分类号_____ 密级_____

U D C _____ 编号_____

硕士学位论文

论 文 题 目 数字图像并行处理的 VLSI 实现研究

学科、专业 计 算 机 技 术

研究生姓名 谭 会 生

导师姓名及 桂 卫 华 教 授

专业技术职务 朱 晓 青 教授级高工

硕士学位论文

数字图像并行处理的 VLSI 实现研究

作 者 姓 名 ： 谭会生
学 科 专 业 ： 计算机技术
学院（系、所）： 信息科学与工程学院
指 导 教 师 ： 桂卫华 教授

中 南 大 学
2004 年 **11** 月

分类号 **VDC**_____ 密级_____

硕士学位论文

数字图像并行处理的 VLSI 实现研究

VLSI Implementation Research in Digital Image Parallel Processing

作 者 姓 名 ： 谭会生
学 科 专 业 ： 计算机技术
学院（系、所）： 信息科学与工程学院
指 导 教 师 ： 桂卫华 教授
副指导教师： 朱晓青 教授级高工

论文答辩日期_____ 答辩委员会主席_____

中 南 大 学

2004 年 **11** 月

摘　要

提高图像处理的速度一直是图像处理中致力于解决又难以突破的关键问题之一，图像并行处理技术则是提高图像处理速度的最有效的技术。EDA 技术的出现和 VLSI 的发展,为经济快速、自行开发高性能的图像并行处理硬件提供了一个全新的设计平台和广阔的发展空间。

本文在分析了邻域处理的算法及其并行数据结构，邻域图像并行处理机等芯片级图像并行处理器设计技术的基础上，重点阐述了作者利用 EDA 技术完成的 3 个图像并行处理的 VLSI 实现设计实例：Sobel 图像边缘检测器，Laplacian 图像边缘检测器以及整数小波变换的 VLSI 实现设计。各设计实例包括算法介绍，系统的总体设计，主要模块的设计思想,有关仿真结果和分析。

在 Sobel 图像边缘检测器的 VLSI 实现设计中，通过设置 FIFO 栈实现了处理数据的缓冲，通过串入并出模块实现了列像素的刷新，通过设置移位寄存器避免了数据的重复输入和像素处理窗口数据的刷新，通过设计并行流水式滤波器和重复设置四个滤波器来提高整个系统的处理速度，因此本系统实现了邻域数据处理的流水式输入，多层次、全过程的流水线处理和并行处理，具有良好的实时处理性能。对一幅（1024×1024）的图像，本系统（50M 时钟）的处理时间为 0.105s，其处理速度较六级指令流水线的 DSP（40M 晶振）系统提高了 10 倍，较 MCU 系统（12M 晶振）提高了 400 倍。

在 Laplacian 图像边缘检测器的 VLSI 的实现设计中，通过使用移位寄存器存放像素值和使用寄存器实现的数据串并转换，既避免了卷积运算数据的高度重复读取，又实现了卷积运算数据的完全“流水式”输入，同时采用了速度最优的高阶分布式算法完成模板的卷积运算,从而使系统在资源，速度上达到了较好的优化。本系统经过初始的延迟后，每时钟周期就可“流水式”输出一个处理结果，具有良好的实时处理性能。若系统时钟为 100MHZ，则处理一幅 1024*1024 的图像的时间仅需 0.01S 左右。

整数小波变换是一种基于提升格式的小波变换，具有计算快捷，节省内存，能对任意尺寸图像进行小波变换等优点，而且可对图像进行有损和无损压缩。在整数小波变换的 VLSI 的实现设计中，通过自顶向下的设计方法简化了设计难度，通过采用多种形式的数据寄存或锁存组合实现了处理数据的缓冲和小波变换的并行处理和流水线处理，通过数据的移位运算简单而快速地实现了数据的乘除运算及有关取整操作，通过设计多级的控制状态机实现了小波变换的复杂时序控制，通过设计通用的小波分解和合成模块以及小波变换存取控制参数的控制模块实现了小波的多级变换。

关键词　图像并行处理，EDA技术，Sobel图像边缘检测器，Laplacian图像边缘检测器，整数小波变换,VLSI实现设计

ABSTRACT

How to increase processing speed has been one of the key problems in digital image processing, which has been studied by many ways and still been difficult to be broken through. Image parallel processing technology has proved to be one of the most effectual technology for increasing image processing speed. The appearance of EDA technology and the development of VLSI have provided a new design platform and amplitude's development space, so that it is possible to more fast and economically develop the high performance image parallel processing hardware.

The Image parallel processing technology of chip design level are firstly analyzed in this paper, such as neighbor domain image processing algorithm and its parallel data structure, neighbor domain image parallel processor etc. Then three practical application cases of VLSI implementation design, which are Sobel image edge detector, Laplacian image edge detector and integer wavelet transform, are mainly discussed through the combination of algorithm’s introduction, the system’s overall design, the design means of main models and the analysis of simulation .

In the VLSI implementation design of Sobel image edge detector, the buffering of input data and data processing are realized by FIFO stack, the updating of vertical pixels are realized by serial-in and parallel-out modules. Repetitive data input are avoided and pixels_processing_window data’s updating are realized by shift-register circuit. The system processing speed is increased by designing parallel and pipelining filters and repetitively using four filters. Thus the pipelining input of neighbor domain processing data and the parallel processing of varies kinds of level and whole procedure data are realized, good real-time capability is also achieved. If the system working clock is 50 MHz, the detector only needs 0.105s to dispose a image with 1024*1024 pixels. Compared with single DSP(40MHz oscillator) and MCU(12MHz oscillator) application system, the system speed is enhanced by 10 and 400 times for each.

In the VLSI implementation design of Laplacian image edge detector, High repetition read of convolution processing data are avoided and overall stream input of convolution processing data are realized through using shift-register to storage pixels and register to deal with serial-in and parallel-out procedure. At the same time, the system is optimized in efficiency and hardware resources by means of using distributed arithmetic with the highest speed to complete the convolution arithmetic.A stream-like processing
result could come out from each clock period, it has good real-time function. If the system

working clock is 100 MHz, the detector only needs 0.01s to handle a image with 1024*1024 pixels.

Integer wavelet transform, a kind of algorithm based on lifting scheme, which has the advantages of fast computing speed, storage economization, can be freely used to dispose any size of image and realize lossy and lossless compression of image. In the VLSI implementation design of Integer wavelet transform ,the design difficulty is decreased by the means of top-down design. Data buffering, parallel processing and piping processing are realized by varies kinds of data registering or latching. Data's multiplication, division and its rounding computation are realized by shift-calculation .Complex sequential control of wavelet transform is realized by multilevel state machine. Wavelet multilevel transform is realized by designing common decomposition and synthesizing modules and parameter access modules.

KEY WORDS Digital image parallel processing, EDA technology, Sobel image edge detection, Laplacian image edge detection, Integer wavelet transform, VLSI implementation design

目　录

第一章 绪论

1.1 研究背景和意义

数字图像处理起源于20世纪20年代，当时通过海底电缆从英国的伦敦到美国的纽约采用数字压缩技术传输了第一幅数字照片。此后，由于遥感等领域的应用，使数字图像处理技术逐步受到关注并得到相应的发展。1964年，美国的喷气推进实验室处理了由大空船“徘徊者七号”发回的月球照片，标志着第三代计算机问世后数字图像处理开始得到普遍应用。由于CT的发明、应用及获得倍受科技界瞩目的诺贝尔奖，使得数字图像处理大放异彩。其后，数字图像处理技术发展迅速，目前已成为工程学、计算机科学、信息科学、统计学、物理学、化学、生物学、医学甚至社会科学等各个学科之间学习和研究的对象。随着信息高速公路、数字地球概念的提出以及Internet的广泛应用，信息传输中的非话业务也会急剧地增长。其中图像信息以其信息量大、传输速度快、作用距离远等一系列优点使其成为人类获取信息的重要来源及利用信息的重要手段。同时数字图像处理科学又是与国计民生紧密相连的一门应用科学，它已给人们带来巨大的经济和社会效益，不久的将来它不仅在理论上会有深入的发展，在应用上亦是科学研究、社会生产乃至人类生活中不可缺少的强有力的工具。在信息社会中，数字图像处理科学无论是在理论上还是在实践上都存在巨大的潜力[1,2]。

随着计算机、集成电路等技术的飞跃发展，图像处理无论在算法上、系统结构上，还是在应用上以及普及的程度上都取得了长足的发展。但是，图像处理依然面临着许多挑战性的问题，其中最主要的问题就是如何提高解决实际复杂问题的综合能力，就当前的技术水平来说，这种综合能力包括图像处理的网络化、复杂问题的求解与处理速度的高速化[3]。

图像并行处理技术是图像处理中的一个重要方面，是提高图像处理速度的最有效技术[3]，其发展水平一直受到图像界的关注，原因在于：一方面，图像并行处理技术的发展难度很大，这种难度不仅在于图像并行处理系统的硬件及系统处理本身，以及它对计算机技术和集成电路等技术的依赖关系，而且在于实际应用的复杂性和应用部门对系统价格的承受能力；另一方面，图像并行处理技术的发展所产生的效应也是非常显著的，它在处理速度上所获得的加速比是令人振奋的，其实际应用系统也将产生很大的经济效益和社会效益。凡是在图像处理技术应用的地方都可以应用图像并行处理技术，目前由于受多方面原因的制约，其应用领域主要集中在军事、工业自动化以及公安部门的刑事侦察上，同时在这些应用领域强有力的推动下，图像并行处理技术也得到了迅速的发展。

现代电子产品正在以前所未有的革新速度，向着功能多样化、体积最小化、功耗

最低化的方向迅速发展。EDA(电子设计自动化)技术正是为了适应现代电子产品设计的要求，吸收多学科最新成果而形成的一门新技术。利用 EDA 技术进行电子系统的设计，具有以下几个特点：（1）用软件的方式设计硬件；（2）用软件方式设计的系统到硬件系统的转换是由有关的开发软件自动完成的；（3）设计过程中可用有关软件进行各种仿真；（4）系统可现场编程，在线升级；（5）整个系统可集成在一个芯片上，体积小、功耗低、可靠性高；（6）从以前的“组合设计”转向真正的“自由设计”；（7）设计的移植性好，效率高；（8）非常适合分工设计，团体协作。因此，EDA 技术是现代电子设计的发展趋势[4,5,6]。

作为数字信号处理算法的实现有多种途径[7,8,9,10,11,12,13,14,15,16]，传统上多采用高级语言编程在计算机上实现，便于使用的还有在基于专用单片机来实现的（一般称为可编程 DSP 单片机）以及在 VLSI 上实现某种算法的专用集成电路芯片（ASIC）等，近年来，随着 EDA 技术的出现和 VLSI 的发展,为经济快速、自行开发高性能的芯片级数字信号处理硬件提供了一个全新的设计平台和广阔的发展空间，国内外比较流行的是在 FPGA/CPLD 芯片中实现复杂算法的运算处理。

边缘可定义为图像中灰度发生急剧变化的区域边界,它是图像最基本的特征，是图像分析识别前必不可少的环节,是一种重要的图像预处理技术。边缘检测主要就是(图像的)灰度变化的度量、检测和定位,它是图像分析和模式识别的主要特征提取手段，它在计算机视觉、图像分析等应用中起着重要的作用，是图像分析与处理中研究的热点问题[17,18]。

小波是定义在有限间隔而且其平均值为零的一种函数。小波变换通过平移母小波（Mother Wavelet）可获得信号的时间信息，而通过缩放小波的宽度（或者叫做尺度）可获得信号的频率特性。因此小波变换是全局变换，在时域和频域都具有良好的局部化性能，而且在应用中易于考虑人类的视觉特性，从而成为图像压缩编码的主要技术之一，并且离散小波变换已经纳入 MPEG-4 和 JPEG2000 编码标准[19,20]。

本课题“数字图像并行处理的 VLSI 实现研究”，就是以 VLSI 芯片 FPGA/CPLD 为开发设计载体，以 EDA 技术为开发设计平台，以图像预处理中的边缘检测和图像小波变换编码中的小波变换的 VLSI 的实现设计为研究对象，以提高图像处理的速度为目标，以并行算法、并行数据、并行结构、并行处理等并行技术的综合应用为实现手段，以期探讨数字图像并行处理的 VLSI 的实现技术。

1.2 国内外研究现状

本课题所涉及的内容包括图像并行处理技术，EDA 技术，图像的边缘检测，小波变换图像编码技术。下面简单地介绍一下它们的国内外的研究现状。

1.2.1 图像并行处理技术

数字图像处理就是用计算机或实时的硬件进行的数字图像信息的各种处理。数字图像处理的主要内容，大体上可分为如下几个方面：图像信息的获取；图像信息的存取；图像信息的传送；图像信息的处理；图像信息的输出和显示。其中图像信息的处理又主要包括如下几项内容：几何处理，算术处理，图像增强，图像复原，图像重建，图像编码，图像识别，图像理解。数字图像处理的方法大致可分为空域法和变换域法。而空域法又可分为邻域处理法和点处理法[1]。

自 20 世纪 60 年代第三代数字计算机问世以后，数字图像处理技术出现了空前的发展，其形势方兴未艾。在该领域中需进一步研究的问题有如下五个方面：（1）在进一步提高精度的同时着重解决处理速度问题；（2）加强软件开发、开发新的处理方法；（3）加强边缘学科的研究工作，促进图像处理技术的发展；（4）加强理论研究，逐步形成图像处理科学自身的理论体系；（5）时刻注意图像处理领域的标准化问题。图像处理技术未来发展大致可归纳为如下四点：（1）图像处理的发展将向着高速、高分辨率、立体化、多媒体化、智能化和标准化方向发展；（2）图像、图形相结合朝着三维成像或多维成像的方向发展；（3）硬件芯片研究；（4）新理论与新算法研究[1]。

图像并行处理技术是图像处理中的一个重要方面,是提高图像处理速度的最有效的技术[1]。通过多年的发展，图像并行处理技术也确定了它在图像处理中的地位。

根据文献[3, 21]可知，图像并行处理技术研究的内容，可分为并行处理算法、并行数据结构、并行系统结构、并行实现手段等四个方面。

并行处理算法的研究[21]主要包括并行算法的概念，并行设计方法，并行计算模型以及性能评估准则等内容。

并行数据结构的研究[3]就是研究图像数据处理的特点，并行数据结构，视频数据的存储，并行处理数据的输入与输出。

并行系统结构的研究就是研究处理单元与处理单元之间、处理单元与存储体之间的通讯问题。常用的并行结构有四种[3]:（1）环型结构；（2）交叉开关结构；（3）树型结构；（4）ChiP 结构；（5）Systolic 结构。流水线和阵列型图像并行处理，邻域图像并行处理机。

并行处理的实现手段，有软件和硬件实现两种形式，而硬件的实现形式[3, 22, 23, 24, 25]又可分为如下几种：（1）基于计算机的图像并行处理；（2）基于 DSP 的图像并行处理；（3）基于 FPGA/CPLD 的图像并行处理；（4）基于 DSP+FPGA/CPLD 的图像并行处理。其中基于计算机的图像并行处理又可分为并行计算机系统、基于 MMX/SSE 技术的图像并行处理、基于集群计算机系统的图像并行处理等三种形式。基于 DSP 的图像并行处理，则主要广泛应用于手机、语言、家电等领域，并在高速图像处理中得到了越来越

多的应用。而基于 FPGA/CPLD 的图像并行处理和基于 DSP+FPGA/CPLD 的图像并行处理，则是近年新出现的形式，它已在数码静止相机，实时监控系统等方面应用。

1.2.2 EDA 技术

什么叫 EDA 技术？由于它是一门迅速发展的新技术，涉及面广，内容丰富，理解各异，目前尚无统一的看法。文献[4]认为，EDA 技术有狭义的 EDA 技术和广义的 EDA 技术之分。狭义的 EDA 技术，就是指以大规模可编程逻辑器件为设计载体，以硬件描述语言为系统逻辑描述的主要表达方式，以计算机、大规模可编程逻辑器件的开发软件及实验开发系统为设计工具，通过有关的开发软件，自动完成用软件方式设计的电子系统到硬件系统的逻辑编译、逻辑化简、逻辑分割、逻辑综合及优化、逻辑布局布线、逻辑仿真，直至对于特定目标芯片的适配编译、逻辑映射、编程下载等工作，最终形成集成电子系统或专用集成芯片的一门新技术，或称为 IES/ASIC 自动设计技术。广义的 EDA 技术,除了狭义的 EDA 技术外,还包括计算机辅助分析 CAA 技术(如 PSPICE，EWB，MATLAB 等)，印刷电路板计算机辅助设计 PCB-CAD 技术(如 PROTEL，ORCAD 等)。在广义的 EDA 技术中，CAA 技术和 PCB-CAD 技术不具备逻辑综合和逻辑适配的功能，因此它并不能称为真正意义上的 EDA 技术。故将广义的 EDA 技术称为现代电子设计技术更为合适。

对于迅猛发展的 EDA 技术的综合应用，从 EDA 技术的综合应用系统的深度来分，可分为如下几个层次[5]：（1）功能电路模块的设计；（2）算法实现电路模块的设计；（3）片上系统/嵌入式系统/现代 DSP 系统的设计。

根据利用 EDA 技术所开发的产品的最终主要硬件构成来分，作者认为，EDA 技术的应用发展将表现为如下几种形式[5]：（1）CPLD/FPGA 系统：使用 EDA 技术开发 CPLD/FPGA，使自行开发的 CPLD/FPGA 作为电子系统、控制系统、信息处理系统的主体。（2）“CPLD/FPGA+MCU”系统：综合应用 EDA 技术与单片机技术，使自行开发的“CPLD/FPGA+MCU”作为电子系统、控制系统、信息处理系统的主体。(3)“CPLD/FPGA+专用 DSP 处理器”系统：将 EDA 技术与 DSP 专用处理器配合使用，使自行开发的“CPLD/FPGA+专用 DSP 处理器”，构成一个数字信号处理系统的整体。（4）基于 FPGA 实现的现代 DSP 系统：基于 SOPC（a System on a Programmable Chip）技术、EDA 技术与 FPGA 技术实现方式的现代 DSP 系统。（5）基于 FPGA 实现的 SOC 片上系统：使用超大规模的 FPGA 实现的，内含 1 个或数个嵌入式 CPU 或 DSP,能够实现复杂系统功能的单一芯片系统。（6）基于 FPGA 实现的嵌入式系统：使用 CPLD/FPGA 实现的，内含嵌入式处理器，能满足对象系统要求实现特定功能的， 能够嵌入到宿主系统的专用计算机应用系统。

1.2.3 图像边缘检测

图像边缘检测，作为一种重要的图像预处理技术，就是研究更好的边缘检测方法和检测算子。边缘检测的主要方法有：（1）微分算子法；（2）样板匹配法；（3）边界及曲线增强技术；（4）连续小波边缘检测；（5）边缘聚焦；（6）纹理边缘检测；（7）神经网络边缘检测。边缘检测的主要算子[17, 18, 26]有：（1）以各种微分算子为基础，结合用模板及门限、平滑等手段提取边缘的算子：① Roberts 算子 ；② Sobel 算子和 Prewitt 算子；③ Kirsch 算子；④ Laplacian 算子；⑤ Canny 算子；⑥ LOG 算子。（2）以传统微分算子为基础的改进算法：① 基于左右导数算子类的边缘提取；② 基于梯度极值的边缘检测算法；③ 基于样条修匀公式的边缘检测。

1.2.4 小波变换图像编码技术

1909 年，哈尔（Alfrad Haar）在函数空间中寻找一个与傅立叶类似的基时发现了小波，并被命名为哈尔小波（Haar Wavelet），他最早发现和使用了小波。随后，科学家们在小波变换 WT（wavelet transform）的概念、系统的小波分析方法、构造正交小波基、小波变换的算法等方面进行了大量的研究分析，特别是在把小波理论引入到工程应用方面，做出了极其重要的贡献，取得了很大发展。由于小波分析在时域和频域同时具有良好的局部化特性，可以完成一些 Fourier 分析无法解决的信号分析与处理，已成为一种新的、应用性很强的信号分析与处理工具，因此现在小波分析的理论与方法已广泛应用于信号处理，语音分析，模式识别，数据压缩，图像配准，数据融合，数字水印、量子物理等方面[19, 27, 28]。

小波变换有连续小波变换和离散小波变换之分。在小波变换的实际使用中，使用的是离散小波变换的快速算法——Mallat 算法和提升算法。根据所使用的滤波器系数是否为整数，又可分为整数小波变换和浮点数小波变换。小波变换的难点是小波基的选择[19, 29]，小波变换的关键是滤波器的正则性与信号的边界处理[30]。小波分析的主要有效工具——MATLAB 软件[31]。

基于小波变换的图像编码与经典的图像编码方法相比，至少具有如下优点[19,32,33]：（1）小波变换本质上是全局变换，重建图像中可以免除采用分块正交变换编码所固有的“方块效应”。（2）小波变换是采用塔式分解的数据结构，与人眼由粗到精、由全貌到细节的观察习惯相一致，这是将 WT 与 HVS 的空间分解特性结合起来以改善图像压缩性能的有利条件。小波变换比经典的变换（DCT）更符合人的视觉特性，通过合理的量化编码产生的人为噪声比同样比特率的 JPEG 方法产生的影响要小得多。（3）小波变换是图像的时—频表示，具有时间—频域定位能力，并可实现图像中平稳成分

与非平稳成分的分离，从而可对其进行高效编码。

小波变换编码算法中，嵌入式零树小波编码[19，20，32，33，34]、基于塔式网格矢量量化的小波变换编码[19，32，35，36]、基于 LBG 算法的小波变换编码[19，32，37，38]与基于提升算法的小波变换编码[19，32，39，40]等具有代表性。其中嵌入式零树小波编码算法是一个简单而有效的,目前国际上最先进的方法之一,可以在相同的压缩倍数下得到最好的复现图像质量,而且是嵌入式编码,能非常精确地控制压缩倍数[20]。而基于提升方法的小波变换编码，不仅具有计算更快捷，能够在当前位置完成小波变换从而节省内存，能对任意尺寸图像进行小波变换等优点，还可以实现从整数到正数的变换，对变换后的数据进行熵编码就能实现图像的无损压缩[19]。

离散小波变换的实现，一般有软件和硬件实现两种，但用软件方法效率相对较低，对许多实时应用而言（例如监控系统、数码静止相机等等），常用硬件的方式实现。离散小波变换的 VLSI 实现，目前主要集中在 VLSI 的架构研究，主要 VLSI 的架构有：脉动 1D-DWT 实现；双缓冲区乒乓处理的 1D-DWT 变换的 VLSI 结构；2D-DWT 的脉动并行架构；基于输入延迟和 RPA 控制的 2D-DWT 的架构；基于提升策略的实现；可编程 2D-DWT 架构等[41-50]。同时也出现了小波变换 VLSI 实现的工程应用研究[22，23，24，25，51，52]，但总的来将尚处于研究应用的初级阶段，有待于加强其研究。

1.3 作者的主要工作

作者首先对“数字图像并行处理的 VLSI 实现研究”所涉及的图像并行处理技术，EDA 技术，图像的边缘检测，小波变换图像编码技术进行了比较系统的学习和分析，接着剖析了邻域处理的算法及其并行数据结构，邻域图像并行处理机等与芯片级图像并行处理器设计有关的图像并行处理技术，最后重点阐述了作者综合运用图像并行处理技术设计的 3 个用 VLSI 实现的应用实例：Sobel 图像边缘检测器的 VLSI 实现设计，Laplacian 图像边缘检测器的 VLSI 的实现设计，整数小波变换的 VLSI 实现设计。各设计实例包括有关算法介绍，系统的总体设计，主要模块的设计思想,有关设计的仿真结果和分析。

第二章 图像并行处理器的设计分析

2.1 图像并行处理技术基础

图像并行处理技术的基本概念是并行性的概念，而并行处理结构和并行处理算法则是实现并行性的基本方法。在图像并行处理的研究中，从算法到结构的转换是非常重要的，因此，在一个图像并行处理系统中，何处运用并行处理技术以及采用何种并行处理技术，是设计图像并行处理系统最为关键的环节。

2.1.1 基本概念

并行处理是计算机界长期研究的一个重大课题。在计算机系统的体系结构中引入并行性所依据的 3 个基本概念是时间重叠（time interleaving）、资源重复（resource replication）和资源共享（resource sharing）[3]。

时间重叠是指多个处理过程在时间上相互错开，轮流重叠地使用同一套硬件设备的各个部分。这种并行性在原则上不要求重复设置硬件设备，以在同一时刻同时进行多种操作的方式提高处理速度。在实现上，这种并行性在高性能处理机中表现为各种流水线部件或流水线处理机。

资源重复是设置多个相同的设备，同时从事处理工作。这种并行性是以数量取胜的方法来提高处理速度。在实现上，这种并行性在高性能处理机中表现为各种多处理机或多处理器系统。

资源共享具有分时系统的基本特性，即多个用户按照一定的时间顺序轮流使用同一套硬件设备。比如某个用户在执行一种任务，而另一个用户正按照一定的时间划分使用中央处理器，这种在工作时间上的重叠，也可视为并行性的一种形式。资源共享促进了计算机软件中的并行性的发展，也推动了计算机网络和分布式处理系统的发展。

从广义上说，并行性既包括了同时性（simultaneity），又包括了并发性（concurrency），前者是指 2 个或 2 个以上的事件在同一时刻发生，后者是指 2 个或 2 个以上的事件在同一时间间隔内发生。

提高计算机运算速度有两种最基本的方法：一种是采用高速运算部件；另一种是运用并行计算。提高图像处理的速度也是遵循这个基本思路来进行的。

常用的并行处理有两种最基本的连接模式：流水线连接和并行阵列连接，其连接模式如图 2.1 所示。

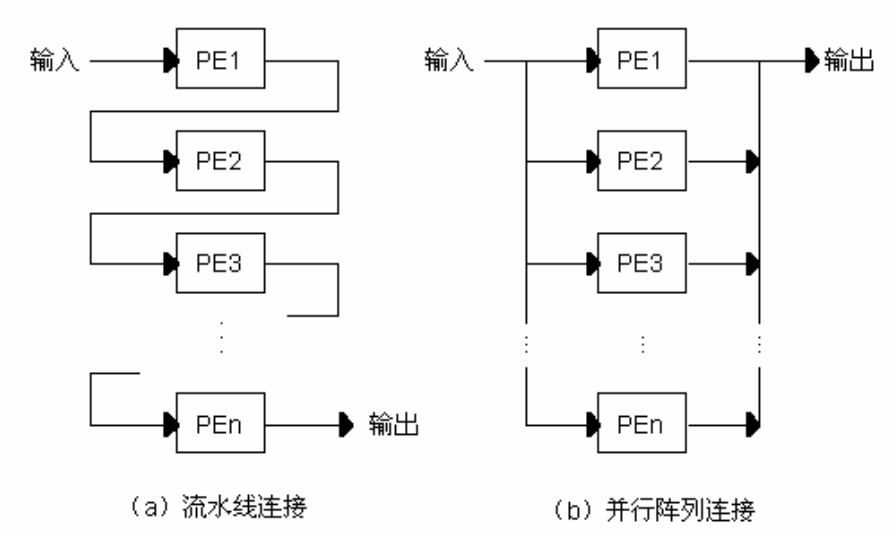


图 2.1　并行处理中的两种最基本的连接模式(PE 表示处理单元)

图 2.1（a）所示的流水线结构里，多种任务在流水线的各级上同时执行，整个任务的速度取决于执行时间最长的子任务的执行时间。图 2.1（b）所示的连接模式是用多个处理单元组成一个并行阵列，每一个处理单元都可以独立执行任务。

