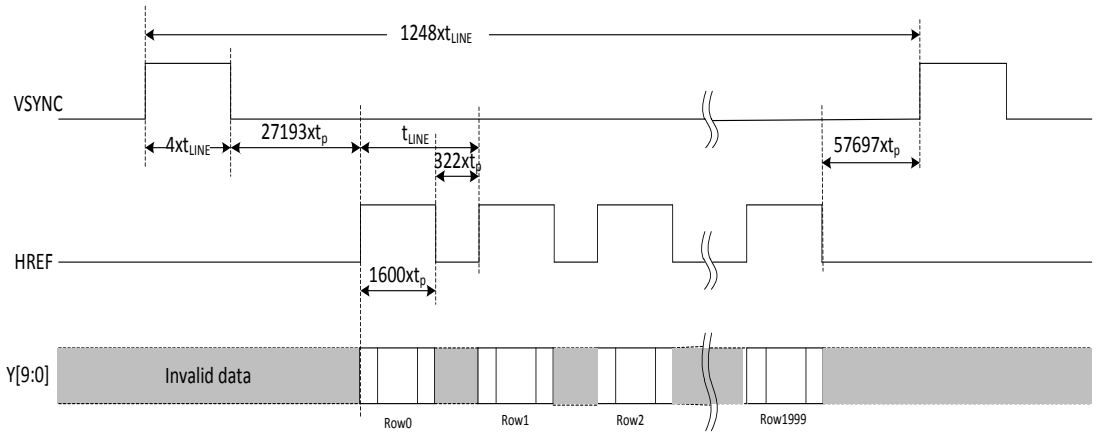


图2-3 OV2640芯片图像输出的帧时序

2.3 视觉信息预处理子系统的数据流

图2-4为视觉信息预处理SOC系统的总体架构示意图。其中FPGA采用Xilinx公司的Virtex-5芯片。CIS（CMOS Image Sensor）是高清视频数据源，采用OmniVision公司的OV2640芯片。该芯片利用SCCB总线进行相关功能寄存器配置，以实现指定大小图像输出，SCCB master是SCCB主机模块。SRAM行缓存模块用来对接收的图像数据进行缓存，以满足滤波模块的数据需求。格式转换模块用来实现YUV图像格式与RGB图像格式之间的转化，此模块属于可选功能，在Microblaze处理器控制下选择是否进行该处理过程。滤波降噪模块采用中值滤波的方法，消除图像数据在传输过程中产生的噪声，提高图像质量。DDR控制器用来进行DDR内存芯片的读写控制，系统使用DDR内存对图像数据进行帧缓存。可重构处理器接口与后处理系统接口用来控制与可重构处理器以及后处理系统的数据交互。

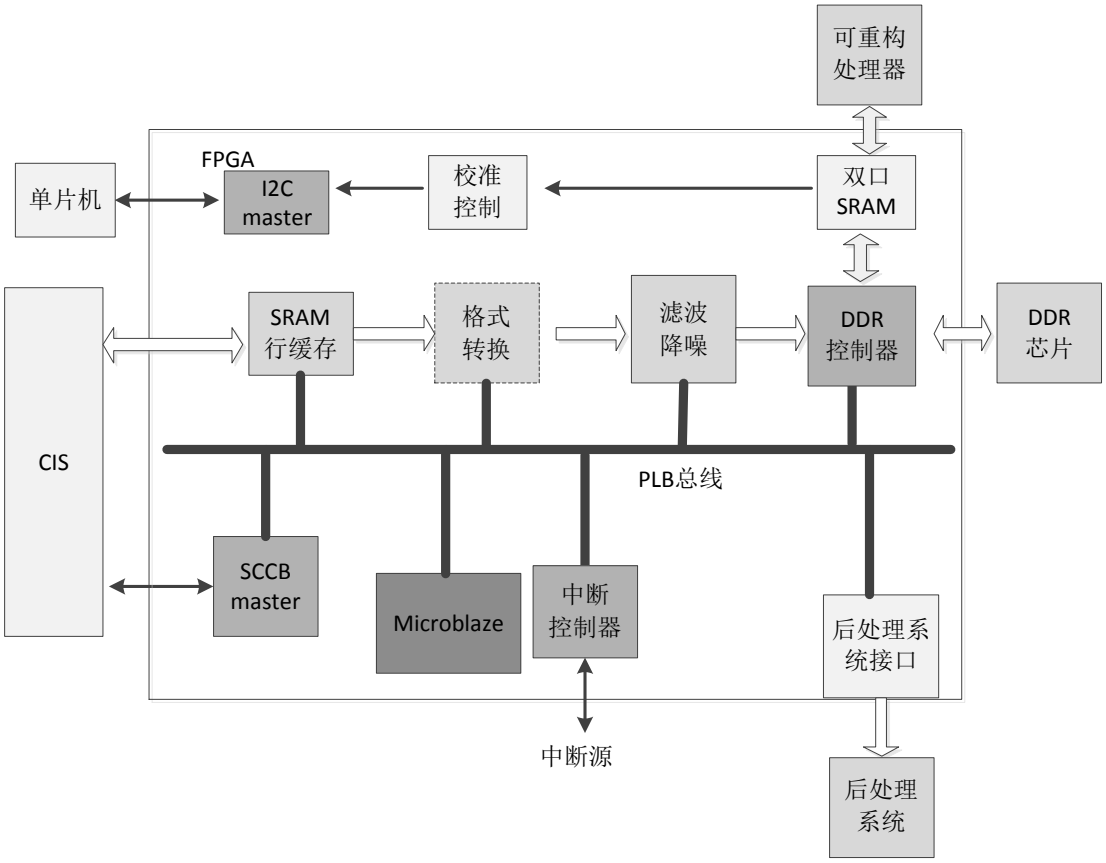


图2-4 视觉信息预处理子系统总体架构

首先是系统启动过程：电源接通以后，开始向预处理子系统，可重构处理器以及后处理系统供电。首先，预处理子系统向CMOS图像传感器，可重构处理器以及后处理系统分别发出并保持一个RESET信号，使各个子系统恢复到初始状态。稳定供电后，撤去RESET信号，然后预处理子系统开始执行自检程序，通过读取各子系统中相关寄存器的状态检查各部分是否正常工作以及是否准备就绪。各子系统初始化完成以后，向预处理子系统发送初始化完成信号。预处理子系统接收到反馈信号后，便确认各个子系统已经准备就绪，可以开始正常工作，初始化过程完成。

初始化完成后，Microblaze处理器将图像传感器的配置命令及配置信息通过PLB总线发送给SCCB 主机模块。SCCB主机接收到配置命令后，首先产生起始信号，然后依次发送图像传感器的设备地址和相应的寄存器地址，接着发送相应的配置数据。对图像传感器的配置主要包括输出图像的分辨率和图像制式两方面内容。其中对图像分辨率的配置主要可以通过改变寄存器COM7（寄存器地址为0x12）以及寄存器HSIZE、VSIZE和SIZEL（寄存器地址分别为0xC0、0xC1和0x8C）的值来实现。通过配置COM7可以使图像传感器输出UXGA、CIF、SVGA三种不

同分辨率的图像，而配置HSIZE、VSIZE以及SIZEL寄存器的值可以实现从SXGA到40×30之间任意分辨率的图像尺寸。而通过配置寄存器IMAGE_MODE（寄存器地址为0xDA），可以实现图像传感器输出YUV422、RAW10、RGB565等不同制式的图像数据，对于YUV422制式的图像还可以通过配置寄存器CTRL0使传感器输出时序为YUYV或者是VYUY。另外还可以改变寄存器R_DVP_SP的值来改变图像输出的帧频大小。

SCCB主机对图像传感器配置完成后紧接着对刚才所配置的寄存器进行读取操作，并将读取到的数值发送给Microblaze处理器。Microblaze处理器接收到该数据后，与之前发送的配置信息对比，若两者不一致，说明配置失败，需要从新发送配置信息，重复上述配置过程。若两者一致，则证明图像传感器配置成功，向图像采集模块发送使能信号，开始接收图像数据。

图像采集电路按照图像传感器的数据输出时序，通过行同步信号和帧同步信号判断图像数据的有效性，从而采集有效图像数据，忽略无效数据。图像数据进入预处理子系统后首先进行制式识别检测，以验证接收的图像分辨率以及帧频是否正确。然后对图像数据进行行缓存处理，以满足后续滤波模块的数据需求。这里采用SRAM进行行缓存，由于传感器输出的最大分辨率为1600×1200，所以需要预留的存储空间为3×1600×16bits。行缓存的数据输入和存储方式如图2-5所示。

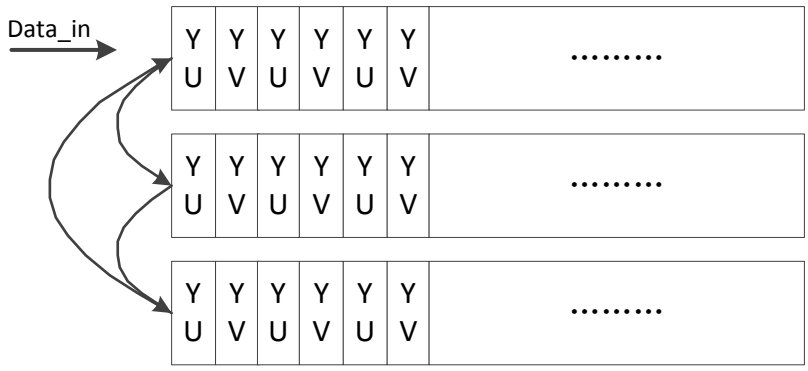


图2-5 图像行缓存格式

图像数据在采集和存储过程中常常会引入图像噪声，这些噪声将严重影响高层次图像处理算法的处理效果，因此图像数据行缓存后将进行滤波降噪处理。中值滤波算法既能够有效抑制图像中的脉冲噪声，同时还能够保护图像细节，因此本文中采用中值滤波的方式消除图像中的噪声。综合考虑降噪效果和硬件资源消耗，滤波窗口采用3×3的大小进行。

由于角点检测、特征匹配等图像处理算法主要是基于灰度图像处理，而静态

图像的显示则一般以RGB24bits格式显示。因此，经过滤波降噪处理后的图像数据根据实际需要选择是否进行制式转换处理。如果需要进行转换，则进入YUV422转RGB模块进行处理。

之后，图像数据进入DDR存储器进行帧缓存处理。由于之前的处理电路与DDR时钟不在同一个时钟域，因此在DDR读写控制端之前需要添加异步FIFO模块。DDR读写控制采用NPI接口IP进行，每一路的读写数据分别占用一个控制端口，这样外部电路只需根据控制器提供的相应信号进行独立的读写控制，而不需要进行仲裁处理。

对于接收的前两帧图像，需要将图像数据以及相应的控制信息（如图像的格式、大小信息，需要进行何种视频算法运算等）发送到可重构处理器进行拼接融合算法的处理。预处理子系统与可重构处理器通过双口SRAM进行通信。可重构处理器根据预先定义的通信协议，正确接收图像数据，并根据相应的指令信息执行Harris 角点检测、归一化互相关、以及RANSAC特征点匹配等视觉信息处理算法，然后将处理后的结果（如是否拼接成功，拼接算法生成的仿射矩阵等）反馈回预处理子系统。

预处理子系统根据可重构反馈回的信息判断是否需要对摄像头的方向、位置进行调整。如果需要调整，则将相应的调整信息通过I2C总线发送给控制云台的单片机。单片机根据接收到的调整信息，通过RS485协议控制云台的步进电机，从而实现摄像头的机械调整。如果不需要对摄像头进行调整，则将图像信息以及拼接融合所需要的必要信息发送给后处理系统，作进一步的处理和最终的输出显示。

Microblaze作为整个系统的主控制器，以中断的方式接受外部命令，从而改变系统的数据流和功能。

Microblaze支持外部中断源（连接到中断输入端口）。如果MSR寄存器的中断使能位IE设置为1，处理器只响应中断。在中断时，完成执行级的指令，而译码级的指令被一个跳转到中断向量（0x10）的分支替换。中断返回地址自动的加载到R14寄存器。此外，处理器通过清除IE位禁止将来的中断。Microblaze只有一个终端输入端口，如果有多个中断，必须添加中断控制器INTC。INTC能够实现设置中断优先级，屏蔽低级中断等功能，中断触发方式包括电平触发（高、低）和边沿触发（上升沿、下降沿） 。本论文采用外部开关模拟OSD菜单的方式向Microblaze发送中断控制命令。FPGA开发板中，dip开关通过GPIO与Microblaze的中断输入端相连。GPIO中有三个与中断相关的寄存器：

- GIE：全局中断使能 （ 1=enable； 0=disable）
- IER：输入中断信号使能 （1=enable； 0=disable）
- ISR：中断状态寄存器（中断清除） 写1表示清除中断标志

Microblaze 接收到外部中断控制信号后，根据相应的中断源和中断信息判断外部指令类型，改变预处理子系统中的状态寄存器，使视觉信息处理系统实现算法演示，可重构性能分析，图像拼接融合等多种功能。

2.4 预处理子系统的输出反馈控制

在视觉信息处理系统完成图像的拼接融合过程中，图像数据进入可重构处理器经过图像角点检测、归一化互相关及随机一致性检测等算法处理后，会计算出一个用于图像拼接的仿射变换矩阵，该仿射变换矩阵的大小为 3×3，矩阵的形式如下：

$$H=\begin{bmatrix} s_x\cos\theta & s_x\sin\theta & 0 \\ -s_y\sin\theta & s_y\cos\theta & 0 \\ \delta_x & \delta_y & 1 \end{bmatrix} \tag{2-1}$$

其中 θ 代表图像旋转角度，δ_x，δ_y 分别代表水平平移量和垂直平移量 S_x、S_y 分别为水平缩放和垂直缩放系数。获取图像的图像传感器之间存在平移、旋转、缩放。变换后图像中的物体形状和大小都会变换但是图像中的平行线变换后还是平行线。

对于同一个图像传感器或者同样型号的图像传感器输出的图像而言，水平缩放系数和垂直缩放系数相等，变换后图像的物体形状不会变化，但是大小会发生变化。本系统就属于这种情形。此时变换矩阵为：

$$H=\begin{bmatrix} r\cos\theta & r\sin\theta & 0 \\ -r\sin\theta & r\cos\theta & 0 \\ \delta_x & \delta_y & 1 \end{bmatrix} \tag{2-2}$$

其中 θ 代表图像旋转角度，δx，δy 分别代表水平平移量和垂直平移量，r 为缩放系数。

如图 2-6 所示，由于机械的偏差，由双路图像传感器输出的两幅图像之间可能存在一定的夹角 θ，如果 θ 的值过大就会造成图像拼接的失败。这时就需要根据可重构处理器反馈回的仿射矩阵调整摄像头所在的云台，使图像传感器的夹角控制在一定范围内。

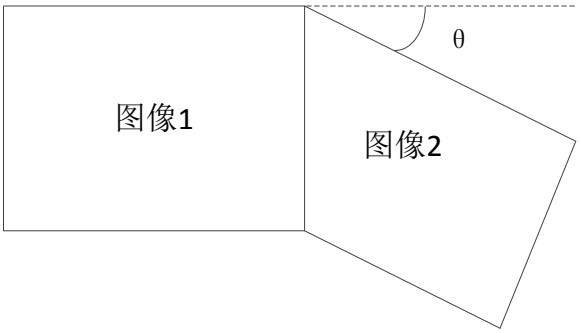


图 2-6 两幅图像存在夹角

根据可重构处理器返回的仿射矩阵，可以计算出两幅图像之间的夹角。由仿射矩阵的组成可以看出，通过计算 H(1,1)与 H(1,2)两个元素的比值，就可以计算出两幅图像夹角的正切值 tanθ。当两幅图像的夹角 θ>T 时，T 为预先设置的阈值，就需要对摄像头进行机械调整。假设两个摄像头之间的距离为 L，当前的夹角为 θ，则需要对摄像头进行调整的幅度为 L×tanθ。预处理子系统将这个值通过 I2C 总线发送给单片机，单片机根据接收到的调整信息，通过 RS485 协议控制云台上的步进电机，使两个图像传感器的夹角缩小到可接受的范围内，从而能完成了预处理子系统对双路图像传感器的反馈控制。

第三章 预处理子系统各主要模块设计

本章将详细介绍视觉信息预处理子系统中图像传感器配置及图像采集模块、滤波降噪模块、格式转换模块和DDR读写控制等模块的具体设计方法。

3.1 图像传感器配置模块设计

3.1.1 SCCB 协议

SCCB协议是OmniVision公司提出的专门用于OV系列图像传感器进行功能配置的一种总线规范，OV图像传感器芯片系列内置SCCB slave模块。SCCB总线是一种3线结构的串行总线，其主机和从机的连接方式如图3-1所示。

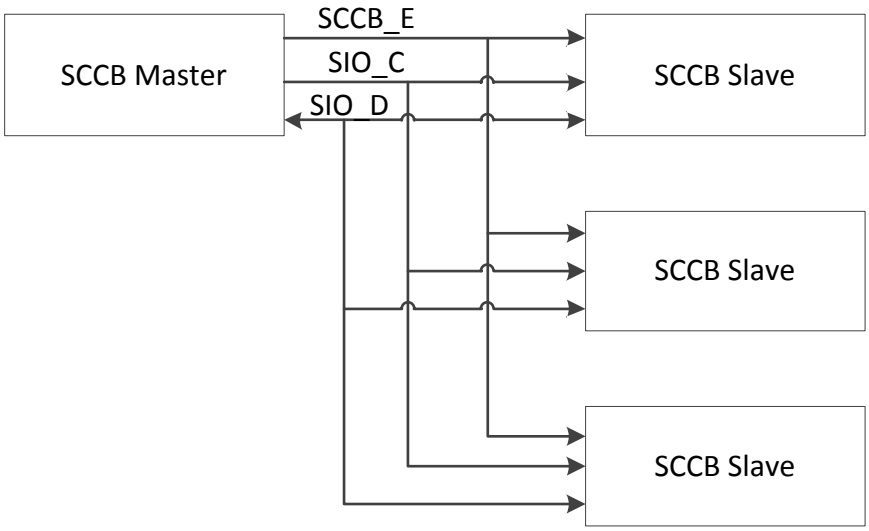


图3-1 SCCB总线3线连接结构

其中SCCB_E是使能信号，低电平有效，在多个图像传感器时，选择对哪一个传感器进行配置。在对单个传感器进行配置或者多个传感器配置相同时，此信号可省去，变成一种2线结构。本论文采用2线结构的SCCB总线对OV2640芯片进行配置。SIO_C是串行时钟线，单向信号。SIO_D是串行数据线，双向信号。SCCB的总线协议如下：

