广东工业大学

硕士学位论文

# 基于ARM的嵌入式静态图像显示系统的研究与实现

姓名：王战盟

申请学位级别：硕士

专业：系统工程

指导教师：谷爱昱

20070425

# 摘要

JPEG 标准是当前在静态图像编解码领域中应用最为广泛的一种图像编解码技术，被广泛应用于图像的存储和传输。随着嵌入式技术的发展，JPEG 图像编解码技术被用到了许多嵌入式系统中，产生了如数码相机、多媒体手机、掌上电脑等许多嵌入式产品。目前 JPEG 编解码多采用专用集成电路 ASIC 来实现，这种方法集成度较高，实现起来相对容易，但成本较高，升级拓展困难。一种更加灵活的方案是基于高性能的处理器，如 ARM、DSP 等，通过采用各种优化算法实现 JPEG 编解码功能，该方案使系统功能拓展和升级可以通过软件升级来实现，具有高度的灵活性和适应性，而且通过软件来实现系统功能的扩展和升级成本也较硬件升级低。本文的研究题目："基于 ARM 的静态图像显示系统的研究与实现"，研究了在 ARM 硬件平台上采用软件编程的方法来实现 JPEG 图像解码和显示。

本文首先介绍了 JPEG 静态图像编码标准的基本原理，然后基于 SAMSUNG 公司的 ARM 处理器 S3C44B0X 设计出用于静态图像解码和显示的硬件平台，并根据该硬件平台设计了相关的引导程序、驱动程序和显示模块的显示程序，为最终实现了一个嵌入式的图像处理系统做好了软硬件基础。接着进一步分析和研究了 JPEG 基本系统的解码过程，JPEG 解码过程主要由头文件信息处理、熵解码、反量化、反离散余弦变换和色度空间转换五个部分组成。在对解码过程中运算量最大，耗时最多的反离散余弦变换 IDCT 部分，结合 S3C44B0X 的流水线操作与并行操作特征和反离散余弦变换原理，将二维 8×8 矩阵的反离散余弦变换运算转换成 16 次的一维 8 点反离散余弦变换运算，并对一维 8 点的反离散余弦变换采用一种快速算法，高效快速地实现了反离散余弦变换。最后实现了基于 ARM 平台的 JPEG 静态图像的解码显示系统，并且从 S3C44B0X 的硬件特性和 C 程序结构方面对解码程序做了的优化。通过从解码的图像质量

和速度两个方面进行测试，系统能够达到图像重构的要求和实时解压显示的目标。

**关键词**：JPEG，嵌入式系统，S3C44B0X，静态图像解码，LCD

# ABSTRACT

JPEG is a kind of image encoding and decoding technique used most widely in the still image field to save and transmit the image information nowadays. With the development of embedded technique, JPEG image encoding and decoding technique has been used in many embedded systems, accordingly many embedded products appear, such as multimedia digital camera, and palmtop computer etc. At present the special integrated circuit ASIC is often adopted to implement JPEG image encoding and decoding. The integrated level of this method is high and easy to realize, but its cost is high and it is difficult to upgrade and develop. Another more flexible scheme is based on a high-performance processor such as ARM, DSP and so on, and all kinds of optimization algorithm are used to implement JPEG image encoding and decoding. The method makes the system upgrading and function expanding realized through the software, which has lower cost than hardware upgrade and has high flexibility and adaptability. The research topic of this article is "The Research and Realization of Still Image Display System Based on ARM" in which based on ARM hardware platform, the software program is adopted to implement JPEG image encoding, decoding and display.

JPEG standard is introduced firstly in the paper then based on ARM processor S3C44B0X which come from SAMSUNG, a hardware processing platform used to finish still image decoding and display has been designed. According to the hardware platform, bottom driver program and display module program have been designed, which provide the software and hardware foundation for realizing the embedded image processing system finally. Further analyses and investigations have been done on the decoding process of JPEG basic system. The process of JPEG image decoding is made up of five parts:tag code information processing, entropy decoding, dequantization, inversed discrete cosine transform and color space

converter. Because of IDCT costing the most quantity of time in the process of decoding, we adopt at row and column method and convert 8×8 two-dimensional inversed discrete cosine transform into 8 point one-dimensional inversed discrete cosine transform, which realizes inversed discrete cosine transform algorithm with high efficiency and speed. Finally JPEG still image decoding and display processing system has been realized based on this hardware platform. In the end the farther optimization has been done to the decoding program from the aspects of hardware characteristic of S3C44B0X and C program structure. Through the test of decoding result to image quality and speed, the system can reach the requirement of image reconfiguration and the target of real-time decoding and display.

Keywords:JPEG; Embedded System; S3C44B0X; Still Image decoding; LCD

# 独创性声明

秉承学校严谨的学风与优良的科学道德，本人声明所呈交的论文是我个人在导师的指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，不包含本人或其他用途使用过的成果。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明，并表示了谢意。

本学位论文成果是本人在广东工业大学读书期间在导师的指导下取得的，论文成果归广东工业大学所有。

申请学位论文与资料若有不实之处，本人承担一切相关责任，特此声明。

指导老师签字：

论文作者签字：

2007年4月15日

# 第一章 绪论

## 1.1 引言

人类从自然界获取信息感受外部世界，是通过听觉、视觉、触觉、嗅觉、味觉等来完成，而绝大部分来自视觉所接收的图像信息，图像信息在人们从自然界获取的信息中占到了80%左右。“图”是物体透射光或反射光的分布，“像”是人的视觉系统对图的接收在大脑中形成的印象或认识。图像是“图”和“像”两者的结合[1]。

图像信息的内涵十分丰富，为了满足人们对不同场合的各种要求，达到一些特定的目标，需要对图像进行“改造”，即对图像进行处理。简单的说，图像处理就是对图像信息进行加工处理，以满足人的视觉心理和实际应用的需要。而利用计算机和其他高速、大规模的集成数字硬件，对将图像信息转换来的数字电信号进行某些数字运算或处理，以期提高图像的质量或达到人们所预期的结果，就称为数字图像处理。如对数据量过大的图像进行压缩编码和解码、对被噪声污染的图像除去噪声，对信息微弱的图像进行增强处理等[2]。图像信息经过数字化后得到的数据量是非常大的，例如，一副640×480个象素组成的24位彩色位图，每个象素的R、G、B分别用8 bit数据量来表示，那么这幅图像需要用640×480×3×8 bit，即7372800个 bit，921600个字节。如此庞大的数据量对图像的计算、存储、传输和显示都带来了极大的不便。然而，人们经过研究发现，图像的数据量虽然庞大，但是数据之间有很大的相关性，可以对图像数据进行压缩去除数据间的冗余，这样就可以使海量的图像数据得到精简，以便对图像进行处理[3]。因此，人们对图像在存储和传输前进行压缩处理，而在接收和显示端对压缩后的图像数据进行接压缩处理以还原和显示图像。

当今社会，人们的生活消费水平越来越高，对消费品的需求也在不断变化。在消费类多媒体产品方面，人们越来越青睐于小巧便携的电子产品，如数码相机、手机、掌上电脑和MP3/MP4播放器等。嵌入式技术的发展，使人们的这些消费需求变为可能。

## 1．2 研究背景

伴随着Internet技术的不断成熟，移动通信的迅猛发展，网络传输以及各种新兴多媒体业务的出现，如视频会议，可视电话，高清晰度电视、视频点播、视频检索、数字图书馆、远程医疗以及视频监控等，人们利用图像这种媒介进行交流的需求越来越广泛，同时也对图像编解码技术提出了新的要求。除了传统的良好压缩性能与重建质量外，人们还要求实现多分辨率编码、嵌入式编码，从而向用户环境提供个性化服务[4]。

与此同时，飞速发展的嵌入式技术（Embedded Technology）可以实现人们对客户终端设备提出的新要求，如使客户终端设备轻巧便利、易于控制或具有某些特定的功能。嵌入式系统就是以应用为中心，以计算机技术为基础，软硬件可裁减，适合应用系统对功能、可靠性、成本、体积和功耗要求的专用的计算机系统[5]。嵌入式系统是将先进的计算机技术、半导体技术和电子技术与各个行业的具体应用相结合的产物。这一点就决定了它必然是一个技术密集、资金密集、高度分散、不断创新的知识集成系统[6]。

近年来，伴随着因特网在全球的普及和消费电子产业的蓬勃发展，静态图像压缩标准 JPEG 标准被广泛地应用于图像信息的传输和存储，多媒体和网络程序、工业生产、遥感、生物医学领域、军事、模式识别和机器人视觉等领域。随着多媒体技术的不断发展，图像及音频视频的处理成为多媒体技术研究的核心，而对各种压缩解压缩方法的研究更是多媒体技术研究的焦点。图像的压缩解压缩是一个很重要的研究领域，这一领域的突破对于通信和多媒体事业的发展将有深远的影响。随着数字信号处理器 ARM、DSP 等芯片性能的不断提高和价格的不断下降，使得嵌入式的实时图像处理变的不再是梦想。

## 1．3 静态图像编码技术的发展历程

图像压缩编码技术可以追溯到 1948 年提出的电视信号数字化，到今天已经有 50 年的历史了。

Kunt 提出了第一代数据压缩编码的概念。他把 20 世纪 1948 到 1988 年这 40 年中出现的以研究去除冗余为目标的编码方法称为第一代编码。

如 PCM、DPCM、亚取样编码法、变换编码中的 DFT、DCT、walsh-Hadamard 变换等方法以及以此为基础的混合编码法属于经典的第一代编码方法。直至五十年代和六十年代的图像压缩技术由于受到电路技术等的制约，仅仅停留在预测编码、亚采样以及内插复原等技术的研究，还很不成熟。A. Huffman 于 1952 年第一次发表了他的论文“最小冗余度代码的构造方法”。从此，数据压缩开始在商业程序中实现并被应用在许多技术领域。

第二代数据压缩编码从 20 世纪 90 年代开始。数学家们因为不满足于 Huffman 编码中的某些致命弱点，设计出另一种更为精确，更能接近信息论中“熵”极限的编码方法即算术编码。在算术编码的基础上又发展了变换编码如金字塔编码法、Fractal 编码、模型基编码。其中最具有代表性的是 LZ77 和 LZ78，将基于这一思路的编码方法称作“字典”式编码。字典式编码不但在压缩效果上大大超过了 Huffman，而且容易实现，其压缩和解压缩的速度也异常惊人。1984 年，LZ78 算法的一个变种－LZW 产生。LZW 继承了 LZ77 和 LZ78 压缩效果好、速度快的优点，而且在算法描述上更容易被人们接受。70 年代末 80 年代初，人们逐渐意识对于多数灰度或是彩色图像乃至声音文件，没有必要忠实地保留其所有信息，在允许一定的精度损失的情况下，可以实现更为有效的压缩方法。到 80 年代末，许多人已经在这一领域取得了不小的收获，提出了一批在压缩效果上让人惊讶不已的声音和图像压缩算法。

第三代数据压缩编码技术主要从 90 年代至今，图像压缩技术的主要成果体现在小波编码、分形编码等，矢量量化编码技术也有较大发展。由于小波变换理论，分形理论，人工神经网络理论，视觉仿真理论的建立，人们开始突破传统的信源编码理论，例如不再假设图像是平稳的随机场。有关于图像编码技术的科技成果和科技论文与日俱增，图像编码技术开始走向繁荣。现代编码技术的特点是:充分考虑人的视觉特性，在恰当地考虑对图像信号的分解与表达时，采用图像的合成与识别方案压缩数据。图像压缩编码向着更高的压缩比和更好的压缩质量的道路前进，进入了一个崭新的、欣欣向荣的大发展时期。在数据压缩编码技术的发展过程中，获得最大成功并被广泛应用在各个领域的就是压缩技术第二代中用于静态连续图像信号的压缩和解压缩的 JPEG 算法。JPEG 标准结合采用了预测、

不定长等多种压缩编码方法，压缩比可以达到 10:1～100:1，而且压缩比可以在一定范围内由用户进行选择[7]。但是采用 JPEG 标准制定的压缩算法的计算量比较大，如对一幅较高精度的真彩色图像使用 JPEG 压缩或者解压缩时，大约要进行上亿次运算操作。为了解决 JPEG 中存在的计算复杂、块效应的问题，近些年来出现了很多新的压缩编码方法，如使用人工神经元网络(Artificial Neural Network, ANN)的压缩编码算法、分形(Fractl)、小波(Wavelet)、基于对象(Object-Based)的压缩编码算法和基于模型(Model-Based)的压缩编码算法等。JPEG2000 是近年来制定的关于静态图像压缩和解压缩的一个新标准，采用基于小波变换的编码算法。

## 1．4 JPEG 静态图像处理的现状及本文的研究意义

伴随 JPEG 标准广泛地应用于图像信息的传输、存储和显示等领域，国内外有许多公司机构和个人在从事基于 JPEG 标准的嵌入式编解码的研究和实现工作。大致可分为以下几类：JPEG 图像处理的集成板卡、基于不同处理器平台的实现、专用的图像处理 IC 芯片、基于可编程逻辑器件的图像编解码处理芯片的设计与实现。

松下电器产业公司基于其面向数字民用产品的综合开发平台“UniPhier”，开发出的 SoC 能够进行除了能够进行 JPEG 编解码外还支持其它一些标准如 MPEG－2 和 MPEG－4 等标准的编解码处理[8]。中星微公司推出的 Vinno I(VC868)芯片，是 0.18 微米综合信号单芯片移动多媒体处理器，内置 JPEG 编码解码、图像信号处理器等，可实现 JPEG 图像的编码解码[9]。上海三意电机驱动有限公司开发出 DM642 图像处理平台能够实现 JPEG 解码[10]。陕西维视数字图像技术有限公司的 MV-100 多媒体动/静态图像采集卡，能够对 BMP、JPEG 等静态图象捕捉和显示[11]。

FR1000 是 Fujitsu 公司生产的主要应用于嵌入式系统的多核处理器,FR1000 将 4 个处理器核（processor element）集成在 1 枚芯片上，各个处理器核之间共享内存和其他外部设备。电子科技大学的章承科利用 FR1000 处理器，将 JPEG 图像划分为 4 个部分，分别在 4 个处理器核上进行解码，实现了 JPEG 图像的解码[12]。在 Fujitsu FR1000VDK 开发环境下，

魏璞实现了一个可在多处理器嵌入式系统环境下运行的 JPEG 解码 FarmWare 软件产品[13]。

电子科技大学的罗凤武在基于 Motorola 公司的 ColdFire 系列的 32 位 DSP 处理器 MCF5272 上实现 JPEG 解码算法的设计与实现[14]。魏忠义等在 ADSP 开发板上实现了 JPEG 图像压缩编码算法，其中的二维 DCT 部分采用行列分解法，一维 DCT 部分使用 DFT 变换方法实现[15]。金燕波等采用以 TI 公司的高速 DSP 芯片 TMS320C6201 为核心的数字信号处理板作为图像压缩器的硬件平台，通过自行开发的压缩程序，实现了图像的实时压缩[16]。中国农业大学的张浩利用 DSP 平台实现了嵌入式手持图像显示系统的软件设计[17]。

随着可编程逻辑器件技术的发展，人们在利用可编程的逻辑器件来实现图像的编解码芯片方面也进行了研究。西南交通大学的刘东用 VHDL 设计实现 JPEG 基本系统硬件编码器[18]。大连理工大学的尹伟用 FPGA 设计了静态图像的编解码。他用硬件描述语言 VHDL，综合与仿真采用 ALTERA 公司的 QUANTUS II 2.0 EAD 工具平台，最后在上海 TrainSillcon 公司的 OPEN-FPGA4.0 实验板上完成了硬件的编程和调试[19]。西北工业大学的汪宇飞在 Altera 公司的 Cyclone FPGA 芯片上实现了 JPEG 高速编码芯片[20]。

另外，还有一些公司设计生产的专门用于 JPEG 编解码的芯片，如 ZORAN 公司的 ZR36060，ZR36050+和 ZR36016，飞利浦公司的 PNX0101,SIGMATEL 公司的 STM 3502，珠海炬力公司的 ATJ2805 等芯片。

以上介绍的各种图像处理的实现方法，无论是建立专门的硬件平台或者板卡，还是解码芯片主要都是利用硬件的方法实现的，其成本较高，不适合一些成本要求较低的情况。而利用可编程的逻辑器件设计的处理芯片功能太过单一，虽然 DSP 处理器有强大数字信号的处理能力，但其对系统控制的能力不如 ARM 处理器。ARM（Advanced RISC Machines）公司是微处理器行业的一家知名企业，设计了大量高性能、廉价、耗能低的 RISC（Reduced Instruction Set Computer）处理器、相关技术及软件。ARM 处理器具有性能高、成本低和能耗省的特点。适用于多种领域，如嵌入控制、消费、教育类多媒体、成像和安全产品等。

随着 ARM 技术的不断发展，ARM 处理器的速度在不断提高，其对数据

处理的能力也越来越强，价格也在不断降低。本文基于 ARM 处理器设计了对成本要求相对较低，而相对基于 DSP 实现的和专门解码芯片的功能较强大的图像处理和显示的嵌入式系统。由于 ARM 芯片强大的控制能力和高速的数据处理能力，使得 ARM 在诸多嵌入式图像编解码实现方法中更具有优势，所以研究静态图像压缩和解压缩标准 JPEG 算法在 ARM 上的实现和显示方法具有重要的价值和现实意义。

## 1．5 论文的主要内容

本文设计出基于 ARM 处理器 S3C44B0X 的嵌入式处理平台，在该平台上实现了 JPEG 静态图像的解码显示处理过程。并根据 S3C44B0X 的硬件特性和 C 语言程序结构方面对系统的程序做了进一步的优化。

本论文的主要工作有以下几个方面：

1．嵌入式图像处理系统的软硬件平台的构建；

2．JPEG 基本系统解码程序的设计；

3．解码程序从 PC 机到目标平台的移植和优化。

全文共分为五章：

第一章　绪论部分，介绍了有关图像视频压缩的国际标准，本课题研究的意义现状以及主要的研究内容。

第二章　对静态图像压缩标准 JPEG 原理做了简单介绍。

第三章　构建嵌入式图像处理的软硬件平台，包括主要硬件电路的设计，系统启动引导程序 Bootloader 的实现，底层驱动程序和显示模块的驱动程序的编写。最后对交叉编译工具 ADS 做了简单介绍。