在图像并行处理中，有两类并行性形式[3]：（1）流水线并行性；（2）数据并行性。其示例如图 2.2 和图 2.3 所示。

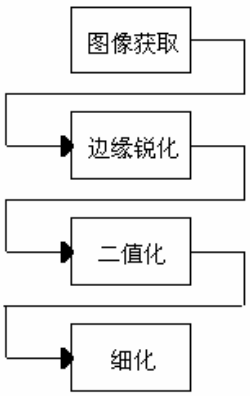


图 2.2　流水线并行性的示例

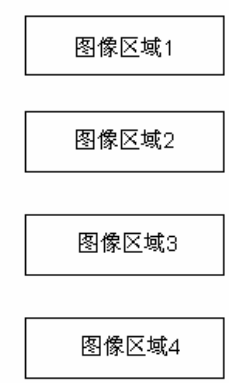


图 2.3　数据并行性的示例

2.1.2　并行结构

并行结构需要解决处理单元与处理单元之间、处理单元与存储体之间的通讯问题。好的并行结构能够充分发挥并行处理的优势，取得接近于 N（处理单元数）倍单个处理器的速度。相反，其速度有可能降至单个处理器的水平。常用的并行结构有四种[3]：(1) 环型结构；(2) 交叉开关结构；(3) 树型结构；(4) ChiP 结构；(5) Systolic 结构。其具体结构如图 2.4～2.8 所示。

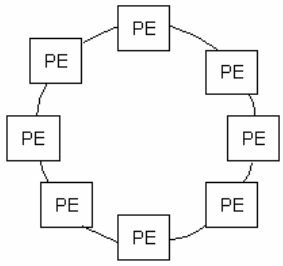


图 2.4　环型结构

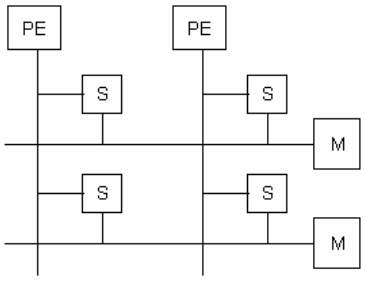


图 2.5 交叉开关结构

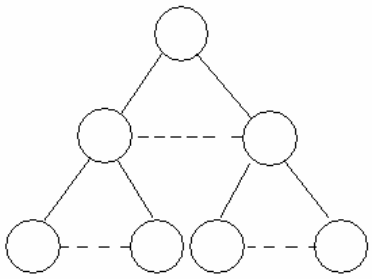


图 2.6　树型结构

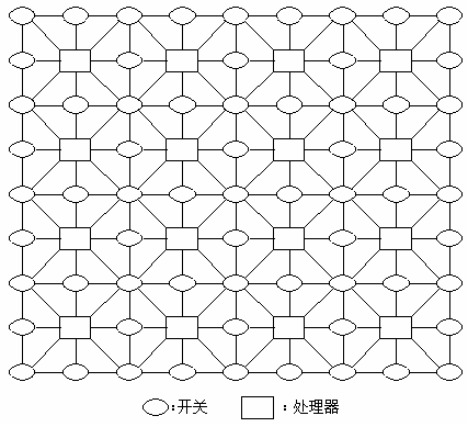


图 2.7　CHIP 结构

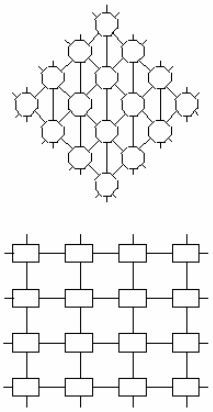


图 2.8　二维 Systolic 结构

2.1.3　并行算法

一个高效的并行处理系统，一般由 3 个部分组成：并行机系统结构、并行软件和并行算法。在给定并行机和软件任务要求的情况下，并行算法将是提高系统效率的一个决定性的因素。例如，一台进行 100%向量运算的向量计算机，运算速度为 1 亿次/s；若算法中向量运算占 75%，则速度会降为 1400 万次/s；如果所有运算全是标量运算，则运算速度只能达到 400 万次/s。可见并行算法对运算速度有较大的影响。

下面将主要讨论并行算法及其设计所涉及的一些基本概念，包括它的设计方法、并行计算模型和性能评估准则等[21]。

并行算法是一些可同时执行的诸进程的集合，这些进程相互作用并协调动作，从而达到对给定问题求解。并行算法与串行算法的不同点在于，它力求经空间复杂度换取时间复杂度。

并行算法有多种分类方法。依据基本运算对象的不同分为数值并行算法和非数值并行算法；依据并行进程间执行顺序关系的不同分为同步并行算法、异步并行算法、独立并行算法；依据各处理机承担的任务粒度的不同分为细粒并行算法、中粒度并行算法、粗粒度并行算法。

利用并行处理机系统求解一个给定的问题，需要根据系统的类型和特征设计并行算法。并行算法设计通常有三种途径：(1)检测和开发现有串行算法中固有并行性而

直接将其并行化；(2)从问题本身的特征出发，设计一个新的并行算法；(3)修改已有的并行算法使其可求解另一类相似问题。其中是从问题目本身的特征出发，设计一个新的并行算法，是并行算法设计的一种重要途径。目前普遍使用的并行算法设计技术有流水线技术、分治策略、平衡树方法、倍增技术以及加速级连策略等。其中分治策略是公认的设计并行算法的根本准则，它主要有数据分割和功能分割两种分割方法；倍增技术是通信算法(如全局通信，一对多通信等)的并行算法设计中常采用的方法。

并行计算模型是对各种并行机(至少某一类并行机)的特征的一种抽取，它使得设计的并行算法具有更强的生命力。迄今为止，已有多种并行计算模型存在，如 PRAM 模型、LogP 模型、C^3模型、BDM 模型等，但其中的每一种只抽象了实际并行机的一个或几个方面尚无一处适用于所有并行机。其中 PRAM 模型是对一类共享存储并行机的特征提取，假设了网络带宽无限，可直接开发原始算法内在细粒度并行。LogP 模型是对一类分布式并行计算机的特征提取，基于点对点通信的计算模型，集中分析了处理机与网络之间的瓶颈。C^3模型是对一类基于消息传递的分布式粗粒度系统的特征提取，集中反映的是网络拥挤和路由影响。BDM 模型是共享存储编程模式与消息传递的分布式存储系统之间的一个桥梁模型，反映的是存储系统中流水线预取等方面的影响。

常用的评价并行算法性能的指标有加速比、效率、可扩展性等。其中加速比定义为最优先串行算法的执行时间与 P 台处理机并行执行时间的比值，效率定义为加速比与 P 的比值，度量算法可扩展性的恒等效率函数定义为$T_1 = f_E(P)$。通常情况下并行算法的设计追求高加速比、效率和强可扩展性。

Amda*HL* 加速比模型与 Gustafson 加速比模型分别是固定问题规模和可变问题规模的并行加速比模型。Amda*HL* 定理认为固定问题规模中并行算法的并行成份固定，因而并行算法的最大加速比不超过串行成份的倒数；Gustafson 定理则认为并行算法中的并行成份随处理机数的增大而增大，因而为获得高的加速比，必须随处理机的增多而增加问题规模。

2.1.4 性能指标

在并行处理中，如何评价并行处理机的性能是一个重要的问题。下面给出几个常用的衡量并行处理性能的参数[3]。

1．并行处理机（器）个数

这是并行处理结构的一个重要参数，当并行处理机（器）个数达不到解决问题的规模时，往往需要采用分块算法。

2．加速比

加速比的定义为 $S_P = T_1 / T_P$ （2.1）

其中T_1为已知最快串行算法在单处理机上的运行时间，T_P为对同一问题用并行算法在

P 台并行处理机上的运行时间。

3．并行处理效率

并行处理效率定义为 $E_P = S_P / P$ （2.2）

其中 P 为并行处理机台数，S_P 为并行处理加速比。

目前图像处理系统尚没有一套完整的性能评价方法。对图像并行处理系统的处理速度有两种主要的评价方法：

1．算法类别及其处理速度

这类指标是最基本的速度指标。算法类别表明该系统能完成何种处理，如图像的加减、直方图统计、Roberts、Sobel、卷积等，卷积还有卷积核的大小这一参数。处理速度是指在确定的图像区域内完成该算法所需的时间。

2．包含图像输入输出的算法类别及其处理速度

这类指标是考核系统实际应用的能力。一种图像加速卡的运行过程是计算机将待处理的数据送入图像加速卡，图像加速卡处理后的结果又返回计算机，考核这种系统的处理速度就应该包含从计算机传送数据开始到确定区间的图像处理完毕并全部送会计算机为止的全过程。

在上述的 2 种评价方法中，显然第 2 种评价方法更能反映图像并行处理系统的综合能力。

对图像并行处理系统性能的评价除了速度这一重要指标之外，还有系统采用的并行处理方式、加速比、主机同步调用能力以及性能价格比。

2.2 图像处理的特征分析

图像处理的算法具有复杂性和多样性。要想提高图像处理的速度，就应该对图像处理的特征和算法的共性有一个概括的了解。图像处理具有六个重要的特征，即图像处理的一致性、分层性、邻域性、行顺序性、并行性、实时性[3]。

1.一致性

图像处理的一致性是指对图像区域里的每一点的处理是按相同规则进行的。以图像求反为例，处于确定区域里的每一个像素都要进行求反运算。这里讲“确定区域”而不统称整幅图像，原因在于偶尔也会出现在整幅图像的不同区域实施不同算法的情况。

2．分层性

图像处理的分层性是指归于许多问题的求解，常常不是一种算法就能够解决的，而要依次使用多种不同的算法，这些算法具有顺序性，而且上一步的处理结果会直接影响下一步的处理。根据所涉及数据的性质，可以把这些算法划分成 3 个处理层次：数据层、信息层及知识层。对一幅图像进行处理，首先是进行原始数据处理，这种处

理属于底层处理，从待处理的数据来讲，这时的数据量大；经过底层处理以后，下一个处理层是信息处理，即提取所需要的信息并对该信息进行处理，如提取物体边界以及计算面积、周长、中心等等；知识层处理则是更高一层的处理，如基于知识的图像查询、文字识别、指纹识别、人面像识别等等。对一幅图像先后进行 3 个层次的处理，其表征该幅图像的数据量将随着处理层次的提升而降低，也就是说，处理层次越高，表征该幅图像的数据量就越少。

3．邻域性

在图像数据层处理里，以图像处理运算所涉及像素区域的区别，根据算法的共性，大体上可以把图像处理划分为点处理、邻域处理、几何变换处理。

点处理是对单一像素进行处理，可以是一帧图像内的点处理，也可以是两个不同帧图像对应像素点的点对点处理，这些处理不需要加入该像素的相邻像素就可以完成。

邻域处理是在对单一像素进行处理时，需要该像素的相邻像素参加运算才能够完成，这一类邻域处理的算法有 Roberts（2×2 邻域）、Sobel（3×3 邻域）等等。邻域处理的另一种形式是该算法必须要求一个数据块参加运算，例如按 8×8 数据块进行的 JPEG 图像压缩，相邻两次处理采用不同的 8×8 数据块。所谓图像处理的邻域性，一般是指图像处理的许多算法属于邻域处理，就是说，许多算法需要邻域数据。

几何变换处理一般是指图像的放大、缩小、旋转、平移等处理，这类处理往往是被处理的像素在处理后的地址发生变化，放大和缩小处理还在像素数量上发生了变化。

4．行顺序性

行顺序性是由电视扫描方式引起的，电视扫描是按照从上到下、从左到右的规律进行的，图像数字化也按这一规律进行，第 1 行第 1 点、第 2 点…，下一行的第 1 点、第 2 点…，直至一帧图像数字化完成。视频数据流具有行顺序性，图像的多种数据格式也具有行顺序性，这种行顺序性正是流水线处理的出发点。

5．并行性

一般说来，数据层的各种算法具有高度的并行性，这是两维的并行处理，采用对数据按顺序串行执行一系列指令的冯·诺依曼结构是完全不适合的。针对两维数据的并行处理，可以采用不同的并行结构，既可以对邻域的像素做并行处理，也可以在更大的区域作并行的点处理。这些并行处理或采用流水线结构，或采用并行阵列连接，其特点是与像素的局部地址无关。可以说，并行处理在数据层的处理中可以大显身手，其效果（在算法类别及其处理速度上）是显著的，甚至是激动人心的，并可作为衡量各种并行处理的结构和各种硬件处理系统性能的指标。

6．实时性

实时性的一个含义是指某些过程和图像信源在时间具有一致性，亦在时间上是确

切的，可称为时域实时性。实时性的另一个含义是指任务实时，亦即在任务上是确切的，可称为空域实时性或任务实时性。

2.3　邻域处理算法及其数据结构

根据所涉及数据的性质，图像处理的算法可划分成 3 个处理层次[3]：数据层、信息层及知识层，其中数据层的处理既是图像处理的基础，同时也是处理数据量最大的。而在数据层的处理中，图像处理可划分为点处理、邻域处理、几何变换处理，邻域处理算法则是使用最多的。下面我们就介绍一下邻域处理的算法及其数据结构[3]。

按照被处理的图像数据是否被重复使用的观点来划分，图像处理中的邻域处理算法可分为两类：一类算法是重复使用图像数据的，大多数邻域图像处理算法属于这一类算法；另一类算法是不重复使用图像数据的，只有比较少的邻域图像处理算法属于这一类算法，如 8×8 的 JPEG 算法。下面介绍 5 种邻域处理算法。

1.Roberts 算子

Roberts 算子用于边缘增强。绝对值输出的 Roberts 算子的数学表达式为

$$G(x,y)=\left|f(x,y)-f(x+1,\ y+1)\right|+\left|f(x+1,\ y)-f(x,\ y+1)\right| \qquad (2.3)$$

式中 $f(x,y)$代表原始图像，$G(x,y)$代表图像 $f(x,y)$在(x,y)点的梯度。

在图像处理中，常常要对式（2.3）的$G(x,y)$进行二值化处理，其数学表达式为

$$C(x,y)=\begin{cases}L_G, & G(x,y)\geq T\\ L_B, & G(x,y)\angle T\end{cases} \qquad (2.4)$$

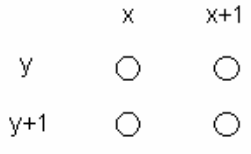
式中$C(x,y)$代表二值图像，$G(x,y)$代表图像 $f(x,y)$的 Roberts 梯度，L_G、L_B为常数，一般取L_G=255，L_B=0，T为阀值。Roberts 算子的数据结构如图 2.9 所示，显然这是一个 2×2 的数据结构。

2. 3×3 卷积

3×3 卷积常用于图像滤波，其表达式为

$$g(x,y)=\sum_{i=-1}^{1}\sum_{j=-1}^{1}H(j,i)\times f(x+j,\ y=i) \qquad (2.5)$$

式中$H(j,i)$为卷积模板，也称为卷积核，其数据结构如图 2.10 所示。3×3 卷积的图像数据结构如图 2.11 所示，显然这是一个 3×3 的数据结构。



H (-1, -1)	H (0, -1)	H (1, -1)
H (-1, 0)	H (0, 0)	H (1, 0)
H (-1, 1)	H (0, 1)	H (1, 1)

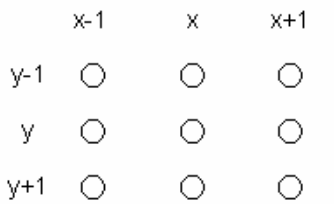


图 2.9　2×2 的数据结构　　图 2.10　3×3 卷积模板　　图 2.11　3×3 数据结构

3. Sobel 算子

Sobel 算子用于边缘增强，该算子是测量沿两个垂直方向的灰度差，然后在把这些测量值组合起来形成边缘强度。如像素处得到的响应分别为$G_x(x,y)$、$G_y(x,y)$，则

$$G_x(x,y)=\sum_{i=-1}^{1}\sum_{j=-1}^{1}H_x(j,i)\times f(x+j,\ y+i) \tag{2.6}$$

$$G_y(x,y)=\sum_{i=-1}^{1}\sum_{j=-1}^{1}H_y(j,i)\times f(x+j,\ y+i) \tag{2.7}$$

和图 2.11 中的 3×3 卷积模板的表示方法类似，式中 x 方向的卷积模板和 y 方向的卷积模板分别如图 2.12 和图 2.13 所示。

-1	0	1
-2	0	2
-1	0	1

图 2.12　x 方向的卷积模板

-1	-2	-1
0	0	0
1	2	1

图 2.13　y 方向的卷积模板

平方根输出的 Sobel 算子数学表达式为

$$R=\left[G_x^2(x,y)+G_y^2(x,y)\right]^{\frac{1}{2}} \tag{2.8}$$

绝对值输出的 Sobel 算子数学表达式为

$$R=\left|G_x(x,y)\right|+\left|G_y(x,y)\right| \tag{2.9}$$

Sobel 算子的数据结构也是 3×3 的数据结构，如图 2.11 所示。

4．3×3 十字中值滤波

3×3 十字中值滤波有多种形式，最常用的 3×3 十字中值滤波是对 5 个相邻像素进行排序，以确定中心点的数值，其数据结构如图 2.14 所示。

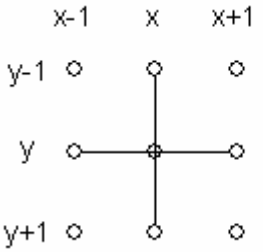


图 2.14　3×3 十字中值滤波数据结构

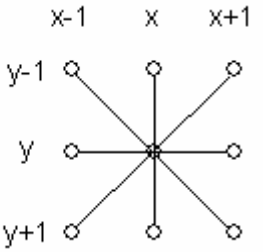


图 2.15　3×3 邻域里 4 方向中值滤波数据结构

图 2.14 所示的 3×3 十字中值滤波的数学表达式为

$$G'(x,y)=\mathrm{med}\left[g(x,y-1),g(x-1,y),g(x+1,y),g(x,y+1)\right] \tag{2.10}$$

在 3×3 邻域里 4 方向的中值滤波的数据结构如图 2.15 所示，其数学表达式为

$$G'(x,y)=\mathrm{med}\left[G_1(x,y),G_2(x,y),G_3(x,y),G_4(x,y)\right] \tag{2.11}$$

式中

$$G_1(x,y)=\mathrm{med}[g(x-1,y),g(x,y),g(x+1,y)]$$
$$G_2(x,y)=\mathrm{med}[g(x-1,y-1),g(x,y),g(x+1,y+1)]$$
$$G_3(x,y)=\mathrm{med}[g(x,y-1),g(x,y),g(x,y+1)]$$
$$G_4(x,y)=\mathrm{med}[g(x+1,y-1),g(x,y),g(x-1,y+1)]$$

5．JPEG 静止图像压缩

JPEG 标准中定义了有失真压缩算法和无失真压缩算法两种。有失真压缩算法又称不可逆编码算法，以离散余弦变换 DCT（discrete cosine transform）为基础，实现对源图像各个部分的变换压缩。无失真压缩算法是以差分脉冲编码调制 DPCM（differential pulse coding modulation）为基础的预测算法，它是一种完全复原的编码方式，恢复后的图像与源图像比较没有失真引入。JPEG 的数据结构是 8×8 的数据结构。

不论是重复使用图像数据的邻域处理算法还是不重复使用图像数据的邻域处理算法，其算法要求的整个邻域数据均称为并行的邻域处理算法的数据结构，如 Roberts 算子的 2×2 数据结构，Sobel 算子的 3×3 数据结构，JPEG 静态图像压缩的 8×8 数据结构等。常用的并行算法数据结构如图 2.16 所示。

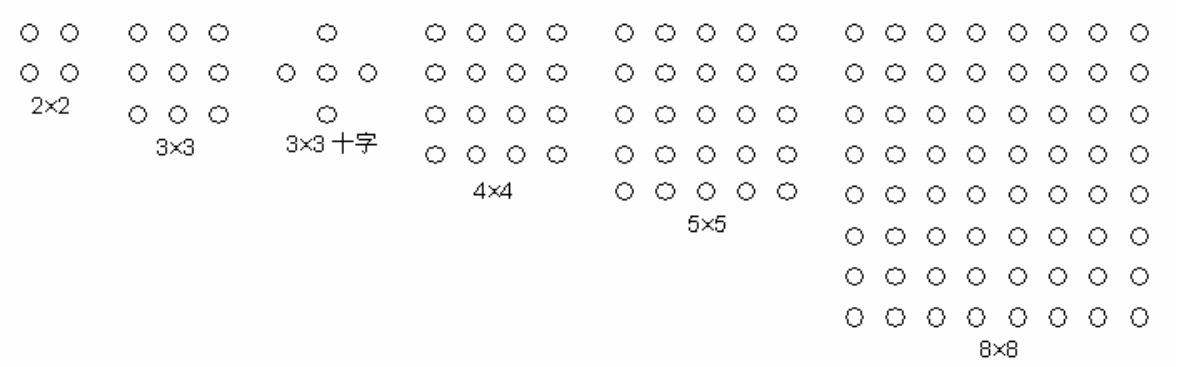


图 2.16　常用的并行算法数据结构

在邻域处理算法中，并行算法要优于串行算法，并行的数据结构也一定优于串行的数据结构。这种优势非常明显，例如执行式（2.5）的 3×3 卷积，串行执行时使用的是单点数据结构，需要按顺序读出 9 个像素并独立执行 9 次乘加运算。使用并行的数据结构，可以在一个点时钟周期内并行地执行 3×3 卷积。

对于邻域图像处理算法，从图像并行处理的数据吞吐量这一指标来考虑，同样是在一个时钟周期内，通过的数据量越多，则系统的处理能力越强。邻域图像处理算法所包含的像素邻域大小不同，显然，处理大邻域的比处理小邻域的系统能力更强，系统能处理多个邻域，则比处理单个邻域的并行性更高。

2.4　邻域图像并行处理机

2.4.1　基本原理与系统结构

要想提高图像处理的速度，就一定要寻找适应并行算法数据结构的系统结构。邻域图像并行处理机正是通盘考虑了并行算法的数据结构、系统的存储结构、系统的处理结构之间的有机联系,从而实现了并行的算法数据结构、并行存取的存储数据结构、并行处理的数据结构三者同一。邻域图像并行处理机的基本原理[3]是数据并行、处理并行，系统的数据结构采用具有并行性、同一性的数据结构。邻域图像并行处理机的并行性、同一性的数据结构简称为 3P（parallel）+3E（equal）数据结构，其示意图如图 2.17 所示。

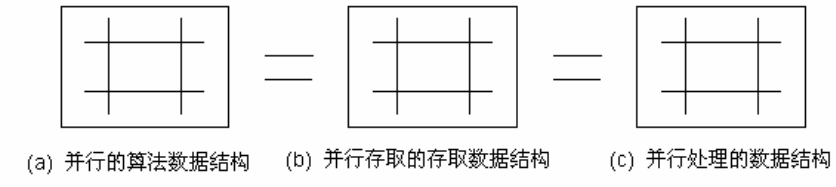


图 2.17　邻域图像并行处理机具有并行性、同一性的数据结构

我们知道，如果处理器能力很强，而数据供应不上，则不能充分发挥处理器的作用；反之，如果数据是高度并行的，而处理器的能力很有限，不能及时地处理这些数据，则同样达不到高速图像处理的目的。当然，这些数据应当是算法所要求的数据。图 2.17 所示的数据结构，强调了数据的并行性和同一性，这种并行性的意义在于高速的数据并行，而这种同一性体现在存储结构和处理结构对算法的数据结构同一。建筑在并行性、同一性数据结构基础上的邻域图像并行处理机处理的核心部分是邻域图像帧存储体和邻域图像并行处理器，其设计思想就是邻域图像帧存储体提供并行的结构化的图像数据，邻域图像并行处理器则高速地对并行的结构化的图像数据进行同一结构的并行处理。邻域图像并行处理机并行处理的本质是多指令、多数据的并行处理。我们所说的图像邻域，就是指一个数据的结合，可以执行同一种算法，也可以执行不同的多种算法。图 2.18 给处了邻域图像并行处理机的系统结构框图[3]。

图 2.18 所示的是一个包括图像输入、输出在内的完整的系统结构框图。在算法的分配上，图像处理硬件完成数据层和某些信息层的处理，计算机软件则完成另一些信息层和知识层的处理。在系统结构上，既采用流水线结构，也采用并行阵列结构。图 2.18 中含有一个处理节点（如图中虚线框所示），每一个节点包括一个邻域帧存储体组（图 2.18 中帧存为帧存储体的简称）和 N 组邻域处理器阵列；每一组邻域处理器阵列由多个邻域处理器构成，完成一个算法或两个算法；N 组邻域处理器完成多个算法，其处理结果可以送入主机，也可以通过级联，把这一个处理节点的处理结果送入下一个处理节点。每一个处理节点可以进行行顺序邻域处理，也可以进行随机邻域

处理。在随机邻域处理结构里，邻域图像帧存储体并行读出邻域图像数据，所形成的邻域图像数据在时间上没有行延迟和点延迟，系统可按帧频速率进行流水线处理；在行顺序邻域处理结构里，邻域图像帧存储体并行读出一列多个相邻行的数据，所形成的邻域图像数据在时间上没有行延迟，只有点延迟，系统可按点频进行流水线处理。在处理节点之间的级联可以有多种选择，如行顺序邻域处理与行顺序邻域处理节点之间的级联等、行顺序邻域处理与随机邻域处理节点之间的级联等。

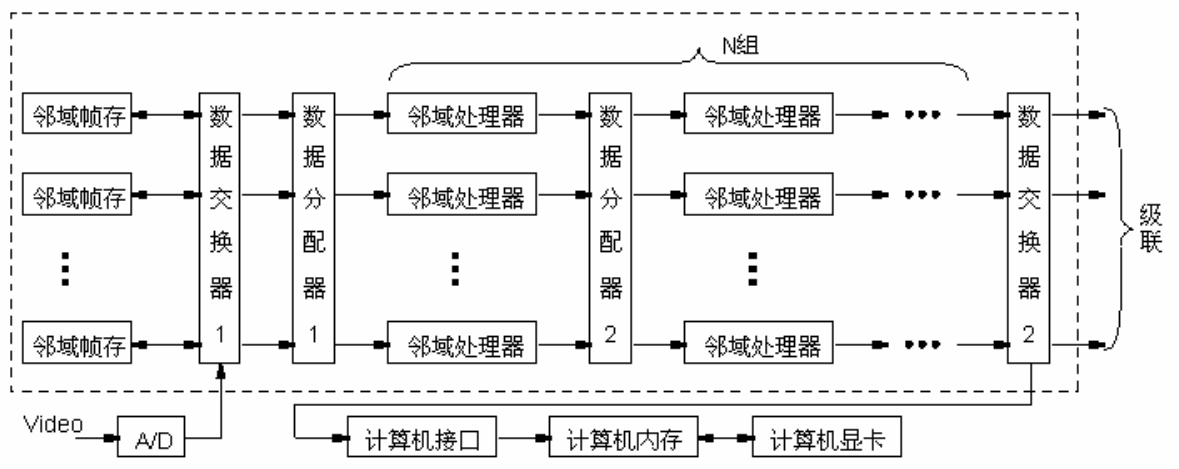


图 2.18　　邻域图像并行处理机的系统结构框图

在每一个处理节点内，至少包括一个邻域图像帧存储体，以便提供结构化的邻域图像数据。在实际系统中，经常设置 A、B 两个邻域图像帧存储体，当输入图像正在存入邻域图像帧存储体 A 时，则读出邻域图像帧存储体 B 的图像，反之亦然，这样便于动态地进行图像处理。当进行多通道视频图像处理时，则要设置更多的邻域图像帧存储体。数据交换器 1 的作用是为邻域图像帧存储体提供输入输出数据通道，一组邻域处理器所包含的邻域处理器的数量取决于算法的难易程度，但必须能独立完成一个完整的算法。数据分配器负责对一组邻域处理器内的邻域处理器进行数据组织，数据交换器 2 把处理结果送入主机或送入下一个处理节点。

图 2.18 所示的邻域图像并行处理机的图像显示由计算机终端完成，它是一种面向计算机内存的系统操作方式。如果增设一个 D/A 电路，把图 2.18 中数据交换器 2 的输出改送到 D/A 上去，再用监视器来显示图像，则可构成一个面向帧存储体的系统。