数据开始传输标志：当SIO_C信号为高电平时，SIO_D信号由高电平变为低电平。

数据停止传输标志：当SIO_C信号为高电平时，SIO_D信号由低电平变为高电平。

数据的传输结构：SCCB总线的数据传输格式如图3-2所示。协议规定，每次传输最多有3个阶段（phase）。每个阶段有9bits，其中前8bis是传输的数据，第9bit是非关心位（Don’t care bit）或者NA bit。

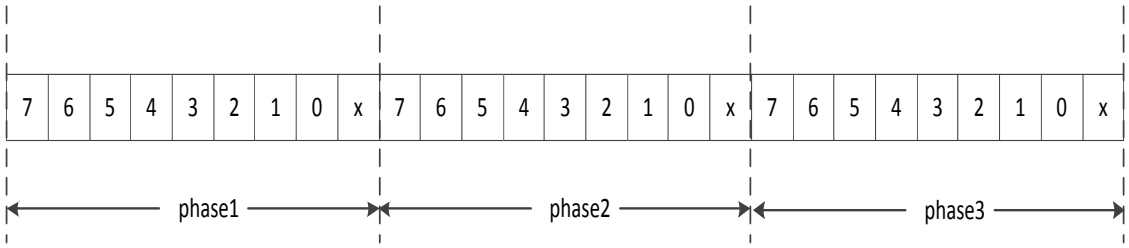


图3-2 SCCB总线数据传输结构

写数据传输过程：通过一次完整的写传输过程，SCCB主机可以将一个字节的数据配置到图像传感器芯片的指定寄存器当中。如图3-3所示，写传输过程包括三个阶段（phase）：ID address，即指定的从机地址，一般固化在图像传感器芯片中，本论文使用的OV2640芯片写传输过程的ID address为60；Sub-address是图像传感器中指定的寄存器地址；Write Data是需要向指定寄存器内配置的数据。每个阶段的第9bit均为Don’t care bit。

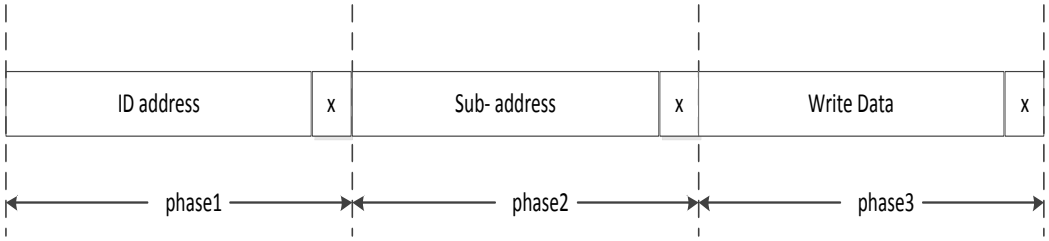


图3-3 3-phase SCCB写数据传输过程

读数据传输过程：读传输过程必须在3-phase 写传输或者2-phase写传输之后，因为单独的读传输过程不能确定寄存器地址，即Sub-address。2-phase写传输过程相当于3-phase写传输过程只有前两个阶段。读传输过程如图3-4所示，其中ID address含义与写传输过程中相同，本论文使用的OV2640芯片写传输过程的ID address为61，Read Data为指定的寄存器中的8位数据。NA bit时主机需要将数据线置为高电平。

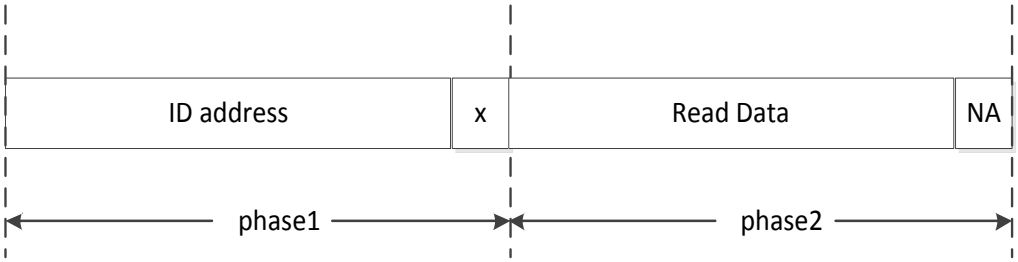


图3-4 SCCB读传输过程

3.1.2 SCCB master 设计

表3-1 SCCB master模块的接口信号

Name	Direction	Description
Clk	In	系统时钟
Rst	In	复位
WR	In	写命令标志位
RD	In	读命令标志位
Din[7:0]	In	输入数据，即待发送数据
Addr[6:0]	In	从设备的地址
Reg_addr[7:0]	In	寄存器地址
Dout[7:0]	Out	输出数据，即接收道德数据
SIO_C	Out	SCCB时钟线
SIO_D	Inout	SCCB数据线

表3-1中列出了SCCB总线主机模块的接口信号。其中系统时钟采用PLB总线时钟。复位信号、读写命令、图像传感器的设备地址、需要配置的寄存器地址及待发送的数据均由Microblaze处理器通过PLB总线给出。

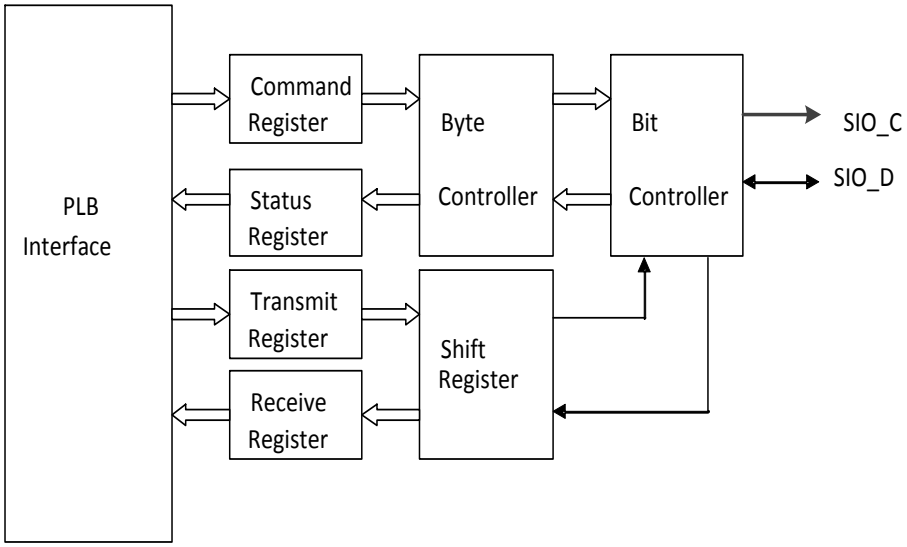


图3-5 SCCB主机结构图

图3-5显示了SCCB总线主机结构设计图。其中Command Register表示读写命令寄存器；Status Register为状态寄存器；Transmit Register为发送数据寄存器；Receive Register为接收数据寄存器；Byte Controller模块负责以字节为单位的传输控制；Bit Controller用来产生开始信号、结束信号以及每一位的传输信号；Shift Register负责完成串并数据转换。

（1）Byte Controller

Byte Controller模块主要用来控制Bit Controller模块，使其按需要产生传输起始标志、传输结束标志、单比特数据以及Don’t care bit或者NA bit等。Byte Controller模块根据状态寄存器的值来产生不同的控制流。SCCB主机的状态转换图如图3-6所示。

复位之后主机处于idle状态，当Microblaze处理器通过PLB总线发出读写命令之后，首先产生起始信号，如果是写命令，则进入3-phase写状态，按照协议规定的3-phase帧结构，依次产生ID address、Sub-address以及相应的写数据信号；如果是读命令，则首先进入2-phase写过程，产生相应的ID address和Sub-address，之后再进入2-phase读过程，读取相应寄存器的值，最后产生结束传输标志，完成一次完整的通信过程。

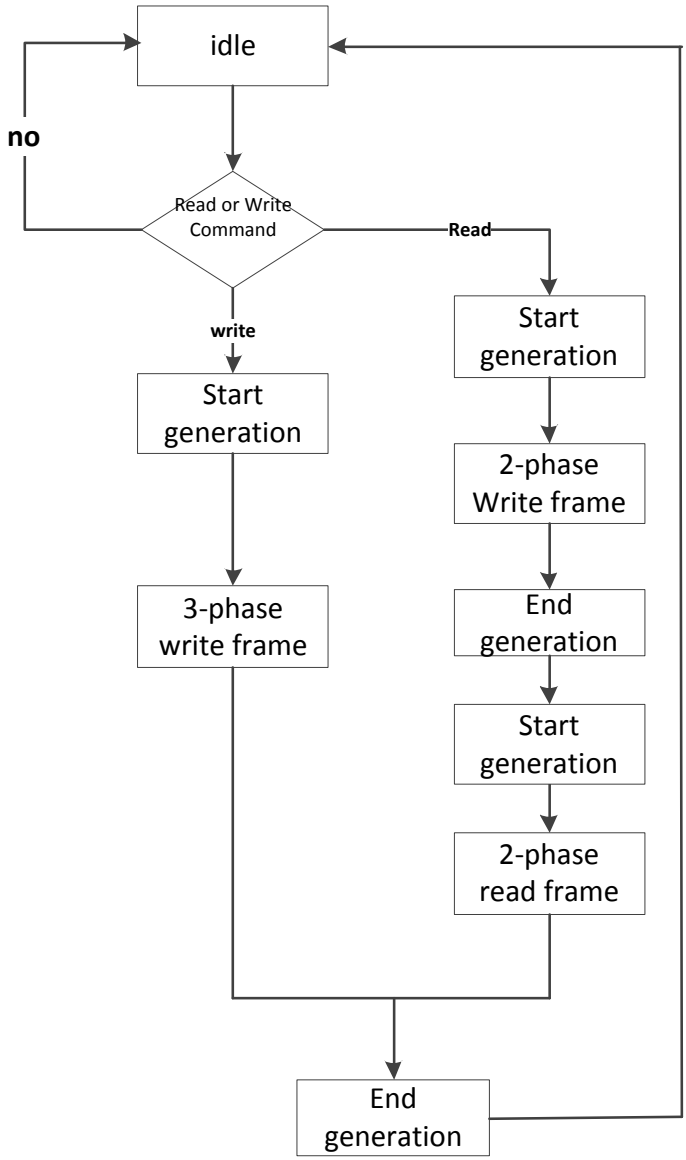


图3-6 SCCB主机状态转换图

（2）Bit Controller

Bit Controller模块用来控制SIO_C和SIO_D两个信号的具体行为，使其产生起始信号、结束信号、数据读写传输所需要的波形条件。如图3-7所示，可以将每1比特的产生分为A、B、C、D 4个过程，通过控制每个过程中SIO_C信号和SIO_D信号的高低，产生开始、结束以及读写传输的条件。例如当需要产生开始信号时，则A状态时将SIO_C和SIO_D信号同时置为高电平；B状态时两个信号均保持不变；C状态时SIO_D信号置为低电平，SIO_C信号保持不变；D状态时将SIO_C信号置为低电平，SIO_D信号则保持不变。这样就产生了起始信号所需要的波形。

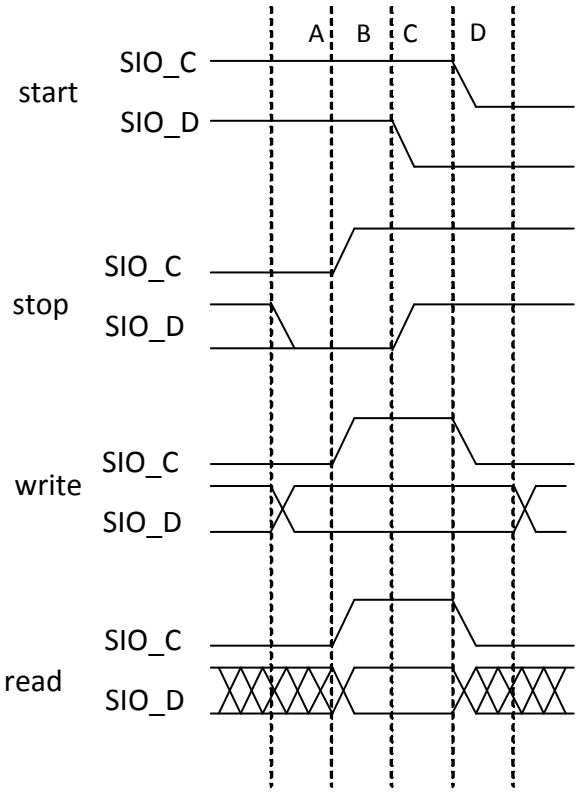


图 3-7 Bit Controller 工作原理示意图

3.2 制式转换电路设计

OV2640图像传感器芯片输出的图像数据格式为YUV422制式，而数字图像处理中经常会用到24-bit RGB制式，因此本论文在预处理子系统中设计了YUV格式转RGB格式的转换电路。YUV格式转换为RGB格式所用的变换公式为：

$$\begin{cases} R = 1.0Y + 0 + 1.402(V - 128) \\ G = 1.0Y - 0.34413(U - 128) - 0.71414(V - 128) \\ B = 1.0Y + 1.772(U - 128) + 0 \end{cases} \quad (3\text{-}1)$$

实际电路设计中，制式转换公式中的乘法运算通过移位和加法运算实现。由于输入的YUV图像数据位宽为8bits，而转换公式中涉及小数运算，因此首先将YUV数据扩展为16bits位宽，即将YUV数据左移8位，其中高8位作为整数部分，低8位作为小数部分。以公式中的1.402×V为例，1.402转化为二进制数等于1.01100111，也就是1 + 1/4 +1/8 + 1/64 +1/128 + 1/256。由此得出1.402×V的计算电路如图3-8所示，将位宽扩展后的V数据分别右移2位，3位，6位，7位，8位，

然后将移位后的数据相加，最后取计算结果的高8位作为输出。

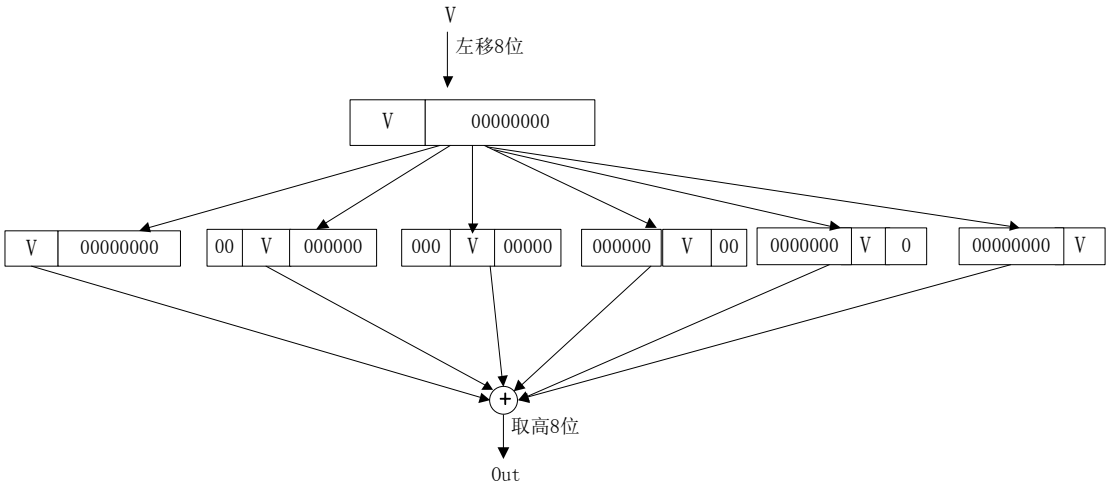


图3-8 制式转换电路示意图

由于YUV422是一种压缩格式，每一个像素点只有16位数据，而RGB格式每一个像素则有24位，因此要进行格式转换之前先要将YUV422格式转换为24位的YUV444格式，转换原理如图3-9所示。每两个YUV422的像素数据转换为两个YUV44格式的像素数据。

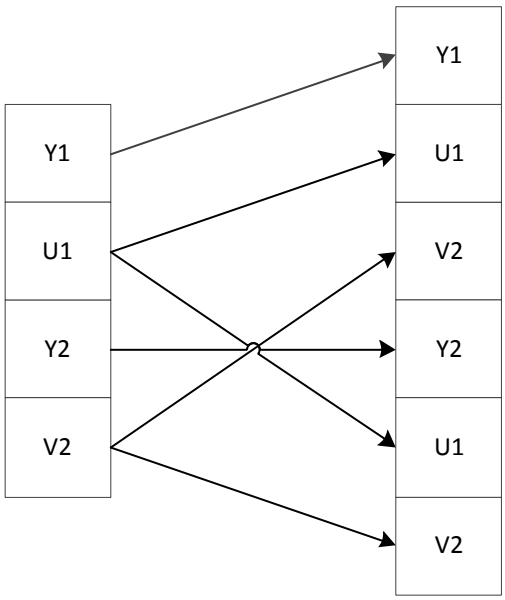


图 3-9 YUV422 转 YUV444 原理

3.3 滤波降噪单元设计

3.3.1 中值滤波模块设计

数字图像在采集、传输和存储过程中常常会引入脉冲噪声。这些噪声不仅会影响图像质量，更会严重影响角点检测、图像拼接、特征识别等高级图像算法的处理效果。因此必须在视觉信息预处理子系统中进行抑制噪声的设计。

中值滤波算法能够有效抑制脉冲噪声，同时保护图像尖锐的边缘，因此被广泛应用在图像降噪处理中。中值滤波是一种基于统计排序的非线性空间域滤波器，算法基本思想是选择一定形式的窗口（通常为3×3,5×5,或者7×7的方形窗口），对滤波窗口中的像素按照其灰度值的大小进行排序，用排序后的像素中值代替窗口中心像素的灰度值。标准中值滤波器定义为

$$Y = Med\{X_i : i \in W\} \qquad (3\text{-}2)$$

其中Med表示取中值运算，W表示截取图像的窗口，X_i表示窗口内像素点的值。

如图3-10，以3×3滤波窗口为例，完成一次中值滤波运算需要对9个像素数据进行排序。本文设计了一种流水线的排序方式，可以实现1个时钟周期内完成中值滤波过程。

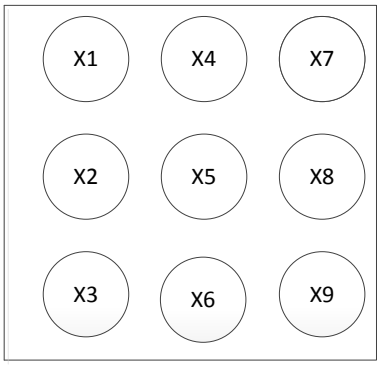


图3-10 3×3滤波窗口

按照传统的排序算法，要完成9个数据的排序，要进行36次的比较。但是如果只是要求算出中值，可以按照图3-11的方式进行排序。即对一个3×3的阵列，首先将所有元素按列方向从大到小排序，然后按照行方向从大到小排序，最后按对角线方向从大到小排序，最终阵列中心位置的元素即为9个数的中值。采用这种简化的求中值的算法，一方面将总的比较次数减少了15次，同时这种算法还有利于流水线的电路设计。