第四章　详细分析了 JPEG 基本系统的解码过程的每一个步骤，在研究分析解码过程中反向离散余弦变换 IDCT 各种算法的基础上实现了本系统所用的 IDCT 算法。

第五章　在 ARM 平台的实现了 JPEG 基本系统的解码及显示，将在 PC 机上设计好的程序移植到目标平台，并对程序做了优化。最后给出了该嵌入式图像处理系统的一些实验测试结果，并对测试结果进行了分析。

结论部分对论文的全部工作进行了总结，在此基础上对进一步继续研究工作做了构想和展望。

# 第二章 JPEG 压缩与解压缩基本原理

## 2.1 引言

JPEG 标准是第一个关于灰度和彩色连续静态图像压缩的国际标准[21]。JPEG 是 Joint Photographic Experts Group 的英文缩写，是一个由国际标准化组织（ISO）和国际电信电报咨询委员会（CCITT）联合组成的专家小组，其任务是制定图像数据库、彩色传真、印刷等方面的连续色调（灰度和彩色）的静止图像压缩编码的国际标准。JPEG 算法标准就是由 JPEG 专家组制定的关于静止图像数据压缩和解压缩的标准，被命名为“ISO 10918－1”，该标准的正式名称为“Digital compression and coding of continuous-tone still images”[22]，如这个名称表示的一样，JPEG 标准是具有连续黑白或彩色协调能力的静止图像的编码技术，既可用于灰度图像又可用于彩色图像，还可用于电视图像序列的帧内图像的压缩。JPEG 主要存储图像颜色变化的信息，特别是亮度的变化信息。在常用的模式中，用有损压缩方式去除冗余的图像色彩数据，在获得极高的压缩率的同时能展现十分丰富、生动的图像，可以用较少的磁盘空间得到较好的图像质量。JPEG 还是一种灵活的格式，具有调节图像质量的功能，允许采用不同的压缩比例对文件进行压缩。和其他具有相同图像质量的文件格式（如 BMP、GIF、TIFF）相比，JPEG 是目前静态图像中压缩率比较高的[23][24]。正是由于这种高压缩比使得 JPEG 格式的文件尺寸较小，下载速度快，使得 Web 页面能够以较短的下载时间提供大量的美观图像，所以目前各类浏览器都支持 JPEG 图像格式，使得它广泛的用于多媒体和网络程序中[25]。

## 2.2 JPEG 的运行模式

JPEG 是适用范围非常广泛、通用性很强的技术，其功能分为四种运行模式，每种模式针对一类特定的应用，用户只要从中选择需要的功能即可。这四种模式是：

1. 基于 DCT 的顺序模式（Sequential DCT-based） DCT（discrete cosine transform），即离散余弦变换，是一种用于图像压缩的高效的编

码方式。基于 DCT 的顺序模式就是对于由 8×8 象素组成的象素块，从左到右进行编码处理，并按照从上到下的顺序进行扫描。编码处理是由二维 DCT 系数的量化和量化系数的熵编码组成的。基于 DCT 的顺序模式运行需要的缓冲条件最小，因而实现的费用最低[26]。

**2. 基于 DCT 的渐进模式**（Progressive DCT-based） 该模式对图像按照由粗到细进行编码，图像重现时由粗糙到清晰，处理的顺序及编码的基本构成与 DCT 顺序模式相同，扫描的顺序也是从上到下，从左到右，但需要对图像进行多次处理扫描。当对图像进行解码时，在第一次扫描后先得到一幅分辨率较低的粗略的图像轮廓，然后在后续的扫描处理后逐渐提高画面质量，直到解码完成，得到清晰的图像。渐进模式传输时间较长，接收端接收到的图像是多次扫描由粗糙到清晰的累进过程。

**3. 无失真模式**（Lossless） 无失真模式能保证解码后精确的回复原图像的采样值，其压缩比低于有失真的压缩编码方法。使用二维差分脉冲编码调制技术的空间预测算法，可处理较大范围的输入像素精度。一个像素值的预测至多用到三个相邻的采样值，然后从实际值中减去预测值，并对差值进行无损的Huffman编码或算术编码。无失真压缩模式可以达到2：1的压缩率。由于不使用DCT变换，对接近象素间的差别进行熵编码，从而不产生图像的失真，可以保证重构图像与原始图像完全相同。

**4. 层次模式**(Hierarchical) 层次模式是上面三种方式的组合，做成具有多种空间分辨率图像的金字塔结构，适用于对原始图像的滤波和划分象素。因为按空间分辨率由高到低的顺序累积起来是金字塔的形状，所以称为金字塔结构。该模式对图像在多个空间分辨率下进行编码，由低品质图像向高品质图像阶段传送时，使用该方法[27]。在信道传送速率低，接收端显示器分辨率也不高的情况下，只需要做低分辨率图像解码，不必进行高分辨率解码[28][29][30]。

## 2.3 JPEG 标准的编码系统

JPEG 算法有两种不同的压缩方式，一种是基于 DCT 的压缩方式，是不可逆的有失真的压缩编码方式；另一种是基于 DPCM，即差分脉冲编码调制的压缩方式，是一种可逆的无失真的编码方式。JPEG 标准中定义了 3

种编码系统[31]：

**1. 基于DCT的基本顺序系统（Baseline Sequential System）** 又称基本系统，采用基于DCT的顺序编解码算法来实现图像的有损压缩。重建图像质量达到人眼难以观察出图像损失的要求。这种系统采用8×8像素块的DCT算法、量化及基于Huffman型的熵编码解码器。该系统的主要处理功能有：

（1） 能处理精度为8bit的图像。

（2） 能以顺序显示的方式呈现解码后的再生图像。

（3） 支持Huffman编码实现熵编码。

**2. 基于DCT的扩展系统(Extended DCT Based System)** 为了满足更为广泛的通信要求，JPEG标准中除了具有基本功能外，还具有扩展功能，其代表性的扩展功能就是累进显示。此外，在扩展功能中还应设置有多种可选择的功能。扩展编码系统使用累进行工作方式，采用自适应算术的编码或者Huffman编码过程，能够处理精度为8bit或12bit的图像。

**3. 无失真系统(Lossless System)** 无失真系统是一种完全能回复原像的可逆系统。由于DCT编码会产生一定的信息丢失，不适宜用于无失真系统。在JPEG算法中，采用DPCM差分脉冲编码调制方式实现无失真系统的可逆编码。这是一种有别于DCT的独立的压缩编码方式，为保证重建图像数据与原图像数据完全相同，无失真系统采用预测编码及Huffman或算术编码。

在JPEG的三种编码系统中，基于DCT变换的基本系统是JPEG最基本的压缩系统，符合JPEG标准的软硬件编码/解码器都必须支持基本系统。另两个系统是可选的扩展，对一些特定的应用项目有很大实用价值[32]。在JPEG中，基本系统是最常用的，本文所讨论和实现的就是基于DCT的基本系统。

## 2.4 JPEG 数据结构

JPEG 文件的数据结构分为两种不同的类型，一种是分层型数据结构，另一种是非分层型数据结构。如图 2－1 所示。一帧图像可以在空间上分为若干部分，每一部分称为一个扫描（Scan）。每个 Scan 可以根据其所含色彩分量的数目分为两类。若只含一种色彩分量，如灰度图像就只含亮度分量，那么这个 Scan 称为无分枝的。若此 Scan 内含有一种以上的彩色分量，如一个 RGB 图像，含有红、绿、蓝三种分量，那么这个 Scan 称为

有分枝的。JPEG 中对一个 Scan 的编解码是分块进行的。一个 Scan 被划分为若干部分，每一部分称为一个“最小编码单元”MCU。JPEG 基本系统的数据类型为非分层型，一幅图像只对应于一帧，一帧只有一个 Scan。

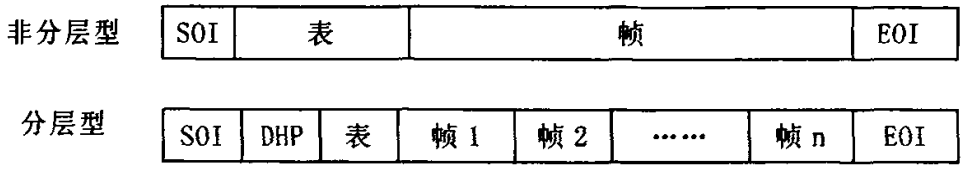


图 2－1 数据结构类型图
Fig. 2－1 The Data Sturctures of JPEG

一个JPEG文件中，除了图像的压缩数据外，还有一些其他信息，如文件大小、色彩信息和各种表的定义等。JPEG文件以数据段为单位来组织所有的文件内容。一个段的长度不大于65535，每个段是由段标识符、段长度和段内容组成的。JPEG文件格式使用 Motorola 格式,对一个字的存储而言,低字节存放的是高位数，高字节存放的是低位数，即高字节在前，低字节在后。段标识符又叫标记码，标记码由两个字节构成，前一个字节是固定值0xFF，而第二个字节是非0字节和0xFF。每个标记之前还可以添加任意数目的0xFF填充字节[33]。标记码有很多，但大多数JPEG文件所用的标记码只是其中一部分，常用的标记码参见附录。

## 2.5 JPEG基本系统

基于离散余弦变换DCT（Discrete Cosine Transform）变换的顺序模式是JPEG的基本系统，所有符合JPEG标准的编解码器都应该支持基本系统，而其他系统则是作为不同应用的选择项。余弦变换是傅立叶变换的一种特殊情况，在傅立叶级数展开式中，如果被展开的函数是实偶函数，那么，其傅立叶级数中只包含余弦项，这种形式被称之为离散余弦变换[34]。DCT变换是目前常用的图像变换算法，它有很多优点：

- DCT是正交变换，可以将图像的空间表达式转化为频率域，只需要要少量的数据点表示图像；
- DCT产生的系数很容易被量化，能获得较好的块压缩；

• DCT算法的性能很好，它有快速算法，如采用快速傅立叶变换可以进行高效的运算，因此它在硬件和软件中都容易实现；

• DCT算法是可逆的，所以利用反向的DCT算法可以解压缩图像；

• DCT是实数域的变换，而且二维的DCT还是可以分离的变换。

以上的DCT的这些特性非常适合于图像的变换，所以被JPEG标准所采用。DCT变换的JPEG基本系统包含编码器、解码器和交换文件格式三部分。

## 2.5.1 编码器

JPEG的基本系统的压缩编码过程的示意图如图2-2所示。编码过程主要有离散余弦变换DCT、量化和熵编码三个过程。DCT变换的对象是像素块，需要将原图像信号划分成许多8×8的像素块，即每个像素块包含的8×8个像素。8×8的像素块经过编码器后得到的就是压缩后的图像数据。

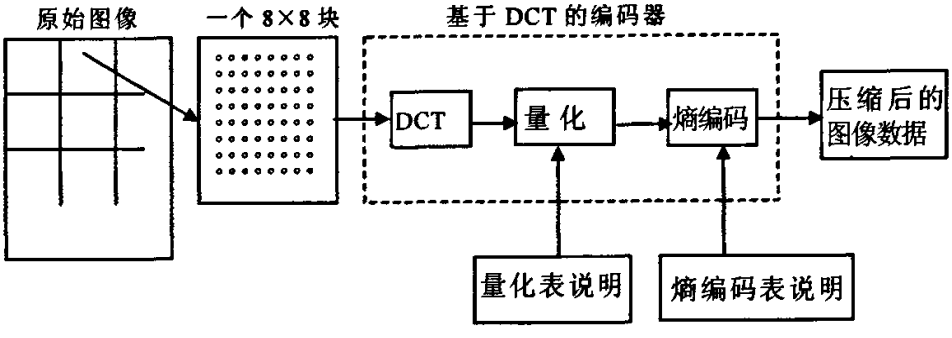


图 2－2 JPEG 编码器
Fig.2－2 The ENCODER of JPEG

编码器的整个压缩编码过程大致分成三个步骤：

**1.对图像数据进行离散余弦变换** 使用离散余弦变换DCT把空间域表示的图像数据转换成频率域表示的图像数据。首先应将输入的原始图像划分许多像素块，每一个像素块均由8×8像素构成。基于块的DCT基本编码能够很好的压缩图像而且几乎不被人眼所察觉，因而在目前大多数的图像和视频压缩标准中被采用。

**2.使用加权函数对DCT系数进行量化** 量化的过程是指用一组预定的容许值之一来代表实际DCT系数值的过程，以便可以用较少的位数对全部

数据编码。在JPEG处理中量化是不可逆的关键步骤，虽然完成一个DCT反变换后由于舍入误差将不能精确的恢复原始输入的数据，但是其结构通常是视觉上完全能接受的[35]。

**3.对量化系数进行熵编码** 为了进一步压缩数据，JPEG先用DPCM即差分脉冲调制编码法对量化后的直流DC系数的差分值进行处理，用RLE即行程编码法对量化后得到的交流AC系数进行处理，最后对DC系数的差值和AC系数进行Huffman编码。Huffman编码可以使用很简单的查表方法来实现。JPEG的标准推荐了四个码表，分别是DC亮度、色度和AC亮度、色度，这四个表是对很多图像统计的结果，具有一定的普遍性[36]。

## 2.5.2 解码器

对压缩的图像数据可以通过解码过程，获得解压缩后重构的图像数据。解码的过程是编码的逆过程，包括熵解码、反量化和反向离散余弦变换（Inverse Discrete Cosine Transform,即IDCT）共三个步骤。图2－3是JPEG基本系统的解码器结构图。

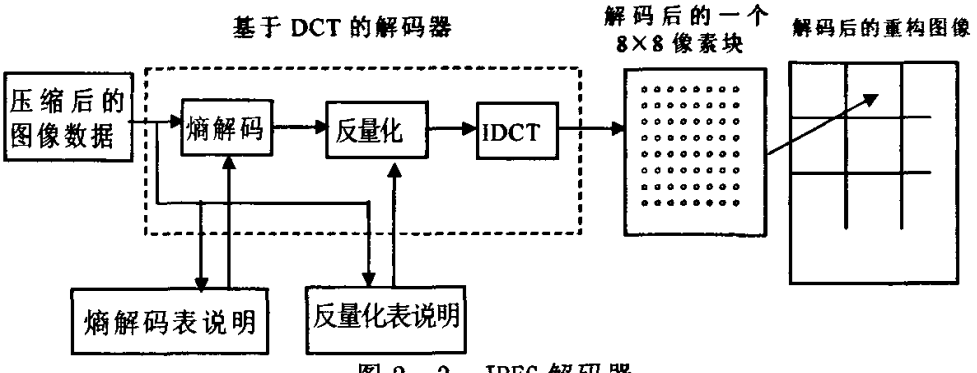


图 2－3 JPEG 解码器

Fig .2－3 The DECODER of JPEG

**1.熵解码** 熵解码的输入信号是被压缩编码的比特流，输出是被解码得到的DCT变换系数的量化值。熵解码包括两个过程，首先对得到的压缩数据的比特流进行Huffman解码，再用DPCM法对得到的直流分量进行处理得到DC系数，用RLE编码法对得到的中间数据进行处理得到交流AC系数。

**2. 反量化** 反量化是将在压缩过程中经过DCT 变换后的频率系数还原出来，其输入是熵解码后的数据，与使用8×8 的量化表进行相乘的运算

来完成反量化。反量化使用亮度和色差各自的量化表对亮度和色度分量分别进行。在反量化时，量化表值及品质控制参数必须满足编码器和解码器的相一致。

**3.反向离散余弦变换（IDCT）** 对反量化后得到的DCT变换系数经过反向离散余弦变换IDCT得到图像的像素。反离散余弦转换的输入是频率域的一个8×8分量系数块，输出则得到空间域的一个8×8像素块。

## 2.5.3 交换格式

在JPEG中除了编码器和解码器外，还有一个过程是格式交换过程。交换格式是压缩图像数据的表示，它是一种简化的格式，主要是用来为不同应用环境之间交换提供方便的，允许JPEG压缩图像在多个平台和应用环境间共享。一幅位图按JPEG压缩算法压缩后，生成的数据按照JPEG文件交换格式生成标准的JPEG文件，现在最常用的是JFIF和JPEG-TIFF。交换格式的过程如图2－4所示。

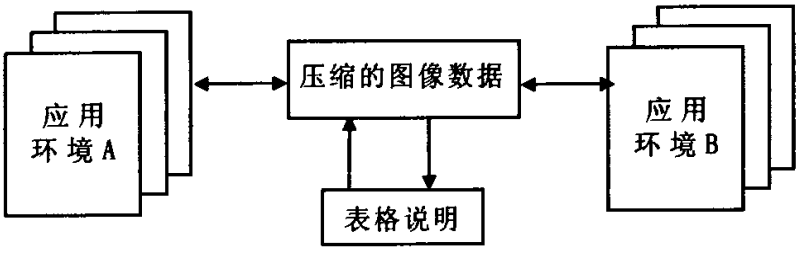


图 2－4 JPEG 交换过程

Fig. 2－4 The Interchange Progress Block Diagram

# 第三章　基于 ARM 的嵌入式系统的构建

## 3.1 图像解压缩的实现方式

图像解压缩的实现方式一般可分为 3 种:纯软件压缩法、硬件实现法以及软硬件结合法。具体说,纯软件压缩法就是在通用计算机上通过编程实现图像的压缩和解压（如在 PC 机上实现 JPEG，MPEG 等），其特点是灵活性强、软件资源丰富、开发周期短，但是设备体积较大，使用场合有一定限制；硬件实现法就是采用专用芯片或可编程器件实现图像的高速实时压缩和解压缩（如实现 JPEG 的 CL550，STI114、L64702，实现 MPEG 的 CL950、STI3500 等芯片），其特点是实时性好、可靠性高，但灵活性差、开发周期长。软硬件结合法则是综合采用硬件和软件措施来实现图像压缩和解压缩，最为有效、方便的方法是使用高性能处理器芯片并配置一些外围器件构成一个图像处理系统，通过编程实现图像的压缩解压缩，其特点是灵活性强、实时性好、可靠性高。本文就是采用这种方法在 SAMSUNG 公司的 ARM7 系列的 S3C44B0X 上实现 JPEG 图像的解压缩，给显示系统提供显示数据。