2.4.2　邻域图像帧存储体的实现

依据上述的邻域图像帧存储体的基本原理来设计一种实现邻域图像并行存取的邻域图像帧存储体，其特征在于：在邻域图像帧存储体的一个读周期里，能够并行地读出行顺序的 M×N 邻域图像数据，或能够并行地读出 M×N 随机邻域图像数据。邻域图像帧存储体首先是一个图像帧存储体，它需要存储器芯片，其存储容量至少可以达到存储一帧视频图像的要求，它还可以存储摄像机送来的经数字化的视频图像，除此

之外，邻域图像帧存储体特别之处还在于它能够实现领域图像的并行存取。邻域图像帧存储体[3]由存储器芯片、扫描时序发生器、视频地址发生器、地址变换器、存储器时序发生器、数据排序电路、数据锁存器、数据变换中心组成，图 2.19、图 2.20 所示邻域图像帧存储体的框图说明了邻域图像帧存储体的各个电路之间的联系。图 2.19 是实现行顺序的 M×N 邻域图像并行存取的邻域图像帧存储体框图。图 2.20 是实现 M×N 随机邻域图像并行存取的邻域图像帧存储体框图。

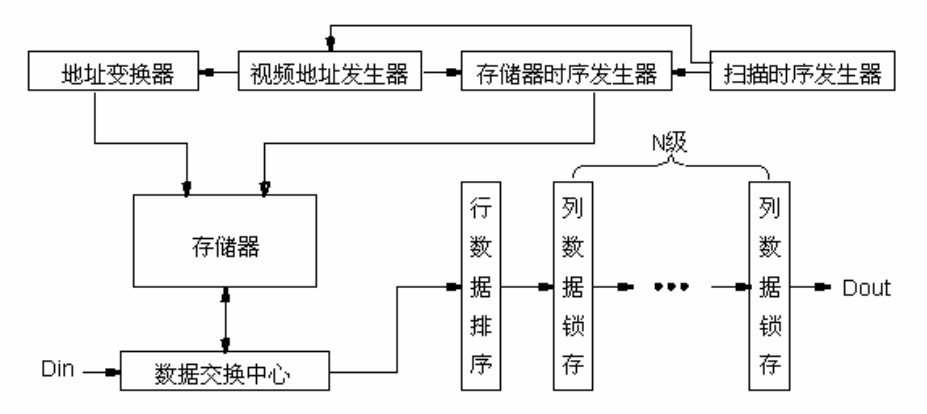


图 2.19　行顺序的 M×N 邻域图像并行存取的邻域图像帧存储体框图

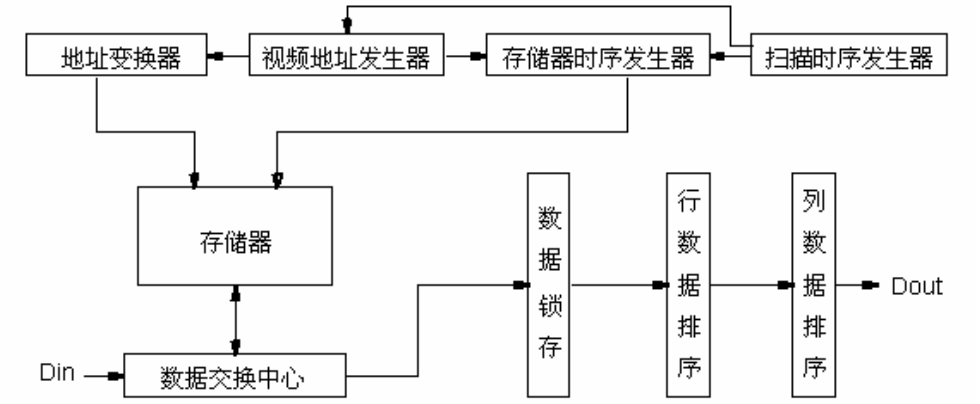


图 2.20　　M×N 随机邻域图像并行存取的邻域图像帧存储体框图

2.4.3　邻域图像并行处理器

由于邻域图像帧存能够提供高速的结构化数据，因此处理器面临着如何适应大吞吐量的结构化数据的问题。如果处理器不充分利用这种结构化的数据，而是串行地进行单数据的处理，其结果显然达不到高速图像处理的目的。然而及时地处理邻域图像帧存储体提供的结构化数据也并非易事，需要从处理器结构上以及处理器能力上来考虑，以期实现处理结构与存储结构的同一，从而达到高速图像处理的目的。

1．邻域图像并行处理机的流水线结构[3]

针对邻域图像帧存储体提供结构化数据这一新特点，构造邻域功能流水线来适应

数据的高吞吐量，同时以处理器的并行阵列连接来适应结构化数据的处理。邻域功能流水线的结构如图 2.21 所示。

邻域功能流水线不同于一般的流水线，后者的数据流是串行的，邻域功能流水线的数据流则是并行的、结构化的。流水线 CLK 可以是视频点时钟，也可以引申为帧频的节拍信号。

邻域功能流水线的处理速度与流水线的节拍时钟有关，同时与功能处理器组的级数有关，前者决定了完成一幅 W×H 点阵图像的处理时间，后者则决定该流水线的最大算法能力。如果在邻域功能流水线里有 N 级功能处理器，完成一幅 W×H 点阵图像的处理时间为 T_{PP}，则每一个功能处理器等效平均耗时 T_P 为

$$T_P=T_{PP}/N \qquad (2.12)$$

邻域功能流水线具有以下 4 个特点：（1）邻域功能流水线的数据流具有邻域性，且各级邻域尺寸可变；（2）邻域功能流水线具有多级功能块；（3）邻域功能流水线的功能是动态可变的；（4）邻域功能流水线能够实现行顺序邻域和随机邻域的邻域图像并行处理。

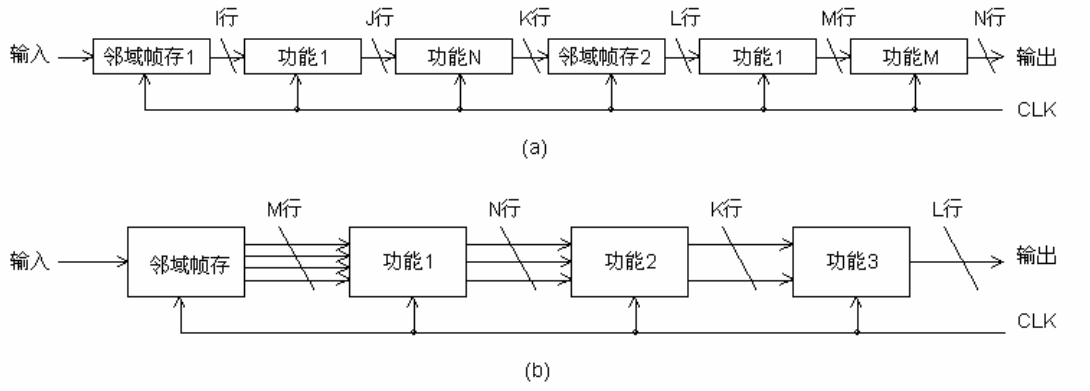


图 2.21　邻域图像并行处理机的两种邻域功能流水线结构

(a)级联型(两级 I≥J≥K,L≥M≥N)　(b)收缩型(三级 M≥N≥K≥L)

邻域图像处理器应具有高速点处理和邻域处理的能力。对于点处理的算法，可以并行进行数据块的点处理。设邻域帧存读周期为 T_0，一次读出的块数据为 N 个像素并即时完成处理，则一幅 W×H 点阵图像进行点处理的时间为

$$T_{PP}= T_0\times W\times H/N+T_{MA} \qquad (2.13)$$

式中 T_{MA} 为帧存额外开销时间，如采用 VRAM 芯片的串口时串出方式的传输时间。

以采用 15ns 的 SRAM 存储芯片为例，如果以一个读周期为读出 4 点并进行点处理的时间，处理 512×512 点阵图像的时间约为 0.98ms，这种速度是实时处理 40 倍。

对于邻域处理类型的算法，由于对每一个像素的处理都要对其邻域数据进行处理，其计算量比点处理要大。设邻域帧存读周期为 T_0，一次读出邻域的 N 个像素并即时完成处理，则一幅 W×H 点阵图像进行处理的时间为

$$T_{PP}= T_0\times W\times H+T_{MA} \qquad (2.14)$$

式中 T_{MA} 为帧存额外开销时间。

2．邻域图像并行处理机的多处理连接[3]

单处理器把难以满足复杂处理的高速处理，为了使每一组处理器达到流水线节拍的处理能力，往往要设置多个处理器；即使单处理器可以达到流水线节拍的处理能力，为了达到更高的处理速度，也需要设置多个处理器。设置多个处理器的方法有多种，图 2.22 给出了邻域处理双处理器横向连接的示意图，图 2.23 给出了邻域处理双处理器纵向连接的示意图。

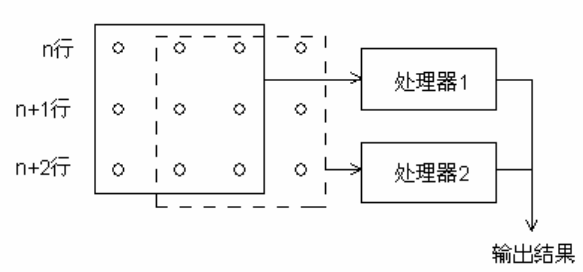
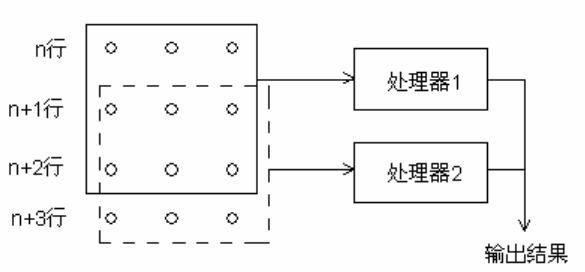


图 2.22 3×3 邻域处理双处理器横向连接图



2.23　3×3 邻域处理双处理器纵向连接

在图 2.22 中，双处理器的连接是横向的，在一个处理节拍里，可同时处理相同行的两点，这种连接主要用于单处理器的处理能力达不到流水线节拍处理能力的情况。在图 2.23 中，双处理器的连接是纵向的，在一个处理节拍里，可同时处理相同列的两点，这种连接主要用于单处理器的处理能力已能达到流水线节拍的处理能力，但为了达到更高处理速度的情况。结合图 2.22 和图 2.23 的连接方式，可以构成纵向、横纵向连接的 4 处理器的连接方式。图 2.22 和图 2.23 只给出了邻域处理的连接情况，对于点处理的多处理器连接，则数据分配和地址变化有所不同。在点处理中，以双处理器横向连接来处理 3×3 邻域，则邻域数据为 6×3（6 列、3 行），以双处理器纵向连接来处理 3×3 邻域，则邻域数据为 3×6（3 列、6 行）。显然，连接方式不同（横向连接、纵向连接和纵向、横纵向连接）、处理的算法不同（点处理或邻域处理），数据的组织也不同，因此需要一个数据分配器来进行数据组织。图 2.24 给出了邻域图像处理器结构的示意图。

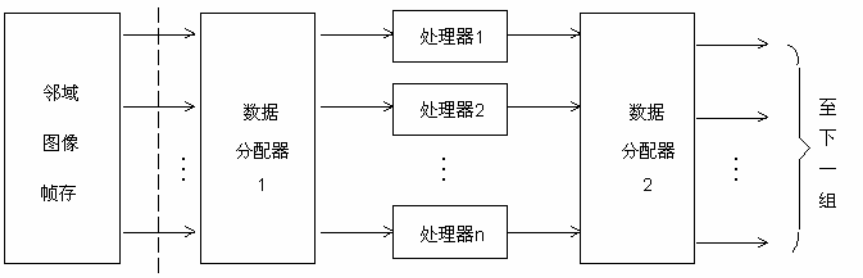


图 2.24　邻域图像处理器结构的示意图

图 2.24 中，虚线左边是邻域图像帧存储体，提供邻域图像数据；右边是邻域图像

处理器的并行结构，显然，这是并行阵列连接的并行结构。在行顺序的邻域图像处理里，采用流水线并行处理，每个流水线节拍都从邻域帧存里读出 1 列多个相邻行的邻域数据,数据分配器 1 根据算法的要求形成 n 个邻域数据并交给 N 个处理器进行处理，由此达到 n 倍的加速。确定处理器个数 N 的依据是单处理器处理该邻域所花费的时间及所要求的处理速率。令行顺序邻域每像素处理节拍为 T_0的硬件处理器个数 N 为

$$N \geqslant T_{PE}/T_0 \tag{2.15}$$

数据分配器 2 的作用是组织前一组邻域处理器的处理结果数据，在行顺序的邻域图像处理里，当多处理器采用横向连接时，数据分配器 2 要进行并到串的转换；而多处理器采用纵向连接时，数据分配器 2 可能直接输出数据，也可能重新组织数据，再送入下一组邻域处理器。

对于随机邻域图像处理，邻域图像帧存储体则需要读出 M×N 个邻域图像数据，再由数据分配器 1 进行数据分配，送给邻域处理器进行并行处理。

第三章　Sobel 图像边缘检测器的 VLSI 实现设计

3.1　Sobel 边缘检测算子

边缘可定义为图像中灰度发生急剧变化的区域边界,它是图像最基本的特征，是图像分析识别前必不可少的环节,是一种重要的图像预处理技术。边缘检测主要就是(图像的)灰度变化的度量、检测和定位,它是图像分析和模式识别的主要特征提取手段，它在计算机视觉、图像分析等应用中起着重要的作用，是图像分析与处理中研究的热点问题。

在过去的 20 年里产生了许多边缘检测器，如 Rorberts 算子,Sobel 算子,Prewitt 算子，Laplacian 算子等[17, 18, 26]。由于 Sobel 算法只涉及加法操作，但却可以得到很好的划分效果，因而是图像处理系统中最常用的边缘检测算法。

Sobel 算法[53, 5]包括带 4 个 3×3 掩码的输入图像数据，即 Sobel 算子，它设置权重来检测水平、垂直、左对角、右对角各个不同方向上密度幅度的不同。这个过程通常被称为过滤。我们来看像素窗口（3×3），如图 3.1 所示：

水平、垂直、左对角、右对角各图像方向上密度幅度的变化可以用如下算子进行计算：

H=(Q0+2Q3+Q6)-(Q2+2Q5+Q8);
V=(Q0+2Q1+Q2)-(Q6+2Q7+Q8);
DR=(Q1+2Q0+Q3)-(Q5+2Q8+Q7);
DL=(Q1+2Q2+Q5)-(Q3+2Q6+Q7);

Q0	Q3	Q6
Q1	[i,j]	Q7
Q2	Q5	Q8

图 3.1　图像窗口像素

H,V,DL,DR 这四个参数用于计算梯度大小和方向。

对梯度大小的一个普遍估计值为：Magnitude＝Max(H,V,DR,DL)。

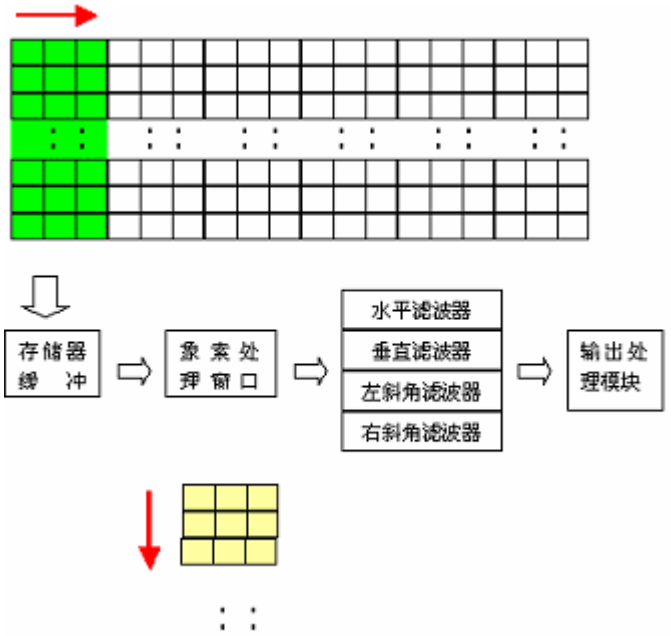
我们通过对图像灰度作直方图分析后，便可以给出区分度阀值 Threshold，区分度阀值往往要借助一定的经验并需要反复调整。如果 Magnitude 大于 Threshold，则该像素被声明为边界像素，否则为一般像素。

3.2　系统的 VLSI 实现设计

3.2.1　系统的总体设计

根据图像处理的知识及分析，我们可得到本边缘检测处理器的工作流程如图 3.2 所示。其处理过程如下：图像处理主处理器将从图像传感器中获取的灰度图(800×600)，按照每三列划分为一帧的原则进行帧窗口划分。帧窗口的图形数据又按照每三行划分为一个像素处理窗口的原则逐一进行处理。像素处理窗口在图形帧窗口内从上

向下移动，步长为一行，而帧窗口整幅图形窗口中从左到右移动，步距为一列。待处理的数据首先送入存储缓冲器中，再送入像素处理窗口中，经像素处理窗口处理后送入各个滤波器进行滤波，最后将有关结果送入输出处理模块处理后输出。



3.2 图像处理流程示意图

根据以上设计思路，我们可把整个系统的实现划分为四个大的模块，其总体结构如图 3.3 所示,其中：

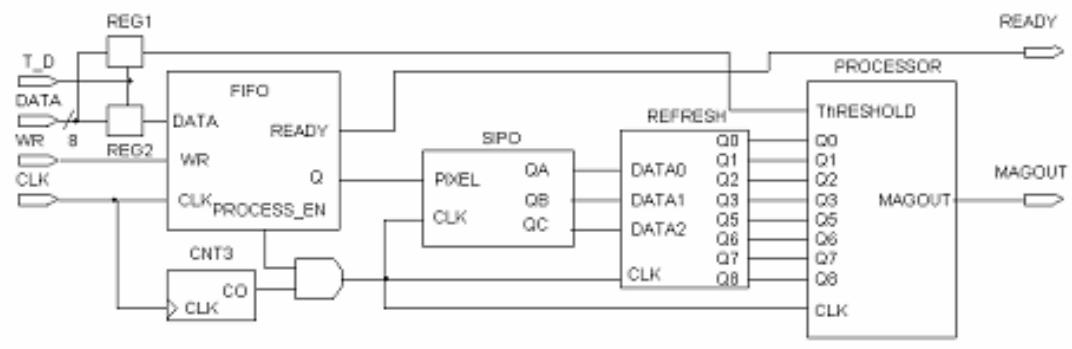


图 3.3　图像边缘检测器总体结构图

帧窗口接收模块(FIFO)：负责接收图像处理主处理器传送过来的一个帧窗口的数据，其本质为一个双端口先入先出栈 FIFO，其数据宽度为 8，深度等于一个帧窗口内的像素点个数(600×3＝1800)。

串入并出模块(SIPO)：负责把 FIFO 内的数据转换成为像素处理窗口的列像素向量，便于像素处理窗口的数据刷新处理。

像素窗口刷新模块(REFRESH)：实现对需要处理的像素数据的刷新。

数据处理模块(PROCESSOR)：它是本图形边缘处理器的核心部分，主要是实现 Sobel 算法，其性能的好坏对整个设计的成败有着关键的作用。本模块拟采用全硬件

并行算法实现，因只有五级串行结构，所以相当于在 5 个时钟周期内就能完成一个像素点的边界判别。

3.2.2　主要功能模块的设计

1. 帧窗口接收模块 FIFO

由于高速设备与慢速设备之间处理速度的差别，它们之间的数据传输一般采用查询方式或中断方式，而数据同步方式则选择了帧同步方式。存储器的类型拟选用 FIFO，堆栈空时向主机发出准备好信号，主机检测到它的数据传输请求时，传送一帧数据，由于 FIFO 的大小与一帧图像的大小是一致的，所以接收完毕后，堆栈满，Sobel 处理器启动边缘检测进程，处理完一帧数据后，堆栈重新变为空，为下一帧数据处理作准备。其内部结构如图 3.4 所示。

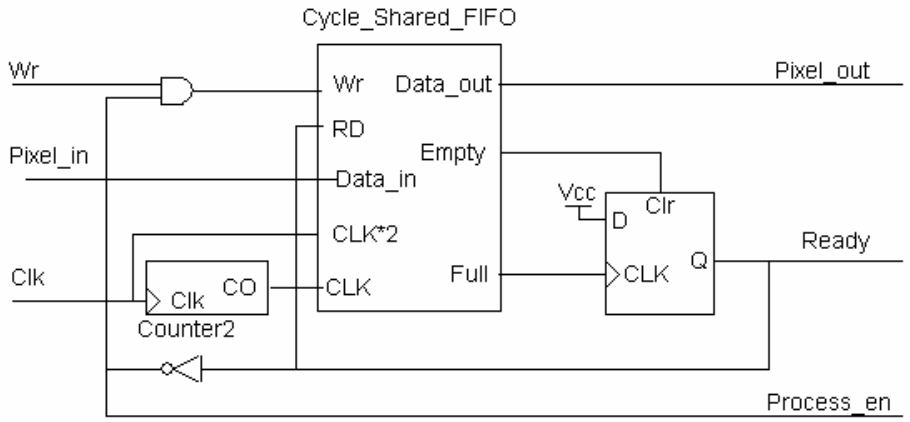


图 3.4 帧窗口接收模块 FIFO 的内部结构图

2. 串入并出模块 SIPO

串入并出模块的主要功能是负责一列像素的刷新，也就是把主处理器传送过来的像素值转换成 3×3 像素窗口的一列。该模块的实现主要是通过六个 D 触发器和一个三进制的计数器组成，当计数器产生进位溢出时，串行输入的三个像素点并行输出。内部结构如图 3.5 所示。

3. 像素窗口刷新模块 REFRESH

像素刷新窗口的主要功能是接收串入并出模块的 3 个并行像素，把窗口中原有的第二列像素推入第三列，第一列推入第二列，新到的并行像素填入第一列。其本质为一个移位寄存器。其内部结构如图 3.6 所示

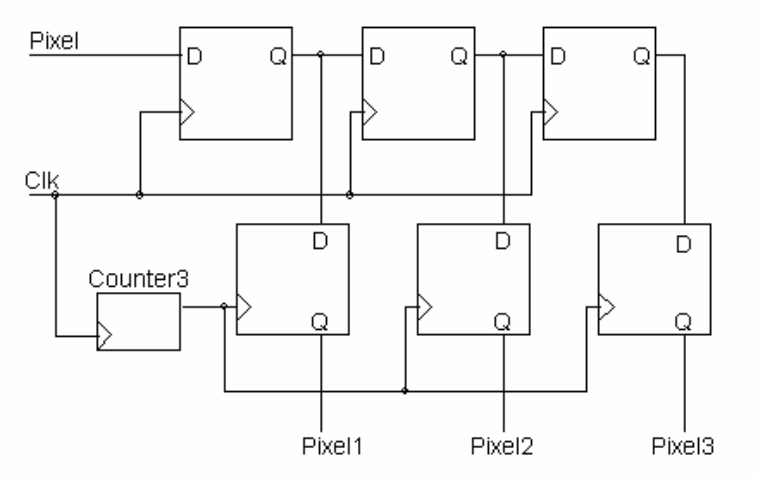


图 3.5 串入并出模块 SIPO 的内部结构图

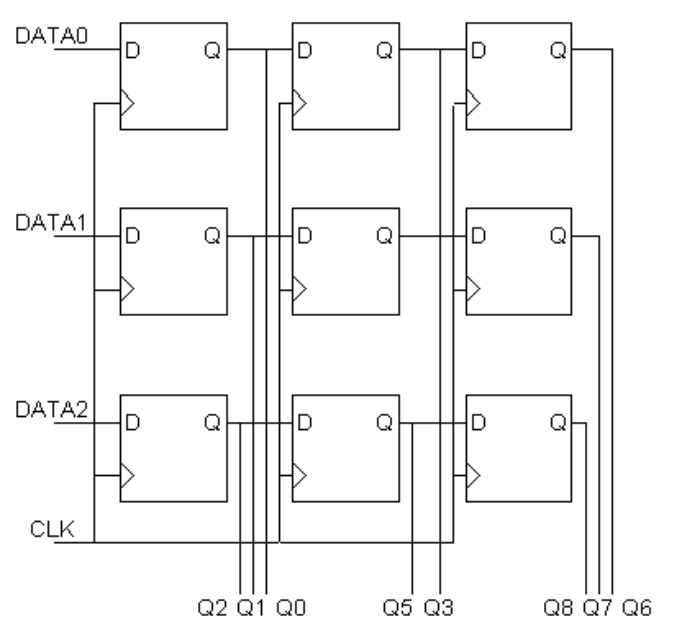


图 3.6　像素刷新窗口 REFRESH 的内部结构图

4.　滤波模块 FILTER

对于 SOBEL 算法的各个滤波器，经变换后可得到：

H=(Q0+Q3+Q3+Q6)-(Q2+Q5+Q5+Q8)；　　V=(Q0+Q1+Q1+Q2)-(Q6+Q7+Q7+Q8)；

DR=(Q1+Q0+Q0+Q3)-(Q5+Q8+Q8+Q7)；　　DL=(Q1+Q2+Q2+Q5)-(Q3+Q6+Q6+Q7)；

因此我们对于滤波模块 FILTER 的设计可采用两级并行流水方案，其内部结构如图 3.7 所示。图中的输入若采用 QA，QB，QC，QD，QE，QF，输出采用 FILTER，则这时表示的是一个通用滤波器，图中的输入 QA，QB，QC，QD，QE，QF 若对应地接上 Q0，Q3，Q6，Q2，Q5，Q8，则这时表示的是水平方向滤波器 H_FILTER，其输出则为 H_FILTER。垂直方向滤波器、左对角滤波器、右对角滤波器与上设计类似。