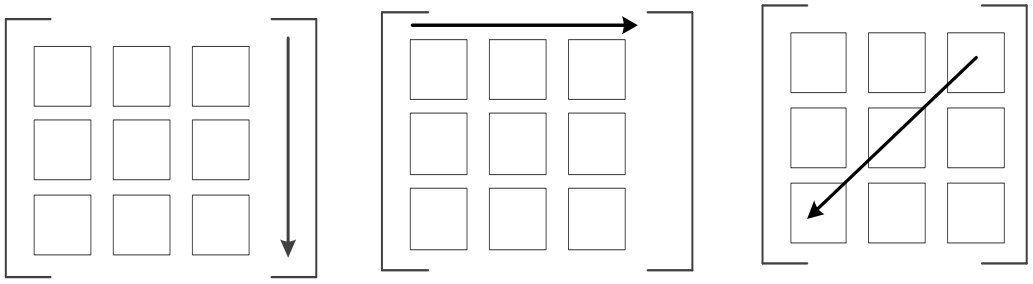


图3-11 简化的求中值算法

根据图3-11所示的排序算法，首先设计了一个对三个数据进行排序的单元模块，然后用这个单元模块搭成如图3-12所示的电路，即完成了中值滤波电路的设计。图中cmp模块即为三数据排序模块，每个时钟周期从前面的行缓存数据中输出三个新的图像数据，进入cmp1。cmp1对这三个数据进行排序，结果送入移位寄存器组X1~X9中，这9个寄存器每3个为一组，且满足x1 > x2 > x3, x4 > x5 > x6, x7 >x8 >x9。然后将x1, x4,x7送入cmp2模块，将x2, x5, x8送入cmp3模块，将x3, x6, x9送入cmp4模块，最后将cmp2输出的最小值，cmp3输出的中间值，cmp4输出的最大值送入cmp5模块，cmp5模块输出的中间值即为中值滤波电路的最终输出结果。采用这样的流水线方式，可以实现每个时钟周期完成一个窗口的滤波。

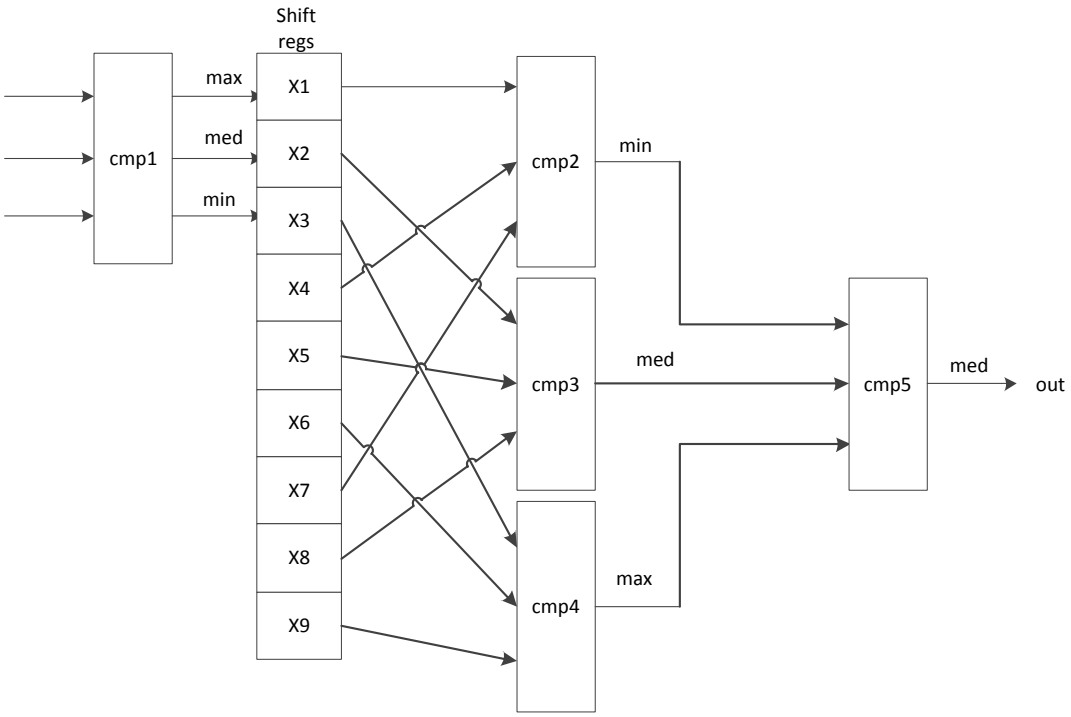


图 3-12 流水线结构的中值滤波电路

3.3.2 中值滤波在可重构处理器上的映射

中值滤波算法需要完成对 n 个元素的排序，冒泡排序法不适合并行实现，因此采用奇偶排序法在可重构阵列上进行算法映射。

奇偶排序算法在 n 个阶段内对 n 个元素(n 是偶数)排序，算法在两个阶段之间交替执行，称为奇数阶段和偶数阶段。在奇数阶段需要 n/2 次比较-交换操作，在偶数阶段需要 n/2 - 1 次比较-交换操作。在奇数阶段，下标为奇数的元素与它们右边相邻的元素比较，如果它们未按照顺序排序，就交换它们；在偶数阶段，下标为偶数的元素与它们右边相邻的元素比较，如果它们未按照顺序排序，就交换它们。经过 n 个阶段的奇偶转换，序列就完成排序。

在算法的每个阶段，每对元素上的比较-交换操作能够同时执行，从而可以将奇偶转换排序并行化。

假设 PE11 为 4×4 可重构阵列上的第一行第一列功能单元，以此类推，则 PE44 是第四行第四列单元。a1 到 a9 为待排序的 9 个图像像素，由于奇偶排序是针对偶数个元素，则先将 a9 不计入排序序列中。将 a1 到 a8 排序之后，取排在第四和第五的两个值与 a9 比较，取中间值即是中值滤波后的结果。

开始时为奇数阶段，则将 a1 与 a2 的比较(a1>=a2)、a3 与 a4 的比较(a3>=a4)、a5 与 a6 的比较(a5>=a6)、a7 与 a8 的比较(a7>=a8)分别映射到 PE11、PE12、PE13、PE14 上，然后将结果（0 或 1）存储到片内存储器中。PE21 与 PE22 分别读取 a1 与 a2 的比较结果，并进行三目运算 OUT1=C?(a1:a2)、OUT2=C?(a2:a1)，则 OUT1 输出 a1 与 a2 中的较大值，OUT2 输出 a1 与 a2 中的较小值，并存储到片内存储器中，后三对的比较同理。记排序后的序列为 b1-b8。

第二阶段为偶数阶段，将 b2 与 b3 的比较(b2>=b3)、b4 与 b5 的比较(b4>=b5)、b6 与 b7 的比较(b6>=b7)分别映射到 PE11、PE12、PE13 上。比较-交换操作与奇数阶段同理。记排序后的序列为 b1，d2~d7，b8。

后面的阶段为奇数阶段与偶数阶段的轮回操作。在 8 个阶段后即可完成 8 个数的排序，这里是按照从大到小的顺序。取出排列在第四和第五位置上的元素，这里记为 j4、j5(j4>=j5)，先将 j4 与 a9 比较，取其中较小值，然后将其与 j5 比较，取其中较大值，则此值即是中值滤波后的结果。

具体映射如图 3-13 所示，其中 PE41-PE44 均未使用。

	PE11	PE12	PE13	PE14	PE21	PE22	PE23	PE24	PE31	PE32	PE33	PE34
1	a1>=a2	a3>=a4	a5>=a6	a7>=a8								
2					1c12?(a1:a2)	1c12?(a2:a1)	1c34?(a3:a4)	1c34?(a4:a3)	1c56?(a5:a6)	1c56?(a6:a5)	1c78?(a7:a8)	1c78?(a8:a7)
3	b2>=b3	b4>=b5	b6>=b7									
4					2c23?(b2:b3)	2c23?(b3:b2)	2c45?(b4:b5)	2c45?(b5:b4)	2c67?(b6:b7)	2c67?(b7:b6)		
5	b1>=d2	d3>=d4	d5>=d6	d7>=b8								
6					3c12?(b1:d2)	3c12?(d2:b1)	3c34?(d3:d4)	3c34?(d4:d3)	3c56?(d5:d6)	3c56?(d6:d5)	3c78?(d7:b8)	3c78?(b8:d7)
7	e2>=e3	e4>=e5	e6>=e7									
8					4c23?(e2:e3)	4c23?(e3:e2)	4c45?(e4:e5)	4c45?(e5:e4)	4c67?(e6:e7)	4c67?(e7:e6)		
9	e1>=f2	f3>=f4	f5>=f6	f7>=e8								
10					5c12?(e1:f2)	5c12?(f2:e1)	5c34?(f3:f4)	5c34?(f4:f3)	5c56?(f5:f6)	5c56?(f6:f5)	5c78?(f7:e8)	5c78?(e8:f7)
11	g2>=g3	g4>=g5	g6>=g7									
12					6c23?(g2:g3)	6c23?(g3:g2)	6c45?(g4:g5)	6c45?(g5:g4)	6c67?(g6:g7)	6c67?(g7:g6)		
13	g1>=h2	h3>=h4	h5>=h6	h7>=g8								
14					7c12?(g1:h2)	7c12?(h2:g1)	7c34?(h3:h4)	7c34?(h4:h3)	7c56?(h5:h6)	7c56?(h6:h5)	7c78?(h7:g8)	7c78?(g8:h7)
15	i2>=i3	i4>=i5	i6>=i7									
16					8c23?(i2:i3)	8c23?(i3:i2)	8c45?(i4:i5)	8c45?(i5:i4)	8c67?(i6:i7)	8c67?(i7:i6)		
17	a9>=j4											
18					c1?(a9:j4)	c1?(j4:a9)						
19		k2>=j5										
20					c2?(k2:j5)	c2?(j5:k2)						
21												
22												

图 3-13 中值滤波算法在可重构处理器上的映射方案

3.4 DDR 读写控制电路设计

由于后续处理的需要，经过降噪处理后的图像要缓存两帧。本设计中采用DDR 存储器进行帧缓存处理。本部分将介绍预处理子系统中 DDR 读写控制电路的设计。

3.4.1 DDR 工作原理

DDR SDRAM（Double Data Rate SDRAM，双倍数据流SDRAM）是在SDR（Single Data Rate）SDRAM的基础上改进而来，其基本结构和工作原理有很多类似的地方。SDRAM的基本操作命令是通过CS#, RAS#, CAS#，WE#，DQM，DQs信号和地址线的组合来完成的。表3-2列出了SDRAM的基本操作命令。

表3-2 SDRAM的基本操作命令（H表示高电平，L表示低电平，X表示高低电平没有影响）

命令名	CS#	RAS#	CAS#	WE#	DQM	地址线	DQs
命令禁止	H	X	X	X	X	X	X
无操作	L	H	H	H	X	X	X
有效/活动（使指定行有效）	L	L	H	H	X	Bank/行	X
读取	L	H	L	H	L/H	Bank/列	X
写入	L	H	L	L	L/H	Bank/列	有效
突发传输终止	L	H	H	L	X	X	活动
预充电	L	L	H	L	X	相应编码	X
自动刷新或者自刷新	L	L	L	H	X	X	X

模式寄存器设置	L	L	L	L	L	操作码	X
写允许/输出允许	-	-	-	-	H	-	有效
写禁止/输出屏蔽	-	-	-	-	H	-	屏蔽

SDRAM在接通电源以后不能立即开始正常工作，必须完成初始化过程。SDRAM的初始化过程如图3-14所示。其中扩展模式寄存器设置只有DDR会进行，SDR初始化过程中不存在。 模式寄存器设置主要完成突发长度，突发传输方式，CAS潜伏期等设置。

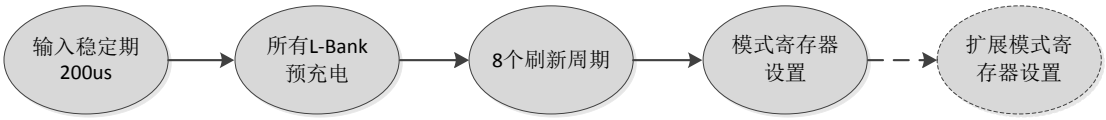


图3-14 SDRAM初始化过程

初始化完成以后，SDRAM 开始正常工作。由于 SDRAM 的行列地址共用一组地址线，所以要对 SDRAM 中的阵列进行操作，必须进行独立的行列寻址。首先要确定行地址，使之处于活动状态，然后再确定列。其中片选和 L-Bank 的定址可以和行有效同时进行。列寻址和读写命令是同事发出的，但是行有效信号之后不能立即发出列寻址信号，二者之间必须有一个间隔，这个间隔被称为 tRCD。如果是读操作，在列寻址（CAS）之后，仍需要经过一定的时间才能有有效数据输出，从 CAS 与读命令发出到第一组数据输出之间的时间成为 CL（CAS Latency，CAS 潜伏期），这个时间可以在模式寄存器中进行设置。CAS 潜伏期只出现在读操作中，如果是写入的操作，数据可以和 CAS 同时发送，而不需要有任何的延迟。不过由于数据并不能即时的写入存储电容，数据的写入需要一定的时钟周期，因此为了保证数据写入的可靠性，在一次写操作完成之后，还要留出一定的写入时间（tWR，Write Recovery Time），之后才能进行下一个读写操作。SDRAM 支持突发读写操作，即读写命令发出后可以连续读写若干个数据，其中 SDR 支持的突发长度有 1、2、4、8 和全页，DDR 支持的突发长度包括 2、4、8，DDR2 则只支持 4、8 的突发长度。当 SDRAM 关闭现有工作行，打开新的工作行的时候需要进行预充电（Precharge）。发出预充电命令之后，要经过一段时间才能发送行有效命令打开新的工作行，这个时间被称为预充电有效周期。另外，SDRAM 还要不断的进行刷新操作，以保留那些很久没有经历重写的存储体中的数据。由于存储单元中电容的数据有效期上限是 64ms，所以每一行的刷新周期是 64ms。

3.4.2 利用 MPMC 对 DDR 进行读写控制

由于DDR操作时序复杂，要不断地进行刷新操作，刷新期间不能进行任何读

写操作，并且还要考虑多个输入输出模块的仲裁，因此本论文使用Xilinx公司提供的MPMC（Multi-Port Memory Controller）实现对DDR存储器的读写控制。

MPMC是一种可参数化配置的存储控制器IP软核，支持SDRAM/DDR /DDR2多种存储器。MPMC提供了8个控制端口，每个端口都可以配置成PLBv4.6，NPI，SDMA，VFBC等多种形式，其中PLBv4.6接口通常用来使DDR通过PLB总线与Microblaze处理器进行通信，如果采用这种接口方案，那么图像数据需要通过PLB总线进入Microblaze处理器，然后处理器再将数据通过PLB总线写入DDR中，这一方面增加了总线上的协议开销延迟以及处理器内读写指令的延迟，还可能造成总线拥挤；SDMA（Soft Direct Memory Access）是一种软件直接读写内存的方式；VFBC（Video Frame Buffer Controller）接口是Xilinx提供的专门用于视频数据存储的读写控制方案，但是VFBC只适用于RGB格式的图像存取，并且VFBC规定的数据帧格式使其不能自由的控制数据的存取地址；NPI（Native Port Interface）是MPMC最底层的控制接口，可以使设计人员比较自由地控制数据的读写地址以及数据传输时的突发长度，因此更符合本系统的设计需求。

NPI接口支持32比特和64比特两种数据位宽，同时支持word（只针对32位NPI）、double-word（只针对64位NPI）、4-word、8-word、16-word、32-word、64-word等多种突发长度的突发传输。本设计采用32 bits位宽，突发长度可配置的方式进行DDR读写控制。

以8-word突发传输为例，图3-15显示了NPI 8-word写传输的时序。其中MPMC_Clk0是存储器时钟，InitDone 信号为高电平表示DDR初始化完成同时MPMC内部FIFO可以使用。AddrReq为地址请求信号，地址请求发送以后，在同一个时钟内MPMC便会发送一个反馈信号AddrReq。Addr为地址信号，其中Addr[11:3]表示列地址信号，Addr[24:12]表示行地址信号，Addr[26:25]表示Bank地址信号。RNW信号表示读写控制，高电平表示读请求，低电平表示写请求。Size信号的值表示不同的突发长度。WrFIFO_Empty和WrFIFO_AlmostFull信号分别表示MPMC内部FIFO的空、满标志，当WrFIFO_AlmostFull信号为高时不能继续进行写数据操作。WrFIFO_BE的值表示数据线上有效的字节，例如WrFIFO_BE=0x3表示WrFIFO_Data只有低16 bits有效。WrFIFO_Push为高电平时表示与之相对应的写数据有效。写数据可以分别在地址请求之前、之时、之后产生。