## 3.2 系统硬件组成

本系统是以 ARM7 为主要硬件平台,基于 SAMSUNG 公司的 S3C44B0X 处理器而设计。系统整体硬件结构图如 3-1 所示。在参考 SAMSUNG 公司的 S3C44B0X DEMO 板设计的基础上，结合本系统的需要，考虑到尽量降低成本节省资源的前提下,本系统主要硬件包括存储电路、LCD 接口电路、USB 、URAT、JTAG 等接口。

### 3.2.1 S3C44B0X 简介

S3C44B0X 是 SAMSUNG 公司推出的 ARM7 系列的微处理器，是目前应用最广泛的 16/32 位嵌入式处理器，它为一般的应用提供了一个低成本、高性价比和高性能的微控制器解决方案。S3C44B0X 体系结构图如 3-2 所示。

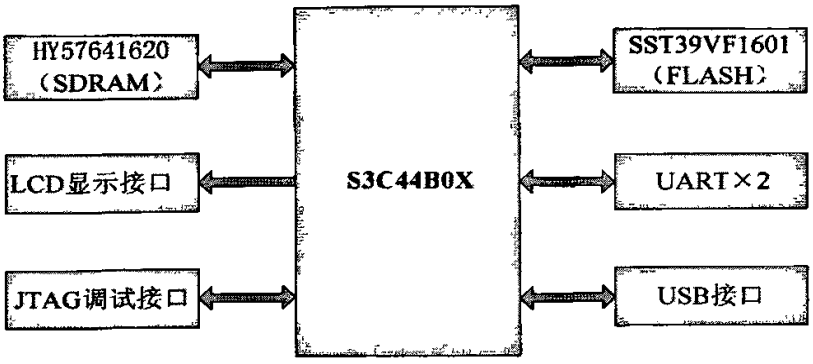


图 3-1 系统硬件结构图

Fig. 3-1 The Hardware Block Diagram of Image Display System

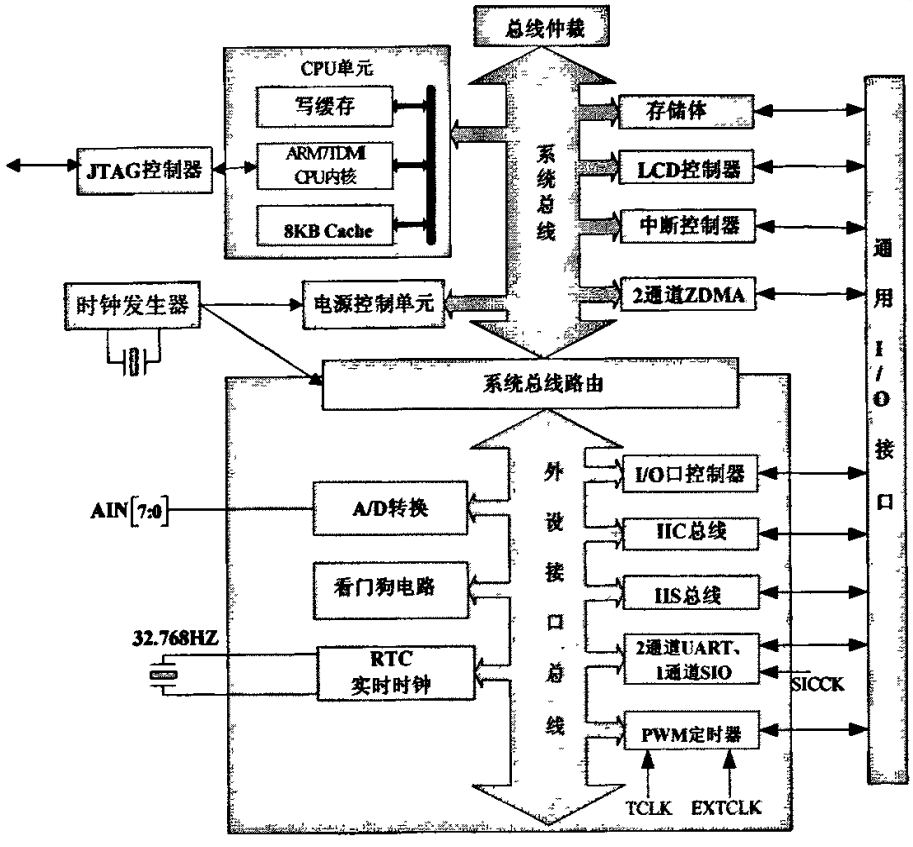


图 3－2 S3C44B0X 结构图

Fig. 3－2 S3C44B0X Block Diagram

为了降低成本，S3C44B0X提供了丰富的内置部件，包括：8KB Cache或内部SRAM、LCD控制器、带自动握手的2通道UART，4通道DMA，71个通用I/O端口，RTC实时时钟、系统管理器（片选逻辑，FP/EDO/SDRAM控制器），带PWM功能的5通道定时器和1个内部定时器，8通道的10位ADC、IIC总线接口，IIS总线接口，同步SIO接口和片上PLL时钟产生器[37] 。

S3C44B0X 使用 ARM7TDMI 内核，采用 0.25μm CMOS 工艺制造。它的低功耗和全静态设计特别适用于对成本和功耗敏感的应用。同时，S3C44B0X 还采用了一种新的总线结构，即 SAMBAII（三星 ARM CPU 嵌入式微处理器总线结构）。S3C44B0X 的另一个出色特性是它的 CPU 内核，是由 ARM 公司设计的 16/32 位 ARM7TDMI RISC 处理器（66MHz），它集成了 Thumb 代码压缩器和一个 32 位的硬件乘法器，同时还支持 ICE 断点调试。

### 3.2.2 存储电路

系统的存储设备包括一片 4M×16bit 位的 SDRAM 和一片 1M×16bit 位的 Flash 存储器。

1. SDRAM 同步动态存储器　　SDRA 由 S3C44B0X 专用的 SDRAM 片选信号 nSCS0 选通。系统使用的 RAM 型号为 HY57641620，是 8M 字节的单

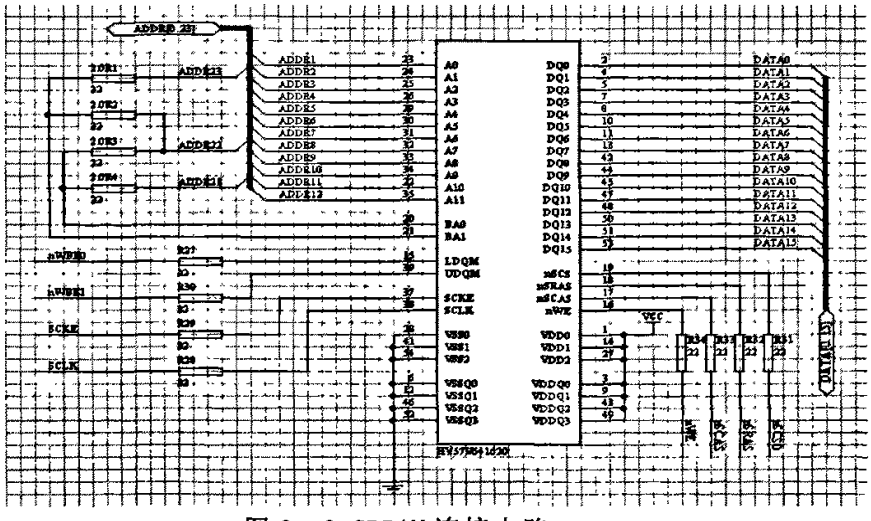

图 3－3 SDRAM 连接电路

Fig.3－3 The Connect Circuit of SDRAM

片 SDRAM。该 SDRAM 分 4 个 BANK，BANK 的地址由 BA0，BA1 决定，每个 BANK

的容量为 1M×16bit,分别用行地址选通脉冲 RAS 和列地址选通脉冲 CAS 进行寻址。行、列地址线复用，行地址线为 A[0-11],列地址线为 A[0-7]，地址空间为 0x0C000000-0x0C7FFFFF。电路连接图如图 3－3 所示。

2. Flash 存储器　　Flash 存储模块是在近十年来应用广泛的一种非易失性半导体存储器，本系统采用 AM29LV160D，容量为 2M 字节，处理器通过片选信号 nGCS0 来对其进行选通。S3C44B0X 的地址线 A1～A20 分别与 Flash 的地址线 A0～A19 相连，组成 1M×16bit 的存储空间,地址为 0X0—0X1FFFFF，即 2M 字节。其中 0X0—0X3FFFF 用来存放系统的启动引导程序代码,0X40000—0X4FFFF 存放启动程序的参数,0X50000—0X1FFFFF 为用户程序区。电路连接图如图 3－4 所示。

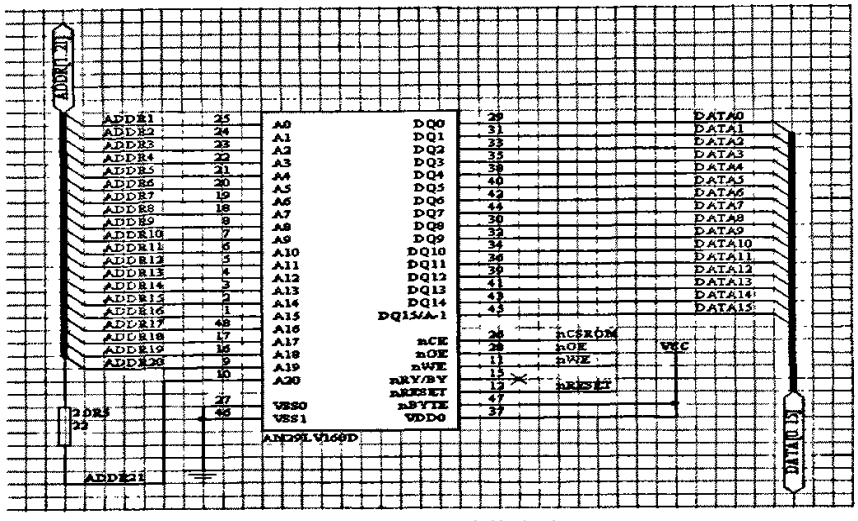


图 3－4 Flash 连接电路

Fig.3－4 The Connect Circuit of Flash

## 3.2.3 显示模块电路

S3C44B0X内置了LCD控制器，其内部结构框图如图3-5所示。LCD控制器的能够支持单色、灰度和最大256色的彩色显示；支持4位单扫描、4位双扫描、8位单扫描的显示模式和多种屏幕尺寸（640×480，320×240，160×160等）；同时还支持具有不同的数据线宽度、不同的接口时序和刷新速率的LCD显示器。LCD控制器的主要工作就是要把定位在系统存储器中的视频缓冲区

的LCD图像数据传送到外部的LCD驱动器，并产生必要的LCD控制信号[38]。LCD接口用24针的接口，电路如图 3－6所示。

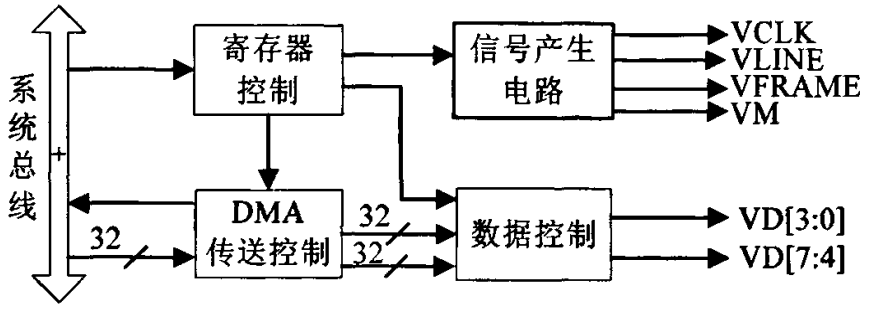


图3-5 LCD控制器内部结构框图
Fig.3-5 LCD Controller Block Diagram

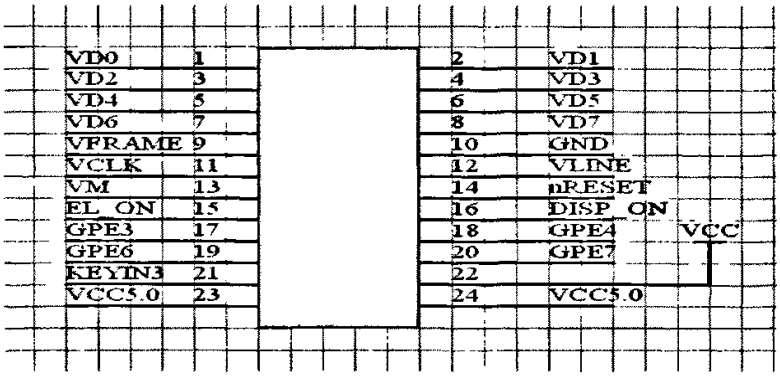


图3－6 LCD接口
Fig.3－6 The Interface Circuit of LCD

本系统所采用的显示器件是sharp公司的STN型256色的LCD显示器LM7M632，具有640×240 的显示尺寸和8位单扫描显示模式[39]，电路连接如图3－7所示。在显示模块中，利用S3C44B0X的端口C和端口D作为显示控制模块部分的各种接口，将端口C中的PC[7：4]作为控制模块中数据端口的VD[4：7]；将端口D中的 PD[3：0]用做控制模块中数据端口的VD[3：0]，将PD4作为帧同步信号VFRAME；PD5作为控制器的VM信号，用来改变行和列的电压极性，从而控制象素点的显示或熄灭；PD6作为同步脉冲信号VLINE；PD7用作象素时钟信号VCLK。控制器输出3.3V的逻辑电压信号VCC，作为显示器的

逻辑电源输入。LCD显示器的Vcon接0～3.3V的可调电压信号，用来改变显示器的对比度；LCD控制模块的信号线经74HC245驱动后与LCD显示模块LM7M632对应的信号相连。

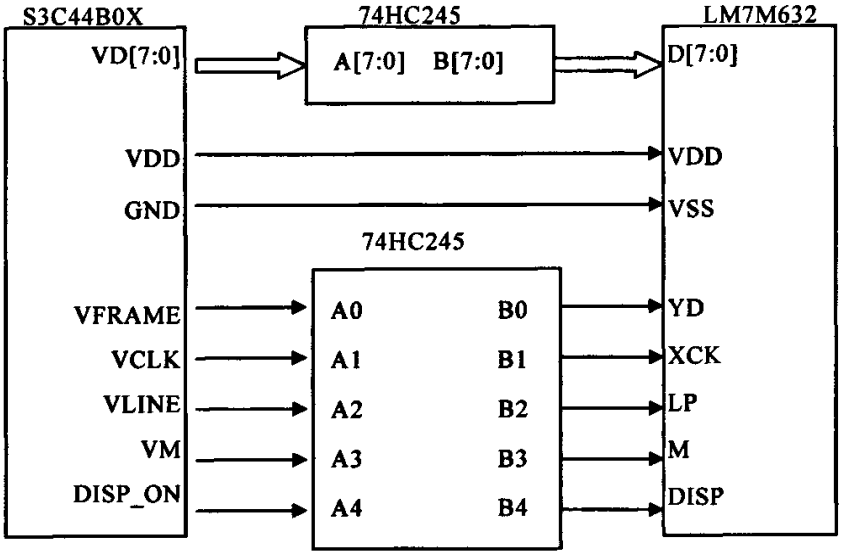


图 3－7：S3C44B0X 与 LCD 硬件连接图
Fig.3－7 The Block Diagram of S3C44B0X and LCD

### 3.2.4 USB 接口

目前移动存储设备越来越被人们广泛的应用，基于 USB 接口的尤盘、移动硬盘等存储工具已经成为人们不可缺少的工具，USB 总线是 Microsoft、Compaq、Intel、NEC 等公司共同制定的一个串行总线标准，它作为一种快速灵活的总线接口，由于其具有使用方便、速度快、支持即插即用、可热插拔、总线供电和低成本等优点而得到广泛应用，并作为一种标准接口普及到多种设备上，考虑到以后系统的功能扩展，设计了 USB 接口。

PDIUSBD12 是一个性能优化了的 USB 器件，通常用在嵌入式系统中以提供高速通用并行接口，同时支持本地 DMA 传输。它采用了模块化的方法，允许使用现存的体系结构使得固件投资减到最少。这种灵活性减少了开发时间、风险和成本，是开发低成本且高效的 USB 外围设备解决方案的一种

最快途径。PDIUSBD12挂起时的低功耗以及LazyClock输出符合ACPI OnNow和USB电源管理设备的要求，低功耗工作允许实现总线供电的外围设备。同时它还集成了像SoftConect、GoodLink、可编程时钟输出、低频晶振和终端电阻等特性，所有这些特性都能在系统实现时节省成本，同时在外围设备上很容易实现更高级的USB功能[40]。本系统采用PDIUSBD12的并行传输和单地址/数据总线方式，用S3C44B0X的nGCS1作为PDIUSBD12的片选信号输入，ADDR0与PDIUSBD12的A0脚相连，选择控制命令输入或数据输入，使用S3C44B0X的EINT0中断。具体的USB接口电路如图3－8所示。

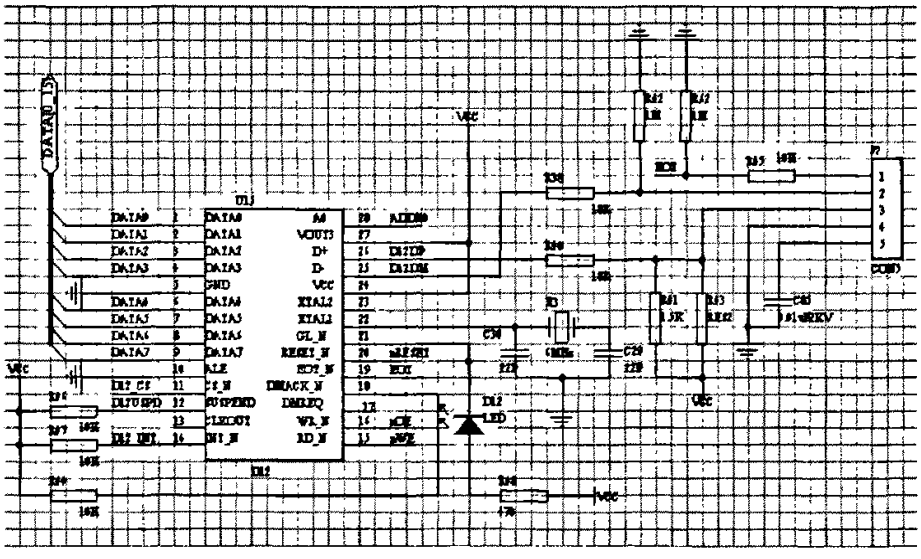

图3－8　USB接口图

Fig.3－8 The Interface Circuit of USB

### 3.2.5　JTAG调试接口和串行通讯接口

JTAG调试接口和串行接口电路部分是在开发阶段用来对系统进行开发调试的，不属于解码系统中的功能模块。

1.JTAG调试接口　　JTAG（Joint Test Action Group）是1985年定制的检测PCB和IC芯片内部测试的一个标准，1990年被修改后称为IEEE的一个标准，即IEEE1149.1－1990。通过在芯片内部封装专门的测试电路TAP（Test Access Port，测试访问口），可对具有JTAG接口芯片进行内部测试及对系统进行仿真、调试。它是开发调试嵌入式系统的一种简洁高