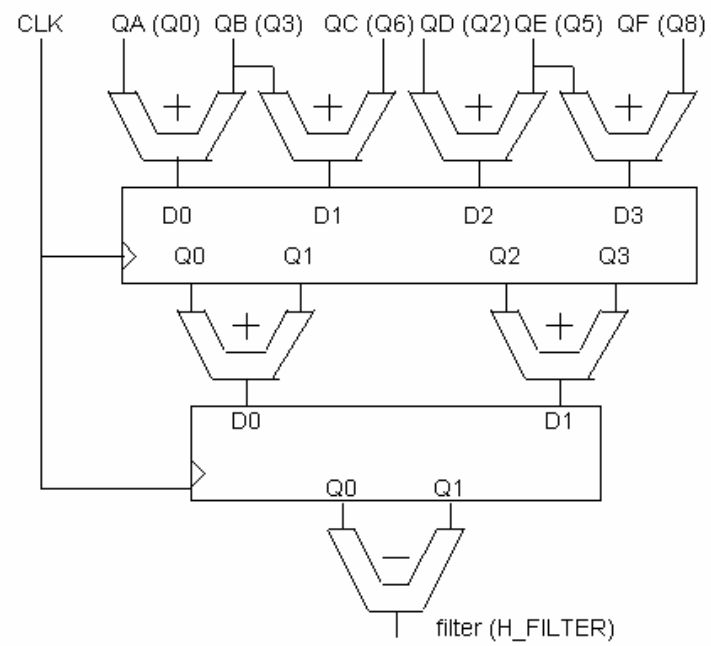


图 3.7　滤波模块 FILTER 的内部结构图

5. 数据处理模块 PROCESSOR

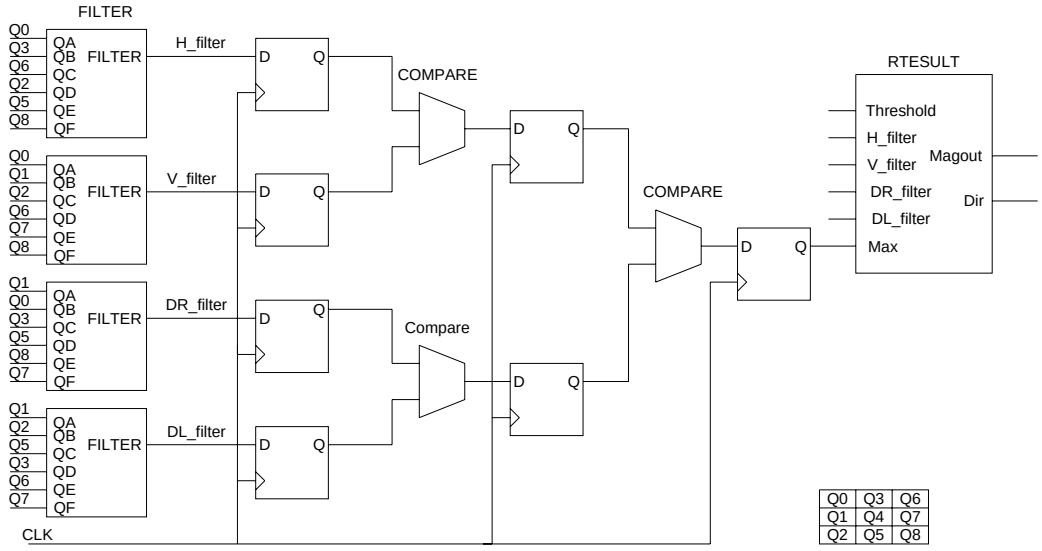


图 3.8　数据处理模块 PROCESSOR 的内部结构图

数据处理模块 PROCESSOR 的主要功能是求出四个方向的图像梯度数据绝对值的最大值，同时判别最大值出现的方向。其内部结构如图 3.8 所示,它包括滤波器 FILTER、比较器 COMPARE、边界判断器 RESULTER、寄存器等。

3.3　系统的 LPM 原理图和 VHDL 程序设计

根据前述的总体设计方案,使用 LPM 原理图为主并辅以 VHDL 进行各个模块和系统

总体程序的设计,其中 SIPO 的 LPM 原理图如图 3.9 所示,REFREASH 的 LPM 原理图如图 3.10 所示，FILTER 的 LPM 原理图如图 3.11 所示，PROCESSOR 的 LPM 原理图如图 3.12 所示，其余模块的 LPM 原理图或 VHDL 程序略。

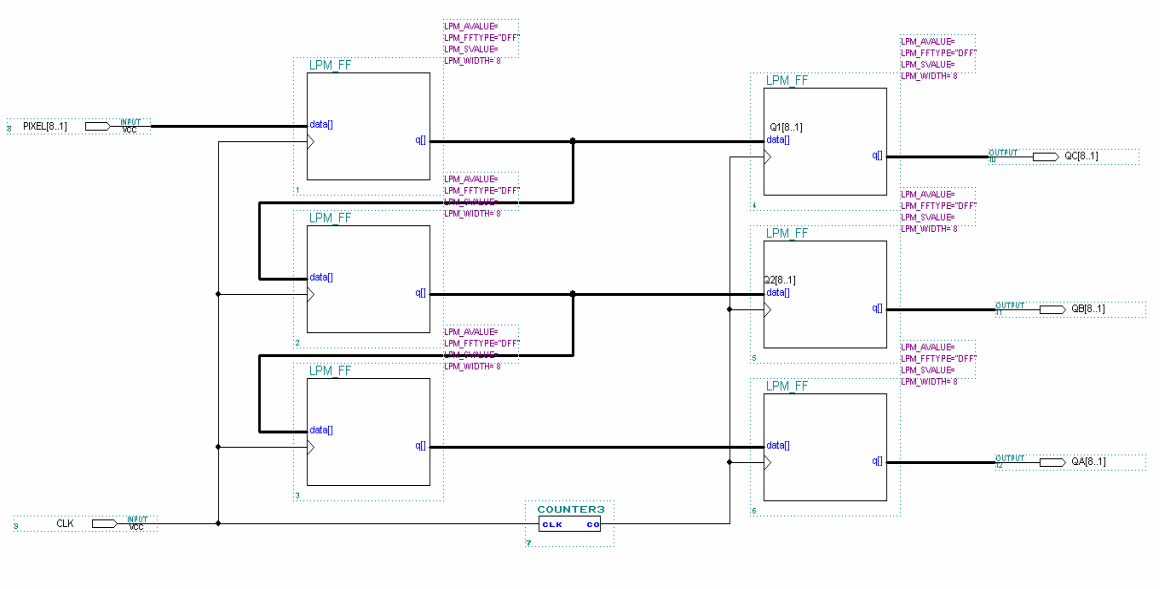


图 3.9　　SIPO 的原理图

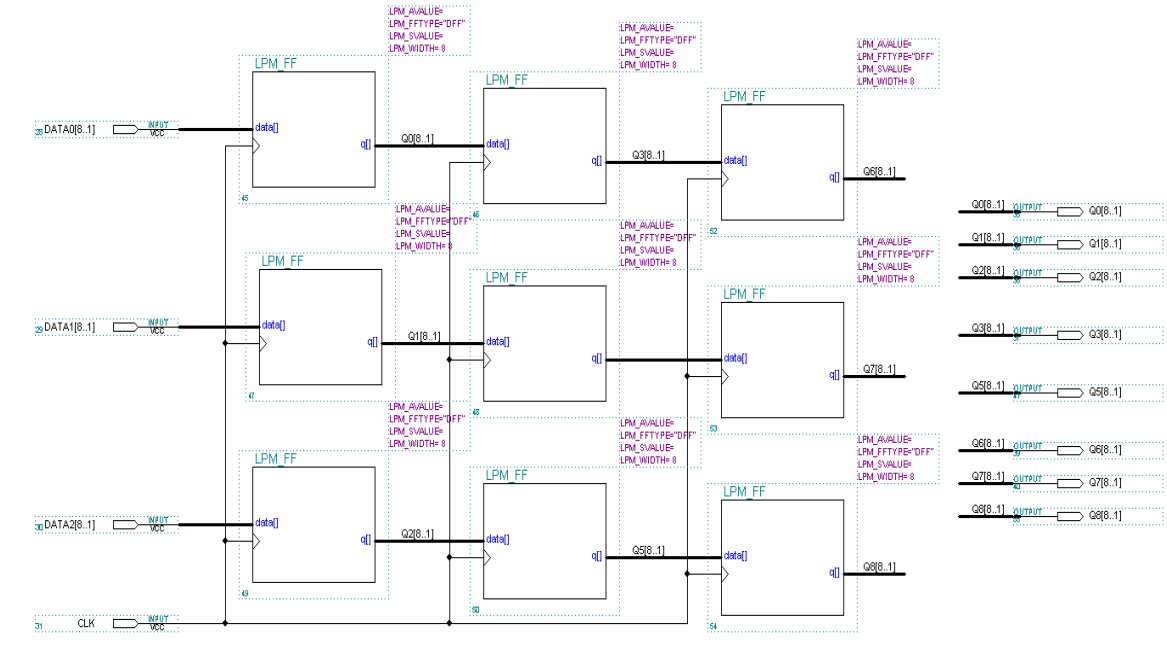


图 3.10　　REFREASH 的原理图

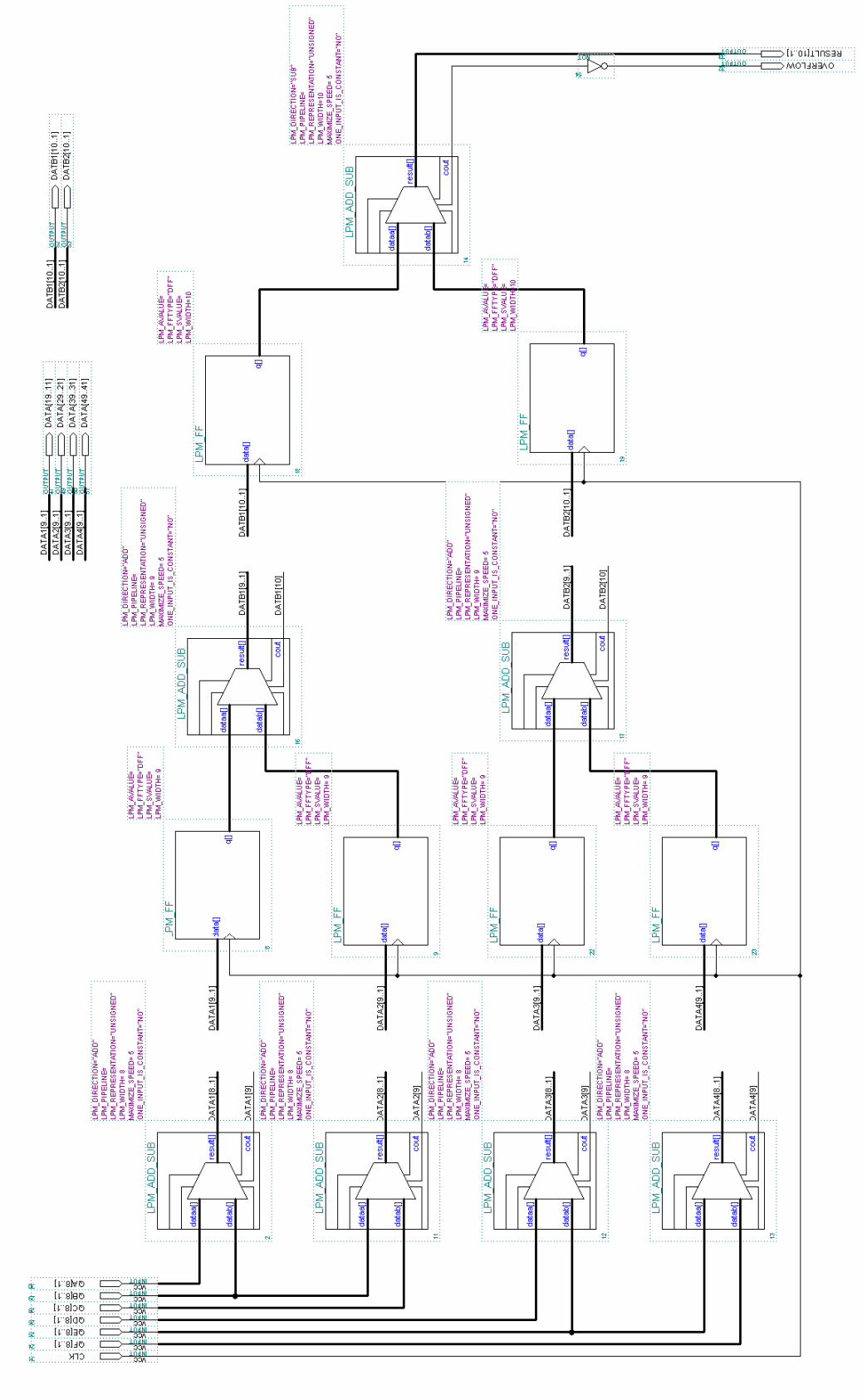


图 3.11　FILTER 的原理图

注：图中的输出端口 DATA[19..11]、DATA[29..21]、DATA[39..31]、DATA[49..41]、DATB1[10..1]、DATB2[10..1]仅供仿真时输出中间结果，调试完后应去掉。

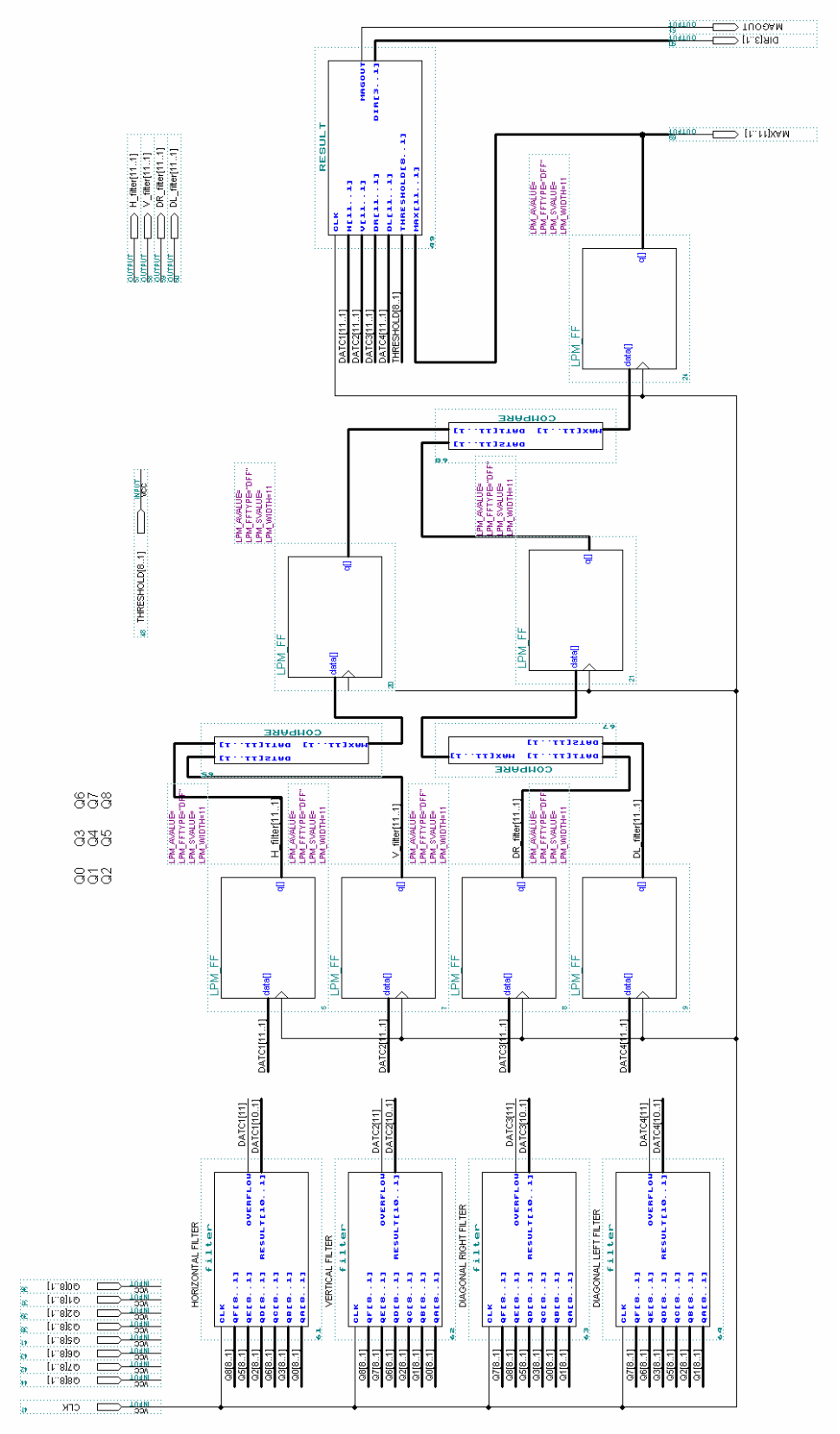


图 3.12　　PROCESSOR 的原理图

3.4　系统的有关仿真结果

整个系统使用 MAX+plus II 10.0，选用 ALTERA 公司的 FLEX10K20 芯片，通过采用 LPM 兆功能块定制方法和 VHDL 自编程的方式完成了设计和调试，经分析仿真结果，系统达到了设计要求。下面是部分仿真结果及分析。

1．FIFO 的仿真

本次设计使用了 Altera LPM 库中的 CSFIFO，即 Cycle_Shared_FIFO。FIFO 用于与主处理器如单片机或 DSP 进行数据接口。为了便于观察系统输出，调试过程中使用的 FIFO 深度值只设置为 4。

FIFO 的仿真结果如图 3.13 所示。从图中可以看出，主处理器写完四个像素点数据后，Q[8..1]按照先入先出的顺序，逐个输出获取的像素数据“12，13，14”，在此期间，ready 信号为无效电平（低），所以外部输入的数据“16，17，18”不能存入 FIFO。当所有的像素点数据全部输出后，ready 信号重新有效，同时 Process_en 有效。综上所述，FIFO 的设计是合理的。

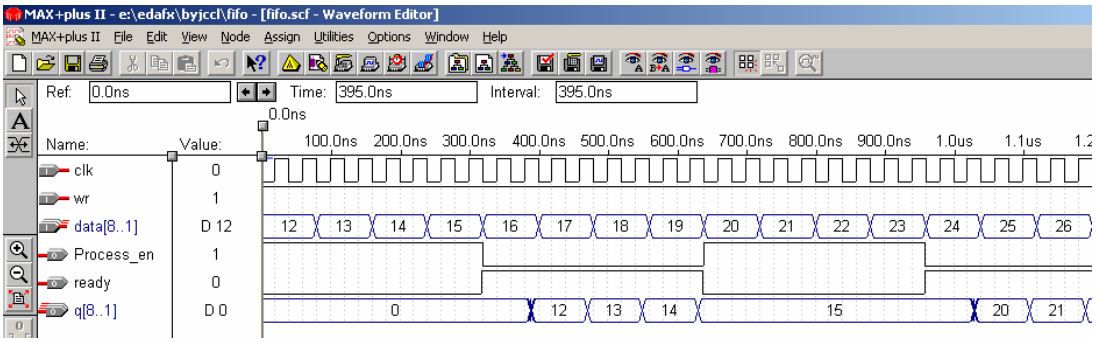


图 3.13　FIFO 仿真结果

2．SIPO 的仿真

SIPO 的仿真结果如图 3.14 所示。由图中可以看出，如我们将串行数据每三个划分为一段，QA、QB、QC 恰好是这一段的并行输出，符合设计期望。

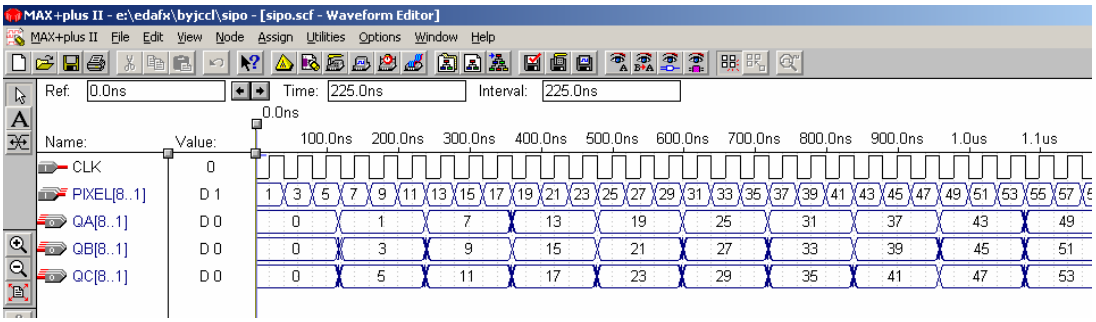


图 3.14　SIPO 仿真结果

3．REFRESH 的仿真

REFRESH 模块在系统中的主要作用是实现像素处理窗口的更新。在每一时钟上升沿，并行提供三个输入像素。REFRESH 模块的仿真结果如图 3.15 所示。如波形图中红色箭头所示，该模块同时实现了并行输入，同步移位寄存，并行数据输出功能，符合

设计期望。

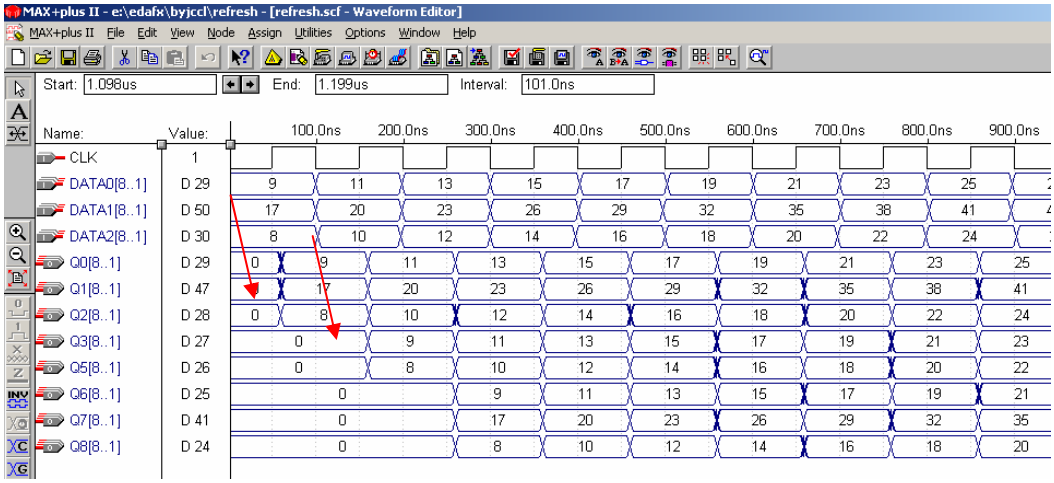


图 3.15　REFRESH 仿真结果

4. FILTER 的仿真

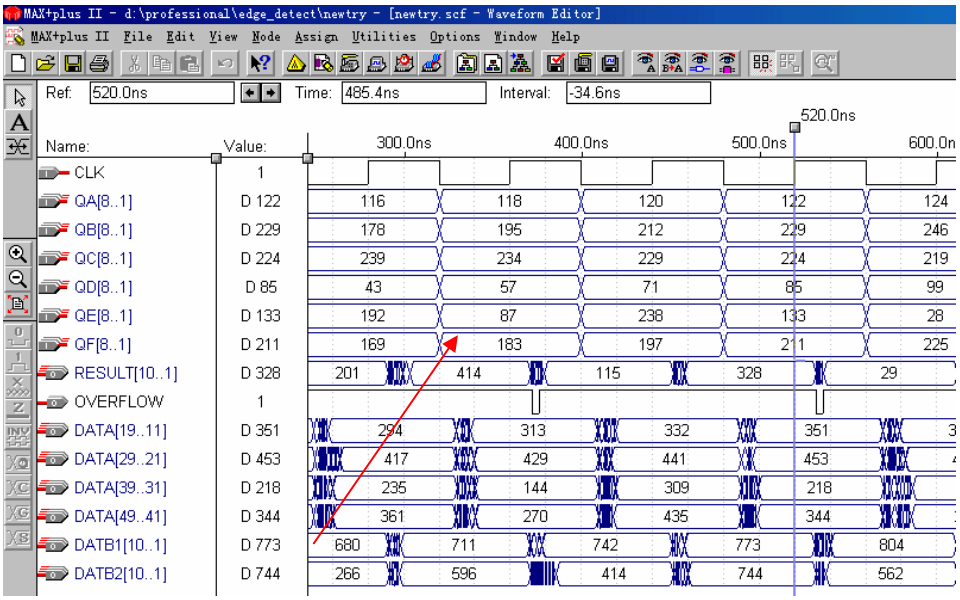


图 3.16　FILTER 仿真结果

滤波器其本质是实现并行流水加法操作。在调试过程中，为了观察到实际的延时，输出信号直接来自流水加法器的输出，而非流水寄存器输出。所以存在部分毛刺。

图 3.16 是它的仿真波形，由图中可以看出：

DATA[19..11]=QA+QB ；DATA[29..21]=QB+QC；DATA[39..31]=QD+QE ；DATA[49..41]=QE+QF；DATB1[10..1]=DATA[19..11]+DATA[29..21]；DATB2[10..1]=DATA[39..11]+DATA[49..41]；RESULT=DATB1[10..1]-DATB2[10..1]。

因此仿真结果符合设计要求。同时我们可以看出，并行流水操作将带来输出的滞后（如红线部分所示）。

3. PROCESSOR 仿真结果

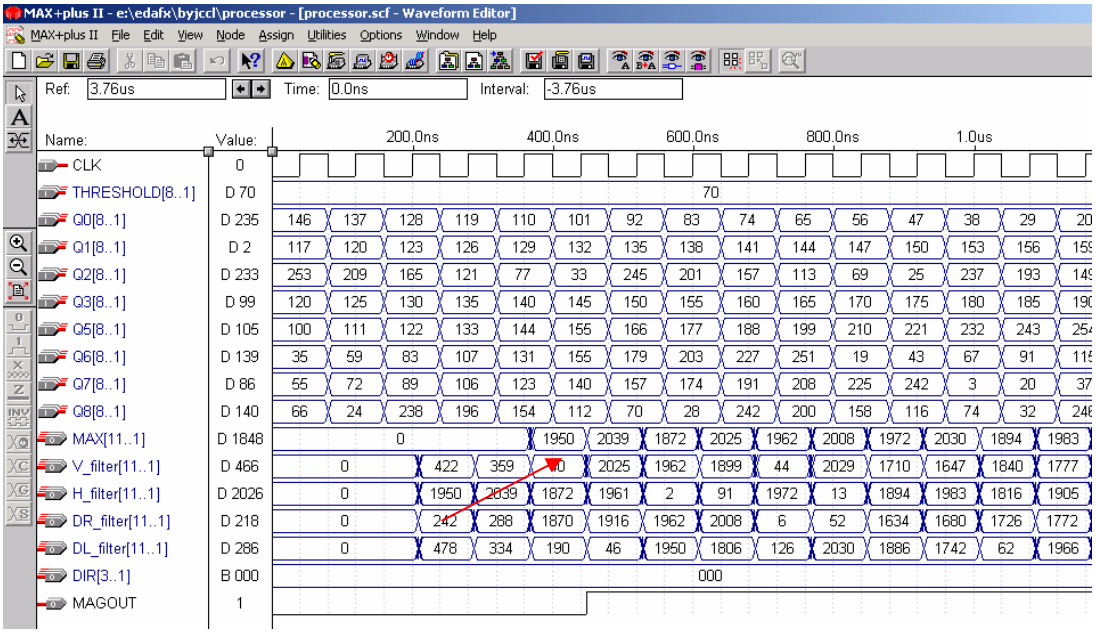


图 3.17　数据处理（边缘检测）模块 PROCESSOR 的仿真结果

图 3.17 是 PROCESSOR 仿真图，我们先计算图中四个方向滤波器的绝对值大小

H_filter＝422 ；　　V_filter＝1950－1024＝926

DR_filter＝242；　　DL_filter＝478

四者中的绝对值最大值是 926，故最大的为 V_filter。观察红线所指的部分，正好为 V_filter 的值 1950。经过验证，上述图中的所有输出满足输出期望。

3.5　结论

在本系统的设计中，通过设置 FIFO 栈解决了高速处理设备与慢速输入设备之间处理速度的矛盾，通过 D 触发器和计数器构成的串入并出模块实现了列像素的刷新，通过设置移位寄存器避免了待处理数据的重复输入和 3×3 像素处理窗口数据的刷新，通过采用两级并行流水线设计滤波器模块和并行设置四个滤波器模块来提高整个系统的处理速度，因此本系统实现了邻域数据处理的流水式输入，多层次、全过程的流水线处理和并行处理，具有良好的实时处理性能。对一幅（1024×1024）的图像，本系统的处理时间为：1024×1024×5/50M＝0.105s，而单独使用 MCS51（12M 晶振）进行处理的时间为：1024×1024×12×（8×4＋8）/12M＝41.943s，单独采用六级指令流水线的 DSP（40M 晶振）器件的时间为：1024×1024×（8×4＋8）/40M＝1.049s，因此本系统的处理速度相对于 DSP 系统提高了 10 倍，相对于 MCU 系统提高了 400 倍。同时该系统集成在一块集成芯片上，体积小，功耗低，可靠性高，并可现场编程，在线升级。

第四章　Laplacian图像边缘检测器的VLSI实现设计

4.1　Laplacian边缘检测算子

常见的边缘检测算子有：Roberts，Laplacian，Kirsch，Sobel，Prewitt等算子。各种检测算子各有其优缺点，其中Laplacian算子是一种典型的边缘检测方法，它是一个3×3模板，对实现硬件的要求不高，并且用于检测屋顶型边缘的效果也不错。研究它的硬件实现具有一定的典型性，并可推广到对其他算子的硬件实现。