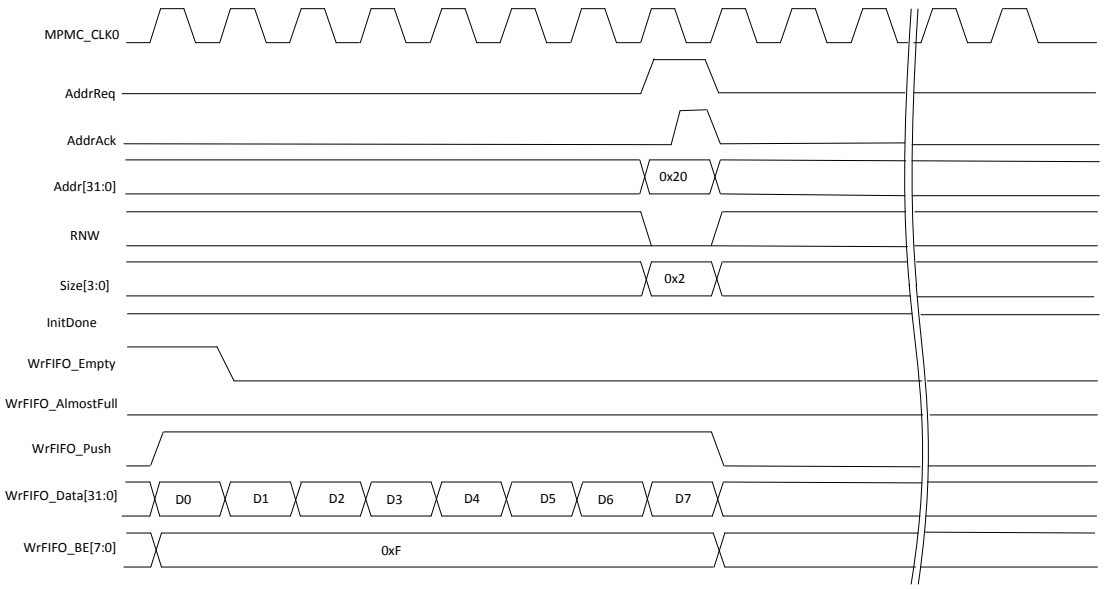


图3-15 32-bit NPI 8-word突发写传输时序

32-bit NPI 8-word读操作的时序如图3-16所示。其中的AddrReq、AddrAck、Addr、RNW、Size信号与写操作中含义相同。与写操作不同的是，读请求只能在读数据之前发出，并且一次读请求后需要23个时钟周期才会有有效读数据产生。

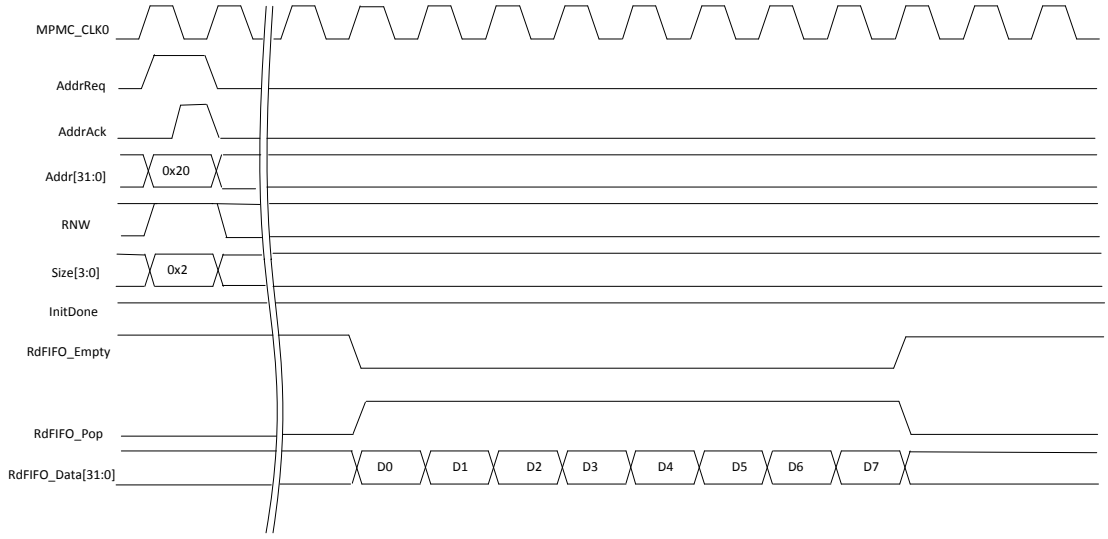


图 3-15 32-bit NPI 8-word 突发读传输时序

3.5 数据通信接口方案设计

一方面从图像传感器输出的图像数据经过预处理子系统后需要发送给可重构处理器进行高级视觉信息处理算法的运算，然后可重构处理器的运算结果反馈

回来。另一方面预处理子系统需要将DDR中缓存的图像数据发送给后处理系统并最终在显示器上显示。

视觉信息预处理子系统与可重构处理器的通信接口及相关描述如表3-3所示，与后处理子系统的接口如表3-4所示。

表3-3与可重构处理器系统的接口

Name	Direction	Description
RP_INIT_DONE	I	可重构初始化结束
WR_RP	O	写可重构
RD_RP	O	读可重构
Instruction[3:0]	O	发送给可重构的指令
RP_Data_Ready	I	可重构运算结束
Data2RP[31:0]	I/O	数据接口
Fifo2RP_clk	I	Fifo 时钟
Fifo2RP_Rd_en	I	Fifo 读使能
Fifo2RP_Wr_en	I	Fifo 写使能
Full_RP	O	Fifo 满标志
Empty_RP	O	Fifo 空标志

表3-4 与后处理子系统的通信接口

Name	Direction	Description
DIS_INIT_DONE	I	后处理初始化完成
WR_Dis	O	写后处理命令
Data2Dis[31:0]	I/O	数据接口
Fifo2Dis_clk	I	Fifo 时钟
Fifo2Dis_Rd_en	I	Fifo 读使能
Empty_Dis	O	Fifo 空标志

（1）与可重构处理器通信流程

上电复位后各子系统各自进行初始化。可重构初始化结束后将RP_INIT_DONE信号置高，预处理子系统接收到该信号后，向可重构发送写命令，同时根据外部中断指令配置Instruction寄存器。可重构接收到写命令之后，一方

面读取Instruction，另一方面将Fifo2RP_Rd_en置高，然后开始读取数据。可重构运算结束后，置高RP_Data_Ready信号，前处理接受到该信号后向可重构发送读指令，RD_RP置高。然后可重构将Fifo2RP_Wr_en置高，前处理系统开始从可重构读数据。Data2RP[31:0]数据结构如下：

- 第0个字：[31:30] 保留
  [29:19] 图像x方向尺寸
  [18:8] 图像y方向尺寸
  [7:0] 保留
- 第1~3个字：保留
- 第4个字及以后：有效数据

（2）与后处理系统通信流程

预处理子系统与后处理子系统属于单向通信，即预处理子系统只会向后处理系统写数据，而不会读数据。预处理系统准备好数据后向后处理系统发送写命令，之后，后处理系统置高Fifo2Dis_Rd_en信号，开始接收数据。Data2Dis[31:0]数据结构如下：

- 第0个字：[31:30] 01=图像数据；10=仿射矩阵数据
  [29:19] 图像x方向尺寸（仿射矩阵则无效）
  [18:8] 图像y方向尺寸（仿射矩阵则无效）
  [7:0] 保留
- 第1~3个字：保留
- 第4个字开始：有效数据

第四章 中值滤波单元的改进设计

4.1 矢量中值滤波

标量中值滤波（Scalar Median Filter，SMF）可以有效地去除灰度图像中的脉冲噪声，但是如果将标量中值滤波应用到彩色图像降噪处理当中，处理效果则不是很理想。这是因为与单信号通道的灰度图像不同，彩色图像是一种含有 R, G, B 三种分量的多通道信号。如果采用标量中值滤波方法分别处理彩色图像的 R, G, B 三种个分量，然后将处理后的图像重新组合作为最终的输出图像，这种方法称为 Vector marginal median Filter (VMMF)[18]。VMMF 虽然可以一定程度上降低彩色图像中的噪声，但是却有一个很严重的缺陷，那就是 VMMF 对 R, G, B 分量处理时是相互独立的，这样处理后的图像中会出现原始图像中并不存在的 R, G，B 组合，也就是说会产生原来并不存在的颜色，造成颜色失真。

矢量滤波器（Vector Filter，VF）很好地解决了 SMF 处理彩色图像时的不足。矢量滤波器将彩色图像中每一个像素的 R, G, B 分量看作一个矢量，在运算中作为整体进行处理。常见的矢量滤波器包括矢量中值滤波器（Vector Median Filter，VMF）[14]，矢量方向滤波器（Basic Vector Directional Filter， BVDF）[19]以及矢量方向距离滤波器（Directional-distance Filter， DDF）[20]等。矢量中值滤波器是其中最常见的一种滤波方法。

矢量中值滤波也是一种基于统计排序的非线性滤波器。矢量中值滤波的滤波过程如下：

首先取一个 n×n 的滤波窗口 W，n 可以取 3, 5, 7, 9 等。令 $X_i=(R_i, G_i, B_i)$（i=1，2, 3…N）表示窗口内每个像素的值。计算滤波窗口中任意两个像素之间的 L_2 范数（即欧氏距离）ρ_{ij}：

$$\begin{aligned}\rho_{ij}(X_i, X_j) &= \left\| X_i - X_j \right\|_2 \\ &= [(R_i - R_j)^2 + (G_i - G_j)^2 + (B_i - B_j)^2]^{1/2} \quad (\mathrm{i, j} = 1, 2, 3, \ldots, \mathrm{N})\end{aligned} \tag{4-1}$$

然后计算每个像素点与窗口内其他像素点的欧氏距离之和 D_i

$$D_i = \sum_{j=1}^{N} \rho_{ij}(X_i, X_j) \quad (\mathrm{i} \neq \mathrm{j}) \tag{4-2}$$

之后对 D_i 进行排序，假设排序结果为 D(1) ≤D(2) ≤D(3) ≤…≤D(N)，对应的像素值为 X(1)，X(2)，X(3)，…, X(N)，则用 X(1)作为输出代替窗口中心点的像素，完成一个像素的滤波。将滤波窗口平移滑动，直至完成整幅图像的滤波。

除了以上使用的 L_2 范数外，有时候还会用 L_1 范数。整个矢量中值滤波过程可以表示为：

$$X_{VMF} = \arg \min_{X_i \in W} \sum_{j=1}^{N} \left\| X_i - X_j \right\|_p \tag{4-3}$$

4.2 矢量中值滤波算法的改进

对于高噪声密度的图像，标准矢量中值滤波的滤波效果不是很理想。随着滤波窗口的增大，矢量中值滤波的降噪能力会随之增强，但是较大的滤波窗口往往会损失更多的图像细节，使图像变得模糊。为了提高矢量中值滤波的滤波效果，人们提出了很多改进方案。

4.2.1 基于噪声检测的矢量中值滤波

标准矢量中值滤波对彩色图像中所有的像素点都会进行滤波处理，然而图像中大部分的像素都不是噪声点，这些像素都不需要进行滤波处理。于是便出现了基于噪声检测的开关矢量中值滤波（switching vector median filter，SVMF）。SVMF 只对检测到的噪声点进行滤波处理，而非噪声点则直接作为窗口的输出，其计算公式如下:

$$out_{(N+1)/2} = \begin{cases} X_{(N+1)/2}, & X_{(N+1)/2} \text{ 是噪声点} \\ X_{VMF} \quad , & X_{(N+1)/2} \text{ 不是噪声点} \end{cases} \tag{4-4}$$

其中 N 为滤波窗口内像素点的个数。开关矢量中值滤波的关键在于正确的噪声检测，既要尽可能的将所有噪声点检测出来，也要防止对过多的非噪声点进行滤波，造成图像模糊。

文献[21]中提出了一种自适应滤波器（Adaptive Vector Median Filter，AVMF）。与 VMF 相同，AVMF 也是首先计算出滤波窗口中每个像素点与其他所有像素的矢量距离之和，然后对矢量和进行排序，假设排序结果为 $D^{(1)} \le D^{(2)} \le D^{(3)} \le \ldots \le D^{(N)}$，对应的像素值为 $X^{(1)}$，$X^{(2)}$，$X^{(3)}$，…，$X^{(N)}$。不同的是 AVMF 对统计

排序结果做了进一步处理：取排好序的像素序列的前 r 个像素，计算其平均值，然后计算中心像素与此平均值的矢量距离 Val。Val 的数学表达式如下：

$$Val = \left\| X_{(N+1)/2} - \frac{1}{r}\sum_{i=1}^{r} X^{(i)} \right\|_{\gamma} \qquad (4\text{-}5)$$

其中，$\| \|_{\gamma}$ 表示 L_1 范数或者 L_2 范数。如果 Val ≥ Tol，Tol 为设定的阈值，那么就认为中心像素点 $X_{(N+1)/2}$ 是疑似噪声点，用矢量中值滤波的输出将其替换；如果 Val < Tol，则认为中心像素点不是噪声，直接作为滤波结果输出。由于像素序列 $X^{(1)}$，$X^{(2)}$，$X^{(3)}$，…, $X^{(r)}$包含了与原始像素最相似的 r 个像素点，因此它提供了一个很好的判断中心像素是否为噪声的判断标准。

AVMF 一定程度上给你改善了标准矢量中值滤波的处理效果，但是利用其提供的方法进行噪声检测时很可能产生错误，尤其是与到图像中的细节或者图像中有较细的线时。于是文献[22]中提出了一种改进的 AVMF（Modified Adaptive Vector Median Filter，MAVMF）。MAVMF 是一种两步（two-phase）的噪声检测方法。第一步与 AVMF 相同，利用 AVMF 提出的方法进行初步噪声检测。对于 AVMF 检测出的噪声点则进一步利用图 4-1 中的 4 个 Laplace 模板判断该点究竟是图像边缘还是真正的噪声点。具体方法是将输入图像与 4 个模板进行卷积操作，其结果中的最小值 z 用来进行边缘检测，如果 z≤T，则该点为图像边缘，而非噪声。

$$\begin{bmatrix} & & & & \\ & & & & \\ -1 & -1 & 4 & -1 & -1 \\ & & & & \\ & & & & \end{bmatrix} \qquad \begin{bmatrix} & & -1 & & \\ & & -1 & & \\ & & 4 & & \\ & & -1 & & \\ & & -1 & & \end{bmatrix}$$

$$\begin{bmatrix} -1 & & & & \\ & -1 & & & \\ & & 4 & & \\ & & & -1 & \\ & & & & -1 \end{bmatrix} \qquad \begin{bmatrix} & & & & -1 \\ & & & -1 & \\ & & 4 & & \\ & -1 & & & \\ -1 & & & & \end{bmatrix}$$

图 4-1 MAVMF 中的 4 个 Laplace 模板

4.2.2 加权矢量中值滤波

除了进行噪声检测之外，还可以利用加权矢量中值滤波（Weighted vector median filter， WVMF）的方法，对滤波窗口内的各个像素点赋予不同的权重，

从而能更加精确的控制滤波过程，改善滤波效果。利用加权矢量中值滤波计算窗口内各像素点之间的矢量距离的方法与标准矢量中值滤波方法一样，只是在计算矢量距离只和时，需要添加相应的权重系数，没有被噪声污染的像素或者与原始像素更相似的像素应当赋予更大的权重系数。WVMF 计算矢量距离之和的公式如下：

$$D_i = \sum_{j=1}^{N} \omega_j \left\| X_i - X_j \right\|_p \qquad \text{i=1, 2, 3, ... , N} \tag{4-6}$$

$$\begin{cases} \dfrac{\partial \omega_j}{\partial \delta(X_j, X_c)} \le 0 \\ \dfrac{\omega_c}{\omega_j} \le \alpha, \ \alpha \ge 1 \end{cases} \qquad \text{j=1,2,3, ... ,N} \tag{4-7}$$

其中 N 表示窗口大小；ω_j 表示像素 X_j 的权重；X_c 表示滤波窗口的中心像素。ω_c 表示中心像素的权重；$\delta(X_j, X_c)$表示像素 X_j 与中心像素之间的位置距离。加权矢量中值滤波的输出与矢量中值滤波相似：

$$X_{WVMF} = \arg \min_{X_i \in W} \sum_{j=1}^{N} \omega_j \left\| X_i - X_j \right\|_p \tag{4-8}$$

加权矢量中值滤波使权重系数较大的像素作为滤波结果输出的概率增大，因此能够更好的控制滤波过程。研究者提出了各种加权方法[23,24,25]，其中中心加权矢量中值滤波方法（Center Weighted Vector Median Filter, CWVMF）[23]最为常见，且滤波效果也很好。CWVMF 只对滤波窗口中心像素赋予一个大于 1 的权重系数，是一种特殊的 WVMF。CWVMF 各像素点的权重系数 ω_j 为

$$\omega_j = \begin{cases} k, & j = (N+1)/2 \\ 1, & \text{其他} \end{cases} \tag{4-9}$$

文献[26,27]指出，随着 k 的增加，CWVMF 对图像细节的保护能力越来越好，但是其对噪声的抑制能力却会减弱。当 k=1 时，CWVMF 与 VMF 相同。

4.3 细节保护型的矢量中值滤波

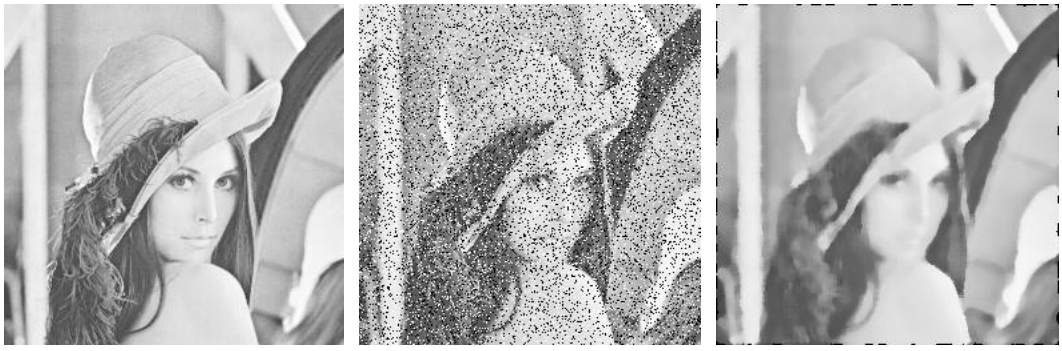
本论文提出了一种改进的矢量中值滤波算法。该算法主要分为两步：首先进

行噪声检测，本文结合 VMMF 算法和相似度函数对噪声图像进行初步的噪声检测；对于疑似噪声点，本文采用了一种基于图像边缘检测的加权矢量领中值滤波算法。

4.3.1 噪声检测

彩色图像中的像素包含 R,G,B 三个分量，每个分量都有可能被噪声污染。VMMF 算法对每一个分量分别采用标量中值滤波进行处理，如之前所述，这种方法会引入人造色彩，但是这种算法却具有很强的降噪能力。本论文采用的噪声检测技术正是利用了这种优势，将 VMMF 处理有的图像最为一幅参考图像。

首先利用 VMMF 算法对噪声图像进行处理，得到一幅参考图像 I'，这幅参考图像被认为是基本不含噪声的图像。图 4-2 显示了 VMMF 算法对 Lena 图像的红色分量处理后的结果。可以看出当噪声密度为 30%时，如果滤波窗口大小取 7×7，VMMF 处理后的图像中几乎不含噪声点。



（a） （b） (c)