效的手段。ARM7TDMI 内部提供了 3 个 JTAG 型扫描链，可以进行调试和配置嵌入式的 ICE－RT 逻辑。

现在许多的高级器件都支持 JTAG 协议，如 ARM、DSP、FPGA 等器件。标准 JTAG 接口是 4 线：TMS、TCK、TDI、TDO，分别为模式选择、时钟、数据输入和数据输出。允许多个器件通过 JTAG 接口串连在一起形成一个 JTAG 链，对各个器件分别测试。现在，JTAG 还经常用于对 Flash 器件进行在线编程 ISP（In－System Programmable）。

目前 JTAG 接口有 2 种链接标准，即 14 针和 20 针接口，其中 14 口标准引脚定义如表 3－1 所示。14 针的 JTAG 接口电路如图 3－9 所示。

表 3－1　JTAG 引脚定义

Table.3－1　PIN Configuration in JTAG

| 引脚号 | 定　义 | 引脚号 | 定　义 |
|---|---|---|---|
| 1，13 | VCC，接电源 | 3 | nTRST,测试系统复位信号 |
| 5 | TDI，测试数据串行输入 | 7 | TMS，测试模式选择 |
| 9 | TCK，测试时钟 | 11 | TDO，测试数据串行输出 |
| 2，4，6，8，10，14 | GND，接地 | 12 | NC，未连接 |

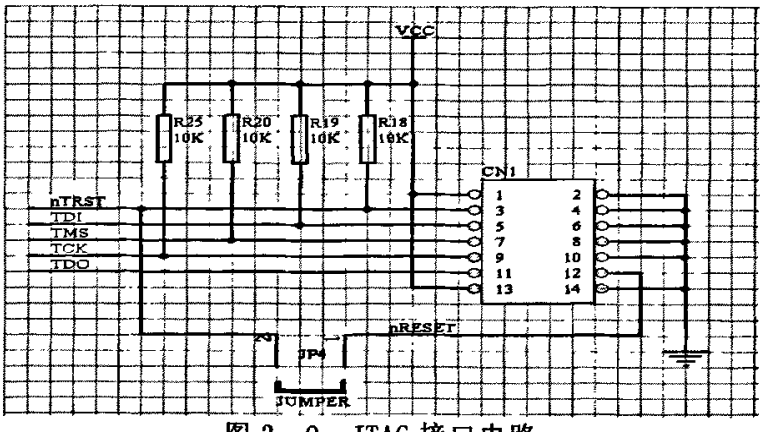


图 3－9　JTAG 接口电路

Fig. 3－9 The Interface Circuit of JTAG

**2. 串行接口 URAT** S3C44B0X 集成了两个 URAT 接口，可与 PC 机或 Modem 进行串行通信。PORTC10～PORTC15 分别作为 nRTS1、nCTS1、TXD1、RXD1、nRTS0 和 nCTS0 信号，GPE1 和 GPE2 作为 TXD0 和 RXD0 信号。两个串行接口都采 MAX2332C 进行电平转换。串行接口电路如图 3－10 所示。

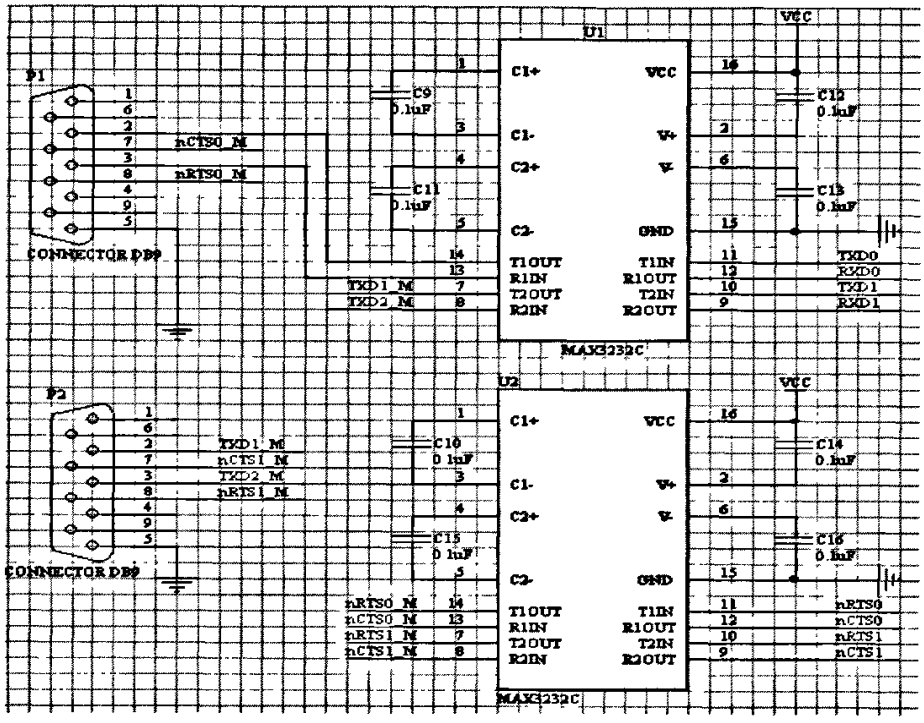


图 3－10 UART 接口电路

Fig. 3－10 The Interface Circuit of UART

## 3.3 系统软件的实现

对嵌入式系统而言，要使系统完成用户制定的任务，运行用户特定的程序，达到预期的目标，只有硬件环境和用户的应用程序是不够的。在硬件电路的基础上，还必须有相应的软件程序，对系统的硬件做成相应的配置，来使硬件进行正常的工作，从而将系统的硬件环境限制到一个由用户指定的状态，以便用户应用程序的顺利运行，完成特定的任务。

### 3.3.1 系统启动程序 Bootloader

嵌入式系统大多是功能比较单一和固定的系统，通常并没有像 PC 机的 BIOS 那样的固件程序，因此在运行嵌入式操作系统或用户的应用程序之前，必须有专门的启动程序来引导硬件系统进入操作系统或主应用程序，因此整个系统的启动加载任务就完全由启动程序来完成[41]。

Bootloader 是本系统加电启运行后的第一段代码，是在解码程序运行之前执行的一段小程序。通过这段代码可以对系统硬件进行初始化，建立内存空间映射图，将系统的软硬件环境带到一个由用户定制的特定状态，为加载解码程序做好准备。通常 Bootloader 是依赖于具体硬件而实现的，每种不同的 CPU 体系结构都有不同的 Bootloader。对于 ARM 核 CPU 而言，不同的 CPU 制造公司为适应不同功能和不同用户的需要会在 ARM 公司提供的 ARM 核基础上扩展一系列外围器件，因此 ARM 核处理器结构一般并不相同，启动代码的编写也有一些区别[42]。除了与 CPU 的体系结构外有关外，Bootloader 还依赖于具体的系统外围硬件的配置，如硬件地址的分配、RAM 和 FLASH 存储器的类型及其他外设的类型等等。本系统采用目前功能较为强大的开源的启动程序 U-boot(Universal Bootloader)，根据本系统的硬件平台对 U-boot 进行裁减和修改，实现了系统的启动引导程序 Bootloader。系统加电或复位后，所有的 CPU 通常都从 CPU 制造商预先安排的地址上取指令，S3C44B0X 在复位后从地址 0x00000000 取它的第一条指令，所以固态存储设备（比如：ROM、EEPROM 或 FLASH 等）被安排这个起始地址上。因此，对基于 S3C44B0X 的图像显示系统，为了保证系统上电或复位时 Bootloader 首先被加载运行，必须将它存放在系统的 FLASH 存储器 AM29LV160D 中的 0x00000000 处，并在硬件设计中把 FLASH 接在 CPU 的 nGCS0 处。这样以来在系统加电启动后，CPU 将首先执行 Bootloader 程序。

Bootloader 由两大部分组成，第一部分是依赖于 CPU 体系结构的代码，如 CPU 的工作模式、堆栈及主要寄存器设置等，用汇编语言来实现，程序简明扼要，以达到快速启动的目的；第二部分是对系统其他硬件的初始化，如存储设备、串口、及网络设备等，一般用高级语言实现，这样比较容易

实现复杂的功能，而且程序代码具有良好的可读性和可移植性。Bootloader的流程如图3－11所示。

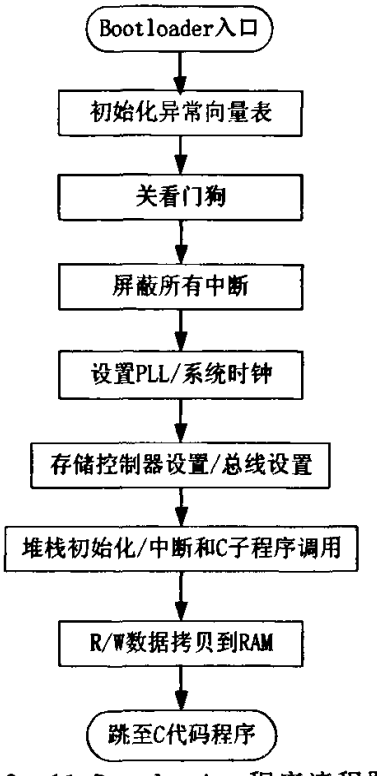


图3－11 Bootloader程序流程图

Fig.3－11 The Flowchart of BootLoader

### 3.3.2 串口通讯

S3C44B0X的通用异步收发器UART（Universal Asynchronous Receiver/Transmitter）提供两个独立的异步串行I/O端口，可以工作在DMA和中断方式下，支持的最高波特率为115200b/s。每个UART通道包含2个16字节FIFO（First In First Out）分别提供给接受和发送。S3C44B0X的UART可以进行以下参数的设置：波特率、红外线收/发模式、停止位、数据宽度和奇偶位校验。每个字收发的格式包括1个起始位，5～8个数据位，1个可选的奇偶校验位和1～2个停止位。每个UART包含一个波特率产生器，发送器，接收器和控制单元。发送器和接收器包含16字节的FIFO和移位寄存器。要被发送的数据，首先被写入FIFO然后复

制到发送移位寄存器。然后它从数据输出端口(TxDn )依次被移位输出。被接收的数据也同样从数据接收端口(RxDn)移位输入到移位寄存器，然后拷贝到FIFO中。

每一个UART的波特率发生器为传输提供了一个串行移位时钟，时钟源可通过S3C44B0X的内部系统时钟MCLK来选择。波特率的时钟由通过时钟源的16位分频产生及由一个URAT波特率除数寄存器UBRDIV指定的16位除数决定， UBRDIVn的值可以按照下式（3.1）确定：

$$UBRDIVn = (round\_off)\left(\frac{MLCK}{波特率\times 16}\right)-1 \qquad (3.1)$$

其中除数的范围为1～（$2^{16}-1$）；(round_off)是对后面的值取整。

本系统中UBRDIVn=(round_off)(60000000/(115200×16))－1=32；

UART 的操作，包括数据发送，数据接收，中断发生，波特率发生等内容。其初始化用函数 Uart_Init(int mclk, int baud)来实现。

```
//串行口初始化
void Uart_Init(int mclk,Int baud)
 {
      int i;
      if(mclk==0)
        mclk=MCLK;
      if(UartChunnel==0)                    //UART0 初始化
        { rUFCON0=0x0;
          rUMCON0=0x0;
          rULCON0=0x3;
          rUCON0=0x245;
          rUBRDIV0= ((int)(mclk/16.0/baud+0.5)-1; }
      else                                  //UART1 初始化
        { rUFCON1=0x0;
          rUMCON1=0x0;
          rULCON1=0x3;
          rUCON1=0x245;
          rUBRDIV1=( (int)(mclk/16./baud + 0.5) -1 ); }
      for(i=0;i<100;i++);                   //延时
 }
```

串行口数据发送程序的流程图如图 3－12 所示，数据接收程序的流程图如图 3－13 所示。

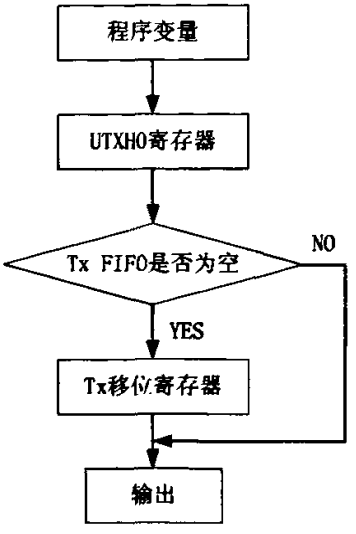


图 3－12　数据发送流程图
Fig.3－12 The Flowchart of Data Transmition in UART

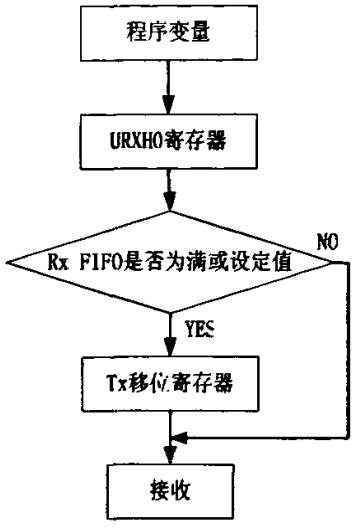


图 3－13　数据接收流程图
Fig.3－13 The Flowchart of Data Reception in UART

## 3.3 LCD 显示模块的驱动

要使LCD正常工作，就必须先完成LCD显示模块的初始化。其过程包括：初始化端口、申请显示缓冲区、初始化LCD控制寄存器。

1. LCD相关的I/O口配置　　由S3C44B0的端口定义知道，S3C44B0X的PC口和PD口作为LCD驱动接口，因此需要将PC口的工作方式设置在第3功能状态，PD口工作在第2功能状态。设置I/O端口控制寄存器的语句如下：

```
//      I/O口控制寄存器设置
        rPDATC=0xffff ;
        rPCONC=0x0f05ff55;
        rPUPC=0x30f0;
        rPDATD=0xff ;
        rPCOND= 0xaaaa ;
        rPUPD = 0xff ;
```

**2. 申请显示缓冲区** 显示缓冲区就是在系统存储器中划出一块区域，用来存放要显示的图像数据。将要显示的图像数据直接放入显示缓冲区就能直接在LCD显示屏上显示出所显示的图像。因此，可以通过直接修改显示缓冲区的内容来完成显示的目的。显示缓冲区的大小与LCD显示屏的屏幕大小相同，为LCD显示屏的象素点的个数。显示缓冲区中的一个字节数据代表LCD上的一个象素点的信息颜色[43]。本系统所采用的LCD显示屏尺寸是640×240，所以所需要的显示缓冲区的大小为640×240×1字节。显示缓冲区的申请代码如下：

```
//申请显示缓冲区
if((U32)frameBuffer256==0)
{
frameBuffer256=(unsigned int(*)malloc(ARRAY_SIZE_COLOR) ;
rNCACHBE1=(((unsigned int ) frameBuffer256)>>12)|
((((unsigned int)( frameBuffer256+ARRAY_SIZE_COLOR) )>>12)<<16) ;
}
```

**3. LCD初始化** 在点亮LCD之前，还应该对LCD控制器相关的寄存器进行初始化，使LCD控制器的配置与外接LCD显示模块特性相匹配，按照用户指定的工作方式运行。LCD初始化包括设置LCD分辨率、扫描频率、显示模式、产生控制信号和控制时序等。

LCD控制器的相关寄存器有LCD控制寄存器1（LCDCON1）、LCD控制寄存器2（LCDCON2）、帧缓冲区起始地址寄存器1（LCDSADDR1）、帧缓冲区起始地址2（LCDSADDR2）、帧缓冲区起始地址3（LCDSADDR3）、红色查找表寄存器（REDLUT）、绿色查找表寄存器（GREENLUT）、蓝色查找表寄存器（BLUELUT）以及抖动模式寄存器（DP1_2、DP4_7、DP3_5 、DP2_3、DP5_7 、DP3_4、DP4_5、DP6_7、 DP4_7、DITHMODE）。通过对以上寄存器的设置，LCD控制器就能够按照设置的工作方式进行工作了。通过对显示缓冲区写入显示数据就可以进行显示了。

## 3.4 交叉开发环境的建立

**1. 交叉开发环境简介** 作为嵌入式系统应用的 ARM 处理器，其应用软件的开发属于跨平台开发，因此，需要一个交叉开发环境。交叉开发

是指在一台通用计算机上进行软件的编辑编译，然后下载到嵌入式设备中进行运行调试的开发方式。用来开发的通用计算机可以是 PC 机、工作站等，运行通用的 Windows 或 Unix 操作系统。开发计算机一般称宿主机，嵌入式设备称目标机。在宿主机上编译好程序，下载到目标机上运行，交叉开发环境提供调试工具对目标机上运行的程序进行调试。

交叉开发环境一般由运行在宿主机上的交叉开发软件和宿主机到目标机的调试通道组成。运行于宿主机上的交叉开发软件最少必须包含编译调试模块，其编译器为交叉编译器。宿主机一般为基于 x86 体系的台式计算机，而编译后的代码必须在 ARM 体系结构的目标机上运行，这就是所谓的交叉编译。在宿主机上编译好目标代码后，通过宿主机到目标机的调试通道将代码下载到目标机，然后由运行在宿主机的调试软件控制代码在目标机上进行调试。