拉普拉斯算子是根据图像 f(x,y)在 x,y 方向上的二阶偏导数定义的一种边缘检测算子,其定义如下[54]:

$$\nabla^2 f(x,y)=\frac{\partial^2 f}{\partial x^2}+\frac{\partial^2 f}{\partial y^2} \tag{4.1}$$

因为图像边缘有大的灰度变化,所以图像的一阶偏导数在边缘处有局部最大值或最小值,这样二阶偏导数在边缘处会通过零点。用拉普拉斯算子检测边缘就是估算拉普拉斯算子的输出,找出它的零点位置。在数字图像中,拉普拉斯算子被定义为:

$$\nabla^2 f(x,y)=f(x,y-1)+f(x,y+1)+f(x-1,y)+f(x+1,y)-4f(x,y) \tag{4.2}$$

如果把它用加权矩阵来表示则可表示成图4.1所示的模板。由于数字图像中离散信号的特点,在连续情况下能获得的精确零点这时可能无法全部检测出来,故拉普拉斯算子输出为零的点并不能表示出完整的目标边缘。为此,在设计中我们定义边缘为满足以下两个条件的像素点的集合:(1)拉普拉斯算子的输出为正;(2)在其八邻点存在拉普拉斯算子的输出为负的点。

0	1	0
1	-4	1
0	1	0

图4.1　Laplacian卷积模板

4.2　图像边缘检测的实现流程

图像边缘检测都有一个共同点：可转化为用一个模板（2×2，3×3，4×4，5×5等）对图像进行卷积。因此图像边缘检测的核心就是如何处理模板的卷积运算。Laplacian算子就是一个的3×3的卷积模板。

3×3卷积运算[55]定义如式（4.3）所示，其中$C_{m,n}$为被卷积后的像素值，$P_{m,n}$为图像的实际像素值，其中$C_{m,n}$为被卷积后的像素值，$P_{m,n}$为图像的实际像素值。由 4.3式可知，完成一次3×3卷积操作需要9次乘法和8次加法操作。

$$C（m,n）=\sum_{i=-1}^{1}\sum_{j=-1}^{1}P_{m+i,n+j}W_{i,j} \tag{4.3}$$

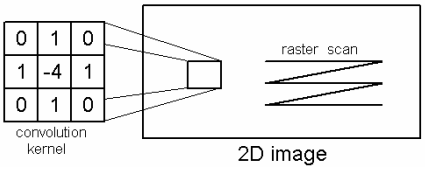


图 4.2　3×3 卷积原理示意图

根据图 4.2 的 3×3 卷积原理示意图，我们可知，图像边缘检测的主要实现流程为：(1)模板在图像中以光栅扫描方式移动；(2)将模板系数与模板下对应像素相乘；(3)将所有乘积相加；(4)将和（模板输出响应）赋给图像中对应模板中心位置的像素完成一次卷积。不断重复上述步骤直至整幅图像被处理完。

4.3　分布式算法的运算原理

分布式算法(Distributed Arithmetic)早在1973年就已经被Croisier提出来了，但是直到FPGA出现以后，才被广泛地应用在FPGA中计算乘积和。它除了用于卷积运算外，还可用于相关、DFT和RNS反演映射的运算中[10]。下面我们就介绍一下DA算法的运算原理。一个线性时不变网络的输出可以用下式表示：

$$y=<c,x>=\sum_{n=0}^{N-1}C[n]\cdot x[n]=c[0]x[0]+c[0]x[0]+\cdots+c[N-1]x[N-1] \tag{4.4}$$

假设系数c[n]是已知常数，x[n]是变量，在有符号DA系统中假设变量x[n]的表达式如下：

$$x[n]=-2^{B-1}\cdot x_{B-1}+\sum_{b=0}^{B-2}x_b[n]\cdot 2^b \qquad x_b[n]\in[0,1] \tag{4.5}$$

式中，$x_b[n]$表示$x[n]$的第b位，而x[n]也就是x的第n次采样。于是，内积y可以表示为：

$$y=<c,x>=\sum_{n=0}^{N-1}C[n]\cdot[-2^{B-1}\cdot x_{B-1}+\sum_{b=0}^{B-2}x_b[n]\cdot 2^b] \tag{4.6}$$

重新分别求和（也就是分布式算法的由来），其结果如下：

$$\begin{aligned}y=&c[0](-x_{B-1}[0]2^{B-1}+x_{B-2}[0]2^{B-2}+\cdots x_0[0]2^0)\\&+c[1](-x_{B-1}[1]2^{B-1}+x_{B-2}[1]2^{B-2}+\cdots x_0[1]2^0)\\&\vdots\\&+c[N-1](-x_{B-1}[N-1]2^{B-1}+x_{B-2}[N-1]2^{B-2}+\cdots x_0[N-1]2^0)\end{aligned}$$

$$
\begin{aligned}
&=(c[0]x_{B-1}[0]+c[1]x_{B-1}[1]+\cdots+c[N-1]x_{B-1}[N-1])2^{B-1}\\
&\quad+(c[0]x_{B-2}[0]+c[1]x_{B-2}[1]+\cdots+c[N-1]x_{B-2}[N-1])2^{B-2}\\
&\quad\vdots\\
&\quad+(c[0]x_{0}[0]+c[1]x_{0}[1]+\cdots+c[N-1]x_{0}[N-1])2^{0}
\end{aligned}
\tag{4.7}
$$

从(4.7)式可以发现，分布式算法是一种以实现乘加运算为目的的运算方法。它与传统算法实现乘加运算的不同在于执行部分积运算的先后顺序不同。分布式算法在实现乘加功能时，是通过将各输入数据的每一对应位产生的部分积预先进行相加形成相应的部分积，然后再对各个部分积累加形成最终结果的，而传统算法是等到所有乘积已经产生之后再来相加完成乘加运算的。与传统串行算法相比，分布式算法可极大地减少硬件电路的规模，提高电路的执行速度。该算法实际上是一个“速度最优的高阶分布式算法”

4.4　系统的 VLSI 实现设计

4.4.1　系统的总体设计

Laplacian 边缘检测器 DETECTOR 总体设计如图 4.3 所示。系统工作原理为：帧存储器按照一定的规则（这里是按行）输出数据，经过 FIFO 输入缓冲，进入两个 32 位的移位寄存器，由 SIPO（串进并出）模块得到所需的 9 个并行数据，送卷积处理器（Convolver）进行卷积，从而得到处理后的数据，最后卷积器输出一个像素值。根据前述的边缘判断规则，我们即可得出这一点是否为边缘点。

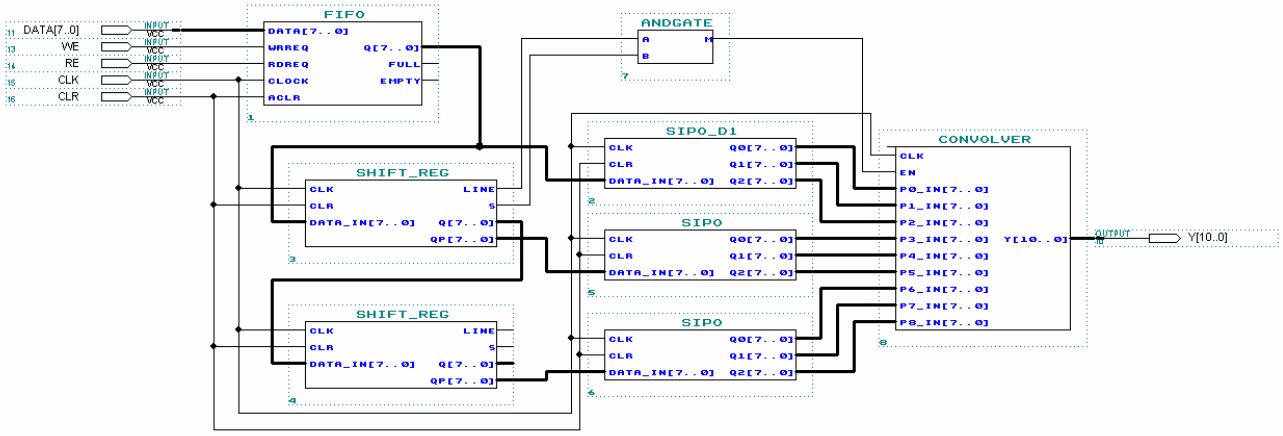


图 4.3　DETECTOR 总体设计图

4.4.2　卷积运算模块的设计

Laplacian 模板是一个 3×3 阶的数，每个像素值为 8 位，因此它和另外的 3×3

模板进行卷积时，输出表达式如下：

$$y =< c, x >= \sum_{n=0}^{8} c[n] \cdot \sum_{b=0}^{7} x_b[k] \cdot 2^b$$

$$= 0 \times P_0 + 1 \times P_1 + 0 \times P_2 + 1 \times P_3 - 4 \times P_4 + 1 \times P_5 + 0 \times P_6 + 1 \times P_7 + 0 \times P_8 \quad (4.8)$$

对于该卷积运算的实现，采用前述的“速度最优的高阶分布式算法”，其硬件实现的原理框图如图 4.4 所示，它是完全流水线式字并行结构，能够达到最大的运算速度。其中的 8 个 ROM 是用于实现 9 个 8 位的数相乘，每个 ROM 都实现一个 9 位的查找表的功能。

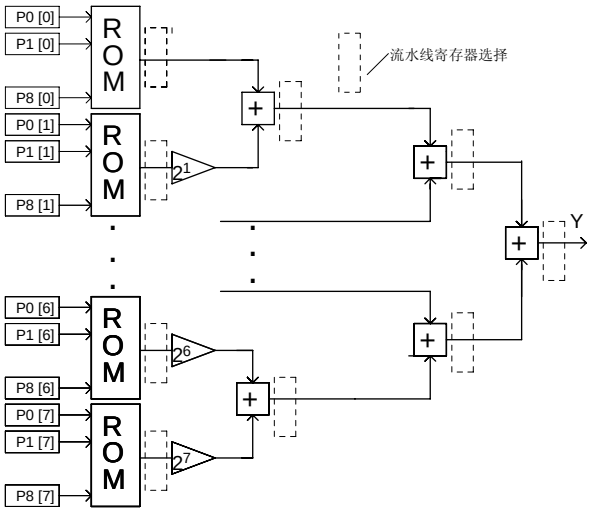


图 4.4　3×3 模板卷积采用速度最优的高阶分布式算法

4.4.3　卷积运算数据“流水式”输入模块的设计

图像的像素是由 CCD（或 CMOS）摄象机经 A/D 转换，再经量化而得到，并放入帧存储体。在图像中，整幅图像像素以帧为单位进行存储。每一帧数据的存储方式如图 4.5 所示。卷积运算扫描像素的获取如图 4.6 所示，该数据输入方式，使用了两个 32 位的移位寄存器存放像素值，避免了卷积运算中对存储器数据的高度重复读取，使用 9 个寄存器实现了数据由串行到并行的转换，实现了完全“流水线”的输入方式。

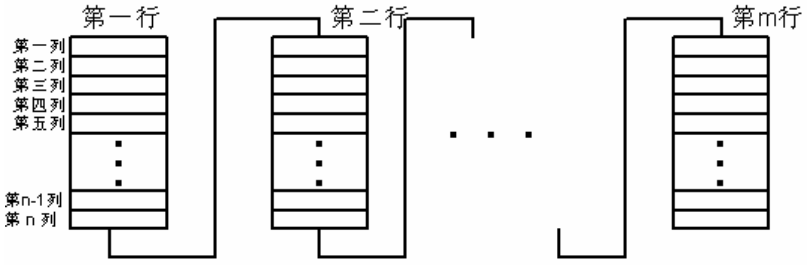
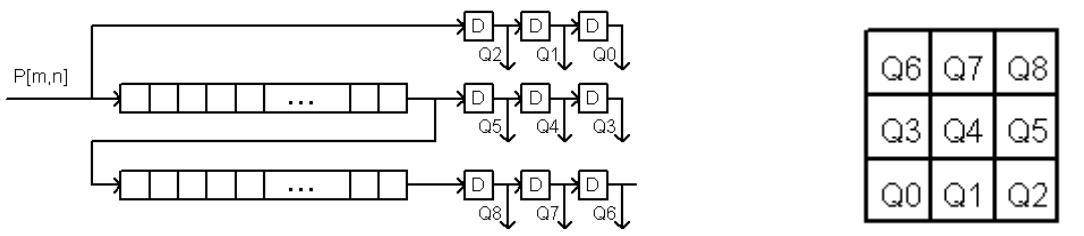


图 4.5　帧数据的存储方式



（a）　　（b）

图 4.6　卷积运算扫描像素的“流水式”输入原理图

（a）“流水式”输入原理　　（b） 输入数据的对应模板

4.5 系统的 VHDL 程序设计

根据系统的总体设计和内部各模块的设计思想，使用 VHDL 进行各个模块和系统总体程序的设计,下面只给出其中的关键程序 CONVOLVER.VHD，其余的 VHDL 程序略。

```
--CONVOLVER.VHD
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
PACKAGE DA_PACKAGE IS
  COMPONENT CASE3S IS
    PORT(TABLE_IN:IN STD_LOGIC_VECTOR(8 DOWNTO 0);
         TABLE_OUT:OUT INTEGER RANGE -4 TO 4);
  END COMPONENT;
END DA_PACKAGE;
LIBRARY WORK;
USE WORK.DA_PACKAGE.ALL;
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
ENTITY CONVOLVER IS
  PORT(CLK ,EN:IN STD_LOGIC;
       P0_IN,P1_IN,P2_IN,P3_IN,P4_IN,P5_IN,P6_IN,
       P7_IN,P8_IN: IN STD_LOGIC_VECTOR(7 DOWNTO 0);
        Y :OUT INTEGER RANGE -1020 TO 1020);
END ENTITY CONVOLVER;
ARCHITECTURE ART OF CONVOLVER IS
  SIGNAL M0,M1,M2,M3,M4,M5,M6,M7:
```

```
STD_LOGIC_VECTOR(8 DOWNTO 0);
SIGNAL Q0,Q1,Q2,Q3,Q4,Q5,Q6,Q7:INTEGER RANGE -4 TO 4;
BEGIN
PROCESS(CLK,EN) IS
  BEGIN
  IF (CLK'EVENT AND CLK='1')THEN
  M0(0)<=P0_IN(0);
  M0(1)<=P1_IN(0);
  M0(2)<=P2_IN(0);
  M0(3)<=P3_IN(0);
  M0(4)<=P4_IN(0);
  M0(5)<=P5_IN(0);
  M0(6)<=P6_IN(0);
  M0(7)<=P7_IN(0);
  M0(8)<=P8_IN(0);
  M1(0)<=P0_IN(1);
  M1(1)<=P1_IN(1);
  M1(2)<=P2_IN(1);
  M1(3)<=P3_IN(1);
  M1(4)<=P4_IN(1);
  M1(5)<=P5_IN(1);
  M1(6)<=P6_IN(1);
  M1(7)<=P7_IN(1);
  M1(8)<=P8_IN(1);
  M2(0)<=P0_IN(2);
  M2(1)<=P1_IN(2);
  M2(2)<=P2_IN(2);
  M2(3)<=P3_IN(2);
  M2(4)<=P4_IN(2);
  M2(5)<=P5_IN(2);
  M2(6)<=P6_IN(2);
  M2(7)<=P7_IN(2);
  M2(8)<=P8_IN(2);
  ……
  M6(0)<=P0_IN(6);
```

```
  M6(1)<=P1_IN(6);
  M6(2)<=P2_IN(6);
  M6(3)<=P3_IN(6);
  M6(4)<=P4_IN(6);
  M6(5)<=P5_IN(6);
  M6(6)<=P6_IN(6);
  M6(7)<=P7_IN(6);
  M6(8)<=P8_IN(6);
  M7(0)<=P0_IN(7);
  M7(1)<=P1_IN(7);
  M7(2)<=P2_IN(7);
  M7(3)<=P3_IN(7);
  M7(4)<=P4_IN(7);
  M7(5)<=P5_IN(7);
  M7(6)<=P6_IN(7);
  M7(7)<=P7_IN(7);
  M7(8)<=P8_IN(7);
  IF (EN='1')THEN
    Y<=Q0+2*Q1+4*Q2+8*Q3+16*Q4+32*Q5+64*Q6+128*Q7;
  ELSE
    Y<=0;
   END IF;
 END IF;
END PROCESS;
LC_TABLE0:CASE3S
  PORT MAP (TABLE_IN=>M0,TABLE_OUT=>Q0);
LC_TABLE1:CASE3S
  PORT MAP (TABLE_IN=>M1,TABLE_OUT=>Q1);
LC_TABLE2:CASE3S
  PORT MAP (TABLE_IN=>M2,TABLE_OUT=>Q2);
LC_TABLE3:CASE3S
  PORT MAP (TABLE_IN=>M3,TABLE_OUT=>Q3);
LC_TABLE4:CASE3S
  PORT MAP (TABLE_IN=>M4,TABLE_OUT=>Q4);
LC_TABLE5:CASE3S
```

```
    PORT MAP (TABLE_IN=>M5,TABLE_OUT=>Q5);
  LC_TABLE6:CASE3S
    PORT MAP (TABLE_IN=>M6,TABLE_OUT=>Q6);
  LC_TABLE7:CASE3S
    PORT MAP (TABLE_IN=>M7,TABLE_OUT=>Q7);
END ARCHITECTURE ART;

--CASE3S.VHD
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;
ENTITY CASE3S IS
   PORT(TABLE_IN:IN STD_LOGIC_VECTOR(8 DOWNTO 0);
        TABLE_OUT: OUT INTEGER RANGE -4 TO 4);
END ENTITY CASE3S;
ARCHITECTURE ART OF CASE3S IS
  SIGNAL K0,K1,K2,K3,K4,K5,
  K6,K7,K8:INTEGER RANGE 0 TO 1 ;
  BEGIN
  PROCESS(TABLE_IN)
    BEGIN
    IF (TABLE_IN(0)='1') THEN
      K0<=1;
    ELSE
      K0<=0;
    END IF;
    IF (TABLE_IN(1)='1') THEN
       K1<=1;
    ELSE
      K1<=0;
    END IF;
    IF (TABLE_IN(2)='1') THEN
       K2<=1;
    ELSE
      K2<=0;
```

```
    END IF;
    ……
    IF (TABLE_IN(7)='1') THEN
       K7<=1;
    ELSE
       K7<=0;
    END IF;
    IF (TABLE_IN(8)='1') THEN
       K8<=1;
    ELSE
       K8<=0;
    END IF;
    TABLE_OUT<=0*K0+1*K1+0*K2+1*K3-4*K4+1*K5+0*K6+1*K7;
  END PROCESS;
END ARCHITECTURE ART;
```

4.6　系统的有关仿真结果

根据前述的总体设计方案，使用 VHDL 进行各个模块和系统总体程序的设计，选择的 FPGA 为 ALTERA/FLEX/EPF10K20TC144-3，使用 MAX+plus II 10.0 进行编译，综合，适配，以及进行有关仿真,系统达到了设计要求。

图 4.7 是卷积运算模块(CONVOLOR)的时序仿真图。从图 4.7 可知,当输入 P1_IN=1，P3_IN=2，P4_IN=1，P5_IN=3，P7_IN=2 时，输出 Y 应该为 P1_IN + P3_IN - 4P4_IN + P5_IN + P7_IN=1+2-4×1+3+2=4，实际输出 Y 为 4，满足设计要求；当输入 P1_IN=3，P3_IN=4，P4_IN=2，P5_IN=9，P7_IN=6 时，输出 Y 应该为 3+4-4×2+9+6=14，实际输出 Y 为 14，满足设计要求；当输入 P1_IN=5，P3_IN=6，P4_IN=3，P5_IN=7，P7_IN=9 时，输出 Y 应该为 5+6-4×3+7+9=15，实际输出 Y 为 15，满足设计要求……,故本模块的设计满足给定的设计要求。

图 4.8 是边缘检测器的时序仿真图（使用时钟频率为 10MHZ），对波形仿真结果分析可知，系统达到了设计功能要求，并且该系统经过初始的两行行延迟和串并转化后（为 72 个时钟周期），以后每个时钟周期就可“流水式”输出一个处理结果，若系统时钟周期 T_{CLK}，对于像素为 N 个点的数字图像，系统的处理时间 $T_N=70*T_{CLK}+N*T_{CLK,}$。处理一幅 1024*1024 的图像的时间，当系统时钟为 10MHZ 时，仅需 0.1S，而系统时钟为 10MHZ 时，仅需 0.01S。

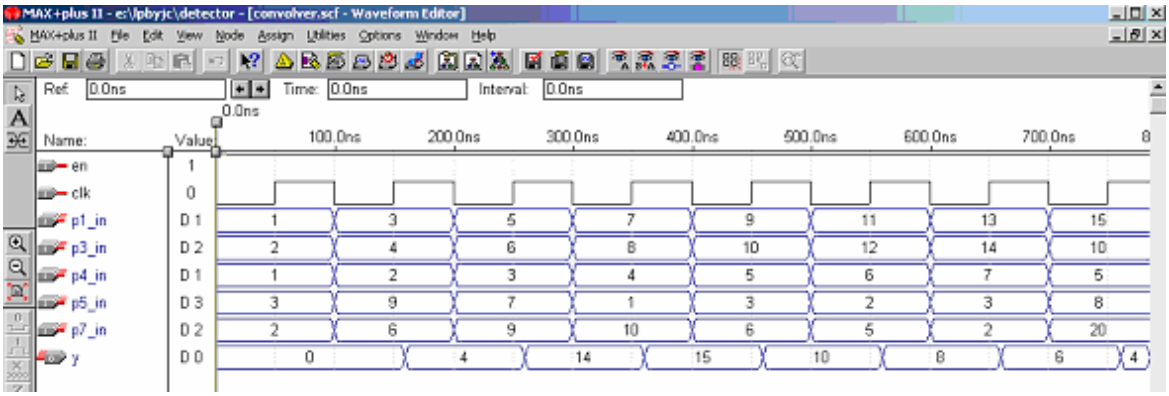


图 4.7　　卷积运算模块(CONVOLOR)的时序仿真图

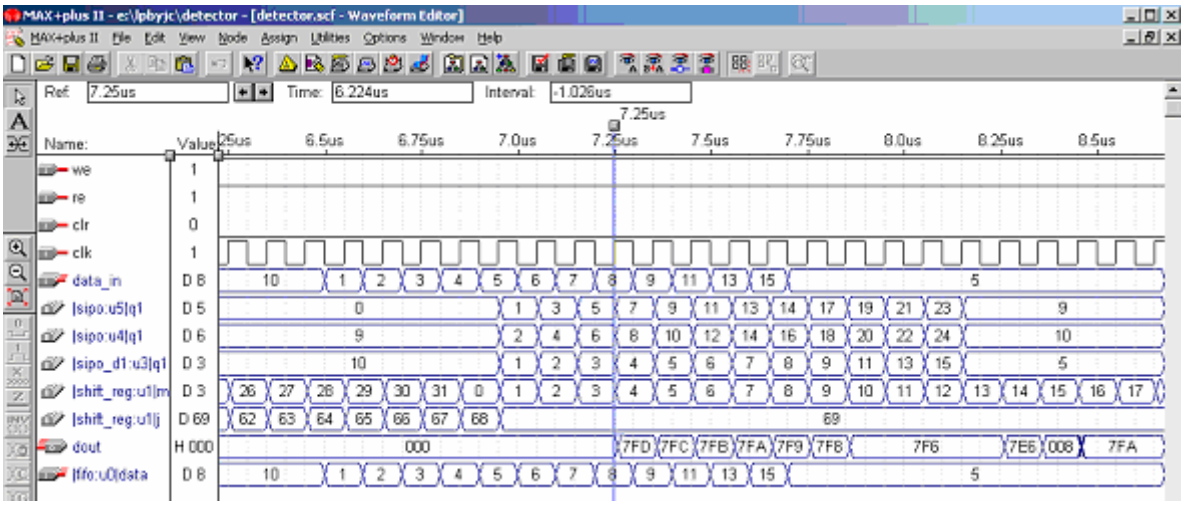


图 4.8　　边缘检测器的时序仿真图

4.7　结论

在 Laplacian 图像边缘检测器的设计中，使用了两个 32 位的移位寄存器存放像素值，使用 9 个寄存器实现了数据由串行到并行的转换，既避免了卷积运算中对存储器数据的高度重复读取，又实现了卷积运算数据的完全“流水式”输入方式，同时采用了速度最优的高阶分布式算法完成模板的卷积运算,从而使系统在资源，速度上达到了较好的优化。该系统经过初始的两行行延迟和串并转化后（为 72 个时钟周期），以后每个时钟周期就可“流水式”输出一个处理结果，具有良好的实时处理性能。若系统工作时钟为 100MHZ，则处理一幅 1024*1024 的图像的时间仅需 0.01S 左右。“流水线”的数据输入方式和分布式卷积运算的设计思想，对于数字图像和数字信号处理的 FPGA 硬件实现，具有广泛的推广应用价值。

第五章 整数小波变换的VLSI实现设计

5.1 整数小波变换

离散小波变换（DWT）虽然简单，但它使用的是非整数滤波器系数，这会导致非整数变换系数。而基于提升方法的整数小波变换（IWT）不仅具有计算更快捷，能够在当前位置完成小波变换从而节省内存，能对任意尺寸图像进行小波变换等优点，而且是可逆的，即可以从整数变换系数中完全重建图像，因而它既可以对图像进行有损压缩（通过量化变换系数），也可以进行无损压缩（对变换系数进行熵编码）。在JPE2000标准指定的两种小波变换中，有一种便是（5,3）整型小波(可逆)变换。

因此为了便于我们理解整数小波变换，下面我们先介绍一下小波变换的提升算法，再介绍整数小波变换的算法。

5.1.1 小波变换提升算法

尽管快速小波变换算法(即 Mallat 算法)在图像压缩编码领域得到了普遍关注，但在实际应用中，仍不难发现其自身存在的弱点[56]：①快速小波变换过程中，需要跟庞大的图像数据做卷积运算，计算复杂，在一定程度上影响了压缩算法的实时性；②内存需求量较大，不适于DSP芯片的实时实现；③对图像的尺寸有要求，并不能对所有尺寸图像进行变换。

Sweldens[57,58]曾提出了一种不依赖Fourier变换的新的小波构造方法——提升方法(Lifting Scheme)，该方法既可保持原有的小波特性，又能克服平移伸缩不变性所带来的局限。这种基于提升方法的第二代小波变换，具有小波的快速算法，可以实现从整数到正数的变换，对变换后的数据进行熵编码就能实现图像的无损压缩[39, 40]。

快速提升小波变换算法的基本思想[56]是将 Mallat 算法中的每一级滤波运算分解为分裂(split)、预测(predict)和更新(update)3个过程，如图5.1所示。其中：

分裂(split)：是将输入信号其序数的奇偶性分为以下2组

$$X_0^0(n) = x(2n)$$
$$X_1^0(n) = x(2n+1)$$

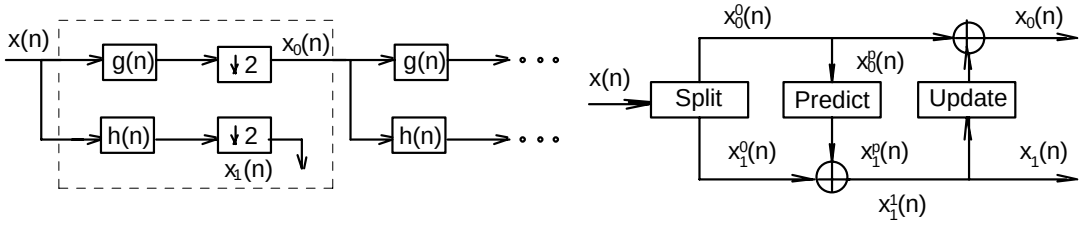
预测(predict)：是将滤波器P作用于偶信号 $X_0^0(n)$ 得到奇信号的预测值 $X_1^p(n)$，再将该预测值与奇信号 $X_1^0(n)$ 相减以得到奇信号的预测误差 $X_1^1(n)$

$$X_1^p(n) = \sum_k p_k X_0^0(n-k)$$
$$X_1^1(n) = X_1^0(n) - X_1^p(n) \tag{5.1}$$