图 4-2 (a)原始图像中的 R 分量；（b）含 30%椒盐噪声图像中的 R 分量；（c）滤波处理后的 R 分量图像

令 $X_i=(X_i^R,X_i^G,X_i^B)$ 表示噪声图像中的像素，$X'_i=(X'^R_i,X'^G_i,X'^B_i)$ 表示参考图像中与之相对应的像素。利用以下相似度公式计算 X_i 与 X'_i 的相似度

$$\mu_i^Q=1-\frac{\rho(X_i^Q,X'^Q_i)}{k} \qquad (4\text{-}10)$$

其中 Q = R, G, B; k 表示参考图像中最大亮度值与最小亮度值之差。ρ 代表 X_i 与 X'_i 的某种矢量距离（通常为 L_1 范数或者 L_2 范数）。如果 X_i 与 X'_i 满足以下条件

$$\left(\mu_i^R>T\right)\&\&\left(\mu_i^G>T\right)\&\&\left(\mu_i^B>T\right) \qquad (4\text{-}11)$$

则认为 X_i 与 X'_i 相似，也就认为 X_i 为非噪声点将其直接输出。否则 X_i 就是疑似噪声点，将进行下一步的滤波处理。公式（4-11）中 T 为预先设置的阈值。

4.3.2 基于边缘检测的加权矢量中值滤波

与图像其它区域相比，当处理图像边缘时更容易产生图像模糊现象。这是因为如果滤波窗口内存在图像边缘，那么位于图像边缘不同侧的像素数量很可能非常相近，这将导致计算 D_i 时，会出现许多非常接近的结果，最终是图像边缘附近变得非常模糊。因此提高图像边缘附近的滤波效果能够有效地提高政府图像的滤波效果。由加权矢量中值滤波的计算公式可以看出，提高滤波窗口内某一个像素的权重系数会增大其作为滤波输出结果的概率。为了改善 VMF 在图像边缘附近的滤波效果，更好的保护图像细节，本论文提出了一种基于图像边缘检测的加权机制

A．图像边缘检测

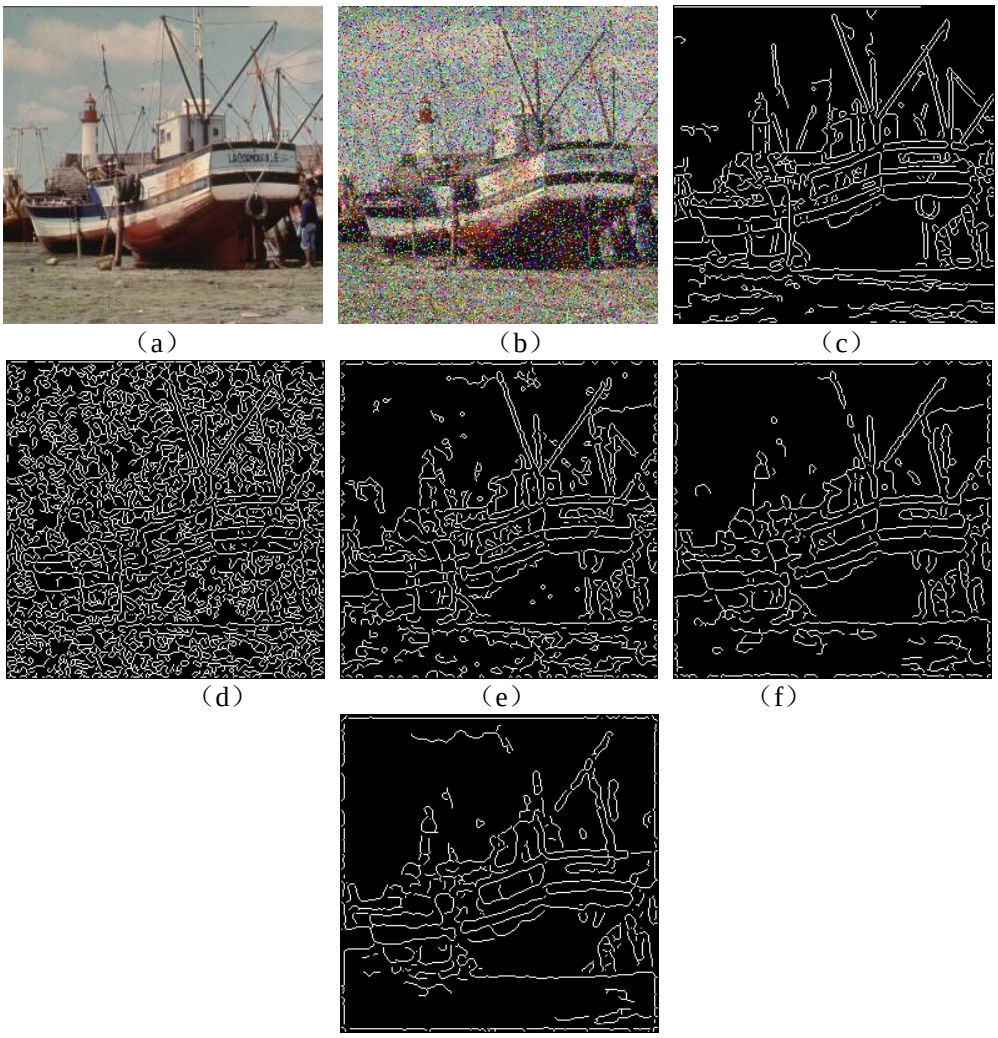
由于待处理图像中含有许多噪声，而这些噪声将严重影响图像边缘检测的效率。另一方面，图像边缘检测通常在灰度图像中进行，所以不必考虑色彩失真的问题。为提高图像边缘检测效率，本论文采取了以下步骤来提取彩色噪声图像中的边缘：

i. 将彩色图像转化为灰度图像；

ii. 利用标量中值滤波消除灰度图像中的噪声。由于随着滤波窗口的增加，中值滤波的降噪能力增强但是会损失更多的图像细。而不论是图像噪声还是图像细节都会影响到接下来的加权矢量中值滤波过程，所以必须再次做一个折中处理。实验结果表明，当噪声密度较低时（$p<0.2$）采用 3×3 的滤波窗口，而噪声密度较大时（$p>0.2$）采用 5×5 的滤波窗口，最终的滤波效果较好。

iii. 采用一定的边缘检测算法检测滤波后的灰度图像的边缘。Canny 检测算子有着更好的边缘检测效果，因此本论文采用 Canny 算子进行边缘检测。

图 4-3 显示了对彩色噪声图像的边缘检测结果。可以看出如果不进行预先的滤波降噪处理，图像中的噪声会造成大量的错误检测结果。当噪声密度较大时，采用 3×3 的滤波窗口降噪后仍会在边缘检测结果中保留一部分噪声，而如果采用 7×7 的窗口滤波，如图 4-3（g） 所示，检测结果中会丢失很多信息。



（a） （b） （c）

（d） （e） （f）

（g）

图 4-3（a）原始图像；（b）噪声图像（p=0.2）；（c）原始图像中的边缘；（d）直接对噪声图像进行边缘提取结果；（e）3×3 窗口滤波后的边缘检测结果；（f）利用 5×5 窗口滤波后的边缘检测结果

B．基于图像边缘检测的加权机制

为了确定滤波窗口中各像素的权重系数，首先根据之前的图像边缘检测结果将滤波窗口中的像素划分为几个不同的组。同一组内的像素将会被赋予同样的权重。如图 4-4 所示，本论文使用 5×5 的滤波窗口对噪声点进行处理。图中深灰色部分代表图像边缘。根据滤波窗口内像素点相对于图像边缘的位置信息，按照以下方法赋予其不同的权重：

1) 如果窗口中心像素在图像边缘上，如图 4-4（a），则位于图像边缘上的像素点划分为同一组，将它们的权重置为 ω_e。窗口内其他像素点划分为另一组，权重为 ω_3。因为图像边缘上的像素与中心像素更相似，所以令 $\omega_e > \omega_3$。

2) 如果窗口中心像素不在图像边缘上，但是滤波窗口内存在图像边缘，如图 4-4（b）（c），则将中心像素的权重设置为 ω_c，同时将窗口内的其他像素点分为三组：位于图像边缘上的像素点为一组，权重设为 ω_2；与中心像素位于图像边缘同一侧的分为另一组，权重设为 ω_1；其余的像素点分为一组，权重设为 ω_3。考虑到各组中的像素与中心像素的相似程度，令 $\omega_c > \omega_1 > \omega_2 > \omega_3$。
3) 如果滤波窗口内不存在图像边缘，如图 4-4（d），则中心像素的权重置为 ω_c，其余像素的权重为 ω_1。这种情况类似于中心加权矢量中值滤波，令 $\omega_c > \omega_1$。

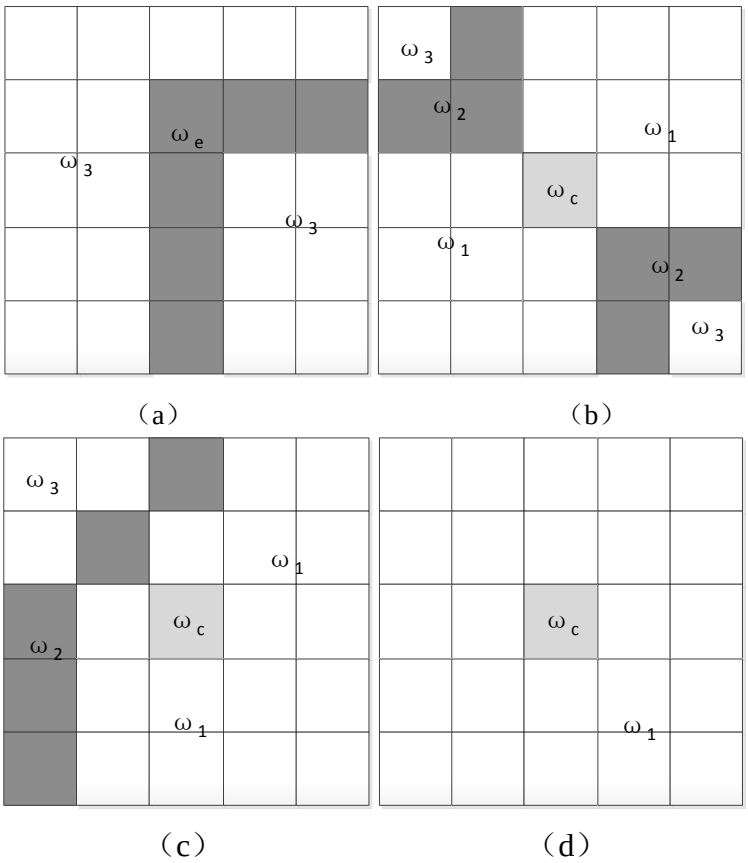
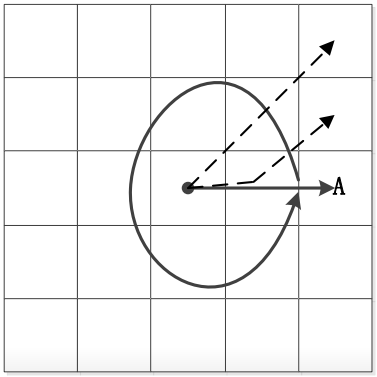


图 4-4 基于图像边缘检测结果对滤波窗口内像素进行的分组

为了实现上述的权重分配结果，本论文采用了如下分配策略：如图 4-5（a）所示，首先定义一个扫描向量 **A** = {a1, a2, a3}，a1,a2,a3 = 0 或 1 。其中 0 代表非边缘像素点（即不在图像边缘上的像素），1 代表边缘像素点。a1 始终位于滤波窗口中心。图 4-5（b）显示了与扫描向量 **A** 相对应的权重向量。例如，如果 **A** = {0, 1, 0}，代表着中心像素是一个非边缘像素点，扫描向量对应的第二个像素为边缘像素点，第三个像素为非边缘像素点。这种情形属于之前描述的情形 2）。

所以这三个像素点所赋予的权重为：中心像素点的权重为 ω_c；第二个像素的权重为 ω_2；第三个像素的权重为 ω_3。将向量 **A** 如图 4-5（a）所示沿逆时针方向扫描滤波整个窗口，这样窗口内的每一个像素都会被赋予指定的权重。



A	ω
{0, 0, 0}	{ω_c, ω_1, ω_1}
{0, 0, 1}	{ω_c, ω_1, ω_2}
{0, 1, 0}	{ω_c, ω_2, ω_3}
{0, 1, 1}	{ω_c, ω_2, ω_2}
{1, 0, 0}	{ω_e, ω_3, ω_3}
{1, 0, 1}	{ω_e, ω_3, ω_e}
{1, 1, 0}	{ω_e, ω_e, ω_3}
{1, 1, 1}	{ω_e, ω_e, ω_e}

（a） （b）

图 4-5 权重分配方法

根据公式（4-7）以及大量的实验统计结果，各权重的比例关系为：$\omega_e/\omega_3=3$, $\omega_2/\omega_3=2$, $\omega_1/\omega_2=1.25$, $\omega_c/\omega_1=3.5$。本论文提出的基于图像边缘检测的开关加权矢量中值滤波算法流程图如图 4-6 所示：

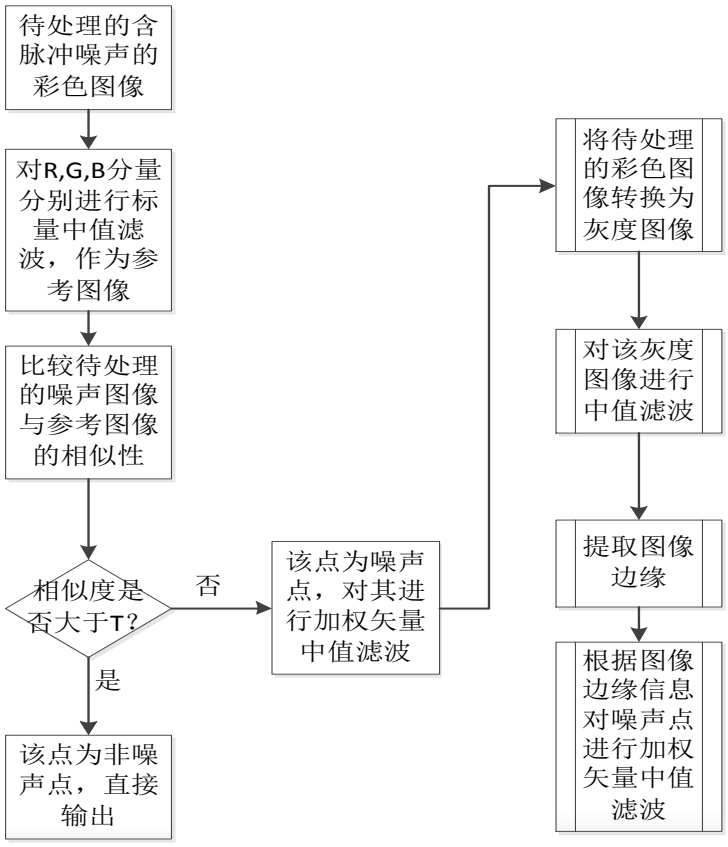


图 4-6 改进的矢量中值滤波算法流程图

4.4 消除冗余计算的矢量中值滤波电路

矢量中值滤波算法虽然能够有效抑制彩色图像中的脉冲噪声且不会引起色彩失真，但是其滤波过程中需要进行大量的乘法运算，加法运算和开方运算，尤其是乘法运算和开方运算十分复杂，占用大量的软、硬件资源，处理时间很长。为了提高矢量中值滤波的运算效率，减少运算时间，研究者分别从算法方面[28,29]和电路结构方面给进行了改进。

从算法角度改进方法中比较有名的是 Samuel Morillas 等人提出的基于快速对等组的矢量中值滤波(PGF, peer group filter)[28]。PGF 算法首先定义了关于滤波窗口 W 内中心像素点 X_c 的 peer group 集合：

$$P(X_c,d)=\{X_i \in W \mid \rho(X_c,X_i)\le d\} \tag{4-12}$$

其中，$\rho(X_c,X_i)$ 是滤波窗口内中心像素点与周围像素点 X_i 之间的欧氏距离；d>0,为预先设定的阈值。也就是说 peer group 是窗口内所有与中心像素点的欧氏距离不大于 d 的像素点的集合。定义 P(X_c，k，d)为 peer group 的子集，表示 peer group 中任意 k+1 个像素点的集合，其中 k 为非负整数。假如滤波窗口中距离中心像素点的矢量距离小于等于 d 的像素点的个数不足 k 个，则判定中心像素点被噪声污染，执行正常的是量滤波过程；否则认为中心像素点不是噪声点，直接将其作为当前窗口的输出。这种方法有效地提高了矢量中值滤波的执行效率，但是随着图像噪声密度的增加，PGF 方法的噪声检测正确率迅速降低，滤波效果有待改进。

采用 ASIC 或者 FPGA 设计实现的硬件加速电路通常比软件算法的执行效率高出数倍。于是 Anis Boudabous 等人基于软硬件相结合的思想，通过将算法中所有的乘法和开方运算并行处理，先后设计了 L_1 范数的矢量中值滤波[30]，L_2 范数（即欧氏距离）矢量中值滤波电路[31],极大地提高了滤波电路的处理速度，缩短了运算时间。其后，又根据类似的方法设计了矢量方向距离滤波（VDDF）[32], 以及矢量方向滤波（VDF）[33], 矢量方向中值滤波（VDMF）[34]等一系列电路。这些电路基本上都利用了 FPGA 中提供的 DSP，例如其设计的 3×3 窗口的并行矢量中值滤波电路中，总共例化了 36 个 DSP 模块，相对于软件算法，虽然提高了运算速度，但也占用了过多的硬件资源。

实际上，由于相邻滤波窗口中存在交叠区域，所以在是量滤波过程中存在着

大量的冗余数据和计算。首先是输入数据冗余，对于 n×n 窗口的滤波，共有 n^2 个输入数据，而实际上除去重复数据，每次只需要新录入 n 个像素数据；然后是欧氏距离冗余，同样由于像素值存在交叠，欧氏距离运算时相邻窗口中也会产生部分相同的欧氏距离（相同的欧氏距离数量为$[n(n-2)]^2/2-n(n-2)/2$）；最后在计算新窗口中的 D_i 时，若 i 位于交叠区域，其输入也有部分欧氏距离来自前一窗口。

根据以上分析，本论文充分利用滤波窗口滑动过程中相邻滤波窗口间数据以及中间结果的对应关系，提出了一种消除冗余计算的矢量中值滤波电路，减少了各种运算单元的数量，节省了硬件资源。