**2. 基于 JTAG 的 ICD 调试通道**　　组成 ARM 交叉开发环境的宿主机到目标机的调试通道一般有以下 3 种：在线仿真器 ICE、基于 JTAG 的 ICD 和 Angel 调试监控软件。

基于 JTAG 的 ICD（In Circuit Debugger）也称为 JTAG 仿真器，是通过 ARM 芯片的 JTAG 边界扫描口进行调试的设备。JTAG 仿真器通过 ARM 处理器的 JTAG 调试接口与目标机通信，通过并行口或串行口、网口、USB 口与宿主机通信。JTAG 仿真器价格比较便宜，连接比较方便。JTAG 边界扫描口与 ARM CPU 核通信，属于完全非插入式调试，无需目标存储器，不占用目标系统的任何应用端口。通过 JTAG 方式可以完成：读/写 CPU 寄存器，访问和控制 ARM 内核；读/写内存，访问存储器和 I/O 口；控制程序单步执行和实时执行；实时设置基于指令地址值或数据值的断点。

基于 JTAG 仿真器的调试是目前 ARM 开发中采用最多的一种方式，本系统调试就采用此种方式。

**3. ADS 交叉开发软件**　　本论文中的软件开发是在 ARM ADS1.2 开发套件中的 GUI 开发环境下完成的。ADS 全称为 ARM Developer Suite，是 ARM 公司推出的新一代 ARM 集成开发工具。ADS 由命令行开发工具，ARM 实时库，GUI 开发环境(Code Warrior 和 AXD)，适用程序和支持软件组成。GUI 开发环境包括 CodeWarrior for ARM 和扩展调试器 AXD。

**(1) CodeWarrior 简介** CodeWarrior是一套完整的集成开发工具，为管理和开发项目提供了简单多样化的图形用户界面。CodeWarrior 是专为基于 ARM RISC 的处理器而设计的，充分发挥了 ARM RISC 的优势，它可加速并简化嵌入式开发过程中的每一个环节，使得开发人员只需通过一个集成软件开发环境就能研制出 ARM 产品。CodeWarrior 集成开发环境用户可以使用 ADS 的 CodeWarrior IDE 为 ARM 和 Thumb 处理器开发用 C,C++，或 ARM 汇编语言的程序代码。用 CodeWarrior 开发一般流程为建立工程，为工程建立或添加文件，编译连接工程，最后调试工程。CodeWarrior 集成开发环境主窗口如图 3－14 所示。

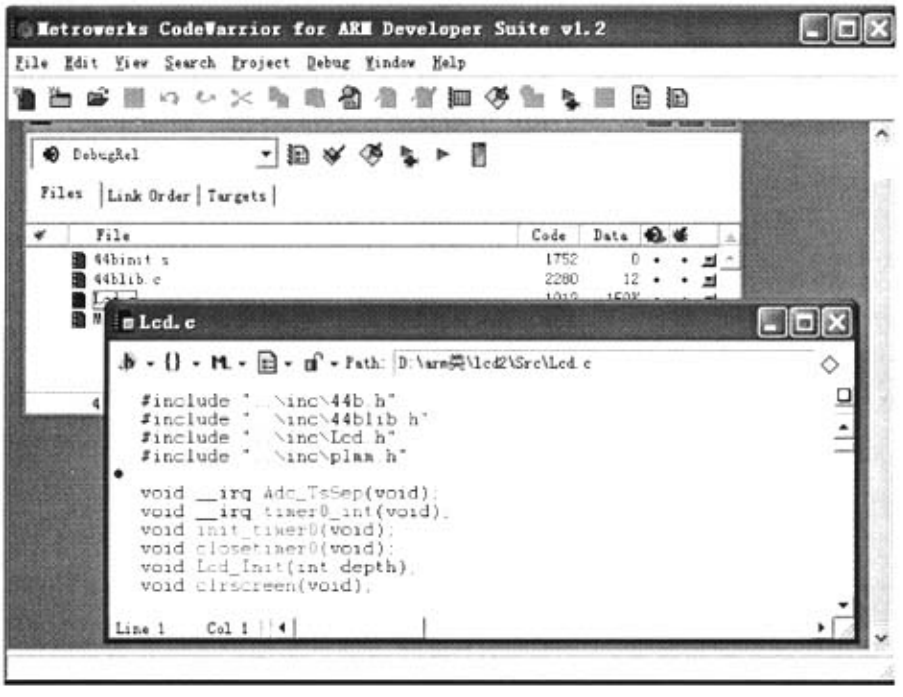


图 3－14 CodeWarrior 集成开发环境主界面

Fig. 3－14 Main Interface of CodeWarrior

**(2)用 AXD 进行代码调试** AXD 调试器为 ARM 的扩展调试器(即 ARM eXtended Debugger)，是 ADS 软件中独立于 CodeWarrior IDE 的图形软件，支持硬件仿真和软件仿真。ARMulator 是一个 ARM 指令集仿真器，集成在 ARM 的调试器 AXD 中，它能对 ARM 的指令集的仿真，为 ARM 和 Thumb

提供精确的模拟，是调试的时候最常用的一种调试工具，用户可以在硬件尚未做好的情况下，开发程序代码。硬件仿真时要在 AXD 的仿真目标配置中选择 ADP，才能进行硬件的仿真调试。本文就是先对程序进行软件仿真调试，调试通过后再将程序下载到硬件系统中，结合硬件系统进行整体调试。要用 AXD 进行代码调试，首先要把编译链接工程后生成的含有调试信息的.axf 映像文件装载到目标内存中。AXD 调试器主窗口如图 3－15 所示。

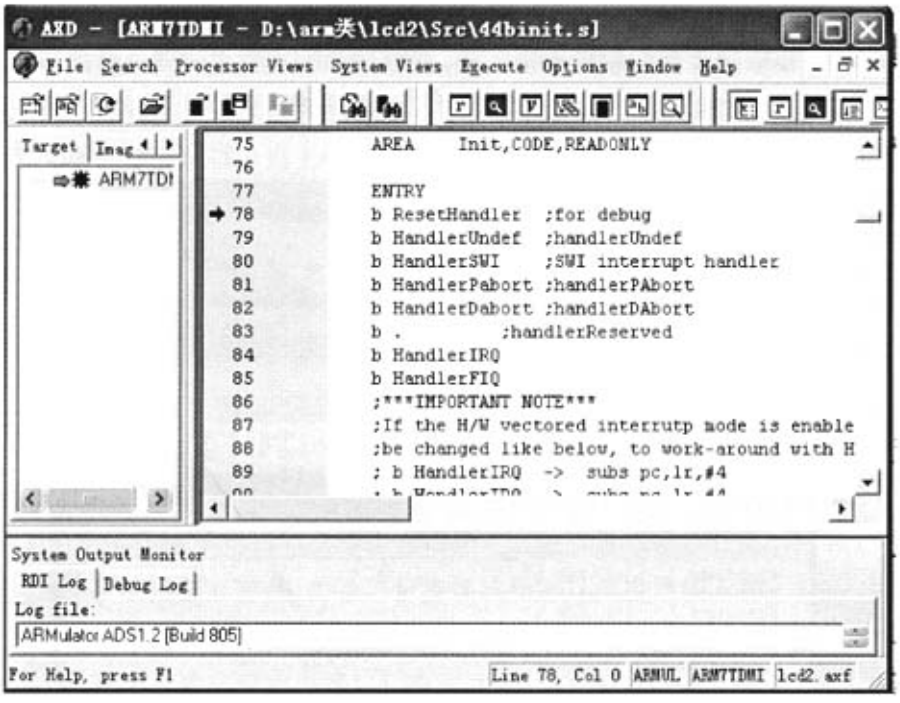


图 3－15　AXD 调试器主界面
Fig. 3－15 Main Interface of AXD Debugger

## 3.5 小结

本章介绍了图像显示系统的硬件电路的设计，包括存储电路、显示电路、USB 接口电路和调试电路等。接着针对该硬件结构，完成了系统的软件设计，包括引导启动程序 Bootloader 的实现、底层驱动程序的实现和显示模块驱动程序的实现。最终构建了一个嵌入式系统，为下一步完成静态图像的显示建立起一个实现的平台。

# 第四章 JPEG基本系统解码算法的分析与研究

## 4.1 引言

在JPEG的几种编码系统中，基于DCT的基本顺序系统是JPEG最基本的压缩系统，又称为基本系统或基线系统(Baseline)。符合JPEG标准的软硬件编码器和解码器都必须支持和实现基本系统。而其他系统作为不同应用的选择项，用在一些具有特定应用的场合。在JPEG中，基本系统是最常用的系统，本文所讨论和实现的解码算法过程就是基于DCT的基本顺序系统。

基于连续DCT编码的JPEG基本过程的解码算法可以由以下几个主要步骤来实现：

1.对JPEG头文件进行分析，从中提取解码所必须的信息,如色彩分量信息，量化表、Huffman解码表等；

2.进行熵解码，通过查找Huffman解码表将压缩图像数据还原成直流DC系数和交流AC系数组成的量化数据块；

3.通过反量化，并通查量化表进行计算，将量化区块反量化成DCT系数；

4.使用反向离散余弦变换(IDCT)，把频率域DCT分量系数反转成颜色空间域表示的图像数据；

5.最后将Y、 Cr、Cb格式的数据转换成RGB格式的BMP图像数据。

## 4.2 JPEG基本系统解码过程分析

JPEG的编解码是以最小编码单元MCU为单位的，实际的编码和解码都是以一个一个的MCU为单位进行的。一个MCU由多个8×8的像素块组成，具体所包含的8×8的数据块的数目是由亮度Y和色度Cr、Cb的采样方式决定的。解码就是逐个对每一个MCU进行循环解码,一直到检查到 EOI 标记。由于一幅图像的高和宽不一定是 MCU 尺寸的整数倍，因此需要对图像的最右边一列或其最下边一行进行复制，扩展其高或宽，使得可以将整个图像划分为整数个 MCU。而在解码输出时，这些复制的行列是要被抛弃的。

### 4.2.1 分析 JPEG 头文件信息

位图文件经过JPEG压缩后，得到的是把各种标记码和编码后的图像数据组成的一帧一帧的数据，通常称为JPEG位数据流(JPEG bit stream)，这样做的目的是为了便于图像文件的传输、存储和JPEG解码。JPEG解码则是一帧一帧的读入JPEG数据流，进行解码。在处理压缩的图像数据前，首先读入的是JPEG的头文件数据，头文件部分给出了JPEG图像的所有信息，有点类似于BMP中的头信息，但要复杂的多。对头文件信息进行处理，获取解码所必须的信息，如图像色彩信息、采样比、宽高尺寸、Huffman表和量化表等，并将这些信息提取出来，这些信息都是在后面解码中要用到的，如果少了这些信息，将不能完成正确的解码。

### 4.2.2 熵解码

在完成了头文件信息的处理后，开始对图像数据解码，首先进行的是熵解码。熵解码是指将经过压缩的图像还原成由直流系数DC和交流系数AC组成的量化数据块的过程。JPEG基本系统对直流系数DC的差值先采用DCPM（Differential Pulse Code Modulation），即差分脉冲编码调制法进行处理，再使用Huffman编码法进行编码；而对交流AC系数是先采用RLE（Run Length Encoding）行程长度编码法进行编码，再用Huffman编码法进行编码，整个过程称为熵编码。所以熵解码就是要利用同样的方法把压缩后的图像数据的直流DC系数和交流AC系数还原出来。

在JPEG解压缩算法过程中，由于Huffman编码中的编码字具有唯一性，因此可以使用很简单的查表（Lookup Table）方法进行解码。解码器在一个扫描行内可使用2个DC系数的Huffman表和2个AC系数的 Huffman表。Huffman码表在压缩的时候已定义在JPEG文件数据中，在处理头文件信息时通过提取得到[44]。Huffman解码后得到由DCPM编码法编码后的直流分量差分值的中间结果的数据和由RLE编码法编码后的交流分量的中间结果的数据，最后再分别用DCPM编码法和RLE编码得到相应的直流DC系数和交流AC系数[45]。下面将以亮度空间Y的解码过程说明熵解码的过程，色度分量Cr、Cb的过程与Y相同。

1. 直流 DC 系数的熵解码　　在压缩的过程中，一个 8×8 的像素块

经过 DCT 变换和量化后得到 8×8 量化系数矩阵，该量化系数矩阵的左上角元素，即第一行、第一列的元素就是直流 DC 系数，除了直流 DC 系数之外的元素都为交流分量，即被称为交流 AC 系数。其中 $AC_{xy}$ 表示某一个交流 AC 系数，下标 $x$ 和 $y$ 分别表示该系数在矩阵中的行号和列号，行列号分别从 0～7，如 $AC_{27}$ 表示位于第二行、第八列的交流系数。

直流 DC 系数代表一个 8×8 的像素块 64 个图像采样值的平均值。相邻的 8×8 像素块的 DC 系数具有很强的相关性，所以 JPEG 标准对 DC 系数采用 DPCM 差分脉冲编码调制法进行编码，不是直接对 DC 系数本身进行编码，而是对相邻 8×8 像素块之间的直流 DC 系数的差值(Diff)进行编码。如果当前的像素块为 K，与 K 相邻的像素块为(K－1)，这两个像素块对应的差值为 Diff，则 Diff＝DC(K)－DC(K－1)。对直流 DC 分量的编码就是对 Diff 进行编码的。

JPEG 基本系统中， Huffman 编码将 DC 系数的差值划分成若干组，并以不同的编号 SSSS 来表示不同的组号，直流 DC 系数差值的分组表见附录。Huffman 编码是用一个变长码 VLC ( Variable Length Code)和一个变长整数 VLI ( Variable Length Impr)来编码差值。

变长码 VLC 是用 SSSS 值对直流 DC 系数的 Huffman 表查找得到；变长整数 VLI 就是差值的大小。若差值为正，则 VLI 就是差值的原码；若差值为负，则 VLI 就是差值的补码减 1；若差值为 0，则不需要 VLI。在对 DC 的差值进行编码时，最终 Huffman 编码值为 VLC 值加上 VLI 的值。DC 差值的亮度 Huffman 编码表（部分）参见附录。

在进行直流 DC 系数的解码时，是对照 Huffman 表将上面介绍的编码过程反过来执行一次，就得到 Huffman 编码前的直流 DC 系数的差分值。JPEG 是从“0”开始对 DC 编码的，所以 DC(0)＝0，然后再将当前的差值 Diff 加上前一个的 DC 值就得到当前的 DC 值，即当前这一像素块的 DC(K)就是:DC(K)=DC(K－1)+Diff 。

**2.交流 AC 系数的熵解码**　　一个 8×8 像素块的 63 个交流 AC 系数是一种二维排列的数据，压缩过程中为了对它进行编码，先将其变换成一维排列的数据，并按照其排列的顺序进行编码。在 JPEG 的基本系统中，对 8×8 像素块的 63 个交流 AC 系数是以“Z”行扫描的方法将它变换成一

维数据进行编码的。“Z”行扫描的的示意图如图4－1所示。经过“Z”形扫描后，交流AC系数的编码顺序为$AC_{01}$，$AC_{10}$，$AC_{20}$，$AC_{11}$，$AC_{02}$，……，$AC_{67}$，$AC_{76}$，$AC_{77}$。其中，$ACxy$表示矩阵中位于第$x$行、第$y$列的交流AC系数。

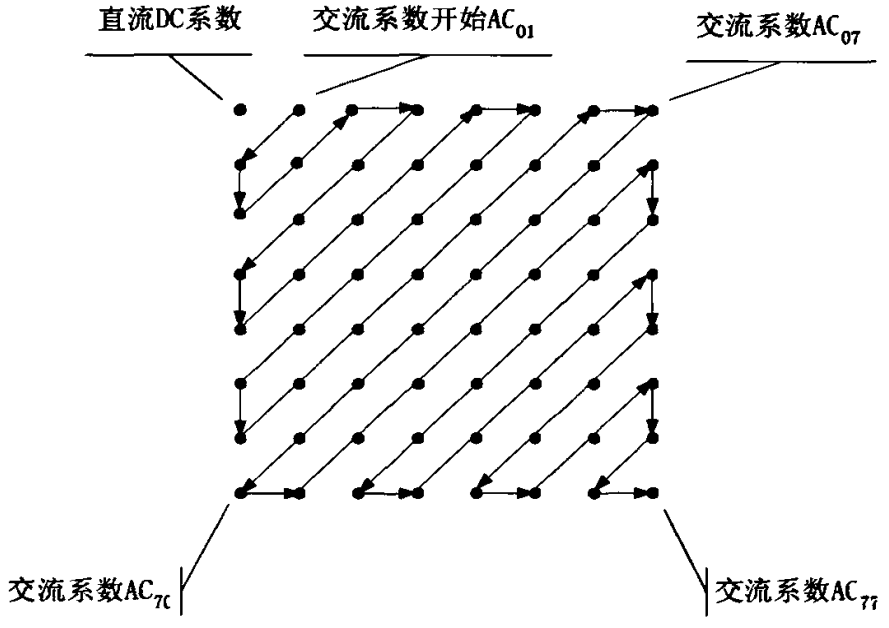


图4－1 “Z”形扫描图

经过量化和“Z”形扫描后的交流AC系数序列中，包括许多无效系数和有效系数，称连续的无效系数（即值为零的AC系数）的个数为行程长度。JPEG在对无效系数和有效系数采取不同的编码处理。进行熵编码的时候，先用RLE行程编码方法对交流AC系数进行处理后得到<Run, Level>的数据格式，其中Run 表示在非零AC 系数前出现零的个数，即行程长度；而Level 表示该非零AC 系数的值。对于AC系数中的有效值，根据AC系数的范围对其进行分组，并用SSSS来表示不同的组号。交流AC系数的分组参加附录2。对于不同组号的差分值，Huffman编码还给予不同的附加字位，即在编码数据后面加上相应的附加位数据。附加位的具体数据是由AC值的大小决定的，若AC值是正数，其附加位数据是自身的二进制数；若AC值是负数，则其附加位数据是是先取

其绝对值的二进制数，再按位求反。接下来，将 <Run,Level> 格式转换为<Run, SSSS><L>格式。其中SSSS为AC的分组号，L为附加位的具体数值。最后查找交流AC系数Huffman编码表进行编码，由Run/SSSS的值查表得到码字，再加上附加位的数值，即为最后的交流AC系数的Huffman编码。附录4为交流AC系数的Huffman编码表（部分）。对交流AC系数进行解码时，就是将上述的编码过程进行一次反向进行，对照Huffman编码表得出的行程值和分组号，再对照交流AC系数值的分组表得到AC值的范围，并根据附加位数据，即可得到编码前的AC值。