更新(update)：是将滤波器 U 作用于奇信号的预测误差，得到偶信号 $X_1^1(n)$ 的预测值 $X_0^P(n)$，然后用来对偶信号进行校正

$$\begin{aligned} X_0^P(n) &= \sum_k u_k X_1^1(n-k) \\ X_{01}^1(n) &= X_0^0(n) - X_0^P(n) \end{aligned} \tag{5.2}$$

式(5.1)和式(5.2)中 k 的取值范围分别对应于滤波器 P 和 U 的阶数。



(a) Mallat 算法　　（b）Lifting 算法

图 5.1　Mallat 算法与 Lifting 算法相互关系示意图

MALLAT 算法中任何一组小波滤波器(h(n)，g(n))，均可以通过因式分解得到相应的一组或多组预测滤波器(P)和更新滤波器(U)，并可在分裂以后进行多次的预测和更新操作。图 5.1(b)仅表示了一组滤波器，其最后结果为

$$X_0(n) = X_0^1(n)$$
$$X_1(n) = X_1^1(n)$$

显然，快速提升小波变换具有计算更快捷、能够在当前位置完成小波变换从而节省内存、能对任意尺寸图像进行变换等优点。

5.1.2　整数小波变换算法

下面以一维整数小波变换为例介绍一下的整数小波变换原理[32]。

假设一个有 N 个整数 x_i 的数据矢量（i=0，1,...，N-1），定义 k=N/2，我们分别计算变换矢量 y_i 的奇数下标和偶数下标分量。首先，讨论 N 为偶数的情况。N/2 个奇数下标分量 y_{2i+1}（i=0，1,...，K-1）等于 x_i 的差值，是细节（高频）变换系数。每个偶数下标分量 y_{2i}（i 在同样的[0，K-1]范围变化）5 个数据项的加权平均，是 N/2 个低频变换系数。通常将其变换为 N/4 个低频系数和 N/4 个高频系数。

奇数下标变换系数满足的基本条件是：

$$y_{2i+1} = -\frac{1}{2}x_{2i} + x_{2i+1} - \frac{1}{2}x_{2i+2}$$

但最后一个系数(i=k-1)例外，它是相邻两项的简单差值　$y_{2k-1} = x_{2k-1} - x_{2k-2}$　。我们可以做如下归纳：

$$y_{2i+1}=\begin{cases}x_{2i+1}-(x_{2i}+x_{2i+2})/2, & i=0,1,\cdots,k-2\\ x_{2i+1}-x_{2i}, & i=k-1\end{cases} \tag{5.3}$$

偶数下标变换系数可以用下面的加权平均来计算：

$$y_{2i}=-\frac{1}{8}x_{2i-2}+\frac{1}{4}x_{2i-1}+\frac{3}{4}x_{2i}+\frac{1}{4}x_{2i+1}-\frac{1}{8}x_{2i+2}$$

但第一个系数（i=0）例外，它满足下式：

$$y_0=\frac{3}{4}x_0+\frac{1}{2}x_1-\frac{1}{4}x_2$$

实际上，可以用 x_{2i} 和两个奇数下标系数 y_{2i-1} 和 y_{2i+1} 计算偶数下标系数 y_{2i}：

$$y_{2i}=\begin{cases}x_{2i}+y_{2i+1}/2, & i=0\\ y_{2i}+(y_{2i-1}+y_{2i+1})/4, & i=1,2,\ldots k-1\end{cases} \tag{5.4}$$

很容易得到逆变换，它用变换系数 y_i 计算与原始数据 x_i 相同的数据项 z_i。首先计算偶数下标元素：

$$z_{2i}=\begin{cases}y_{2i}-y_{2i+1}/2, & i=0\\ y_{2i}-(y_{2i-1}+y_{2i+1})/4, & i=1,2,\ldots k-1\end{cases} \tag{5.5}$$

然后计算奇数下标元素：

$$z_{2i+1}=\begin{cases}y_{2i+1}+(z_{2i}+z_{2i+2})/2, & i=0,1,\ldots k-2\\ y_{2i+1}+z_{2i}, & i=k-1\end{cases} \tag{5.6}$$

按式（5.3）和和（5.4）算出的变换系数通常都不是整数，因为要被 2 和 4 除。根据式（5.5）和（5.6）重建的数据通常也不是整数。上述特定的 IWT 的主要特征是用了截断。利用截断（用符号“$\lfloor\ \rfloor$”表示），可以生成整形变换系数 y_i 和整重建数据项 z_i。式（5.3）到（5.6）可修改为：

$$y_{2i+1}=\begin{cases}x_{2i+1}-\lfloor(x_{2i}+x_{2i+2})/2\rfloor, & i=0,1,,\ldots k-2\\ x_{2i+1}-x_{2i} & i=k-1\end{cases}$$

$$y_{2i}=\begin{cases}x_{2i}+\lfloor y_{2i+1}/2\rfloor, & i=0\\ x_{2i}+\lfloor(y_{2i-1}+y_{2i+1})/4\rfloor, & i=1,2,\ldots,k-1\end{cases} \tag{5.7}$$

$$z_{2i}=\begin{cases}y_{2i}-\lfloor y_{2i+1}/2\rfloor, & i=0\\ y_{2i}-\lfloor(y_{2i-1}+y_{2i+1})/4\rfloor, & i=1,2,\ldots,k-1\end{cases}$$

$$z_{2i+1}=\begin{cases} y_{2i+1}+\lfloor (z_{2i}+z_{2i+2})/2 \rfloor, & i=0,1,2,\ldots,k-1 \\ y_{2i+1}+z_{2i}, & i=k-1 \end{cases}$$

计算 y_i 时，截断会造成信息的丢失。但是，计算 z_i 时也用到了截断，它可恢复丢失的信息。因此方程（5.7）的确是可精确重建原始数据项的前向和后向 IWT。

5.2　整数小波分解的 VLSI 设计

5.2.1　5-3 小波分解模块 WVT53 的总体设计

1．5-3 小波分解模块(WVT53)的输入输出接口

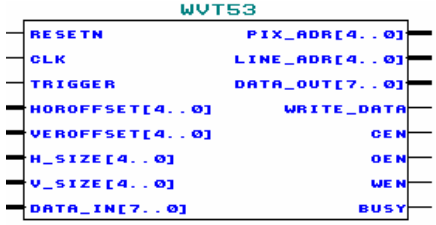


图 5.2　5-3 小波分解模块(WVT53)的输入输出接口图

5-3 小波分解模块(WVT53)的输入输出接口如图 5.2 所示。其中输入端口有：RESETN——系统复位信号，CLK——系统同步时钟信号，TRIGGER——系统工作的触发信号，HOROFFSET——5 位的存储器中的图像数据的水平偏移量，VEROFFSET——5 位的存储器中的图像数据的垂直偏移量，H_SIZE——10 位被处理图像数据的水平尺寸（每行的像素值），V_SIZE——10 位被处理图像数据的垂直尺寸（每列的像素值），DATA_IN——8 位数据输入端口。输出端口有：DATA_OUT——8 位数据输出端口，WRITE_DATA——向外部存储器输出数据使能信号，CEN——芯片选择使能信号，OEN——输出使能信号，WEN——写使能信号。

2．5-3 小波分解模块(WVT53)的总体结构设计

5-3 小波分解需完成如下主要功能：（1）从外部存储器中读取指定的图像数据；（2）将指定的图像数据进行 5-3 小波分解；（3）将小波分解的有关数据存到内部的工作存储器中；（4）根据要求向外部存储器输出数据。

根据分析，我们可将 WVT53 分解为如下几个组成部分：5-3 小波分解处理核 CD53_DOWN，外部存储器读写控制状态机 WR_EXTRAM_SM，小波分解后数据写入内部存储器的控制状态机 W_INTRAM_SM，数据处理的行计数器 CNT_LINES，数据处理的列(像素)计数器 CNT_RPIXS，高频系数计数器 CNT_AC，低频系数计数器 CNT_DC，各种操作结束控制器 FINISH_CTRL，数据和地址选择器 MUX1、MUX2、MUX3。其组成原理框图如图 5.3 所示。

其中 5-3 小波分解处理核 CD53_DOWN——用于将指定的图像数据进行 5-3 小波分

解，外部存储器读写控制状态机 WR_EXTRAM_SM——用于根据外部输入的控制信号和内部有关模块的反馈信号向外部存储器和内部有关模块发出相应的控制信号，小波分解后数据写入内部存储器的控制状态机 W_INTRAM_SM——根据内部的 CD53_DOWN 和 FINISH_CTRL 反馈的有关控制信号向内部的有关模块发出相应的控制信号，数据处理的行计数器 CNT_LINES——根据外部输入的 H_SIZE 和内部 WR_EXTRAM_SM 发出的行数增加指令计算应读取存储器的行数，数据处理的列(像素)计数 CNT_RPIXS——根据外部输入的 V_SIZE 和内部 WR_EXTRAM_SM 发出的行数增加指令计算应读取存储器的列数，高频系数计数器 CNT_AC——根据外部输入的 V_SIZE 和内部 W_INTRAM_SM 发出的控制信息计算高频系数存入内部存储器的数据个数和产生存储地址，低频系数计数器 CNT_DC——根据外部输入的 H_SIZE 和内部 W_INTRAM_SM 发出的控制信息计算低频系数存入内部存储器的数据个数和产生存储地址，各种操作结束控制器 FINISH_CTRL——根据内部各模块发出的操作结束反馈信号向内部 WR_EXTRAM_SM 和 W_INTRAM_SM 发出相应的操作控制信号，数据和地址选择器 MUX1、MUX2、MUX3——用于有关数据的锁存和输出选择。

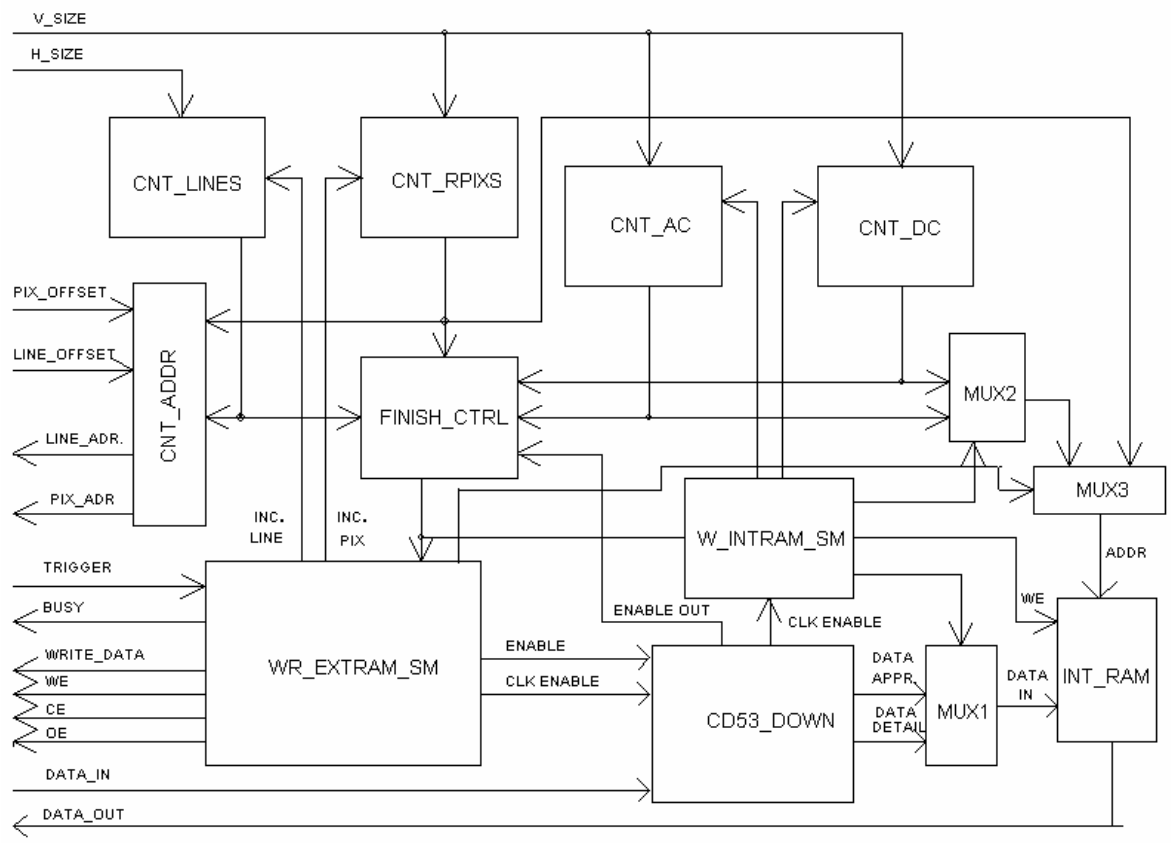


图 5.3　　5-3 小波分解模块(WVT53)的组成原理框图

5.2.2　5-3 小波分解处理核 CD53_DOWN 的设计

1．5-3 小波分解处理核 CD53_DOWN 的输入输出接口

5-3 小波分解处理核 CD53_DOWN 的输入输出接口如图 5.4 所示。其中输入端口有：ENABLEN——系统工作使能信号，低电平有效，CLK——系统同步时钟信号，RESET——系统复位信号，CLK_EN——系统工作使能信号，DATA——8 位数据输入端口。输出端口有：DOUT——8 位高通滤波输出数据，COUT——8 位低通滤波输出数据，CLK_EN——工作时钟使能输出，ENABLEOUTN——输出使能信号。

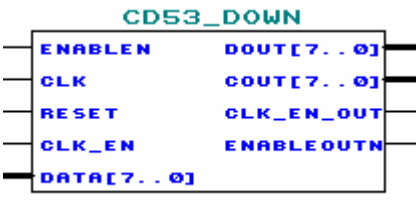


图 5.4　　5-3 小波分解处理核 CD53_DOWN 的输入输出接口图

2．5-3 小波分解处理核 CD53_DOWN 的设计

根据 5.1 节所述，我们可知（5，3）整数小波分解的算法为：

$$y_{2i+1}=\begin{cases}x_{2i+1}-\lfloor(x_{2i}+x_{2i+2})/2\rfloor, & i=0,1,,\ldots k-2\\ x_{2i+1}-x_{2i} & i=k-1\end{cases}$$

$$y_{2i}=\begin{cases}x_{2i}+\lfloor y_{2i+1}/2\rfloor, & i=0\\ x_{2i}+\lfloor(y_{2i-1}+y_{2i+1})/4\rfloor, & i=1,2,\ldots,k-1\end{cases}$$

因此整个数据处理可分为五个过程：（1）高通滤波数据的输入组织；（2）高通滤波数据的处理；（3）低通滤波数据的输入组织；（4）低通滤波数据的处理；（5）数据输出的组织。而高通滤波数据的输入组织，是通过 3 个数据锁存器的串级使用，将外部串行输入的数据转换成 3 个并行输出的数据流。低通滤波数据的输入组织，主要是通过锁存器实现低通滤波数据处理的缓冲。高通和低通滤波数据的处理运算含有加法、减法和除法，加、减法运算比较简单，而除法运算用硬件直接实现比较复杂，这里考虑到除 2 和除 4 的特定运算，使用算数右移的方法来实现除法和取整，其原理如图 5.5 所示。数据输出的组织，可通过锁存器实现数据的按要求输出。根据以上设计

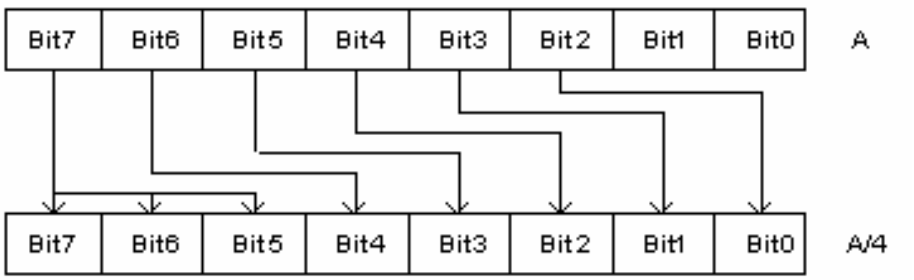


图 5.5　　5-3 小波分解中的除法及取整原理框图

思想，我们可得到 5-3 小波分解数据处理原理框图如图 5.6 所示。同时要使这 5 个部分能够协调的工作，整个系统应有一个组织指挥中心协调各部分工作，因此我们应该

给整个系统加上一个控制模块 CTRL_SM，这里我们可通过一个状态机来描述。

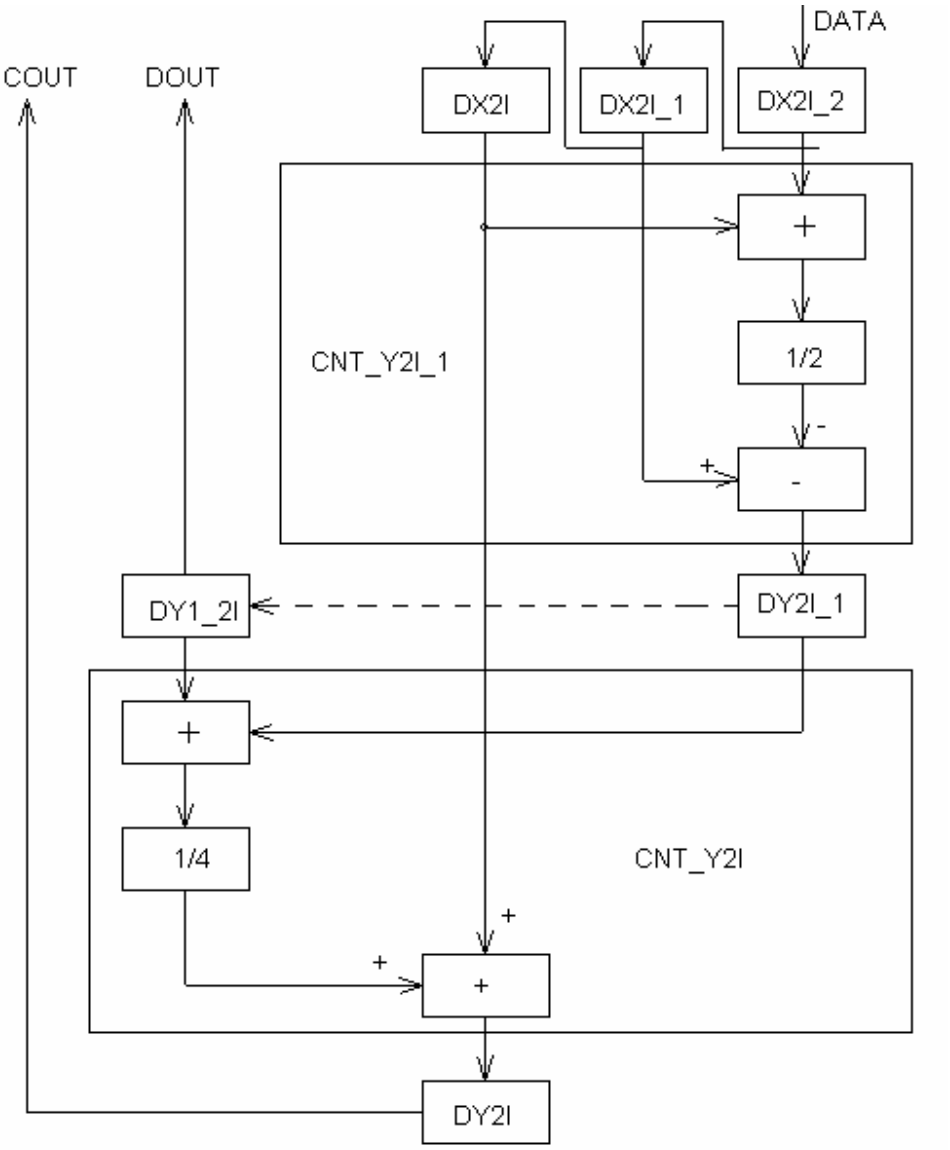


图 5.6 5-3 小波分解数据处理原理框图

综上所述，我们可得到 5-3 小波分解模块(CD53_DOWN)的组成原理框图如图 5.7 所示。其中 CTRL_SM 为 5-3 小波分解控制状态机 CTRL_SM，CNT_Y2I_1、CNT_Y2I 分别为高通和低通滤波数据处理模块，DX2I,DX2I_1,DX2I_2,DY1_2I,REG0,REG1,REG2 为数据寄存器/锁存器，MUX1，MUX2 为数据选择器。

5.2.3 外部存储器读写控制状态机 WR_EXTRAM_SM:

外部存储器读写控制状态机的工作状态为：初始状态 IDLE，读行状态 READ LINE，读列（像素）状态 READ PIXEL，产生时钟使能状态 GENERATION CLK_EN，计数器复位状态 RESET COUNTER，写行状态 WRITE LINE，写列（像素）状态 WRITE PIXEL（1），WRITE PIXEL（2）。处理状态图如图 5.8 所示。

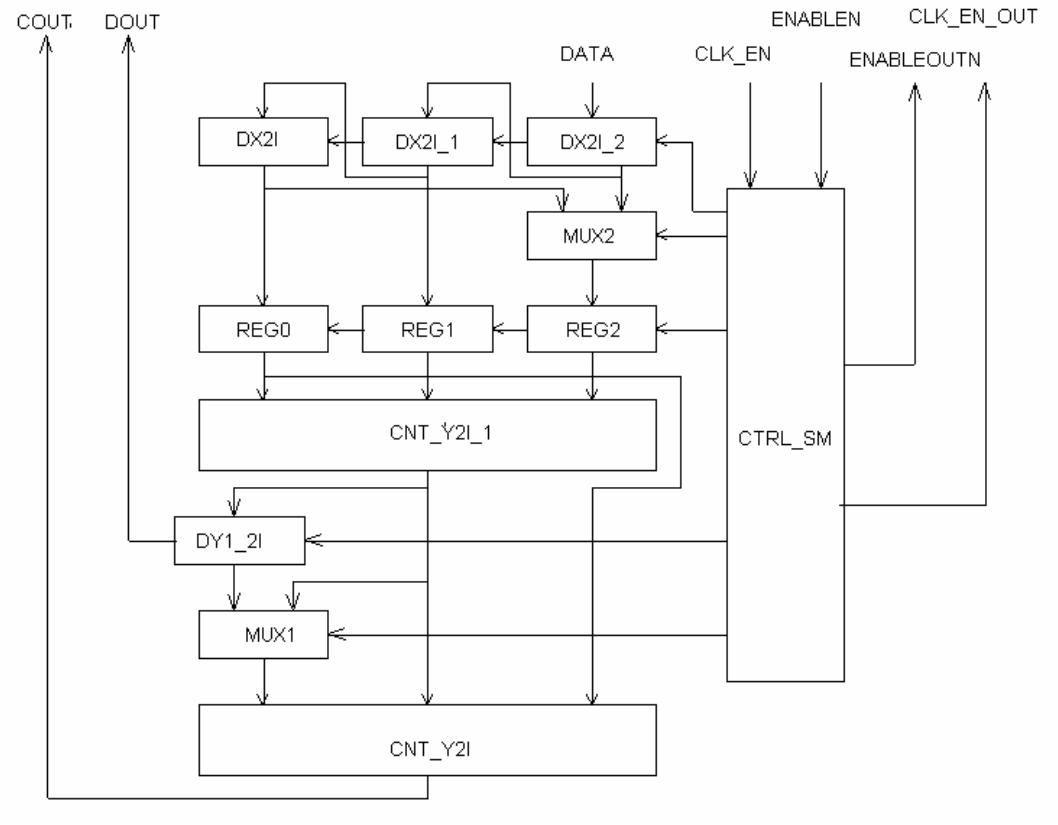


图 5.7　　5-3 小波分解模块(CD53_DOWN)的组成原理框图

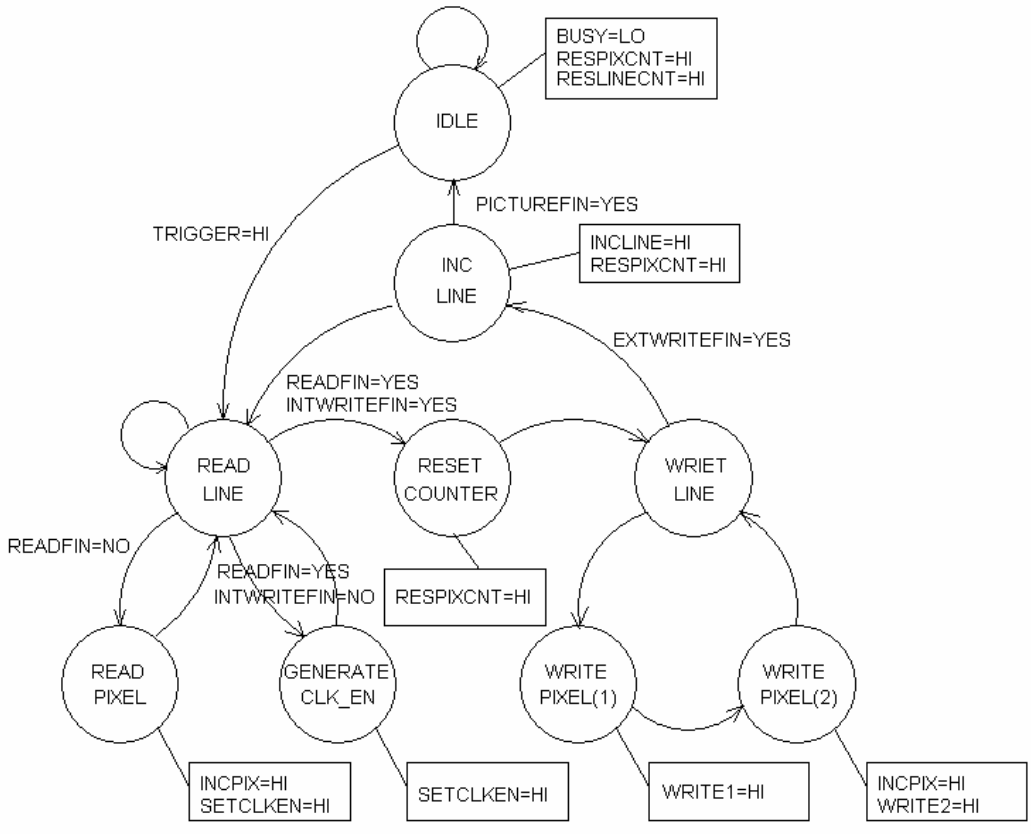


图 5.8　　外部存储器读写控制状态机 WR_EXTRAM_SM

5.2.4　5-3 小波分解模块 WVT53 的 VHDL 程序设计

根据以上的设计分析，我们首先将比较复杂的 5-3 小波分解处理核（CD53_DOWN）、内部存储器 INT_RAM 单独进行 VHDL 程序设计和调试，调试完毕后，再进行 5-3 小波分解模块 WVT53 的 VHDL 程序设计，该程序既包括对前面已经设计模块的调用，又包括一系列用于描述有关处理功能的进程或语句（组）。具体的 VHDL 程序略。

5.2.5　5-3 小波分解有关模块的仿真及分析

图 5.9 是 5-3 小波分解处理核 CD53_DOWN 的仿真结果。表 5.1 是 5-3 小波分解仿真结果 DOUT 分析表，表 5.2 是 5-3 小波分解仿真结果 COUT 分析表。从表中可以看出该模块的程序设计是正确的。