根据矢量中值滤波算法，本电路主要分为三大部分：36 个欧氏距离 ρ_{ij}（L_2 norm）的运算；欧式距离的求和运算（D_i）；排序电路。本文主要针对前两个部分提出改进。

4.4.1 矢量距离计算中的去冗余技术

如图 4-7，本设计中的滤波窗口大小为 3×3，图中的数字代表窗口内每个像素的编号。窗口从左向右滑动，1,2,3,…,9 代表第一个窗口，4,5,6,…,12 代表第二个窗口。括号中的数字显示了窗口滑动过程中后一窗口与前一窗口各像素点的对应关系。对每一个滤波窗口而言共需要计算 36 个欧氏距离，即 1-2,1-3,1-4，…,7-8,7-9,8-9。但是由图 4-7 可以看出，相邻的滤波窗口内有 6 个像素点是相同的。这样处理相邻两滤波窗时，由公式（3-1）算得的欧式距离中，有 15 个结果是重复的，包括 4-5,4-6,4-7,4-8,…,7-8,7-9,8-9。也就是说，每次在处理新的窗口时，实际上只需计算 21 个新的欧氏距离运算单元，即 4-10,4-11,4-12,5-10，…，10-11,10-12,1-12，通过复用之前保存的 15 个结果便可实现 36 个欧氏距离的运算。

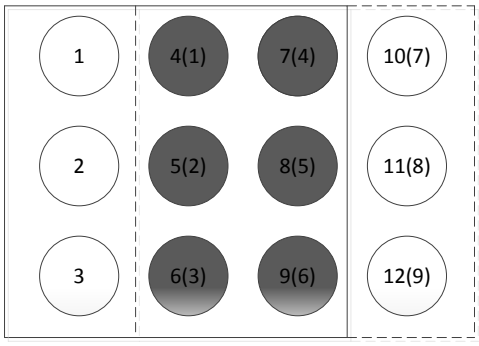


图 4-7 数据复用原理

令 $A_n=(R_i-R_j)^2+(G_i-G_j)^2+(B_i-B_j)^2$,n=1,2,3，…,21。在欧氏距离的运算中，每 3 次平方运算对应 1 次开方运算。矢量中值滤波算法中，最占资源的是开方运

算，因此设计中将 21 个 An 运算单元分为 3 组，每个时钟周期产生 7 个 An 输出。这样整个电路只需 7 个平方根计算单元（图 4.8 中的 sqrt1~sqrt7）。

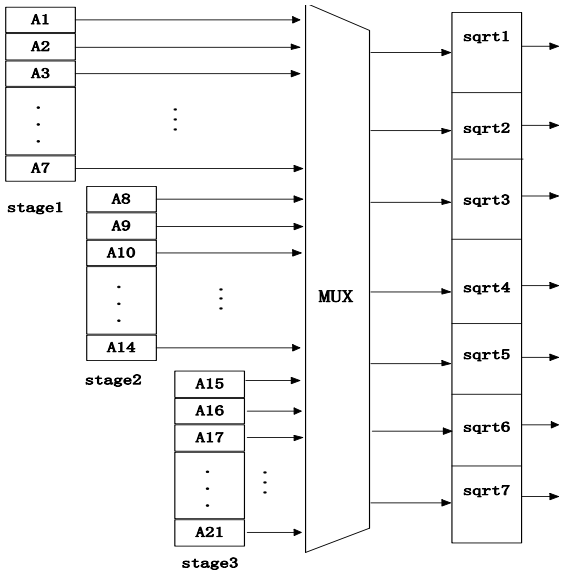


图 4-8　欧氏距离运算电路结构

矢量中值滤波中的欧氏距离计算电路如图 4-8 所示。当处理图像的每一行的第一个像素时，首先计算前 15 个欧氏距离，然后计算后 21 个欧氏距离，与之合并成 36 个欧氏距离。之后每次只需计算新的 21 个欧氏距离，并通过移位操作便可得到所有需要的欧氏距离。为简化后续选择电路，每个窗口算得的 36 个欧氏距离的存储寄存器位置必须相一致，即新窗口算得的 ρ_{ij} 与前一窗口算得的 $\rho_{i'j'}$ 必须存放在同一寄存器中。表 4-1 列出了寄存器与数据的对应关系。由图 4-8 可以看出，本设计中所需减法运算为 21×3=63 次，平方运算为 21×3=63 次，加法运算为 21×2=42 次，开方运算仅为 7 次。

表 4-1 寄存器与欧氏距离对应关系

寄存器	i,j	寄存器	i,j	寄存器	i,j	寄存器	i,j	寄存器	i,j	寄存器	i,j
norm1	1,7	norm7	3,7	norm13	5,7	norm19	7,8	norm25	1,4	norm31	3,4
norm2	1,8	norm8	3,8	norm14	5,8	norm20	7,9	norm26	1,5	norm32	3,5
norm3	1,9	norm9	3,9	norm15	5,9	norm21	8,9	norm27	1,6	norm33	3,6
norm4	2,7	norm10	4,7	norm16	6,7	norm22	1,2	norm28	2,4	norm34	4,5
norm5	2,8	norm11	4,8	norm17	6,8	norm23	1,3	norm29	2,5	norm35	4,6
norm6	2,9	norm12	4,9	norm18	6,9	norm24	2,3	norm30	2,6	norm36	5,6

4.4.2 矢量和计算中的去冗余技术

除了相邻窗口中欧氏距离的结果存在冗余，本文还注意到在计算 Di 时也存在大量的冗余计算。为了消除冗余，本设计使用了 8 数据输入、6 数据输入、4 数据输入三种加法器。图 4-9 中列出了新窗口中所有的 Di 计算公式（其中 4,5 表示图 4-7 中像素点 4 和 5 之间的欧氏距离，其余的与之类似）。其中实线框中表示可以复用的前一窗口的求和操作，虚线框中的数字部分表示可以被下一窗口复用的求和操作。通过观察可以发现，处理每一个窗口时其虚线框与下一窗口中的实线框部分是一一对应的。由此可以推出 Di 的互联电路结构如图 4-10 所示，其中 D_1，D_2，D_3 使用 4 输入加法器，D_4，D_5，D_6 使用 6 输入加法器，其产生的 sum_reuse 分别作为 D_1，D_2，D_3 的 sum_temp 输入，D_7, D_8, D_9 使用 8 输入加法器，其输出结果 sum_reuse 分别作为 D_4，D_5，D_6 的 sum_temp 输入。这样将所有的 D_i 分解成不同输入加法，利用图 4-10 的三类加法器，对于虚线框中的计算结果以 sum_reuse 输出并保存，而对于实线框的计算结果则调用上一个窗口保存的 sum_reuse 由 sum_temp 端口输入。即可实现去冗余的目的。

假如没有复用，则计算每一个 D_i 需要 7 个加法器，总共需要 9×7=63 个加法器。而通过本文设计的加法电路，总共只需 3×3+3×5+3×7=45 个加法器。

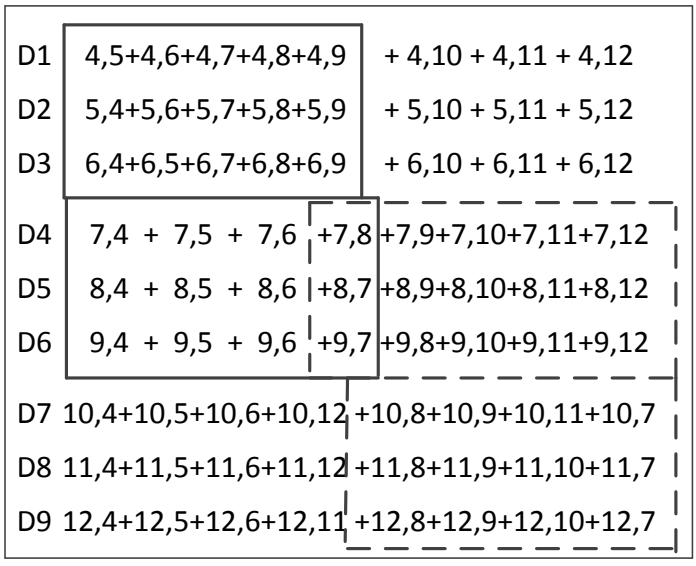


图 4-9 部分和复用原理

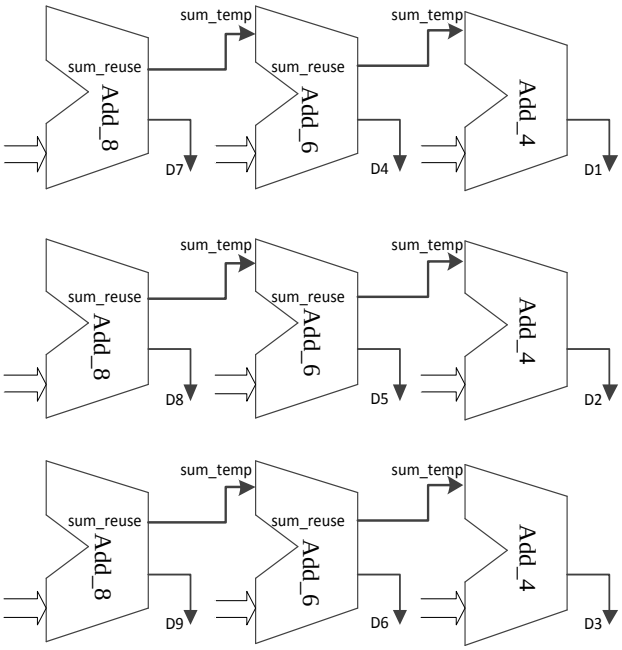


图 4-10 D_i 互连电路结构

4.4.3 改进的矢量中值滤波整体电路

本文设计的矢量中值滤波电路结构如图 4-11 所示。其中 L2 Norm Operation 部分为前文所提到的欧氏距离计算电路，norm1…norm36 对应每个 3×3 滤波窗口需要求得的 36 个欧氏距离，D1… D9 即为图 4-4 中的 D_i 互连电路，sort and decoder 是排序和解码电路。

当系统稳定运行后，1 个周期用于数据录入，产生 X1…X9,每 3 个周期产生 An 计算结果。开方电路需要 12 个时钟周期(开方电路使用 Xilinx 公司提供的 IP，运算所用周期数可配置)。从数据输入到得出 36 个欧氏距离结果共需要 18 个时钟周期，然后经过 3 个周期的求和运算得出 D1-D9,再经过 5 个周期的排序和解码运算得出输出结果。所以对于每一次新的滤波窗口的处理时间为 18+3+5=26 个时钟周期。另外，对于图像中每行的行头，需要首先计算出 15 个欧氏距离，再计算出另外 21 个欧氏距离，以实现复用。由此产生的额外时间开销为 18 个时钟周期。

通过本文设计的电路，矢量中值滤波中计算 ρ_{ij}（1，2,…,9；j=1,2，…，9；i≠j）时，减法单元节省了 45 个，乘法（平方）器节省了 45 个，加法器节省了 30 个，开方单元减少了 29 个。而计算 D_i（i=1，2,…,9）时，加法器减少 18 个。图 4-12 列出了本设计与没有复用时基本运算单元的使用数量的对比情况。

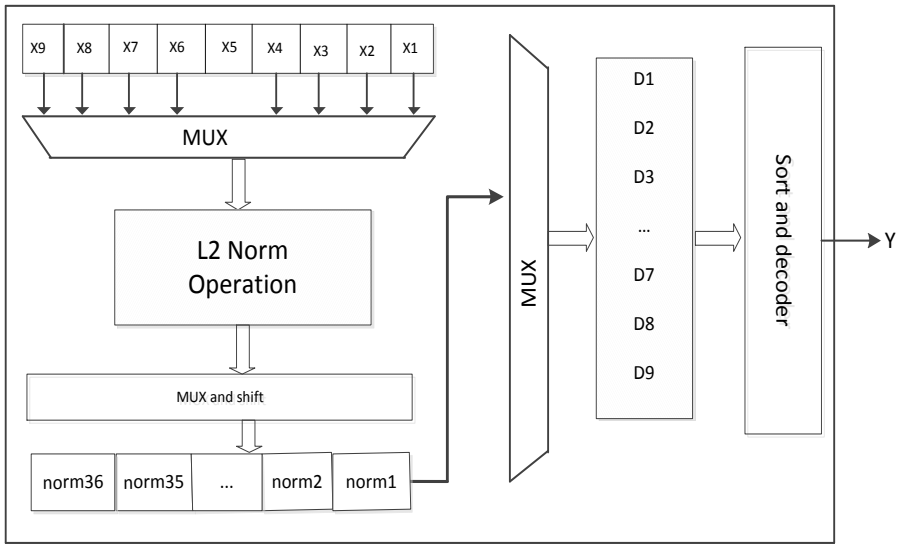


图 4-11 矢量中值滤波电路

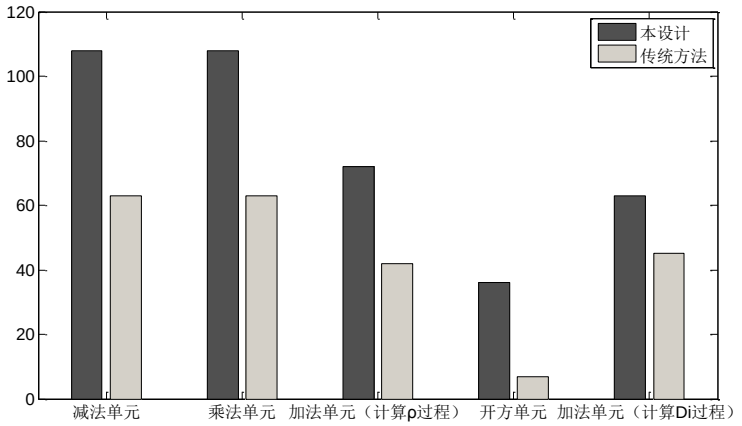


图 4-12 本设计与传统方法中基本运算单元数量对比

第五章 实验仿真结果及分析

本论文的硬件仿真结果是基于 Xilinx LX110T FPGA 开发平台以及 ISE13.3 集成开发环境和 Modelsim10.0a 联合仿真完成；软件仿真结果是基于 Win7 系统下 Matlab R2010a 完成。

5.1 预处理子系统验证平台

本论文所使用的 LX110T FPGA 开发板是一款高性能的系统开发平台。开发板上有丰富的硬件资源，主要包括：一块 Xilinx Virtex5 FPGA，一块 256MB 大小的 DDR2 SODIMM 模块，两块 32MB 的 Flash PROM，以及以太网接口，USB 接口 RS232 端口等。通过 ISE 集成开发环境中的 EDK、SDK 套件可以实现以 Microblaze 处理器为主控制器，PLB 为系统总线以及典型系统外设和自定义外设 IP 的 SOC 系统开发。

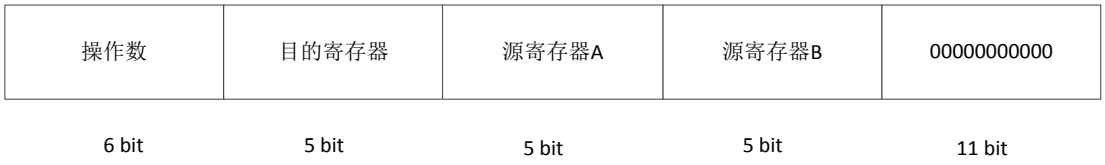
5.1.1 Microblaze 处理器核

Virtex-5 FPGA提供Microblaze处理器软核以方便客户进行SoC设计。Microblaze处理器软核是RISC精简指令集处理器，设计人员可以根据设计定制处理器核的可选配置。Microblaze处理器有32个32位的通用处理器和2个必选特殊寄存器（程序计数器和状态寄存器）以及16个可选的特殊功能寄存器（取决于用户配置）。Microblaze处理器采用哈佛存储结构，即指令和数据采用独立的访问地址空间，指令和数据的地址空间都是32 位长度，所以Microblaze可以独立的访问4GB的地址空间的指令和数据存储器。Microblaze对存储器和I/O设备进行统一编址，可以通过三种接口方式访问存储器和I/O设备，即：本地存储器总线（LMB）；处理器本地总线（PLB）；Xilinx的CacheLink（XCL）。

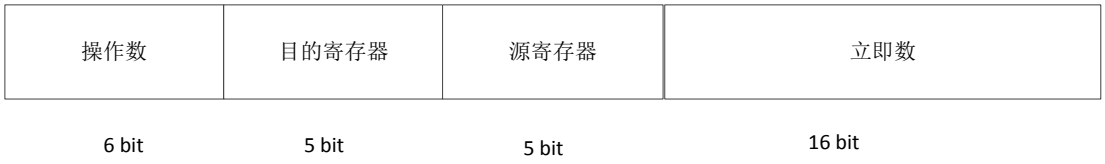
（1）Microblaze处理器的指令

Microblaze的指令长度为32位，指令字包含三个操作数。与MIPS指令集类似，Microblaze的指令集可以分为寄存器类型和立即数类型两类。寄存器类型的指令包含两个源寄存器操作数和一个目的寄存器操作数。立即数类型的指令包含一个源寄存器、一个16位的立即数（可通过imm指令扩展到32位）和一个目的寄存器。两种指令格式如图5-1所示。通过这两种指令，Microblaze可以完成算术操作、逻

辑操作、分支操作加载/存储操作等。



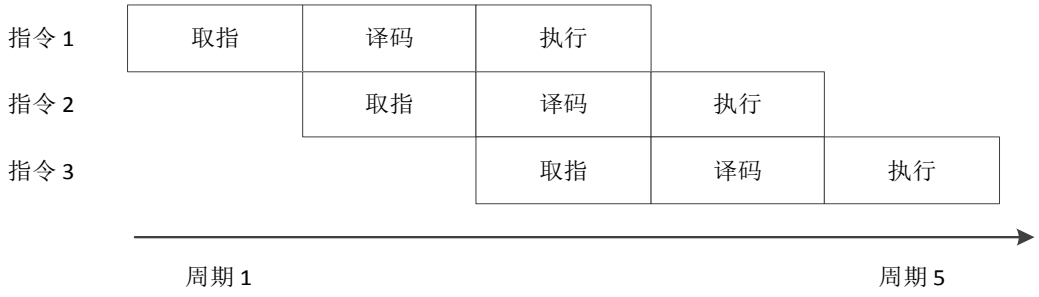
（a）寄存器类型指令结构



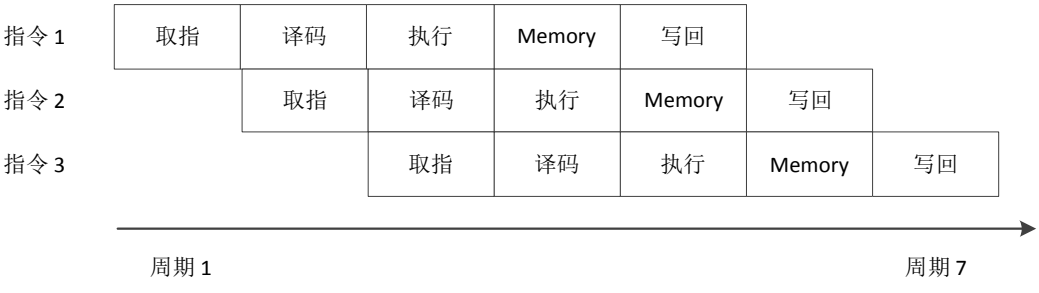
（b）立即数类型指令结构

图5-1 Microblaze处理器指令结构

Microblaze处理器采用流水线的方式执行指令。一般的指令需要一个时钟周期完成，对于少数需要多个时钟周期完成的指令，则采用流水线断流的方式实现。设计过程中，如果减少硬件开销为主要目的，采用面积优化，则流水线为三级，即取指，译码和执行。相反，如果设计重点考虑处理器性能，那么就不使用面积优化，将流水线分为取指、译码、执行、memory读写和写回五级。三级流水线结构和五级流水线结构分别如图5-2（a）和图5-2（b）所示。