熵解码的最后的一个步骤，则是经过“Z”形的反扫描，也就是将一维的DCT 的AC 系数按照“Z”形扫描的顺序，回存到8×8 大小的量化系数区块的相应位置上，其具体位置由译码出来<Run, Level>格式来决定。

至此，整个熵译码的过程到此已完成，接下来便是对得到8×8大小的量化系数块进行反量化的工作。

### 4.2.3 反量化

反量化是将8×8的量化系数块与加权函数进行计算，将在压缩过程中经过正向DCT 变换后得到的DCT系数还原出来。这个加权函数拥有当前数

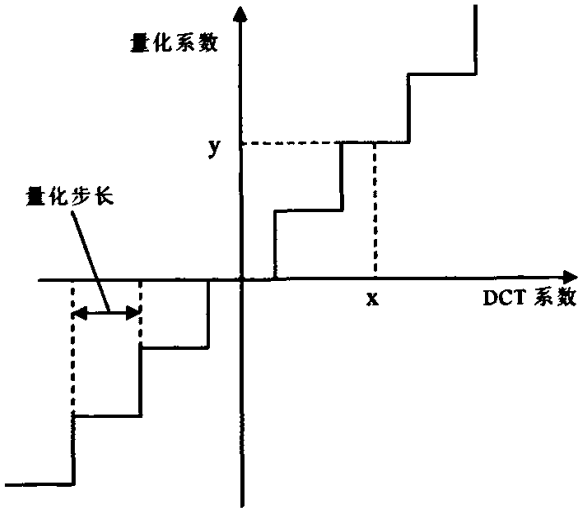


图 4－2 线性均匀量化器特性曲线
Fig. 4－2 The Curve of Linearity and Uniformity Quantizer

据值的最佳视觉效果。具体来讲，这个加权函数就是量化表。JPEG标准对于有损压缩算法采用线性均匀量化，线性均匀量化的特性曲线如图4－2所示。

图中横轴是输入的带量化的DCT系数，纵轴是量化之后的DCT系数，台阶在横轴上的投影叫做量化步长，量化步长是按照系数所在的位置和每种颜色分量的色调来确定的。

在对经过正向DCT变换后得到的8×8个变换系数进行量化时，是利用量化表来完成的。JPEG标准推荐了一套量化表，分别为亮度分量的量化表和色度分量的量化表，这两张量化表是从大量实验中得出的，在压缩和解压缩时也可以不用标准的量化表。量化表是控制JPEG压缩比的关键，基于人眼对高频成分的敏感程度远低于低频成分，并且在图片的像素和像素之间会有一个色彩的过渡过程，大量的图像信息被包含在低频空间中，因此，在量化过程除掉了一些高频分量，损失了高频部分的一些细节。

量化表对经过DCT变换得到的8×8系数矩阵中不同位置的变换系数采取不同的量化步长。假设8×8系数矩阵中某一个位置的系数为$Sxy$（$x,y$分别为该系数所在的行和列的数），相应位置的量化表中的量化步长为$Qxy$，经过量化得到的量化系数为$Rxy$，则对 $Sxy$ 的量化如式（4.1）所示：

$$Rxy = round\left(Sxy / Qxy\right) \tag{4.1}$$

其中$round$表示取整运算，即将系数值$Sxy$与量化表中的量化步长$Qxy$进行除法运算，然后取与其结果最接近的正数值，即为量化后的量化系数为$Rxy$。

在解码过程中，进行反量化的处理，就是用量化后的系数和量化表中的相应位置的量化步长相乘，结果得到量化前的经过DCT变换的8×8系数矩阵中的相应的系数。在解码中除了标准的量化表外，也可以采用文件中的量化表，或者自己设计量化表。反量化的计算公式 如式（4.2）所示：

$$Sxy = Rxy \times Qxy \tag{4.2}$$

由此可见，量化过程是一个有损的处理过程，对经过量化后的量化系数进行反量化处理，并不能得到图像的原始数据，只是能得到与原始图像数据相接近的数据。

### 4.2.4 反离散余弦变换IDCT

使用反向离散余弦变换IDCT把频率域DCT分量系数反转成空间域表示的图像数据。反向离散余弦变换的输入是经过反量化得到的8×8的DCT分量系数，输出则是8×8的图像像素块。

经过DCT变换，就把图片里点和点间的规律呈现出来了，更方便压缩。JPEG是对每8×8个像素点为一个单位处理的。所以如果原始图像的宽和高不是8的倍数，都需要扩展补成到8的倍数，以便于以块为单位进行处理[46]。JPEG里是对亮度分量Y和色度分量Cr、Cb分别做DCT变换的。

因为图像像素值的范围是0～255，而DCT变换的输入变量范围是－128～127，在完成DCT变换前，需要对每个像素值减去128将像素范围变换为－128～127。这种偏移对于像素块的交流AC系数特性没有影响。二维DCT正向离散余弦变换的公式如式（4.3）所示：

$$F(u,v)=\frac{1}{4}C(u)C(v)\sum_{i=0}^{7}\sum_{j=0}^{7}\left\{f(i,j)\cos\left[\frac{(2i+1)u\pi}{16}\right]\cos\left[\frac{(2j+1)v\pi}{16}\right]\right\} \qquad (4.3)$$

式中：$i,j=0$，1，2,…,7,

表示该像素在像素块矩阵中的行列号，即在像素块中的位置；

$f(i,j)$：输入的像素值，$-128\le f(i,j)\le 127$

$u,v$表示DCT系数的在DCT系数矩阵中的行列号，即DCT系数的位置。

$F(u,v)$：变换后的DCT系数，$-1024\le F(u,v)\le 1023$

$C(u),C(v)=\frac{1}{\sqrt{2}}$　　当$u,v=0$

$C(u),C(v)=1$　　当$u,v\neq 0$

每个8×8的图像像素块，实际上是64点的离散信号，该信号是空间二维参数$i$和 $j$的函数。DCT把这些信号作为输入，然后把它分解成64个正交基信号，每个正交基信号对应于64个独立二维频率域中的一个。DCT的输出也是一个 8×8的矩阵，称为DCT系数，每个系数值由64个输入信号唯一的确定。

在进行解码过程中，反向离散余弦变换的输入是经过反量化得到的8×8的DCT系数，输出则是重构图像的8×8像素块。经过反向的离散余弦变换

IDCT后把图像由频率域转换到空间域像素点的值。二维反向离散余弦变换IDCT的公式如式（4.4）。

$$f(i,j)=\frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7}\left\{C(u)C(v)F(u,v)\cos\left[\frac{(2i+1)u\pi}{16}\right]\cos\left[\frac{(2j+1)v\pi}{16}\right]\right\} \qquad (4.4)$$

式中：$u,v=0,1,2,\mathrm{L}\ ,7$

$i,j$表示该像素在像素块矩阵中的行列号，即在像素块中的位置

$u,v$表示DCT系数在DCT系数矩阵中的行列号，即DCT系数的位置

$C(u),C(v)=\frac{1}{\sqrt{2}}$　　当$u,v=0$

$C(u),C(v)=1$　　当$u,v\neq 0$

$f(i,j)$：　　输入的像素值，$-128\le f(i,j)\le 127$

$F(u,v)$：　　变换后的DCT系数，$-1024\le F(u,v)\le 1023$

由式(4.4)可看出，当转换的是直流DC系数的值时，$u$、$v$ 参数值为0，而且$C(u)$、$C(v)$值为$\frac{1}{\sqrt{2}}$，将参数值代入离散余弦反转换，计算出空间域系数的结果，若要计算的是AC 系数值时，则$C(u)$，$C(v)$等于1，再将其代入公式，计算还原成空间域的像素点的值。

经过反离散余弦变换后的各像素值的范围是－128～127，而图像的像素值是位于0～255间的无符号数，因此，需要对IDCT变换后输出的像素值再加上128，则像素块返回原有的电平。经过IDCT变换后得到像素值若超过了指定的精度，则还应进行“限幅”处理，具体方法如式（4.5）：

$$\begin{cases} Pxy=255 & \text{当 } Pxy>255 \\ Pxy=0 & \text{当 } Pxy<0 \end{cases} \qquad (4.5)$$

反向离散余弦变换需要冗长的运算，所以它是整个图像解压缩过程最花时间的部份。如果以纯软件实现图像解压缩，约有75%的时间需要花在反向离散余弦变换的计算。

### 4.2.5 反取样

JPEG文件使用的色彩模型是YCrCb色彩模型，而不是最常用的RGB模式。因此在对图像压缩前先要对图像进行色彩空间的转换，从RGB转换到YCrCb

模式。YCrCb色彩模式表示图像的优点是亮度信号和色度信号Cr、Cb是分离的[47]。由于人的眼睛对图片上的亮度Y的变化远比色度C的变化敏感，所以在压缩的过程中会将图像的亮度信息全数的保留，对于色度则不需要。我们完全可以用一个点保存一个8位的亮度值，每2×2个点保存一个Cr、Cb值，而在人眼看来，不会感觉到图像有太大的变化。所以，原来用RGB模型，保存4个点需要4×3＝12个字节，而用YCrCb模型仅需要4＋2＝6个字节，当然JPEG格式允许对每个点的色度C值都记录下来。JPEG就是通过对图像亮度和色度的不同的取样比来对亮度和色度取样的。若在压缩过程中，采用的是Y：Cr：Cb＝4：1：1的取样方式，所谓的“4”就是从一个16×16 大小的Y亮度区块中有4个8×8 的像素块，在取样时是全数保留，而“1”所表示的则是Cr、Cb 色度在取样过程中，是从16×16大小的区块中分别抽取出一个8×8的像素块，这6个8×8 的像素块就称为最小编码单元MCU，因此，还原回去后会以3个分别表示Y、Cb、Cr 的16×16大小的区块来表示，其结果如图4-3所示。

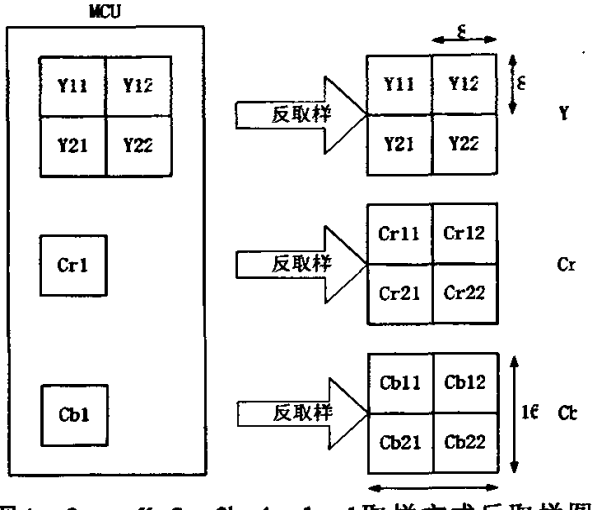


图4－3　Y:Cr:Cb=4：1：1取样方式反取样图

Fig.4－3 The Reverse Sampling about Y:Cr:Cb=4：1：1

## 4.2.6 YCrCb色彩模式转换成RGB色彩模式

最后将YCrCb色彩模式数据转换成RGB色彩模式的BMP位图格式数据，这是JPEG图像格式还原成BMP图像格式的最后一个步骤。色彩模式转换并不

包含在JPEG的标准算法中。在YCrCb颜色空间中，每个分量、每个象素的电平规定为255级，用8位代码表示。

将256级的YCrCb色彩模型转换成RGB色彩模型的计算公式如式（4.6）：

$$\begin{cases} R = Y + 1.402 \times (Cr - 128) \\ G = Y - 0.34414 \times (Cb - 128) \\ B = Y + 1.772 \times (Cb - 128) \end{cases} \quad (4.6)$$

将亮度、色度的像素区块中不同位置所对应的数据，代入到公式（4.6）中，即可将我们所需要的BMP格式的图像转换出来。

到此，整个解码过程完成，经过以上的各个步骤，从输入的图像压缩数据开始到最后得到重构的 RGB 图像。因为本论文是基于 ARM 嵌入式系统的解码和显示，所以先对基本系统解码的过程做了详细的分析。下面将以本文的 ARM 嵌入式的硬件平台为基础，对 JPEG 基本系统中 IDCT 部分用到的一些标准算法进行研究分析，使之更适合于本 ARM 嵌入式硬件平台。

## 4.3 基于 S3C44B0X 的 IDCT 算法的选择

在JPEG解码过程中的反向离散余弦变换IDCT中运算量很大,要进行大量的数乘法和加法运算，而且浮点数的指令要比整数的多得多。所以IDCT部分耗费时间最多，占了整个解码时间的75%以上 ，是决定解码速度主要部分。因此在一些高速或实时应用场合能否快速实现IDCT就成为一个关键因素。为了解决DCT/IDCT计算冗长，减少运算的复杂度，实现快速的DCT/IDCT计算，在过去的几十年里，出现了许多计算DCT的不同算法。

### 4.3.1 DCT算法发展

1977年，W H，Chen等人根据变换矩阵具有对称性，第一次用稀疏矩阵分解法得到DCT-II的快速算法[47]。1984年B.G.Lee 提出一种使用余割因子的DCT分解算法，得到Cooley－Tukey式的简单结构，受到广泛重视[48]。1983年日本学者森川等人利用切比雪夫多项式的逐级分解和多项式同余的概念得到一种DCT算法。这种算法不仅结构简单，且使用余弦做乘子，因而比前述算法优越，但实现结构比较复杂。 1987年，几乎同时有三篇论文论述了DCT的递归算法，其中H.S. Hou的快速算法比较具有代表性。近几年出现的

新算法大多是这几种算法的改进[49]。Duhamel将DCT看成是一种基于循环卷积的算法，并证明对于一维的8点DCT，其乘法的理论下限值是11次，C. Loeffler的DCT快速算法[50]达到了这一极限，它是将DCT运算转为旋转运算,具体实现了这种11步乘法的算法。从此以后，一维DCT快速算法的研究进展缓慢。2000年，Trac. D. Tran提出了一种DCT的近似快速算法，没有用到乘法，但使用了移位运算[51]。在国内，赵耀等人也提出了一种“基于矩阵分解的DCT快速算法”，在该方法中用到了12次乘法[52]。另外，对于三维或者更高维的DCT快速算法也有人研究，在文献[53]中提出了一种快速的三维DCT快速算法。

### 4.3.2 常用的DCT算法

1.多项式变换法　多项式变换法基于多项式环的特性(Polynomial transform based DCT implementation.)，通过蝶形加法运算进行列 DCT 计算，从而减少乘法数量。基于多项式变换的算法仍然利用二维 DCT 的行列分解特性，并且采用常见的一维 DCT 算法计算行 DCT，然后通过蝶形加法运算计算列 DCT。Duhame和Guillemot[53]提出的算法是目前直接多项式变换法中最有效的算法，其加法次数与行一列法相当，乘法次数仅为行一列法的一半。多项式变换法不仅涉及复数运算，并且计算结构很复杂，不容易实现。

2.查表法　运用查表法就是提前将计算过程中所需乘法运算的结果都提前算好并以表格的形式放到内存中，在运算过程中，直接将所需的乘法结果查找出来，尽量减少乘法运算的次数，甚至不用乘法运算，只进行必要的系数加法运算，从而完成整个IDCT运算。查表法的运算时间与变换对象的个数成正比，因为在整个运算过程中不使用乘法运算只有加法运算，因此有较快的速度。文献[54][55]中提出了基于查表法实现二维DCT算法。查表法采用了用空间换取时间的策略，但对嵌入式系统而言，资源有限，对于实现较复杂的运算，用查表法不太实际。本系统若用查表法，设计查询表的大小为440K，如此庞大的一张表，不但占用宝贵的硬件资源，而且要设计完整的查询表，其工作量之大也是显而易见的。所以在现实应用中，使用查表法完成整个二维IDCT运算比较少，可结合其他算法在局部运算中使用。

**3.行－列法** 在二维DCT快速算法中，现在用的较多的是行－列法，即利用二维IDCT具有的可分离性，可以把二维IDCT变成水平方向和垂直方向的两个一维IDCT进行计算。先对每行(列)进行一维IDCT计算，再对每列(行)进行一维IDCT计算，因此，一个 N×N大小的数据块的DCT要计算2N次一维IDCT运算。

一维8点的IDCT变换公式为式（4.7）

$$f(x)=\frac{1}{4}\sum_{u=0}^{7}\left[C(u)F(u)\cos\frac{(2x+1)u\pi}{16}\right] \qquad (4.7)$$

其中： $u=0,1,2,\mathrm{L}\ ,7$

$$C(u)=\frac{1}{\sqrt{2}} \qquad u=0$$

$$C(u)=1 \qquad u\neq 0$$

与式(4.4)的二维DCT公式比较，得到两次8点的一维IDCT复合运算，式(4.8)：

$$f(i,j)=\frac{1}{8}\cdot\sum_{u=0}^{7}\left\{\sqrt{2}C(u)\left[\sum_{v=0}^{7}\sqrt{2}C(v)F(u,v)\cos\frac{(2j+1)v\pi}{16}\right]\cos\frac{(2i+1)u\pi}{16}\right\} \qquad (4.8)$$

其中：式中：$u,v=0,1,2,\mathrm{L}\ ,7$

$i,j$表示该像素在像素块矩阵中的行列号，即在像素块中的位置

$u,v$表示DCT系数在DCT系数矩阵中的行列号，即DCT系数的位置

$C(u),C(v)=\frac{1}{\sqrt{2}}$ 当$u,v=0$

$C(u),C(v)=1$ 当$u,v\neq 0$

$f(i,j)$: 输入的像素值，$-128\le f(i,j)\le 127$

$F(u,v)$: 变换后的DCT系数，$-1024\le F(u,v)\le 1023$

行－列法将2维8×8的DCT变换分离成16次一维的DCT变换进行，通过采用一些一维8点的快速DCT变换算法，可以减少乘法和加法运算的次数，从而提高二维DCT变换的运算速度。