图 5.10 是 5-3 小波分解模块(WVT53)的 VHDL 程序仿真结果，经分析，仿真结果是正确的。

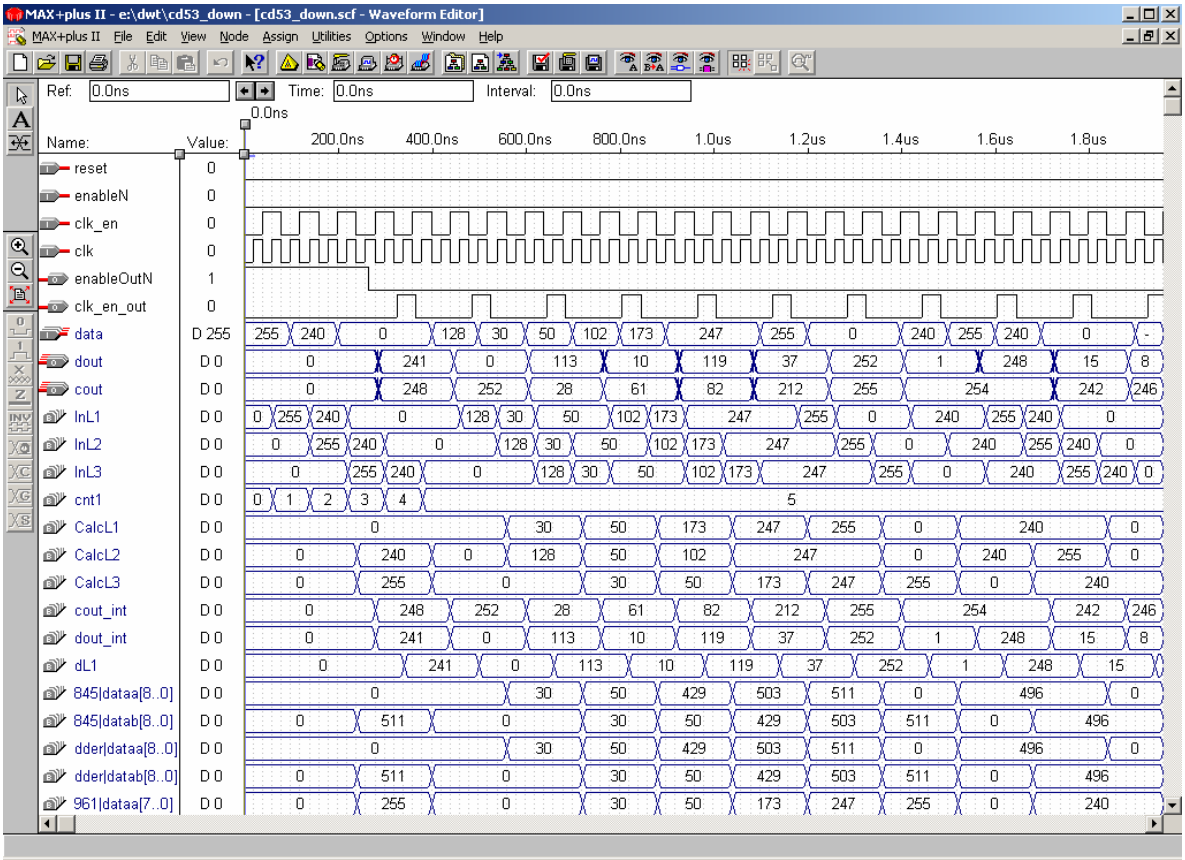


图 5.9　　5-3 小波分解处理核 CD53_DOWN 的仿真结果

表 5.1　　5-3 小波分解仿真结果 DOUT 分析表

序号	P2I 补/P2I 原	P2I_1 补/P2I_1 原	P2I_2 补/P2I_2 原	C2I_1 补/C2I_1 原	DOUT 补/DOUT 原
1	255/-1	240/-16	0/0	241/-15	241/-15
2	0/0	0/0	0/0	0/0	0/0
3	0/0	128/-128	30/30	113/-143	113/-143
4	30/30	50/50	50/50	10/10	10/10
5	50/50	102/102	173/-83	118/118	119/119

表 5.2　　5-3 小波分解仿真结果 COUT 分析表

序号	DOUT 补/DOUT 原	P2I 补/P2I 原	C1_2I 补/C1_2I 原	C2I_1 补/C2I_1 原	COUT 补/COUT 原
1	241/-15	255/-1	0/0	241/-15	252/-4
2	0/0	0/0	241/-15	0/0	252/-4
3	113/-143	0/0	0/0	113/-143	28/28
4	10/10	30/30	113/-143	10/10	61/61
5	119/119	50/50	10/10	119/119	82/82

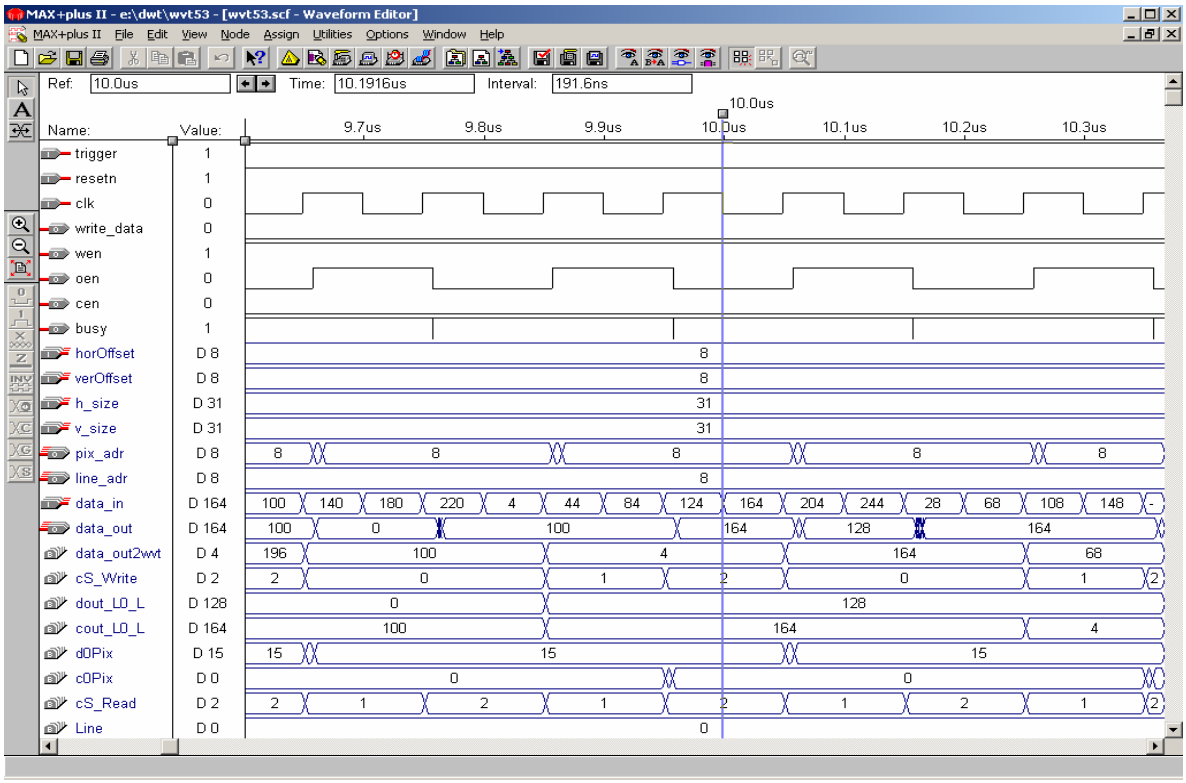


图 5.10　小波分解模块(WVT53)的仿真结果

5.3　整数小波合成的 VLSI 设计

5.3.1　5-3 小波合成模块 IWVT53 的总体设计

1．5-3 小波合成模块(IWVT53)的输入输出接口

5-3 小波合成模块(IWVT53)的输入输出接口如图 5.11 所示。其中输入端口有：RESETN——系统复位信号，CLK——系统同步时钟信号，TRIGGER——系统工作的触发信号，HOROFFSET——5 位的存储器中的图像数据的水平偏移量，VEROFFSET——5 位的存储器中的图像数据的垂直偏移量，H_SIZE——10 位被处理图像数据的水平尺寸（每行的像素值），V_SIZE——10 位被处理图像数据的垂直尺寸（每列的像素值），DATA_IN——8 位数据输入端口。输出端口有：DATA_OUT——8 位数据输出端口，WRITE_DATA——向外部存储器输出数据使能信号，CEN——芯片选择使能信号，OEN——输出使能信号，WEN——写使能信号。

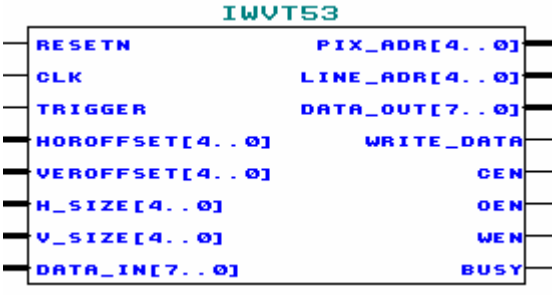


图 5.11　　5-3 小波合成模块(IWVT53)的输入输出接口图

2．5-3 小波合成模块(IWVT53)的总体结构设计

5-3 小波合成需完成如下主要功能：（1）从外部存储器中读取指定的图像数据；（2）将指定的图像数据进行 5-3 小波合成；（3）将小波合成的有关数据存到内部的工作存储器中；（4）根据要求向外部存储器输出数据。

根据分析,我们可将 IWVT53 分解为如下几个组成部分：5-3 小波合成处理核 CD53_UP,外部存储器读写控制状态机 WR_EXTRAM_SM,数据处理的行计数器 CNT_LINES,数据处理的列(像素)计数器 CNT_RPIXS，高频系数计数器 CNT_AC，低频系数计数器 CNT_DC，各种操作结束控制器 FINISH_CTRL，选择器 MUX1。其组成原理框图如图 5.12 所示。

其中 5-3 小波合成处理核 CD53_UP——用于将指定的图像数据进行 5-3 小波合成，外部存储器读写控制状态机 WR_EXTRAM_SM——用于根据外部输入的控制信号和内部有关模块的反馈信号向外部存储器和内部有关模块发出相应的控制信号，数据处理的行计数器 CNT_LINES——根据外部输入的 H_SIZE 和内部 WR_EXTRAM_SM 发出的行数增加指令计算应读取存储器的行数，数据处理的列(像素)计数 CNT_RPIXS——根据外部输入的 V_SIZE 和内部 CD53_UP 发出的时钟使能信号计算应存入存储器的地址，高频系数计数器 CNT_AC——根据外部输入的 V_SIZE 和内部 W_INTRAM_SM 发出的控制信息计算从外存读入小波合成处理核进行小波合成的高频系数，低频系数计数器 CNT_DC——根据外部输入的 H_SIZE 和内部 W_INTRAM_SM 发出的控制信息计算从外存读入小波合成处理核进行小波合成的低频系数，各种操作结束控制器 FINISH_CTRL——根据

内部各模块发出的操作结束反馈信号向内部 WR_EXTRAM_SM 发出相应的操作控制信号，选择器 MUX——用于有关控制信息的锁存和输出选择。

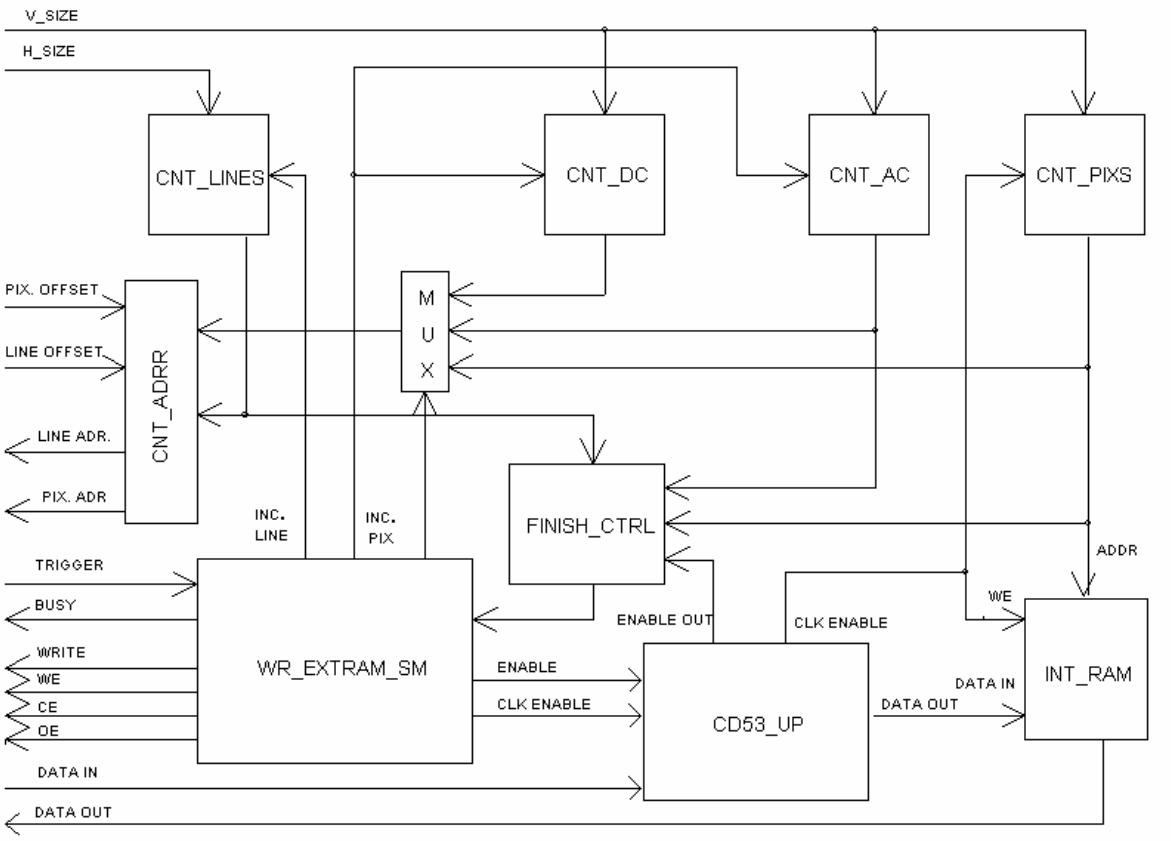


图 5.12　　5-3 小波合成模块(IWVT53)的组成原理框图

5.3.2　5-3 小波合成处理核 CD53_UP 的设计

1．5-3 小波合成处理核 CD53_UP 的输入输出接口

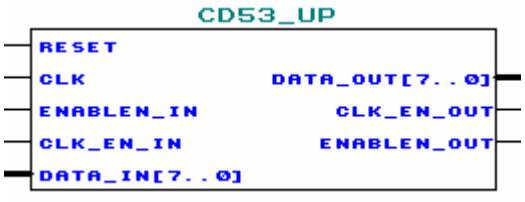


图 5.13　　5-3 小波合成处理核 CD53_UP 的输入输出接口图

5-3 小波合成处理核 CD53_UP 的输入输出接口如图 5.13 所示。其中输入端口有：RESET——系统复位信号，CLK——系统同步时钟信号，ENABLEN_IN——系统输入工作使能信号，低电平有效， CLK_EN_IN——系统时钟输入工作使能信号，DATA_IN——8 位数据输入端口。输出端口有：DATA_OUT——8 位小波合成数据输出，CLK_EN_OUT——工作时钟使能输出，ENABLE_OUTN——输出使能信号。

2．5-3 小波合成处理核 CD53_UP 的设计

根据 5.1 节所述，我们可知（5，3）整数小波合成的算法为：

$$z_{2i}=\begin{cases} y_{2i}-\lfloor y_{2i+1}/2\rfloor, & i=0 \\ y_{2i}-\lfloor (y_{2i-1}+y_{2i+1})/4\rfloor, & i=1,2,\ldots,k-1 \end{cases}$$

$$z_{2i+1}=\begin{cases} y_{2i+1}+\lfloor (z_{2i}+z_{2i+2)}/2\rfloor, & i=0,1,2,\ldots,k-1 \\ y_{2i+1}+z_{2i}, & i=k-1 \end{cases}$$

因此整个数据处理可分为五个过程：（1）高通滤波数据的输入组织；（2）高通滤波数据的处理；（3）低通滤波数据的输入组织；（4）低通滤波数据的处理；（5）数据输出的组织。而高通滤波数据的输入组织，是通过 3 个数据锁存器的串级使用，将外部串行输入的数据转换成 3 个并行输出的数据流。低通滤波数据的输入组织，主要是通过锁存器实现低通滤波数据处理的缓冲。高通和低通滤波数据的处理运算含有加法、减法和除法，加、减法运算比较简单，而除法运算用硬件直接实现比较复杂，这里考虑到除 2 和除 4 的特定运算，使用算数右移的方法来实现除法和取整，其原理如图 5.5 所示。数据输出的组织，可通过锁存器实现数据的按要求输出。根据以上设计思想，我们可得到 5-3 小波合成数据处理原理框图如图 5.14 所示。同时要使这 5 个

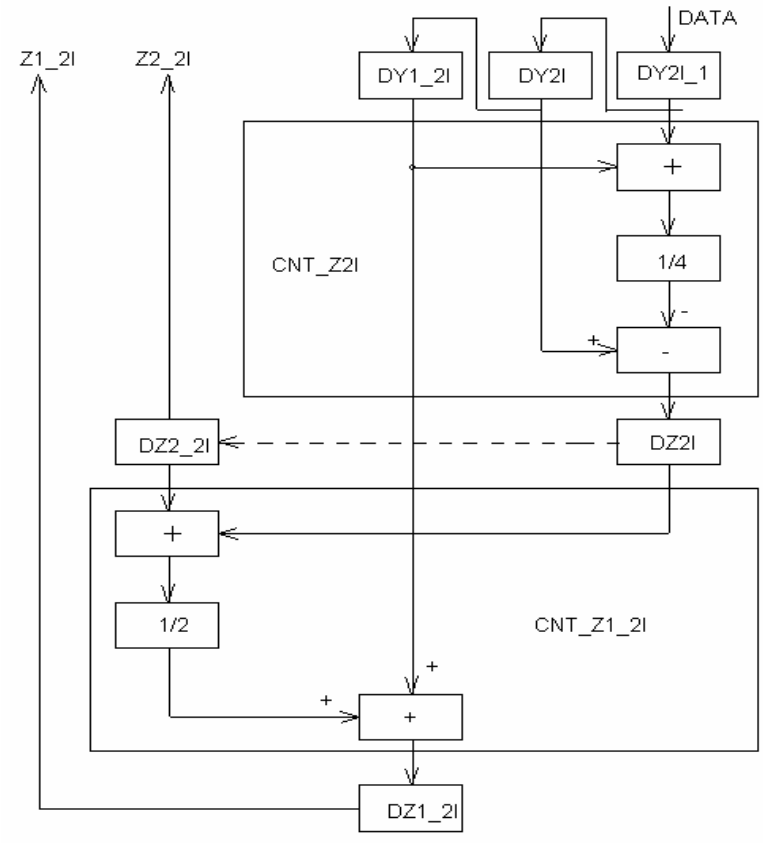


图 5.14　　5-3 小波合成数据处理原理框图

部分能够协调的工作，整个系统应有一个组织指挥中心协调各部分工作，因此我们应该给整个系统加上一个控制模块 CTRL_SM，这里我们可通过一个状态机来描述。

综上所述，我们可得到 5-3 小波合成模块(CD53_UP)的组成原理框图如图 5.15 所

示。其中 CTRL_SM 为 5-3 小波合成控制状态机 CTRL_SM，CNT_Z2I、CNT_Z1_2I 分别为高通和低通滤波数据处理模块，DY1_2I,DY2I,DY2I_1,DZ2_2I,REG0,REG1,REG2 为数据寄存器/锁存器，MUX1，MUX2,MUX3 为数据选择器。

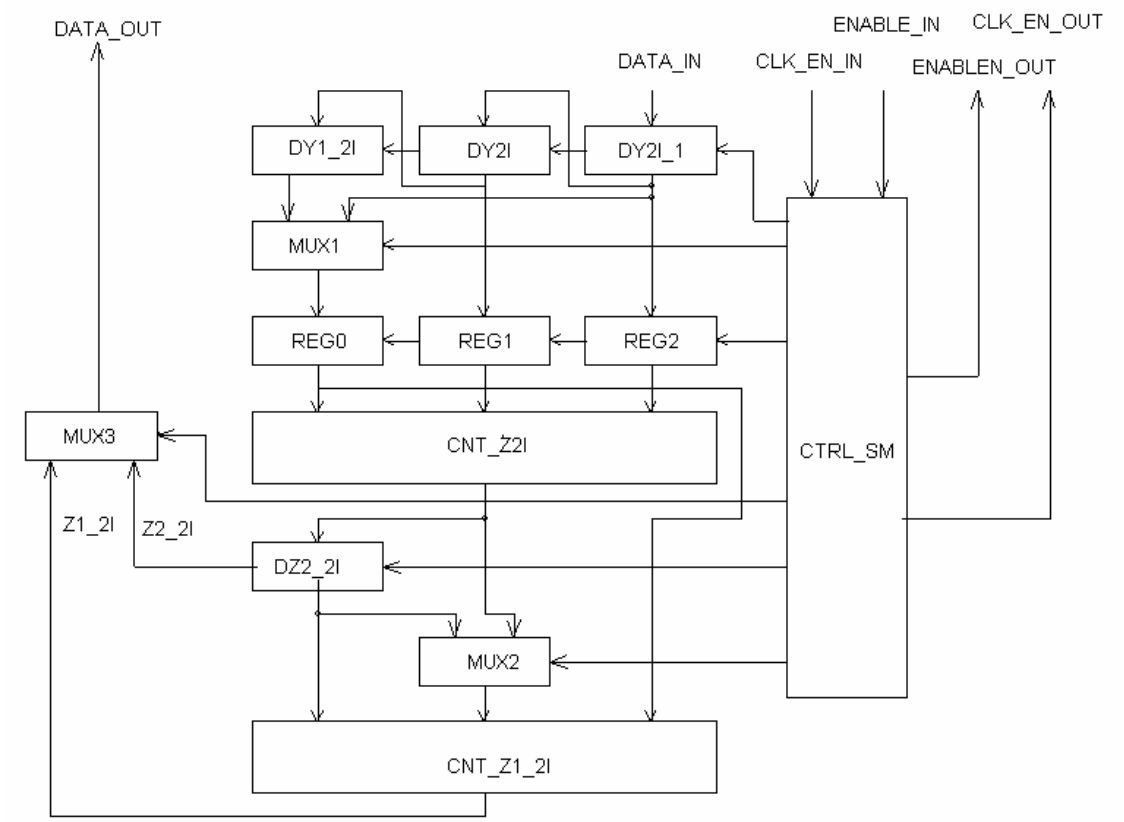


图 5.15　5-3 小波合成模块(CD53_UP)的组成原理框图

5.3.3　外部存储器读写控制状态机 WR_EXTRAM_SM

外部存储器读写控制状态机的工作状态为：初始状态 IDLE，读行状态 READ LINE，读列（像素）状态 READ PIXEL，产生时钟使能状态 GENERATION CLK_EN，计数器复位状态 RESET COUNTER，写行状态 WRITE LINE，写列（像素）状态 WRITE PIXEL（1），WRITE PIXEL（2）。处理状态图如图 5.16 所示。

5.3.4　5-3 小波合成模块 IWVT53 的 VHDL 程序设计

根据以上的设计分析，我们首先将比较复杂的 5-3 小波合成处理核（CD53_UP）、内部存储器 INT_RAM 单独进行 VHDL 程序设计和调试，调试完毕后，再进行 5-3 小波合成模块 WVT53 的 VHDL 程序设计，该程序既包括对前面已经设计模块的调用，又包括一系列用于描述有关处理功能的进程或语句（组）。具体的 VHDL 程序略。

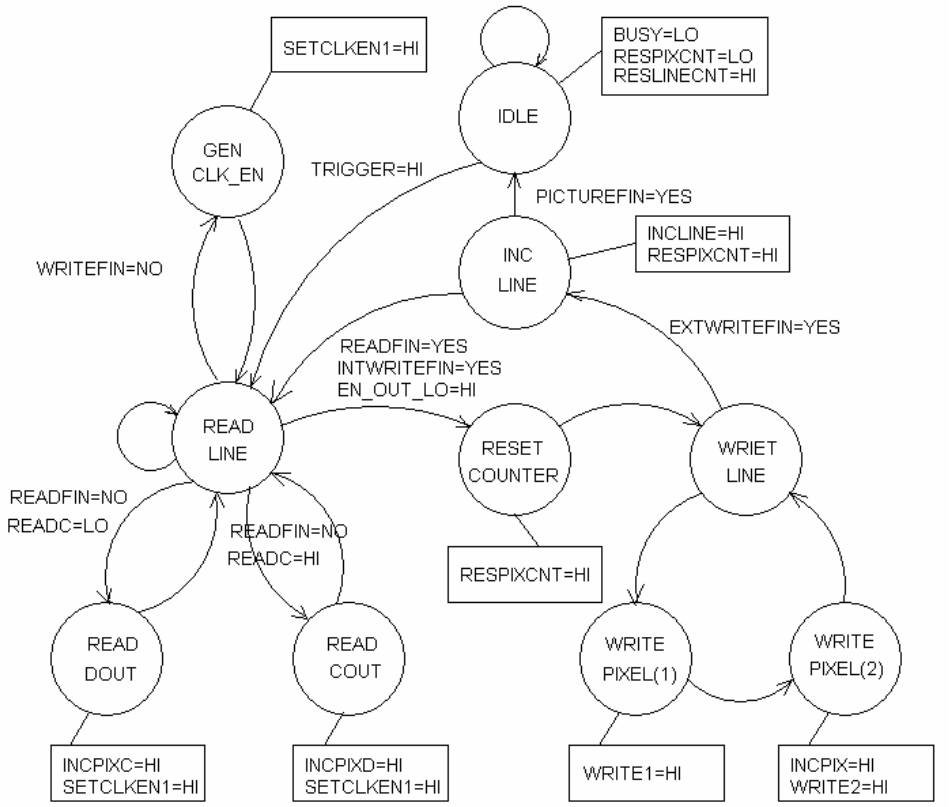


图 5.16　外部存储器读写控制状态机 WR_EXTRAM_SM

5.3.5　5-3 小波合成有关模块的仿真及分析

图 5.17 是 5-3 小波合成处理核 CD53_UP 的仿真结果。表 5.3 是 5-3 小波合成仿真结果 DOUT_2I 分析表，表 5.4 是 5-3 小波合成仿真结果 DOU_2I_2 分析表。从表中可以看出该模块的程序设计是正确的。图 5.18 是 5-3 小波合成模块(IWVT53)的 VHDL 程序仿真结果，经分析，仿真结果是正确的。

表 5.3　　5-3 小波合成仿真结果 DOUT_2I 分析表

序号	P2I 补/P2I 原	P1_2I 补/P1_2I 原	P2I_1 补/P2I_1 原	C2I 补/C2I 原	DATA_OUT 补/DATA_OUT 原
1	180/-78	4/4	4/4	178/-78	178/-78
2	84/84	4/4	164/164	105/105	106/106
3	244/-12	164/-92	68/68	249/-7	250/-6
4	148/-108	68/68	228/-28	138/-118	138/-118
5	52/52	228/-28	132/-124	89/89	90/90

表 5.4　　5-3 小波合成仿真结果 DOUT_2I_1 分析表

序号	DOUT 补/DOUT 原	P2I_1 补/P2I_1 原	C1_2I 补/C1_2I 原	C2I_2 补/C2I_2 原	COUT 补/COUT 原
1	106/106	4/4	178/-78	106/106	18/18
2	250/-6	164/-92	106/106	250/-6	214/-42
3	138/-118	68/68	250/-6	138/-118	6/6
4	90/90	228/-28	138/-118	90/90	214/-42
5	234/-32	132/-124	90/90	234/-22	166/-90