（a）三级流水线结构



（b）五级流水线结构

图5-2 Microblaze处理器直观性指令采用的流水线结构

（2）Microblaze处理器的中断和异常

Microblaze支持复位、用户异常、中断、断点和硬件异常等事件。这些事件的优先级按照从低到高依次为：用户异常、中断、断点、非屏蔽断点、硬件异常、复位。与这些事件相关的存储器地址和保存返回地址的寄存器在表5-1中列出。

表5-1 向量地址和保存返回地址的寄存器

事件	向量地址	保存返回地址的寄存器
复位	0x00000000~0x00000004	—
用户异常	0x00000008~0x000000c	Rx
中断	0x00000010~0x00000004	R14
非屏蔽断点	0x00000018~0x0000001c	R16
硬件断点		
软件断点		
硬件异常	0x00000020~0x00000024	R17或BTR
保留	0x00000028~0x0000004f	—

5.1.2 PLB 总线

PLB（Processor Local Bus）是Xilinx公司提供的总线IP，用来进行Microblaze处理器与外设之间的通信。PLB总线IP包括一个总线控制单元，一个看门狗计时器，以及独立的地址通道、写数据通道和读数据通道。PLB总线支持多个总线主机，支持多个（可配置）从设备，并且提供32-bit，64-bit及128bit多种数据位宽选择；包含四种优先级的主机请，仲裁时间为3个时钟周期；PLB复位电路可产生与PLB总线时钟同步的复位信号，也可产生与外部复位信号同步的复位信号。图5-3显示了一种基于PLB总线的系统结构，其中包含一个总线仲裁器和一些必要的总线控制和选择逻辑，该系统中包含三个总线主机和三个从设备。

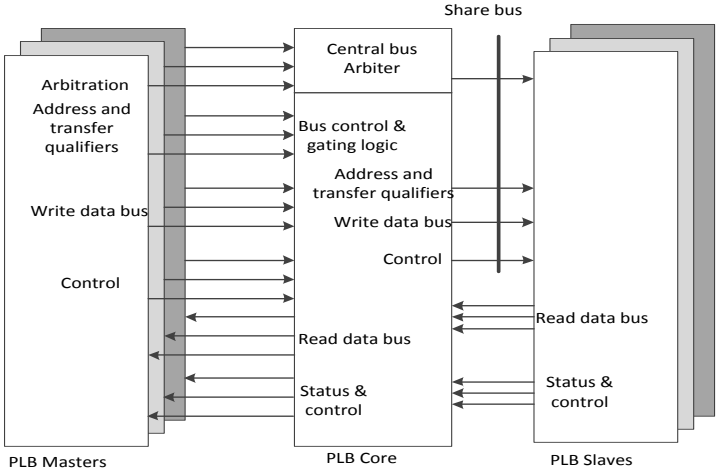


图5-3 PLB总线结构

5.2 硬件电路仿真结果

5.2.1 预处理子系统仿真结果

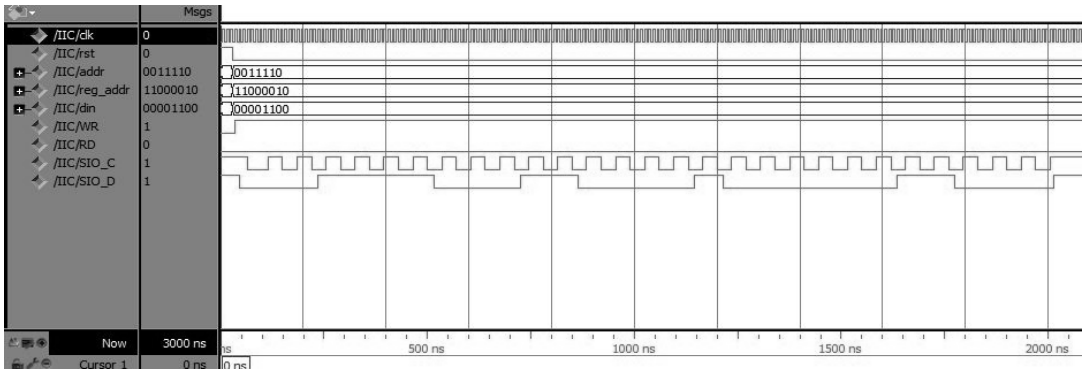


图5-4 SCCB master仿真结果

图5-4为SCCB master电路的仿真结果。图中显示了向OV2640图像传感器芯片内地址为C2（Hex）的寄存器写入0C（Hex）的过程。这是一次3-phase的写传输过程。可以看出，SCCB master正确产生了起始信号，紧接着发送了设备地址和写标志位，然后发送了寄存器的Sub-address和相应的配置数据，每一个字节之后都跟着一位Don’t Care Bit，最后产生传输结束标志。

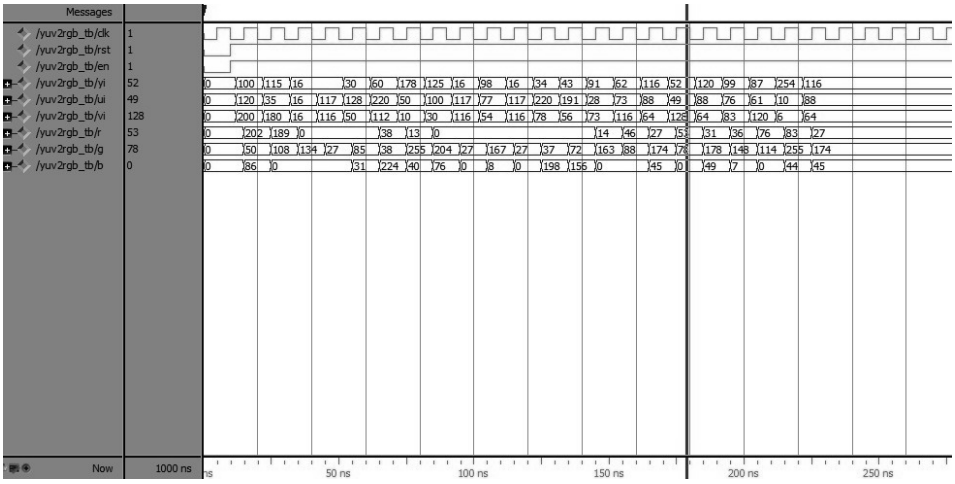


图5-5 图像格式转换电路仿真结果

图5-5为图像格式转换电路的输出结果，完成了YUV格式转换成RGB格式的功能。由于转换公式中设计小数运算，设计中将输入的8位数据扩展为16位，低8位作为小数部分，计算完成之后再截取高8位作为有效输出。根据公式（3-1），可以从仿真结果看出，全部数据均转换正确。

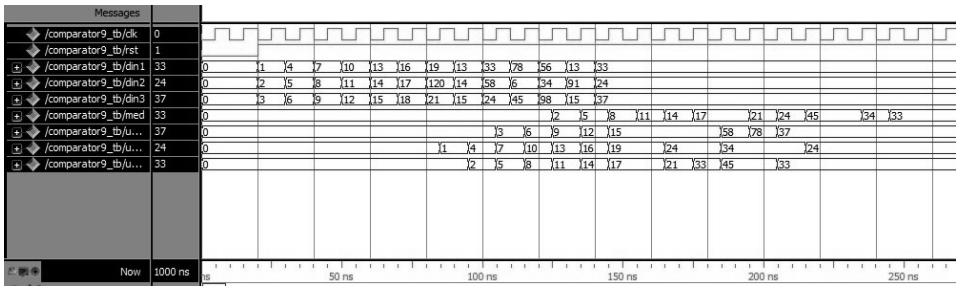


图 5-6 标准中值滤波仿真结果

图 5-6 为标准中值滤波电路的仿真结果。可以看出每个时钟周期输入 3 个数据，经过 3 个周期后完成第一个窗口的数据输入，再经过 8 个时钟周期开始输出第一个窗口的滤波结果，之后每个时钟周期可以完成一个 3×3 窗口的滤波。由仿真结果可以看出，每个窗口的滤波结果都正确。

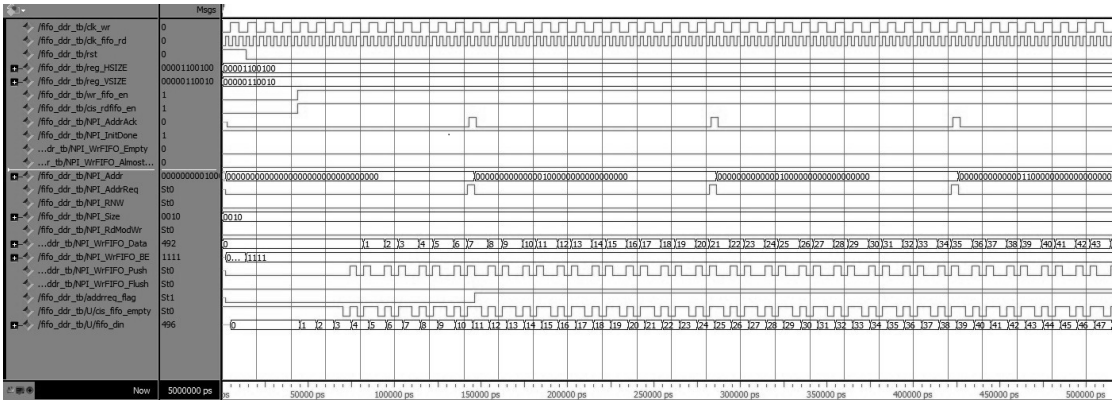


图5-7 DDR写传输控制电路仿真结果

图5-7为DDR写传输控制模块电路的仿真结果。由于DDR时钟和前面的电路不是同一个时钟域，因此在DDR读写控制模块之间使用了异步FIFO。当异步FIFO不为空时，DDR控制端的WrFIFO_Push信号才能有效。图中实现了32bit NPI 8-word突发写传输的控制电路，通过在FPGA开发板上实验，并利用SDK软件调试工具，通过Microblaze处理器读取DDR中的值，结果证明读出的数据与写入的数据一致。

预处理子系统place&rout后的报告如表5-2所示

表5-2 预处理子系统place&rout报告

Device Utilization Summary:		
Number of BSCANs	1 out of 4	25%
Number of BUFGs	6 out of 32	18%
Number of BUFIOs	8 out of 80	10%
Number of DCM_ADVs	1 out of 12	8%
Number of DSP48Es	3 out of 64	4%
Number of IDELAYCTRLs	3 out of 22	13%
Number of LOCed IDELAYCTRLs	3 out of 3	100%
Number of ILOGICs	106 out of 800	13%
Number of LOCed ILOGICs	8 out of 106	7%
Number of External IOBs	202 out of 640	31%
Number of LOCed IOBs	202 out of 202	100%
Number of IODELAYs	80 out of 800	10%
Number of LOCed IODELAYs	8 out of 80	10%
Number of OLOGICs	196 out of 800	24%
Number of PLL_ADVs	1 out of 6	116%
Number of RAM18X2SDPs	1 out of 148	1%
Number of RAM36_EXPs	51 out of 148	34%
Number of Slices	3045 out of 17280	17%
Number of Slice Registers	6001 out of 69120	8%
Number used as Flip Flops	5984	
Number used as Latches	0	
Number used as LatchThrus	17	
Number of Slice LUTs	4632 out of 69120	6%
Number of Slice LUT-Flip Flop pairs	7960 out of 69120	11%

5.2.2 改进的 VMF 电路仿真结果

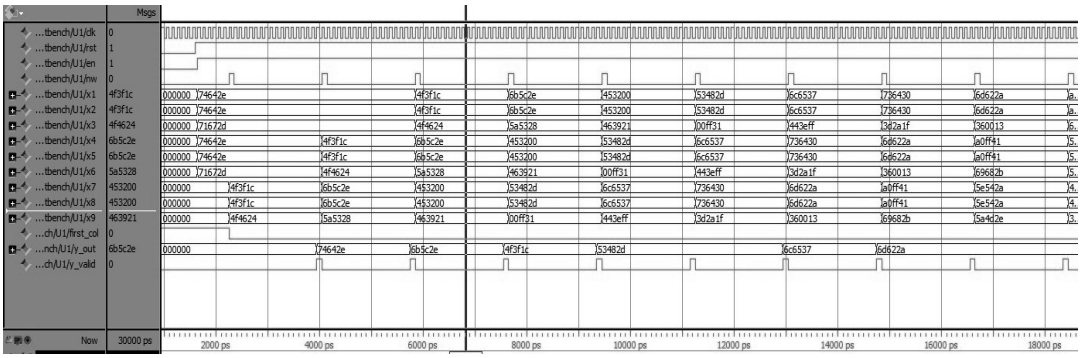


图5-8 改进的VMF电路仿真结果

图5-8为改进的矢量中值滤波电路仿真结果。其中X1~X9为24bits的图像数据。将图中数据在Matlab中进行软件VMF运算，实验结果证明两者输出结果一致。

本文将含有10%椒盐噪声的标准测试图像如lena还选用了MAE、PSNR作为图像质量的客观评价指标，对处理后的图像进行评测。MAE、PSNR的计算公式为：

$$MAE = \frac{\sum_{i=1}^{N}\sum_{k=1}^{M}\left\|x_{ik}-o_{ik}\right\|_1}{3\times NM} \tag{5-1}$$

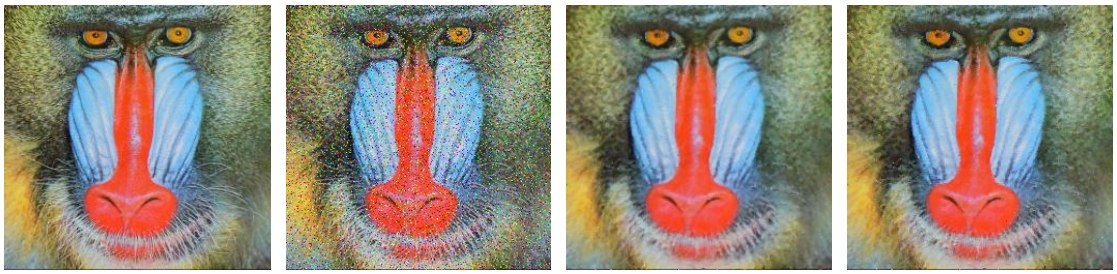
$$MSE = \frac{\sum_{i=1}^{N}\sum_{k=1}^{M}\left\|x_{ik}-o_{ik}\right\|_2^2}{3\times NM} \tag{5-2}$$

$$PSNR = 20\log_{10}\left(\frac{255}{\sqrt{MSE}}\right) \tag{5-3}$$

其中N，M分别代表图像的宽度和高度，x_{ik}表示滤波后图像中的像素，o_{ik}表示原始图像中的像素。$\|\cdot\|_1$ 表示L_1范数，$\|\cdot\|_2$表示L_2范数。表5-3中列出了各图像的MAE及PSNR结果，可以看出本文设计的电路处理结果与软件处理结果完全一致。即证明本文设计的电路完全实现了矢量中值滤波的功能。图5-9分别显示了软件处理方法和本文设的降噪系统处理后的效果图，可以看出本文设计的矢量中值滤波电路的滤波效果与Matlab滤波效果相同.