表4－1列出了一些常用的一维的快速DCT算法所使用的运算次数。AAN的算法[56]计算8×8的二维DCT变换只需要80次乘法和464次加法。但是AAN算法计算得到的是放大了的DCT变换结果，而且在固定精度的定点运算中由于

尺度转换合并到量化中导致计算结果不精确。FEIG算法在AAN算法的基础上发展而来，FEIG算法计算8×8的二维DCT总共只需要462次加法、54次乘法和6 次左移操作。FEIG较AAN算法速度更快，但是具有与AAN 算法类似的缺点，并且结构更加复杂。由于AAN、FEIG系列算法计算结果精度不高，同时算法结构较为复杂，多应用于基于通用CPU的软件方案。

表 4－1 常用一维 IDCT 算法计算复杂度比较

Table.4－1 Computational complexity of 1－D IDCT algorithms

| 算法类型 | 一维 8 点 DCT | | 二维 8×8 DCT | |
|---|---|---|---|---|
| | 加法次数 | 乘法次数 | 加法次数 | 乘法次数 |
| 一维 DCT 直接运算 | 56 | 64 | 896 | 1024 |
| Lee－Huang 算法 | 29 | 12 | 464 | 192 |
| Loeffler算法 | 29 | 11 | 464 | 176 |
| Chen 算法 | 26 | 16 | 416 | 256 |
| Feig算法 | 29 | 12 | 464 | 192 |
| AAN 算法 | 29 | 5 | 464 | 80 |

经过以上分析研究，在本系统中采用行列分解法将二维8×8 IDCT运算转化成16次的一维8点IDCT运算，用Loeffler的多项式算法来实现8点一维的快速IDCT运算。

### 4.3.3 Loeffler算法

Loeffler算法将8点的一维DCT运算分为串行的四级运算，各级内部的运算可并行处理。因为二维的IDCT要分解成16次一维的8点IDCT，所以用Loeffler的算法最终需要176次乘法和464次加法。图4－4为Loeffler算法的流图，$x_0 \sim x_7$是输入数据，$y_0 \sim y_7$是输出的DCT系数。流图中有三种运算因子:蝶形因子、旋转因子和倍乘因子，分别如图4-5所示。

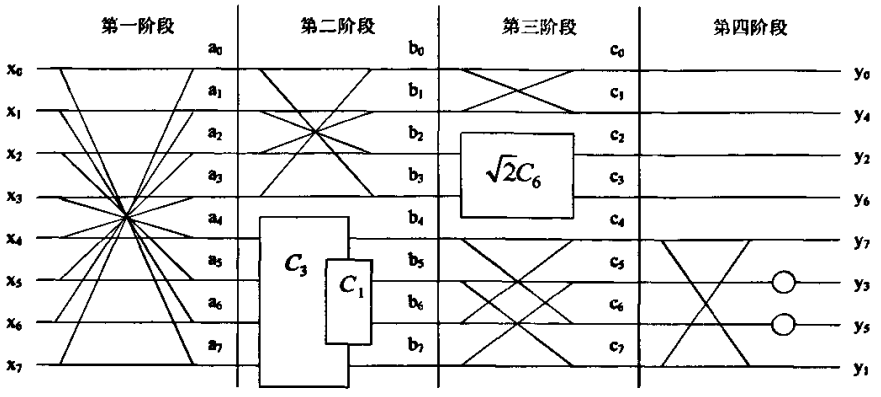


图4－4　一维DCT Loeffler算法蝶形数据流图

Fig. 4－4 The Data Flow of 1－D DCT Loeffler Algorithm

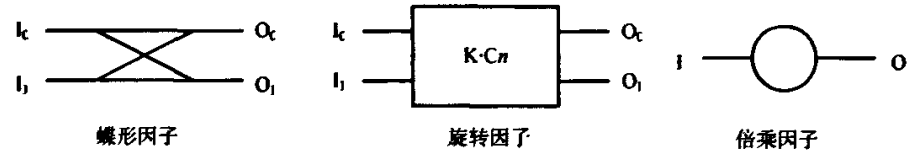


图4－5 Loeffler 算法的运算因子

Fig.4-5 The Computation Factors of Loeffler Algorithm

蝶形因子的运算关系为式（4.9），完成一次蝶形运算要2次加法：

$$\begin{cases} O_0 = I_0 + I_1 \\ O_1 = I_0 - I_1 \end{cases} \tag{4.9}$$

旋转因子的运算关系为（4.10）：

$$\begin{cases} O_0 = I_0 \cdot \cos\dfrac{n\pi}{2N} + I_1 \cdot K \cdot \sin\dfrac{n\pi}{2N} \\ O_1 = -I_1 \cdot K \cdot \sin\dfrac{n\pi}{2N} + I_1 \cdot K \cdot \cos\dfrac{n\pi}{2N} \end{cases} \tag{4.10}$$

直接计算旋转因子要4次乘法和2次加法，可将其变形如下式（4.11）：

$$\begin{cases} O_0 = \left[K \cdot \sin\dfrac{n\pi}{2N} - K \cdot \cos\dfrac{n\pi}{2N}\right] \cdot I_1 + K \cdot \cos\dfrac{n\pi}{2N} \cdot (I_0 + I_1) \\ O_1 = -\left[K \cdot \cos\dfrac{n\pi}{2N} + K \cdot \sin\dfrac{n\pi}{2N}\right] \cdot I_0 + K \cdot \cos\dfrac{n\pi}{2N} \cdot (I_0 + I_1) \end{cases} \quad (4.11)$$

对一个旋转因子而言，K值和n值都是确定的，所以，式（4.11）中 $K \cdot \cos\dfrac{n\pi}{2N}$、$\left[K \cdot \sin\dfrac{n\pi}{2N} - K \cdot \cos\dfrac{n\pi}{2N}\right]$、$\left[K \cdot \cos\dfrac{n\pi}{2N} + K \cdot \sin\dfrac{n\pi}{2N}\right]$的值都是确定的，即相当于一个常数值，所以将旋转因子运算公式变形为式（4.11）后，完成一次旋转因子则需要3次乘法和3次加法。倍乘因子的运算关系较简单，只要进行一次乘法运算：

$O = \sqrt{2} \cdot I$

根据以上各运算因子的计算公式及算法流图，可得到一维8点DCT的结果，可以验证共需要29次加法和11次乘法。

## 4.4 小结

本章首先对JPEG基本系统的解码过程做了详细的论述，为后面解码程序的编写打好基础。接着对一些常用的IDCT变换的算法进行了各自算法复杂度和优缺点的的比较与分析，最后，根据本系统的硬件平台选择出IDCT转换的方法，即将二维8×8的IDCT转化为16次的一维IDCT来实现，一维的IDCT算法选用Loeffler的算法。

# 第五章　图像显示在ARM平台上的实现与优化

本章将介绍JPEG静态图像显示系统在ARM7嵌入式平台上的具体实现。在此基础上，将解码显示软件在ARM7上进行移植并在具体的硬件环境下进行优化，最后对得到实验测试结果进行了分析。

## 5.1 JPEG静态图像显示软件的设计方法及总体结构

### 5.1.1 软件的设计方法

首先在PC机上，借助ADS开发工具用通用的C语言对JPEG解码和显示软件进行编写和软件仿真，实现基本的解码和显示功能。虽然用C语言实现的程序，除了PC机外，在其它一些常用的处理器平台上，也能够正常运行，但就具体的某个应用平台来讲，每个处理器有其各自不同的特点，程序运行的性能可能会有很大的差异。所以，为了充分发挥处理器的优势和进一步提高程序的性能，在实际应用时，还必须根据处理器的特点对软件进行必要的优化处理。本课题中的图像显示最终的运行平台是S3C44B0X，所以，在PC机上实现了软件的功能后，还要将它移植到S3C44B0X上进行运行，并根据S3C44B0X的硬件环境和系统资源的特点采取优化和改进措施，使其在基于S3C44B0X的硬件平台上实现静态图像的实时解码和显示。

对于嵌入式的实时图像解码应用，最重要的问题之一是解码的实时性，即较低的延迟和较快的速度。在此基础上，尽可能追求更好的图像质量。所以，进行嵌入式的实时解码器设计，很多情况下需要在解码器的性能和解码速度之间进行恰当的折衷。

### 5.1.2 软件的总体结构和流程

JPEG静态图像的显示软件的总体结构如图5-1所示，包括解码程序和显示处理两大部分。首先，要对JPEG解码的过程进行初始化，即读取标记码数据对头文件信息进行处理；然后进行正式的解码过程，包括熵解码、反量化、

反向离散余弦变换IDCT；执行完解码过程得到重构图像的YCrCb数据，并将YCrCb颜色空间的数据转换为24位RGB颜色空间的数据，就得到了重构的图像数据。由于S3C44B0X的LCD控制器不支持真彩色24位RGB显示，只支持8位RGB332格式即256色的彩色显示模式，所以还要将24位RGB数据转化到256色的数据格式。最后将转化之后的256彩色数据送入LCD显示缓冲区，在LCD上显示出解码重构图像。解码程序采用了以MCU行为单位的解码顺序，每一幅BMP图像在JPEG编码的时候都被划分成了若干个MCU，每个MCU可以由一个或多个8×8的分块组成，在解码的时候，我们需要按照顺序将压缩的图像数据以8×8的分块为单位分别进行熵解码、发量化、和IDCT变换以及颜色空间转换，将经过这些操作的所有MCU分块的数据存放到一个缓冲lpPtr中，重复上面过程直到结束，得到每一个像素行的位图数据，从而完成解码，

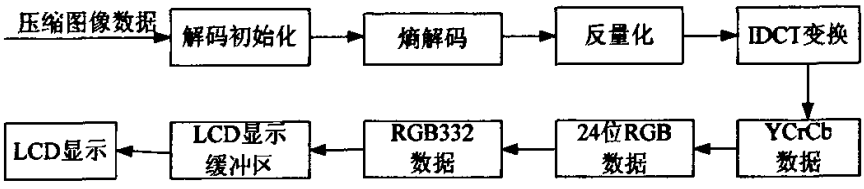


图5-1 JPEG静态图像显示软件结构图

Fig. 5-1 Decoding and Display Block Diagram

## 5.2 解码及显示程序的实现

解码过程主要包括熵解码、反量化、反离散余弦，而熵解码的算法、反离散余弦变换的算法是解码程序设计的核心，直接决定了解码的性能。实际上，解码过程的绝大部分处理时间都用于实现这些算法运算上，其算法的复杂程度和优化程度最终决定着解码程序的速度。

基于嵌入式系统应用的图像显示对于解码器实时性的要求，及可用的系统资源的限制，在解码程序的设计过程中尽量采用了一些目前被广泛使用的较为成熟的算法。相对而言，这些算法实现起来复杂程度不是很高，因而对系统资源也没有太高的要求。同时，由于这些算法的成熟性，有一些比较好的可替代的快速算法可以采用。当然，从解码效率或图像的质量上来看，这些算法不一定是最好的，但从解码程序的实时性以及总体性能的综合考虑来看，这些算法的选择是比较合理的。

### 5.2.1 熵解码算法的实现

熵解码是解码过程中最重要的一环，主要是对读入的图像的数据进行的DC直系数和AC交流系数的Huffman解码。

JPEG算法中提供了标准的Huffman码表，也允许用户选择自适应的Huffman码表。考虑到每副图像各自不同的特点，压缩编码时会根据各自的特点生产Huffman码表，所以决定采用自适应的Huffman码表。采用自适应的Huffman码表，首先要统计输入图像的特性，先生成码树，再做反推得到各级Huffman码表。

在每个JPEG文件里定义了一张表来描述Huffman树，定义在DHT标记后面，Huffman代码的长度限制在16bit内。一般一个JPEG文件里会有4个Huffman码表，用于直流DC系数的Huffman码表（包括一个亮度表和一个色度表）；用于交流AC的Huffman码表(包括一个亮度表和一个色度表)。每个Huffman码表分别是这样保存的：前面的16字节对应长度为1到16的Huffman码字的个数，接下来是这16个数字之和个字节，对应字节就是对应Huffman码字的等价数字。图5－2为Huffman解码一个码字的流程图。

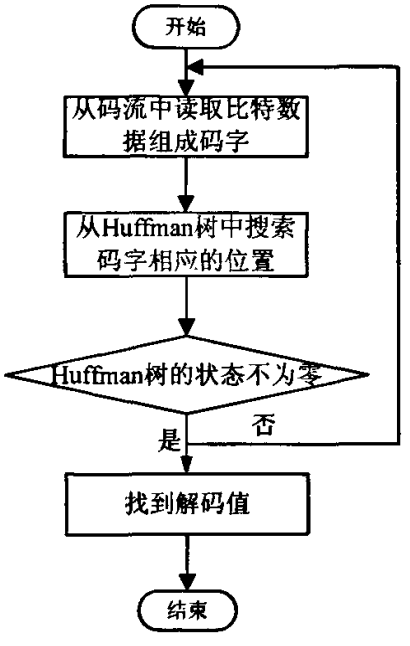


图5-2：Huffman解码一个码字的流程

Fig.5-2 The Flow Chart of One Code in Huffman Decoding

解码时，在Huffman解码端收到的是图像数据经过压缩后行成的码流。先从码流中读入第一个比特位数据，在Huffman树中进行搜索，如果Huffman树的值为零，说明该比特数据不是完整的Huffman编码，然后再读入下一个比特位数据加上前面的比特位数据组成新的码字，然后再在Huffman树中进行搜索，如果Huffman树的值还为零，就重复前面的过程；如果Huffman树的值不为零，就找到了码字的位置，根据码字的位置找到解码的值。在程序中用子程序DecodeElement()实现Huffman一个码字的解码，解码输出是一个8位值。如果在进行Huffman解码时碰巧产生了一个0xFF，那么就用0xFF 0x00代替。即是说在JPEG图像解码时碰到0xFF00就把它当作FF处理。

### 5.2.2 反量化的实现

JPEG的头文件里包含了量化表，定义在标记DQT后，以标记码0xFFDB开始，一个标记码开始定义一张量化表。JPEG文件里一般含有两个量化表，一个是用于亮度分量的量化表，一个是用于色度分量的量化表。在位图被压缩称为JPEG文件后，都会生成自己的量化表，因此，在解码时可以使用每个文件自身所带的量化表，也可以采用JPEG标准推荐的量化表。考虑到每个文件的特点，决定在解码时采用文件自带的量化表来对熵解码出来的系数进行反量化，反量化的操作就是对解码系数乘上对应的量化阶矩。反量化程序实现起来比较简单，在程序中用函数IQtIZzBlock()来实现。反量化完的数据存buffer2[8][8]缓存中。

```
//反量化
  void  IQtIZzBlock(short  *s)
   {
     short i,j;
     short tag;
     short *pQt;
     int buffer2[8][8];
     for(i=0; i<8; i++)                    // 开始反量化
  {
    for(j=0; j<8; j++)
       { tag=Zig_Zag[i][j];
        buffer2[i][j]=(int)s[tag]*(int)pQt[tag]; }
  }
```

### 5.2.3 反离散余弦变换IDCT的实现

IDCT的运算量很大,其中要进行大量的浮点乘法和加法运算，因而在JPEG解码过程中IDCT所占时间最多 ,占到了整个解码时间的75%以上,在实时应用中能否快速实现IDCT就成为决定JPEG解码速度一个关键因素。本文采用行列分解法先将二维IDCT分解成一维8点的IDCT，对于一维8点IDCT采用Loeffler的快速算法。完成一次二维8×8的IDCT运算总共要进行16次的8点一维IDCT运算。实现二维IDCT运算的流程图如图5－3 所示。

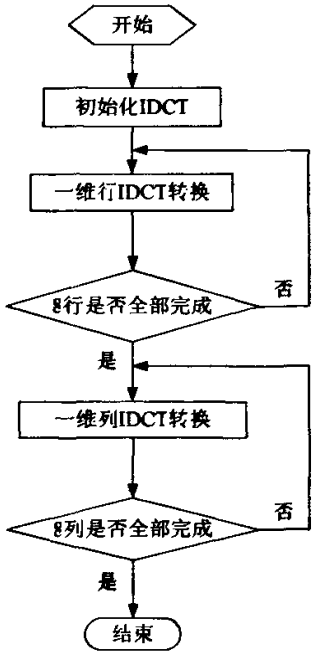


图5－3　二维IDCT运算流程图

Fig. 5－3 The Flow Chart of 2-D IDCT

在IDCT的初始化中，对IDCT过程中用到的中间缓存区进行划分，并将并将其内容清零。开始二维IDCT转换时，先读入经过反量化后的数据进行8次一维的行变换，将变换后的数据存入中间数据缓冲区。在8次一维的行转换完成后，对存在中间数据缓冲区的数据再进行8次的一维列IDCT变换。得到的就是二维IDCT转换的结果。对于每次一维的行IDCT变换和一维的列IDCT变换，采用Loeffler的快速算法,将一维8点IDCT运算分为四个阶段进行。因为DCT是正交变换，IDCT可以通过DCT转置来获得，对应在DCT的数据流图中，

就是数据流的方向取反，各乘积项大小和位置不变，所以根据图4-4将输出作为输入，数据流向从右到左，蝶形因子和倍乘因子运算方向变反，旋转因子位置和大小都不变，则得到IDCT的蝶形图，如图5-4所示。

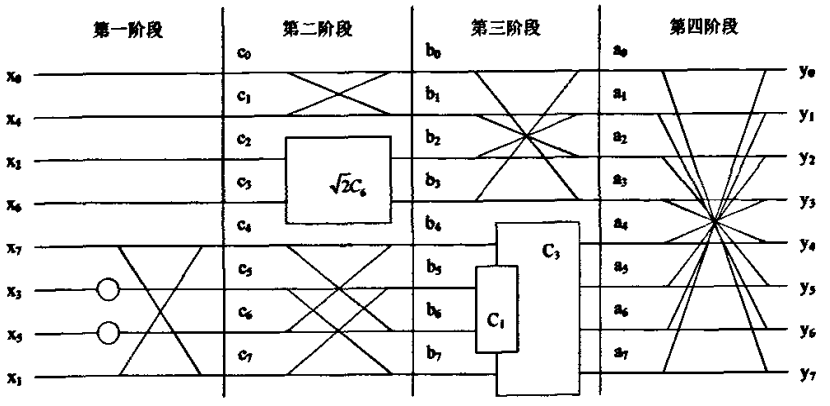


图5-4 一维IDCT变换Loeffler算法蝶形数据流图

Fig.5-4 The Data Flow Chart of 1—D IDCT Loeffler Algorithm

根据蝶形图，按数据流向进行各项加法和乘法运算，共11次乘法和29次加法。一维的行IDCT变换在程序中用函数idctrow()来实现，一维的列IDCT变换在程序中用函数idctcol()来实现。完成8次行变换和8次列变换总共进行176次乘法和464次加法。