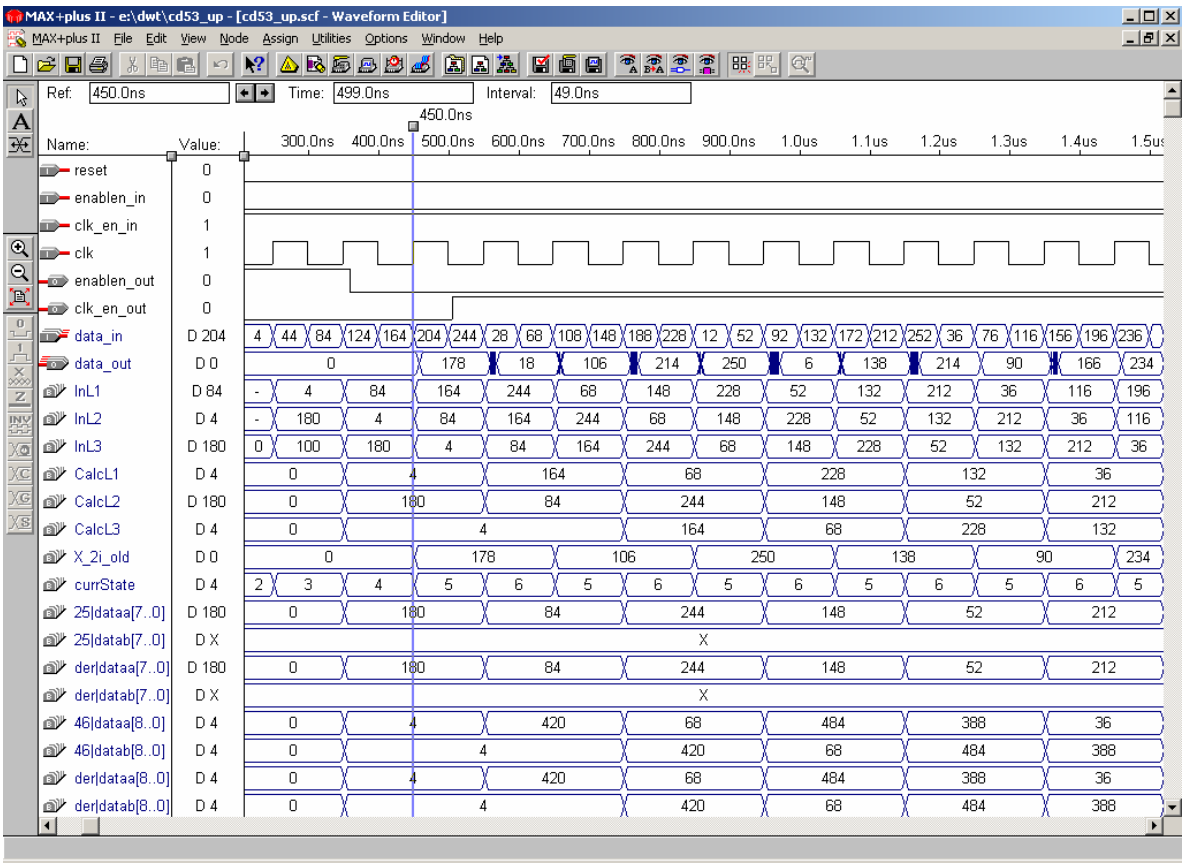


图 5.17　　5-3 小波合成处理核 CD53_UP 的仿真结果

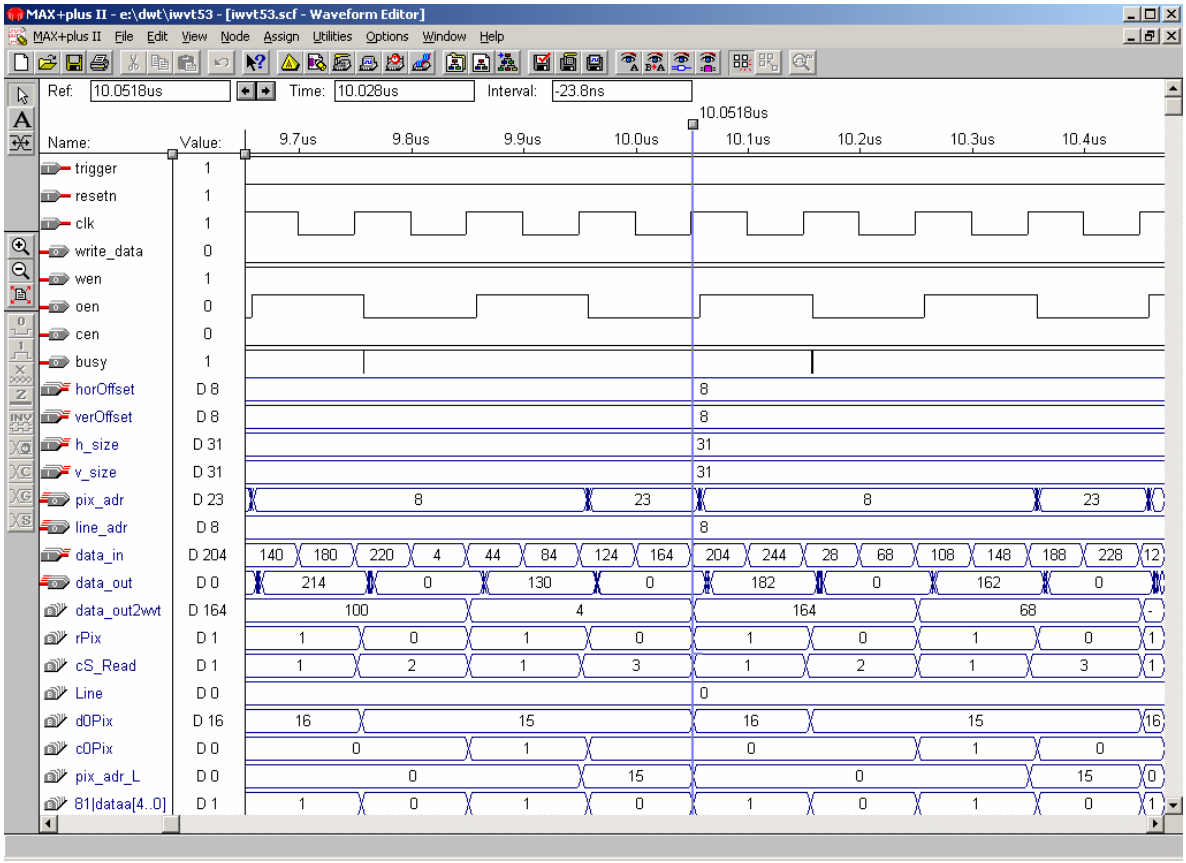


图 5.18　　小波合成模块(IWVT53)的仿真结果

5.4 多级小波变换的实现讨论

前面5.2节和5.3节设计的是小波变换某一级处理的模块，但是在实际的小波变换图像处理中，小波变换往往需要进行多级处理，那么对于本系统的设计怎样扩充成多级小波变换（包括分解和合成）的系统呢？

我们可增加一个控制模块并将前面已经设计好的小波分解模块和小波合成模块组成一个整体即可扩充成多级小波变换（包括分解和合成）的系统，而系统中的小波分解和合成处理可在控制模块中增加两个外接的控制开关来完成处理的选择。

增加的控制模块主要完成如下功能：（1）对小波变换的处理设置一个变换级数的计数器，以记录系统当前变换所处的级数。（2）根据当前处理的级数，计算本级分解或合成数据处理的数据存取控制参数。（3）根据系统所处的工作状态，控制系统水平变换和垂直变换的数据存取控制参数的赋值。

下面根据以上设计思想给出了执行上述有关功能的有关进程或语句。

```
--小波变换级数计算进程，程序中的LEVEL为级数计数器
PROCESS (STATECLK, RESET, INCLEVEL,DATAREADY,DECLEVEL) IS
  BEGIN
  IF ((RESET=HI) OR (DATAREADY=LO)) THEN
    LEVEL <= "00";
  ELSIF STATECLK'EVENT AND STATECLK=HI THEN
    IF INCLEVEL = HI THEN
      LEVEL <= LEVEL + 1;
    ELSIF DECLEVEL = HI THEN
      LEVEL <= LEVEL - 1;
    END IF;
  END IF;
END PROCESS;

--不同级数小波变换数据存取控制参数的计算进程，H_SIZE为被处理图像数据的
--水平尺寸（每行像素值），V_SIZE为被处理图像数据的垂直尺寸（每列像素值）
PROCESS (LEVEL) IS
  BEGIN
  CASE LEVEL IS
    WHEN "00" =>
                H_SIZE_2 <= H_SIZE;
```

```
                V_SIZE_2 <= V_SIZE;
    WHEN "01" =>
                H_SIZE_2 <= ("0" & H_SIZE(9 DOWNTO 1));
                V_SIZE_2 <= ("0" & V_SIZE(9 DOWNTO 1));
    WHEN "10" =>
                H_SIZE_2 <= ("00" & H_SIZE(9 DOWNTO 2));
                V_SIZE_2 <= ("00" & V_SIZE(9 DOWNTO 2));
    WHEN "11" =>
                H_SIZE_2 <= ("000" & H_SIZE(9 DOWNTO 3));
                V_SIZE_2 <= ("000" & V_SIZE(9 DOWNTO 3));
    WHEN OTHERS =>
                H_SIZE_2 <= ("0000" & H_SIZE(9 DOWNTO 4));
                V_SIZE_2 <= ("0000" & V_SIZE(9 DOWNTO 4));
  END CASE;
END PROCESS;

--小波变换水平分解和垂直分解数据存取控制参数的选择
ADR_HOR <= (LINE_ADR(3 DOWNTO 0) & PIX_ADR(4 DOWNTO 0));
ADR_VERT <= (PIX_ADR(3 DOWNTO 0) & LINE_ADR(4 DOWNTO 0));
H_SIZE_WVT <= V_SIZE_2 WHEN (V_TRANS_WVT_BUSY=HI) ELSE H_SIZE_2;
V_SIZE_WVT <= H_SIZE_2 WHEN (V_TRANS_WVT_BUSY=HI) ELSE V_SIZE_2;
VEROFFSET_WVT <= HOROFFSET WHEN (V_TRANS_WVT_BUSY=HI) ELSE VEROFFSET;
HOROFFSET_WVT <= VEROFFSET WHEN (V_TRANS_WVT_BUSY=HI) ELSE HOROFFSET;
ADR_WVT <= ADR_VERT WHEN (V_TRANS_WVT_BUSY=HI) ELSE ADR_HOR;

--小波变换水平合成和垂直合成数据存取控制参数的选择
ADR_HOR_I  <= (LINE_ADR_I(3 DOWNTO 0) & PIX_ADR_I(4 DOWNTO 0));
ADR_VERT_I <= (PIX_ADR_I(3 DOWNTO 0) & LINE_ADR_I(4 DOWNTO 0));
H_SIZE_I <= H_SIZE_2 WHEN (H_TRANS_IWVT_BUSY=HI) ELSE V_SIZE_2;
V_SIZE_I <= V_SIZE_2 WHEN (H_TRANS_IWVT_BUSY=HI) ELSE H_SIZE_2;
VEROFFSET_I <= VEROFFSET WHEN (H_TRANS_IWVT_BUSY=HI) ELSE HOROFFSET;
HOROFFSET_I <= HOROFFSET WHEN (H_TRANS_IWVT_BUSY=HI) ELSE VEROFFSET;
ADR_IWVT <= ADR_HOR_I WHEN (H_TRANS_IWVT_BUSY=HI) ELSE ADR_VERT_I;
```

5.5 结论

在本系统的设计中，在小波变换算法的选择上，选择了一种优化的小波变换算法——基于提升格式的整数小波变换，该变换不仅具有计算快捷，能够在当前位置完成小波变换从而节省内存，能对任意尺寸图像进行小波变换等优点，而且是可逆的，它既可以对图像进行有损压缩，也可以进行无损压缩。在整数小波变换的 VLSI 的实现设计中，通过自顶向下的多级分层设计方法简化了模块的设计难度，通过采用多种形式的数据寄存或锁存满足了系统数据串行输入和小波变换处理数据的邻域性并行要求以及后续处理需利用先前处理结果的数据要求，从而实现了处理数据的缓冲和小波变换的并行处理和流水线处理，通过数据的移位运算简单而快速地实现了数据的乘除运算及除法运算结果的取整操作，通过设计多级的控制状态机实现了小波变换的复杂时序控制，通过设计通用的小波分解和合成模块以及小波变换存取控制参数的控制模块，实现了小波的多级变换。整个系统处理快捷，节省内存，能对任意尺寸图像进行小波变换。

第六章 总结与展望

6.1 全文总结

提高图像处理的速度一直是图像处理中致力于解决又难以突破的关键问题之一，图像并行处理技术则是提高图像处理速度的最有效的技术。EDA 技术的出现和 VLSI 的发展,为经济快速、自行开发高性能的图像并行处理硬件提供了一个全新的设计平台和广阔的发展空间。

本文在分析了邻域处理的算法及其并行数据结构，邻域图像并行处理机等芯片级图像并行处理器设计技术的基础上，重点阐述了作者利用 EDA 技术完成的 3 个图像并行处理的 VLSI 实现设计实例：Sobel 图像边缘检测器，Laplacian 图像边缘检测器以及整数小波变换的 VLSI 实现设计。各设计实例包括算法介绍，系统的总体设计，主要模块的设计思想,有关仿真结果和分析。

通过三个数字图像并行处理的 VLSI 的实现设计可知，为了提高图像处理器的速度和节约图像处理器的硬件实现资源，总的设计原则是：根据系统的设计要求，选用合适的并行处理算法、采用合适的并行数据结构、构建多层次的并行系统结构、运用多种并行实现手段。

在系统 VLSI 实现的具体设计中，主要应解决三个关键模块的设计，即构建高速而节约资源的数据处理模块，构建满足并行处理要求的数据输入和输出模块，构建能使整个系统协调工作的时序控制发生器。而构建高速而节约资源的数据处理模块，需要研究的内容有：处理算法的选择，算法并行性的挖掘，各运算处理的先后顺序的确定，前后处理数据的相互利用，系统中各处理模块的平衡，流水线处理、分布式计算、重复设置并行处理部件等并行实现手段的综合应用等。构建满足并行处理要求的数据输入和输出模块，需要研究的内容有：数据缓冲、数据寄存/锁存、串并转换、并串转换、内外存储器的设置，实际上就是根据并行处理的要求构建满足并行数据处理要求的数据存取系统。构建能使整个系统协调工作的时序控制发生器，需要研究的内容有：系统内各模块的工作时序要求、系统中各模块间的工作时序要求和协调等，实际上就是整个系统的控制信号系统。

6.2 后继工作展望

本文虽然是硕士学位论文研究工作的总结，但是实际上只是我跨入“数字图像并行处理的 VLSI 实现”这个有着深远的实际意义和广阔的应用前景的科学研究殿堂的良好开端，真正的研究工作才刚刚起步，我将继续完善和深入如下研究：（1）小波变

换及其它数字图像处理算法研究与改进：通过对现有有关算法的对比研究和改进研究，以期改进算法和提出新的算法;(2)深化小波变换及其它数字图像处理算法的 VLSI 的实现研究：通过 VLSI 架构的对比，主要模块的实现设计，以及实例设计，实现 VLSI 实现设计的突破；（3）VLSI 实现研究方法的沟通和融合：将基于 MATLAB 的计算机辅助分析与基于 EDA 技术的实现，通过模型的对接，数据的互通，结果的对比，实现研究方法的沟通和融合；（4）工程应用研究：通过选择某个方面的工程应用重点，先进行点的突破，再由点到面进行推广应用。

参考文献

[1] 阮秋琦. 数字图像处理学[M]. 北京：电子工业出版社，2001

[2] [美]Rafael C. Gonzalez，Richard E. Woods. 数字图像处理（第二版）（英文版）[M]. 北京：电子工业出版社，2002

[3] 苏光大.图像并行处理技术.北京：清华大学出版社，2002

[4] 谭会生,张昌凡.EDA 技术及应用（第二版）[M]. 西安：西安电子科技大学出版社，2004

[5] 谭会生,瞿遂春.EDA 技术综合应用实例与分析[M]. 西安：西安电子科技大学出版社，2004

[6] 谭会生.EDA 技术基础[M].长沙:湖南大学出版社,2004

[7] 张欣. VLSI 数字信号处理——设计与实现[M]. 北京：科学出版社，2003

[8] 潘松，黄继业，王国栋. 现代 DSP 技术[M]. 西安：西安电子科技大学出版社，2001

[9] [日]谷隆嗣（编），伊藤秀男，野口孝树，藤原洋（著），崔东印（译）. VLSI 与数字信号处理[M]. 北京：科学出版社，2003

[10] (美)Uwe Meyer-Baese 著.刘凌,胡永生 译 .数字信号处理的 FPGA 实现[M].北京：清华大学出版社，2002.6

[11] 戴逸民,梁晓霞,裴小平.基于 DSP 的现代电子系统设计.北京:电子工业出版社,2002

[12] 王波. 利用可编程逻辑器件设计 LED 显示屏:[优秀硕士学位论文]. 南京：南京理工大学，2002

[13] 胡旸.宽带无线数据传输 64QAM 突发模式解调器设计:[优秀硕士学位论文].杭州:浙江大学，2003

[14] 黄晓雷. 单板、单片高速数据采集系统研究与设计:[优秀硕士学位论文]. 成都：电子科技大学，2001

[15] 王天明. 多通道通用数据采集系统的设计与实现:[优秀硕士学位论文]. 哈尔滨：哈尔滨工程大学，2003

[16] 贺今朝. 一种基于 FPGA 的模糊控制器的研究:[优秀硕士学位论文]. 大连：大连理工大学，2002

[17] 杨述斌. 图像边缘检测技术概述[J]. 武汉化工学院学报 2003,25(3):73～76

[18] 孙慧,周红霞，李朝晖. 图像处理中边缘检测技术的研究[J]. 电脑发与应用,2002,15(10): 7～9

[19] 李弼程，罗建书. 小波分析及其应用[M]. 北京：电子工业出版社，2003

[20] 孟军，魏同立，吴金等. 数字图像离散小波变换的原理与硬件实现分析[J]. 东

南大学学报（自然科学版），2002，32（6）：842～847
[21] 张艳.分布并行算法设计、分析与实现:[优秀博士学位论文].成都:电子科技大学,2001
[22] 王向前．高效图像数据压缩及其硬件实现:[优秀硕士学位论文]．西安：西安电子科技大学，2001
[23] 龚惠民．视频解码系统设计:[优秀硕士学位论文]．杭州:浙江大学，2002
[24] 叶丰．基于小波压缩的多路实时监控系统的设计与实现:[优秀硕士学位论文]．杭州:浙江大学，2002
[25] 姜勇．基于FPGA的实时图像处理系统的研究:[优秀硕士学位论文]．长春:长春理工大学，2002
[26] 宋芳莉．图像边缘检测中的方法研究:[优秀硕士学位论文]．西安:西北大学，2002
[27] 秦前清，杨宗凯．实用小波分析[M]．西安：电子科技大学出版社，1994
[28] 葛伟．基于小波的图像处理技术的实现与探讨[J]．现代计算机，2002，5：6～11，20
[29] 鲁业频．几种常见编码方法的比较[J]．电视技术，2002，4：16-19
[30] 屈稳太,诸静.图像变换中的两个关键技术——滤波器的正则性与信号的边界处理[J].浙江大学学报(工学版)，2003，37（3）：185～189，207
[31] 胡昌华,张军波,夏军等.基于MATLAB的系统分析与设计——小波分析[M].西安：电子科技大学出版社，1999
[32] [美]David Salomon（著），吴乐南（译）.数据压缩原理与应用（第二版）[M].北京：电子工业出版社，2003
[33] 吕晓琪,张晟羽中．基于小波变换的图像压缩技术[J]．包头钢铁学院学报,2002,（21）：59-63
[34] Shopiro J M．embeded image coding using zerotree of wavelet coefficients. IEEE Trans. Singal Processing，1993，41：3445-3462
[35] Michel Barlaud et al．Pyramidal lattice vector quantization fo multiscale image coding. IEEE Trans. Image Proc.，1994，3（4）：367-380
[36] Antonini M et al. Image coding using lattice vector quantization of wavelet coefficients. Proc. IEEE ICASSP，1991：2273-2277
[37] Antonini M et al. Image coding using vector quantization in the wavelet transform domain. Proc. IEEE ICASSP，1990：2297-2300
[38] Amir Averbuch，Danny Lazar，Moshe Israeli. Image compression using wavelet transform and multiresolution decompressition. IEEE Trans. IP，1996，5（1）：4-15

[39] Caldrbank R，Daubechies I，Swelens W，et al. Wavelet transforms that map integers to integers. Appl. Comput. Harmon，Anl.，1998，5（3）：332-369
[40] Daubechies I，Swelens W. factoring wavelet transforms into lifting steps. J. Fourier Anal. Appl.，1998，4（3）：245-267
[41] 余刚,齐文新,陈朝阳. 离散小波变换的 VLSI 架构[J].计算机与数字工程,2003,31(2):18-23
[42] Wishwanath M，Owens R M，Irwin M J，VLSI architectures for the discrete wavelet transform[J]. IEEE Trans Circuits Syst II，1995，42（5）：305-316
[43] 吴晓冬，李永明，陈弘毅. 一维离散小波/小波包变换的 VLSI 结构[J]. 半导体学报，1999，20（3）：206-213
[44] A.Grzeszczak,M,K.Mandal,S.Pauchanathan and T.Yeap.VLSI implementation of discrete wavelet transform.IEEE Trans. On VLSI Systems，1996，4(4)：421-432
[45] 乔世杰,王国裕. 离散小波变换的 VLSI 实现[J]. 微电子学，2002，31（2）：143-145
[46] M.Martina et.al. A VLSI architecture for IWT(integer wavelet transform),”Procesding Of 43rd Midwest Symposium On Circuits And Systems,Lansing Mi,USA,Aug.2000
[47] M.Ferretti And D.Rizzo,A parallel architecture for the 2-D discrete wavelet transform with integer lifting scheme,in journal of VLSI signal processing 28,165-185,2001
[48] 刘雷波,王学进,孟鸿鹰等. JPEG2000 小波变换器的 VLSI 结构设计[J]. 电子学报，2002，30（11）：1609-1612
[49] Chien-Yu Chen,Zhong-Lan Yang,Tu-Chih Wang And Liang-Gee Chen.A programmable parllel VLSI architecture for 2D-discrete wavelet transform,journal of VLSI signal processing 28,151-163,2001
[50] 乔世杰,王国裕,智贵连. 二维离散小波变换的 VLSI 实现[J]. 微电子学，2001，31（5）：363-366
[51] 刘雷波,王学进,孟鸿鹰等. JPEG2000 小波变换器的 VLSI 结构设计[J]. 电子学报，2002，30（11）：1609-1612
[52] 刘雷波,李德建,王学进等. JPEG2000DWT 变换器和 EBCOT 编码器的 VLSI 结构设计[J]. 清华大学学报(自然科学版) 2003，43（4）：573-576
[53] （美）James R. Armstrong F. Gail Gray （著）. 李宗伯,王蓉晖（译）. VHDL 设计表示和综合. 北京：机械工业出版社，2002
[54] 吕俊白. 基于 Laplacian 算子的一种新的边缘检测方法[J]. 小型微型计算机系统,2002,23(9):1133～1135.

[55] 郑兆青，陈朝阳，沈绪榜. 一种 3×3 卷积器的设计[J]. 微电子学与计算机 2003.1:70-72

[56] 王向阳,杨红颖. 基于快速提升小波变换的多阈值嵌入零树小波图像编码算法[J]. 测绘学报，2002，31（4）：333-338

[57] Sweldens W. The Lifting Scheme：a construction of second generation wavelets[technical report]. USA：University of South Carolina，1995

[58] Sweldens W. The lifting scheme：a custom-design construction of biorthogonal wavelets. Appl. Comput. Harmon. Anal.，1996，3（2）：186-200

致 谢

本学位论文是在我的导师——中南大学信息科学与工程学院院长、博士生导师桂卫华教授的指导和关怀下进行的。从选题，论证到论文工作的结束，自始至终都倾注了导师的心血。也正是他的远见卓识，热情鼓励和富有启发的建议，对论文研究工作的顺利完成起了关键的指导性作用，同时也使我拓宽和深化了原有的研究领域，找到了“数字图像并行处理的 VLSI 实现研究”这样一个理论研究与实际应用良好结合的研究点，迈进了一个有着深远的实际意义和广阔的应用前景的科学研究殿堂。作为导师，他崇高的威望，渊博的学识，严谨的学风和一丝不苟的工作精神，使我不断追求，不敢懈怠，也进一步培养了我踏实肯干、刻苦钻研、勇于创新的工作作风和治学精神，它将使我终身受益。

三年学业的顺利完成，离不开我的导师桂卫华教授和其它老师的辛勤培养和指导，离不开中南大学提供的学习环境，离不开我的工作单位——株洲工学院的支持，更离不开我的家人的大力支持。在此，我首先向我的导师桂卫华教授、副导师朱晓青教授级高级工程师表示衷心的感谢！其次就是向在我读硕期间曾给予我帮助的中南大学信息科学与工程学院吴敏教授，阳春华教授，喻寿益教授、唐朝辉教授，徐德智教授、王雅琳副教授以及其他老师表示诚挚的谢意！向给予了我在职学习机会和工作支持的株洲工学院院长、博士生导师张晓琪教授，株洲工学院副院长张昌凡教授，株洲工学院人事处凌四立处长、彭涛处长，株洲工学院电气工程系瞿遂春主任，龙文瑞书记等株洲工学院的各级领导表示衷心的感谢！最后向大力支持我的所有家人表示衷心的感谢！

谭会生

2004 年 11 月 8 日于中南大学

作者攻硕期间完成的论文及科研工作

本人在攻硕期间共完成论文 8 篇，其中核心刊物 2 篇，出版著作 3 部，其中 2 部为专著/编著，并入选为西安电子科技大学面向 21 世纪高等学校信息工程类专业系列教材。具体情况如下：

[1] 谭会生.基于 EDA 技术的单片交通控制器的设计[J].中国包装工业，2002,(5):163-163

[2] 谭会生,娄长华.制造资源计划 MRP-II 的应用开发[J].中国包装工业，2002,(6):125-127

[3] 谭会生,邬沛军.基于 Delphi 的销售信息管理系统的开发[J].湖南大学学报，2002,(6):95-97,101.

[4] 谭会生. 现代电子设计技术研究[J].株洲工学院学报，2002，(4)：111～112。

[5] 谭会生.UML 在实验教学智能咨询系统开发中的应用[J]。包装工程，2003，2：35-38。

[6] 谭会生.现代电子设计技术实验室建设探讨。中华教育杂志（香港），2004，1：45-48。

[7] ★谭会生,桂卫华,徐德智.论基于B/S分布式系统中数据库访问接口[J].企业技术开发，2004,23(1):3-6

[8] 谭会生,张昌凡(编著).EDA 技术及应用（第二版, 面向 21 世纪高等学校信息工程类专业系列教材）[M].西安:西安电子科技大学出版社,2004.

[9] 谭会生,瞿遂春(著).EDA 技术综合应用实例与分析(面向 21 世纪高等学校信息工程类专业系列教材)[M].西安:西安电子科技大学出版社,2004.

[10] 谭会生(主编),毛旭光,余建坤等.EDA 技术基础(湖南大学出版社电气信息类规划教材）[M].长沙:湖南大学出版社,2004

[11] ★谭会生，桂卫华，刘展良.基于 EDA 技术的图像边缘检测协处理器的设计[J].包装工程（全国中文核心期刊），2004,25(6):102-104,107。

注:凡打★的文章是以“中南大学信息科学与工程学院”作为第一署名单位的名义发表的。