表5-3 软件实现与本设计处理后的MAE、PSNR结果

图像	软件实现		本设计	
	MAE	PSNR	MAE	PSNR
lena	5.59	26.84	5.59	26.84
peppers	4.20	28.25	4.20	28.25
baboon	9.79	24.27	9.79	24.27



（a） （b） （c） （d）

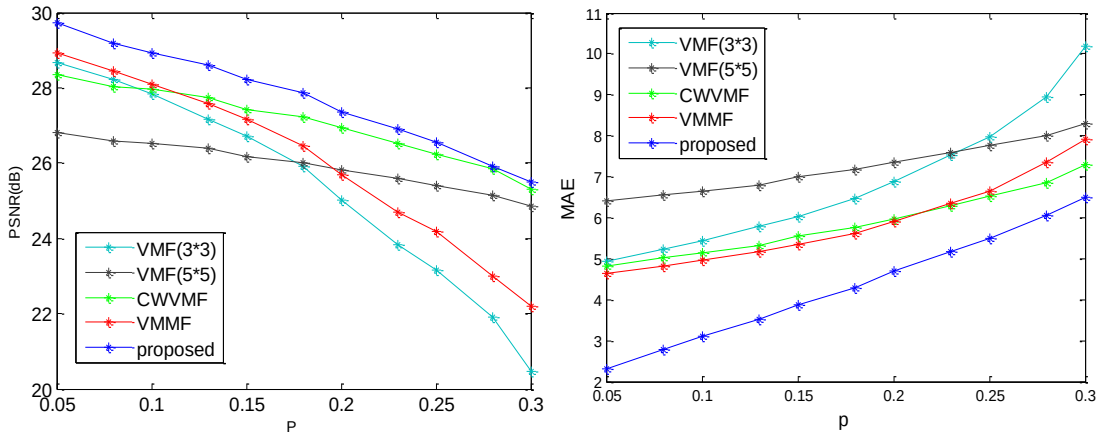
图 5-9 （a）原始图像；（b）加入 10%椒盐噪声的图像；（c）Matlab 软件算法处理后的图像；（d）本论文中的 VMF 电路处理后的图像

5.3 中值滤波改进算法的实验结果

为了验证本论文提出的改进算法的处理效果，本论文进行了大量的仿真实验。采用MAE、MSE、PSNR作为评价指标，分别与标准的VMF算法（3×3窗口及5×5窗口）、CWVMF算法（5×5窗口）及VMMF算法（3×3窗口）的处理结果进行了详细的对比。

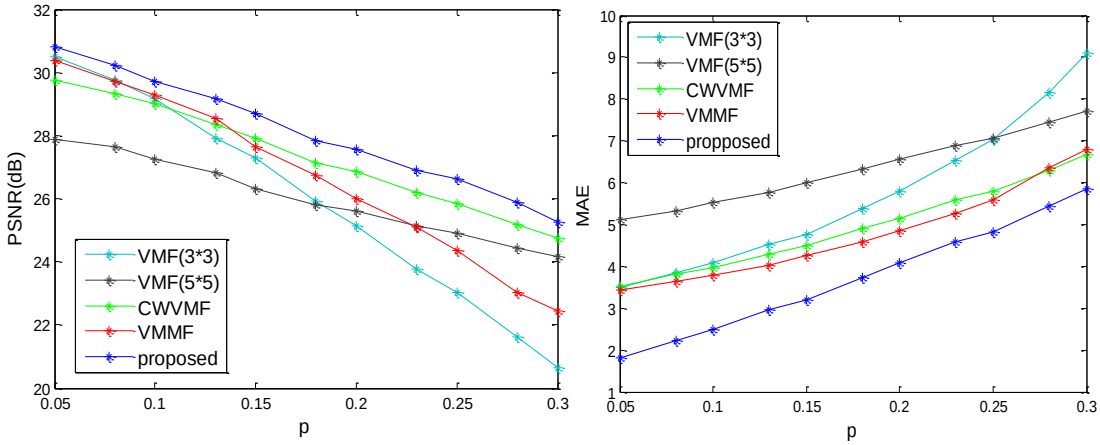
图5-10和图5-11为根据实验结果绘制的MAE、PANR曲线。其中横坐标表示噪声密度。从图中可以看出当噪声密度较小时，采用3×3滤波窗口的VMF比采用5×5窗口的VMF处理效果要好，这是因为噪声密度不大时，3×3的滤波窗口便能基本滤除所有噪声，而5×5窗口的滤波却损伤了图像细节。随着噪声密度的增加，由于较大滤波窗口的VMF算法有更强的降噪能力，所以处理效果较好。对于5×5窗口的CWVMF，由于其具备一定的保护图像细节的能力，所以低噪声密度时可以达到与3×3窗口的VMF算法相近的滤波效果。VMMF算法有着很强的降噪能力，但是会引入人工色彩，造成色彩失真。与这些算法相比，本论文提出的矢量中值滤波算法（采用5×5的滤波窗口）无论在低噪声密度时还是高噪声密度时，处理效果均比较好。表5-4和表5-5列出了采用不同滤波算法对图像Boats和Hat进行处理后的MAE、MSE和PSNR的值。

采用不同滤波算法对测试图像进行处理后的图像由图5-12至5-13所示，从直观上显示了各种不同滤波算法的处理效果。



（a） （b）

图5-10 Lena图像采用不同滤波算法处理后的结果：（a）PSNR结果；（b）MAE结果



（a） （b）

图5-11 对Peppers图像采用不同滤波算法处理后的结果：（a）PSNR结果；（b）MAE结果

表5-4 采用不同滤波算法对不同噪声密度的Boats图像处理后MAE、MSE和PSNR结果

	5%			10%			20%			30%		
filter	MAE	MSE	PSNR	MAE	MSE	PSNR	MAE	MSE	PSNR	MAE	MSE	PSNR
none	6.344	853.95	18.816	13.065	1770.1	15.650	25.382	3424.1	12.785	38.19	5151.1	11.012
VMF (3×3)	5.124	102.57	28.020	5.675	126.37	27.114	7.280	235.33	24.414	10.480	582.72	20.476
VMF (5×5)	7.656	202.45	25.067	7.965	214.10	24.824	8.617	241.64	24.299	9.538	288.66	23.526
CWVMF (5×5)	5.845	144.07	26.544	6.174	154.43	26.243	7.058	187.89	25.391	8.297	252.32	24.111
VMMF (3×3)	4.959	102.28	28.032	5.330	122.06	27.265	6.386	199.45	25.132	8.596	443.26	21.664
Proposed (5×5)	**3.086**	**97.13**	**28.257**	**3.890**	**112.54**	**27.618**	**5.712**	**166.43**	**25.918**	**7.488**	**240.09**	**24.327**

表5-5 采用不同滤波算法对不同噪声密度的Hat图像处理后MAE、MSE和PSNR结果

	5%			10%			20%			30%		
filter	MAE	MSE	PSNR	MAE	MSE	PSNR	MAE	MSE	PSNR	MAE	MSE	PSNR
none	6.348	858.52	18.793	12.706	1717.1	15.78	25.619	3467.8	12.730	38.253	5181	10.986
VMF (3×3)	2.243	27.36	33.759	2.592	39.26	32.191	3.779	123.90	27.199	6.480	426.41	21.832
VMF (5×5)	3.134	44.96	31.603	3.337	50.85	31.067	3.932	72.56	29.523	4.609	98.49	28.196
CWVMF (5×5)	2.248	29.36	33.453	2.491	35.30	32.653	3.186	57.68	30.519	4.073	95.31	28.339
VMMF (3×3)	2.104	25.67	34.036	2.337	34.83	32.711	3.053	85.07	28.832	4.763	273.44	23.762
Proposed (5×5)	**0.845**	**17.33**	**35.742**	**1.315**	**24.66**	**34.210**	**2.353**	**48.69**	**31.256**	**3.545**	**94.17**	**28.392**

(a) (b) (c)

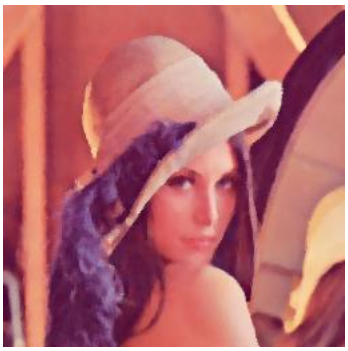
(d) (e) (f)

(g) (h)

图5-12 对图像Boats采用不同滤波算法处理后的结果，其中噪声密度p=0.20。（a）原始图像；（b）噪声图像；（c）提取的图像边缘；（d）3×3窗口的VMF处理结果；（e）5×5窗口的VMF滤波后的图像；（f）5×5窗口的CWVMF滤波后的图像；（g）3×3窗口的VMMF滤波后的图像

(a) (b) (c)


(d)


(e)


(f)


(g)


(h)

图5-13 对图像Lena采用不同滤波算法处理后的结果，其中噪声密度p=0.10。（a）原始图像；（b）噪声图像；（c）提取的图像边缘；（d）3×3窗口的VMF处理结果；（e）5×5窗口的VMF滤波后的图像；（f）5×5窗口CWVMF滤波后的图像；（g）3×3窗口的VMMF滤波后的图像

第六章 总结与展望

6.1 总结

多媒体技术的飞速发展，视觉信息系统中的数据量也越来越大，对处理器的计算性能提出了更高的要求。可重构处理器克服了ASIC设计功能单一、灵活性差的缺点，同时又比传统的处理器架构有更高的计算效率，因此成为了集成电路设计领域的一个研究热点。为了验证可重构处理器的性能，本文设计了一种基于可重构处理器的视觉信息预处理子系统。

本文首先介绍了可重构处理器的研究现状以及视觉信息预处理子系统中需要解决的问题和常用的解决方法。然后介绍了视觉信息处理系统的整体架构，分析了高清视频数据源、预处理子系统、可重构处理器子系统以及后处理子系统的功能和具体的工作流程，重点分析了预处理子系统中的数据流。

预处理子系统以OV2640芯片作为高清视频数据源，以Xilinx FPGA LX110T作为预处理子系统的硬件开发平台。该系统通过SCCB总线对图像传感器进行配置，使其输出指定格式的图像数据。对图像数据采集后，首先进行中值滤波降噪处理，以消除图像噪声对后续高级视觉信息处理算法的影响。然后对图像数据进行格式转化，以适应不同算法对不同图像格式的需要。经过处理后的图像数据在DDR存储器中进行一定的帧缓存之后根据需要分别发送给可重构处理器和后处理系统进行进一步处理。

本文分别设计了视觉信息预处理子系统中的各个模块电路并进行了仿真验证。根据SCCB的协议规范，设计了SCCB master模块电路，仿真结果显示本文所设计的master可以正确输出协议规范的起始、结束信号和数据结构。根据中值滤波算法原理，设计了流水线结构的中值滤波电路，仿真结果显示，对于输入的图像数据，本设计可以实现1个时钟周期内完成中值滤波运算。根据YUV与RGB格式的转换公式，设计了图像格式转换电路。在分析了DDR工作原理的基础上，利用Xilinx公司提供的内存控制器IP，设计了DDR的读写控制电路，实现图像数据的缓存处理。根据整个视觉信息处理系统的要求，定义了预处理子系统与可重构处理器以及后处理系统的接口规范。为了实现外部命令对系统工作模式的控制，本文采用Microblaze处理器中断控制的方式，实现系统的功能选择、任务调度。

相对于标量中值滤波，矢量中值滤波更适合彩色图像的降噪处理。本文设计

了一种消除冗余计算的矢量中值滤波电路，一方面通过并行大量的运算单元及流水线结构，提高了矢量中值滤波的运算速度，另一方面通过复用相邻滤波窗口内的重复数据及中间结果，节省了硬件资源。

同时本文还提出了一种改进的矢量中值滤波算法。该算法利用噪声图像与VMMF处理后的图像的相似性进行噪声检测，对于检测为噪声的像素，则根据图像边缘检测结果进行加权中值滤波处理。实验结果显示与传统的算法相比，该滤波算法在抑制脉冲噪声的同时，能够更好的保护图像细节。

6.2 展望

实验结果表明本设计的正确性和可行性，完成了预期的目标。但是本文所设计的预处理子系统仍有需要改进的地方。

该系统中对于工作模式的改变采用的是外部开关产生的中断实现的，并没有实现OSD菜单控制工作模式的改变。在接下来的研究设计中可以通过OSD菜单方式实现对整个系统的控制。

其次，由于完整的视觉信息处理系统所需要的可重构处理器还在研究设计中，本文所设计的相关接口规范还只是理论上可行，实际工作仍需要根据具体情况作调整。

另外本文采用的高清视频源在高清模式下最大帧频仅为为15fps，不能完全满足人类视觉效果的要求，需要进一步改进。

参考文献

[1] Compton K, Hauck S. Reconfigurable Computing: A Survey of Systems and Software. ACM Computing Surveys，2002，34(2):171-210.

[2] G. Estrin. Organization of computer system: the fixed plus variable structure computer. Western joint IRE-AIEE-ACM computer conference, 1960: 33~40.

[3] Lo C C, Tsai S T, Shieh M D. Reconfigurable architecture for entropy decoding and inverse transform in H. 264. IEEE Transactions on Consumer Electronics, 2010, 56(3): 1670-1676.

[4] Nunez-Yanez J L, Spiteri T, Vafiadis G. Multi-standard reconfigurable motion estimation processor for hybrid video codecs. IET Computers & Digital Techniques, 2011, 5(2): 73-85.

[5] Lai Y K, Lai Y F. A reconfigurable IDCT architecture for universal video decoders. IEEE Transactions on Consumer Electronics, 2010, 56(3): 1872-1879.

[6] Rossi D, Mucci C, Campi F, et al. Application space exploration of a heterogeneous run-time configurable digital signal processor. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2013, 21(2): 193-205.

[7] Chen T W, Chen Y L, Cheng T Y, et al. A multimedia semantic analysis SoC (SASoC) with machine-learning engine. 2010 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC). 2010: 338-339.

[8] Komuro T, Tabata T, Ishikawa M. A reconfigurable embedded system for 1000 f/s real-time vision. IEEE Transactions on Circuits and Systems for Video Technology, 2010, 20(4): 496-504.

[9] Kim H E, Yoon J S, Hwang K D, et al. A Reconfigurable Heterogeneous Multimedia Processor for IC-Stacking on Si-Interposer. IEEE Transactions on Circuits and Systems for Video Technology, 2012, 22(4): 589-604.

[10] Lo W Y, Lun D P, Siu W C, et al. Improved SIMD architecture for high performance video processors. IEEE Transactions on Circuits and Systems for Video Technology, 2011, 21(12): 1769-1783.

[11] Llamocca D. A Dynamically Reconfigurable Pixel Processor System Based on Power/Energy-Performance-Accuracy Optimization. IEEE Transactions on Circuits and Systems for Video Technology, 2013, 23(3): 488-502.

[12]陈守水，基于偏微分方程的图像降噪及质量评价研究. [博士学位论文]. 上海：上海交通大学，2008.

[13]邱宇，基于双边滤波的图像去噪及锐化技术研究. [博士学位论文].重庆：重庆大学, 2011.

[14]J. Astola, P. Haavisto, Y. Neuvo, Vector median filters. Processing of IEEE, 1990, 78 (4): 678–689.

[15]Chan R H, Ho C W, Nikolova M. Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization. IEEE Transactions on Image Processing, 2005, 14(10): 1479-1485.

[16]Akkoul S, Ledee R, Leconge R, et al. A new adaptive switching median filter. IEEE Signal processing letters, 2010, 17(6): 587-590.

[17]Dong Y, Xu S. A new directional weighted median filter for removal of random-valued impulse noise. IEEE Signal Processing Letters, 2007, 14(3): 193-196.

[18]Morillas S, Gregori V. Robustifying vector median filter. Sensors, 2011, 11(8): 8115-8126.

[19]Trahanias P E, Karakos D, Venetsanopoulos A N. Directional processing of color images: theory and experimental results. IEEE Transactions on Image Processing, 1996, 5(6): 868-880.

[20]Karakos D G, Trahanias P E. Generalized multichannel image-filtering structures. IEEE Transactions on Image Processing, 1997, 6(7): 1038-1045.

[21]Lukac R. Adaptive vector median filtering. Pattern Recognition Letters, 2003, 24(12): 1889-1899.

[22]Wang G, Li D, Pan W, et al. Modified switching median filter for impulse noise removal. Signal Processing, 2010, 90(12): 3213-3218.

[23]Ma Z, Wu H R, Feng D. Partition-based vector filtering technique for suppression of noise in digital color images. IEEE Transactions on Image Processing, 2006, 15(8): 2324-2342.

[24]Shen Y, Barner K E. Fast adaptive optimization of weighted vector median filters. IEEE Transactions on Signal Processing, 2006, 54(7): 2497-2510.

[25]Novoselac V, Zovko-Cihlar B. Image noise reduction by vector median filter. IEEE ELMAR, 2012 Proceedings, 2012: 57-62.

[26] Viero T, Oistamo K, Neuvo Y. Three-dimensional median-related filters for color image sequence filtering. IEEE Transactions on Circuits and Systems for Video Technology, 1994, 4(2): 129-142, 208-10.

[27] Lukac R. Adaptive color image filtering based on center-weighted vector directional filters. Multidimensional Systems and Signal Processing, 2004, 15(2): 169-196.

[28] Samuel Morillas, Valent Gregori, Guillermo Peris-Fajarne. Isolating impulsive noise pixels in color images by peer group techniques. Computer Vision and Image Understanding, 2008, 110(1): 102-116

[29] Smolka B, Lukac R, Chydzinski A, et al. Fast adaptive similarity based impulsive noise reduction filter. Real Time Imaging, 2003, 9 (4): 261-276.

[30] Boudabous A, Ben Atitallah A, Kadionik P, et al. HW/SW FPGA Implementation of Vector Median Filter. Research in Microelectronics and Electronics Conference, 2007. PRIME 2007. Ph. D. 2007: 101-104.

[31] Anis Boudabous, Lazhar KHRIJI, A. BEN ATITALLAH, et al. Efficient Architecture and Implementation of Vector Median Filter in Co-Design Context. Radioengineering, 2007, 16(3): 113-119

[32] Anis Boudabous, Ahmed Ben Atitallah, Lazhar Khriji, et al. FPGA implementation of vector directional distance filter based on HW/SW environment validation. International Journal of Electronics and Communications, 2011, 65(3): 250-257

[33] Atitallah A B, Boudabous A, Khriji L, et al. Reconfigurable architecture of VDF filter for multidimensional data. International Journal of Circuit Theory and Applications, 2012: 1-12.

[34] Boudabous A, Atitallah A B, Khriji L, et al. HW/SW Design-Based Implementation of Vector Median Rational Hybrid Filter. Int. Arab J. Inf. Technol., 2010, 7(1): 70-74.

发表论文专利和参加科研情况说明

发表的论文：

［1］ Xu Jiangtao，Wang Lei, Shi Zaifeng.“A Switching Weighted Vector Median Filter Based on Edge Detection ”. Signal Processing. SCI（2012 年影响因子 1.851） 已录用

［2］ “一种针对彩色图像传感器信号的降噪电路”.已完成。

发明专利

1. 一种利用边缘检测的开关加权矢量中值滤波方法

参与的科研项目：

1. 本人参与了国家高技术研究与发展计划（“863”计划）资助项目：基于可重构处理器的视觉信息处理系统设计
2. 本人参与了科技部国家国际科技合作专项项目：用于智能终端主控芯片的高端嵌入式 CPU 关键技术引进与联合开发

致　谢

本论文的工作是在我的导师徐江涛副教授的悉心指导下完成的。近三年的研究生学习生活中，经常会遇到各种各样的困难，徐江涛老师每次都能在我最迷茫和困惑的时候给予最热心的指导、支持和鼓励。在此衷心感谢三年来徐江涛老师对我的关心和指导。

史再峰老师悉心指导我们完成了实验室的科研工作，给我提出了许多宝贵的意见。史老师严肃认真、精益求精的科研态度对我今后的工作、学习产生了深远的影响。在此向史再峰老师表示衷心的谢意。

实验室的姚素英老师和高静老师在学习和生活上也给予了无微不至的关怀，在此向她们表达我衷心的感谢。

在实验室科研及撰写论文期间，庞科学姐、李健学姐以及徐艳、郭美菊等同学对我的研究工作给予了热情帮助，在此向他们表达我的感激之情。

另外也感谢我的家人，他们的理解和支持使我能够在学校专心完成我的学业。

基于可重构处理器的视觉信息预处理子系统研究

作者：王磊
学位授予单位：天津大学

引用本文格式：王磊 基于可重构处理器的视觉信息预处理子系统研究[学位论文]硕士 2013