## 5.2.4 色彩模式转化

将IDCT结果得到的YCrCb色彩模式的数据转化为RGB色彩模式的数据，根据转换公式(4.6)来实现，程序中为函数StoreBuffer()。

由于S3C44B0的LCD控制器不支持24位RGB真彩色显示模式，只支持8位RGB332显示，所以必须将24位RGB数据转化到256彩色，程序中函数void tranRGBto256()实现其功能。最后将转化之后的256彩色数据送入LCD显示缓冲区，LCD显示图像。

```
//24位RGB转化为256色
void tranRGBto256()
{
    int i,iRed,iGreen,iBlue;
    pImmBitmap1=(BYTE*)(malloc(sizeof(BYTE)*(ImgWidth*ImgHeight)));
    for (i=640*240; i>0; i--)
```

```
{ iBlue= *lpPtr++;
  iGreen =*lpPtr++;
  iRed= *lpPtr++;
  iRed = (iRed*7+127)/255;
  iGreen = (iGreen*7+127)/255;
  iBlue = (iBlue+42)/85;
  plmmBitmap1[i]=iBlue+(iGreen<<2)+(iRed<<5); }
```

## 5.3 优化

由于在嵌入式实时系统中，对程序的运行速度要求很高，因此在此平台上的程序不仅要保证正确性，而且要经过运行性能测试，确认其运行速度是否可以满足系统要求。在实际程序开发中，程序的开发是在 PC 机上完成，仅保证了逻辑上的正确性，没有根据具体的运行平台特点，使代码达到平台最优。所以在 ARM 上的程序开发中，除了算法实现阶段外，代码优化也是必不少的步骤。众所周知，代码优化会降低原代码的可读性。所以通常只对经常被调用且对性能影响较大的部分进行优化。结合程序运行的硬件平台的特点从以下几方面对代码进行优化，使平台能充分发挥出其能力。

### 5.3.1 除法运算的处理

ARM7处理器硬件上不支持除法指令，编译器是通过调用C库函数来实现除法运算的[67]。除法和模运算（/和%）执行起来比较慢，所以在程序中尽量避免使用。在不能避免除法运算时，尽量使除数和被除数都是无符号的整数，有符号的除法执行起来会更慢，因为他们先要取得除数和被除数的绝对值，在调用无符号除法运算，最后再确定结果的符号。对于重复对同一个除数的除法，把除法变为乘法运算，即乘以它的倒数。

### 5.3.2 浮点运算的处理

S3C44B0处理器硬件上不支持浮点运算。这样在一个对价格敏感的嵌入式应用系统中，可节省空间和降低功耗；但这并不意味着不能计算浮点运算，因为C编译器在软件上支持浮点运算。实际上，C编译器要把每一个浮点操作

转换为一个子程序调用。C库函数中的子程序使用整形运算来模拟浮点操作。这些代码是用高度优化的汇编语言编写的。执行一次浮点运算需要很多条汇编语句来完成，执行起来还是要比相应整形运算慢的多。在头文件JPEG.H中，我们定义了W1、W2、W3、W5、W6、W7，它是为解码中需要大量浮点运算IDCT准备的。实际上，它是在IDCT之前对每一个系数左移12位，即扩大$2^{12}$，在IDCT之后，再右移12位，从而避免了浮点运算。

### 5.3.3 循环程序的处理

记数循环是程序中十分常用的流程控制结构。在C中，类似下面的for循环会经常用到：

```
for (loop=1;loop<=limit;loop++)
```

这种累加计数的方法符合一般的自然思维习惯， 所以比下面的递减计数方法使用得多：

```
for (loop=limit;loop!=0;loop--)
```

这两者在逻辑上并没有效率差异， 但是映射到具体的体系结构中，就产生了很大的不同，累加法比递减法多用了一条指令[58]，当循环次数比较大的时候，这两段代码就会在性能上产生出明显的差异。分析其中的本质原因， 当进行非零的常数比较时，必须用专门的CMP指令来执行；而当一个变量与零进行比较时，ARM 指令可以直接利用条件执行的特性（NE）来进行判别。因此， 在用到循环执行时，采用了递减至零的方法来设置循环条件。

### 5.3.4 处理器内部寄存器的利用

编译器会试图对C函数中的每一个局部变量分配一个寄存器。如果几个局部变量不会交迭使用，那么编译器会对它们分配同一个寄存器。当局部变量多于可用的寄存器时，编译器会把多余的变量存储到堆栈。由于这些变量被写入了存储器，所以被称为溢出或者替换变量，就像虚拟存储器的内容被替换到硬盘一样。与分配在寄存器中的变量相比，对溢出变量的访问要慢得多。

理论上，C编译器可以分配14个变量到寄存器而不溢出。实际上，一些编译器对某些寄存器有特定的用途（如用r12作为临时过渡寄存器使用，r13

用作堆栈指针，r14用作连接寄存器来保存子程序的返回地址），编译器就不再分配任何变量给它了。因此，将函数内部循环所用局部变量的数目减到过12个以下，这样，编译器就可以把这些变量都分配给ARM寄存器。

### 5.3.5 函数调用的处理

ARM处理器的过程调用标准ATPCS（ARM-Thumb Procedure Call Standard)定义寄存器组中的{r0～r3}作为参数传递和结果返回寄存器，如果参数的个数超过四个，则使用堆栈进行传递[59]。我们知道内部寄存器的访问速度是远大于存储器的，所以在程序中尽量使参数传递在寄存器里面进行，即应尽量把函数的参数控制在四个以下，也可以将几个相关的参数组织在一个结构体中，用传递结构体指针来代替多个参数。另外，在调用函数时，如果函数体很小，只用到很少的寄存器（很少的局部变量），可以把调用函数和被调用函数放在同一个C文件中，并且要先定义后调用，这样编译器就知道了被调用函数生成的代码，并以此对函数调用进行优化。

## 5.4 软件的移植

所谓移植，就是使在某个处理器平台上运行的程序能够在另一个微处理器平台或微控器上运行[60]。对本系统而言，就是要将在PC机上编的JPEG解压缩算法程序，使之在ARM微处理器平台上能够运行。因为在C语言和ADS开发工具中，数据类型完全相同，不同的只是这些代码被不同的编译器生成不同的OBJ目标文件，在不同的硬件平台上运行，所以移植相对比较容易。

首先，在交叉编译环境ADS下将Bootloader代码、初始化代码、LCD驱动代码、主程序代码等添加到工程文件中，设置好运行环境，包括入口地址、编译生成二进制文件名jpegdecode.bin及路径等，然后编译工程文件，调试修改程序，直到没有错误为止，这时生成了二进制文件，将此二进制文件烧入FLASH中的某个地址空间，占用空间的长度为该文件大小。另外，程序是固化在Flash存储器中的，要在SDRAM中运行，因此在Bootloader 中要给程序指定一个在SDRAM中运行的地址，使程序能够在SDRAM中正确运行。

## 5.5 测试及结果分析

在图像解码的过程中，我们最关心的就是速度、图像质量以及对资源的占用问题。在将优化之后的JPEG解压缩和显示程序移植到ARM平台上之后，需要测试程序在实际环境中的运行速度和效果。

1.解码速度的测试　为了测一幅JPEG图像的解码时间，我们利用了看门狗定时器，编写了两个函数来实现此功能，这两个函数分别是Timer_Start(int divider)和Timer_Stop()。

Timer_Start(int divider)是用来启动定时器， Timer_Stop()的作用是使定时器停止。

```
//看门狗定时器启动
void Timer_Start(int divider)
{
 rWTCON=((MCLK/1000000-1)<<8)|(divider<<3);
//禁止看门狗定时器，确定时钟除数因子
 rWTDAT=0xffff;
 rWTCNT=0xffff;
rWTCON=((MCLK/1000000-1)<<8)|(divider<<3)|(1<<5);
}
//开门狗定时器停止
int Timer_Stop()
{
 int k;
 rWTCON=((MCLK/1000000-1)<<8);
 k=(0xffff-rWTCNT);
 return k;
}
```

由于系统时钟频率MCLK=60MHz,看门狗计数寄存器rWTCON倒计数一次的时间：$t\_watchdog=\frac{1}{\frac{MCLK}{(预分频值+1)\times 64}}$，而预分频值为rWTCON寄存器的位[15：8]的内容，值为59，除数因子为1/64，所以t_wachdog=64 $\mu$ s。Timer_Stop()运行结果与t_wachdog的乘积就是解码时间。

图5-5是一幅320×240大小的压缩前的BMP格式的灰度图像，图5-6是对应图5-5的压缩率为4.66的解码图像，解码时间约为0.327秒。图5-7是对应图5-5的压缩率为9.5的解码图像，解码时间约为 0.385秒。

图5-5：LINA.bmp原始位图格式图像
Fig.5-5 The BMP Image of LINA.bmp

图5-6：图5-5压缩率为4.66的解码图像
Fig.5-6 The Reconstruction Image of LINA.bmp
(Compression Ratio:4.66)

图5-7：图5-5压缩率为9.5的解码图像
Fig.5-7 The Reconstruction Image of LINA.bmp
(Compression Ratio:9.5)

图5-8是一幅640×240大小的压缩前的BMP格式的真彩色图像，图5-9是对应图5-8的压缩率为24.86的解码图像，解码时间约为1.269秒。图5-10是对应图5-8的压缩率为9.87的解码图像，解码时间约为1.148秒。

图5-8：butterfly.bmp原始位图格式图像
Fig.5-8 The BMP Image of butterfly.bmp

图5-9：图5-8压缩率为24.86的解码图像
Fig.5-9 The Reconstruction Image of butterfly.bmp
(Compression Ratio:24.86)

图5-10：图5-8压缩率为9.87的解码图像
Fig.5-10 The Reconstruction Image of butterfly.bmp
(Compression Ratio:9.87)

从测出的解码时间来看，压缩比越大，解码时间越长。因为在图像压缩时，压缩比越大，丢弃的图像数据越多，在解码时需要将丢弃的数据进行还原，压缩比越大进行运算的次数就越多，所以解码时间就越长。真彩色图像的解码时间明显大于灰度图像的主要原因是灰度图像只有一个亮度分量，解码时候不需要进行反取样提取MCU和色彩空间的转换运算，而真彩色图像有一个亮度分量和两个色度分量，解码的时候需要要对各分量进行反取样提取MCU及色彩空间的转换，虽然反取样子程序和色彩空间转换子程序耗时不长，但是中间涉及到大块的数据复制，从而使整个解码时间明显增大。

最后通过对大量不同大小、不同压缩率图片进行测试，解码一幅图像的时间都在1.5秒以内，所以软件可以满足实时解码显示的要求。

**2. 图像重构质量的测试** 对解码图像的质量用峰值信噪比PSNR（Peak Signal to Noise Ratio）的评估。信噪比越高则图像质量越好，反之图像质量越差。一般说来，当PSNR值在28 以上时，图象质量差异不太显著，当高于35～40时，则肉眼分辨不出差异[61]。

峰值信噪比的定义如式（5.4），给定一幅大小为M×N的数字化图象 $f(x,y)$ 和参考图象 $f_0(x,y)$，则图象 $f(x,y)$ 的 $PSNR$ 为式（5.1）：

$$PSNR=10\cdot lg\frac{f_{max}^2}{\frac{1}{MN}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}[f(x,y)-f_0(x,y)]^2} \tag{5.1}$$

引入另一常用质量评价指标均方误差 $MSE$ ，其表达式为（5.2）：

$$MSE=\frac{\sum_{x=0}^{M-1}\sum_{y=0}^{n-1}[f(x,y)-f_0(x,y)]^2}{MN} \tag{5.2}$$

则最后PSNR为式（5.3）：

$$PSNR=10\cdot lg\frac{f_{max}^2}{MSE} \tag{5.3}$$

其中 $f_{max}$ 是函数 $f(x,y)$ 的最大灰度值，例如，常用的8bit的灰度图像中，则 $f_{max}$ 的最大值为255。

而在彩色数字图像中，由于图像的颜色用 $RGB$ 3基色的组合表示，每个颜色分量需用一个字节表示，于是相应的峰值信噪比可以表示为式（5.4）：

$$\overline{PSNR}=\frac{1}{3}(PSNR_R+PSNR_G+PSNR_B) \tag{5.4}$$

其中 $PSNR_R$、$PSNR_G$、$PSNR_B$ 分别为图像的 $R$、$G$、$B$ 帧的峰值信噪比。

在表（5－1）中列出了两副不同图像压缩率的解码图像的信噪比。

图5-6是对应图5-5的压缩率为4.66的解码图像，信噪比为35.542 dB，图5-7是对应图5-5的压缩率为9.5的解码图像，信噪比为27.073dB。从解码得到的图像结果看，当压缩率为4.66时，几乎看不到图像有什么细节损失，而当压缩率为9.5时，Lena帽子上的层次明显不清，局部失真。图5-9是对应

图5-8的压缩率为9.87的解码图像，信噪比为33.96 dB；图5-10是对应图5-8的压缩率为24.86的解码图像，信噪比为26.78 dB。从解码后的图像可以看出，当压缩率增大时，解码之后的图像和原图像有一定的差异。

表5-1： 图5-5、图5-6不同压缩率解码图像的信噪比
Table.5-1 Two Images' PSNR With Different Compression Ratio

| 图像名称 | 压缩率 | 文件大小（K） | 峰值信噪比（dB） |
|---|---|---|---|
| 5-5 | 4.66 | 16.3 | 35.542 |
| | 5.13 | 14.8 | 26.785 |
| | 5.59 | 13.6 | 21.522 |
| | 9.5 | 8.0 | 17.850 |
| 5-8 | 9.8 | 45.6 | 34.261 |
| | 12.78 | 35.2 | 31.596 |
| | 18.29 | 24.6 | 29.089 |
| | 22.73 | 19.8 | 27.073 |

从以上结果可以看出，解码重构图像的质量是和图像的压缩率及峰值信噪比密切相关的，随着压缩率的增大，重构图像的质量会逐步降低。这是由于我们采用的是有损压缩，图像中重复或不重要的信息会被丢失，因此容易造成图像数据的损伤。因此，根据使用用途选择合适的压缩比是十分必要的。如果对恢复图像质量要求不是很高，可以选择较大的压缩比，反之，要选择较小压缩比。通过对以上实验结果的分析，本系统在解码速度和重构图像的质量方面都能满足实际使用的要求。

## 5.5 小结

本章论述了静态图像显示系统软件的设计方法流程，在此基础上实现了解码和显示程序。针对ARM的硬件特性和C语言的语法结构，从除法运算的处理、浮点数的处理、循环程序的处理、对内部寄存器的使用和函数调用五个方面对程序进行了优化。将程序移植到嵌入式平台，对程序运行进行了测试，并对测试结果从解码速度和重构图像质量两个方面进行了分析。

# 结论

本论文首先对JPEG标准进行了介绍并详细论述了JPEG解码过程，研究了基于ARM的JPEG静态图像的解压缩技术，设计并成功实现了图像解压缩以及显示系统。

## 1. 本论文完成的主要工作有：

### （1）构建了基于ARM处理器平台的嵌入式系统

设计了基于ARM7 S3C44B0X的嵌入式系统的硬件平台，主要包括：存储模块、显示模块、USB模块。在硬件平台基础上，完成了嵌入式系统的软件支持，软件部分主要有：系统的启动引导程序Bootloader、显示模块的驱动、外围设备的驱动，包括存储器、UART等。本系统中的Bootloader是采用较流行的u-boot进行裁减和修改而实现的。

### （2）编程实现了JPEG图像解压缩

对图像编码原理及JPEG编解码标准做了认真研究，在Windows平台下利用交叉编译环境用C语言实现了静态JPEG图像解码器。在软件设计中，考虑到算法实现的复杂程度和嵌入式系统实时性的要求，选用了比较成熟高效的算法。

### （3）解码程序从PC机到目标平台的移植

基于设计的嵌入式系统环境，完成了JPEG图像解压缩程序到ARM平台的移植、调试以及优化。由于PC机和ARM内部结构的不同，移植到ARM平台上程序效率低下，因此针对ARM的硬件特点，对程序进行了部分优化。程序的优化主要从ARM的硬件特性和C语言的语法特性两个方面进行。

实验结果表明：本论文设计实现的基于IDCT变换的JPEG基本系统解码程序能够正确完成JPEG图像的解码，解码速度较快，恢复图像的质量良好，可以应用于处理静止图像的场合(如数码相机、数字手持设备等)，也对研究M-JPEG、MPEG、MPEG-II、H. 26x有一定的参考价值。

通过这些工作使我对JPEG解码过程、ARM ADS平台和程序优化的方法有了深入的理解，掌握了程序优化思想以及基于嵌入式系统软件开发等等。通过该项目的工作，更加深入的理解JPEG的编解码技术，通过对解码算法的优

化，使JPEG解码速度更快，更适合于在ARM平台等能力受限的嵌入式系统上应用。这些为我今后的学习和研究工作打下了良好的基础，在硕士研究生学习阶段收获的知识也将会受用终生。

## 2. 前景展望

当今网络与移动多媒体技术的飞速发展，对诸如图像、视频、音频的实时处理需求不断增强，而嵌入式也是未来的发展主流，因此嵌入式图像处理系统未来的应用会有更加广阔的前景。随着人们对多媒体产品性能要求的不断提高，本系统的进一步研究将从系统的性能提高和功能扩展两个方面进行。

（1）系统性能方面

虽然本文所设计的解码器速度基本能满足实时解码的要求，但人们对嵌入式多媒体设备的运行速度要求越来越高，针对ARM特性对程序可再做进一步优化，充分发挥ARM的潜能是十分必要的。对解码过程所使用的算法实现进行改善，对运算耗时比较多的IDCT部分，可以用内嵌汇编来实现，那样软件运行的效率将会有很大提高。这也是提高解码速度的一个突破口。

（2）系统功能方面

本文设计的系统已经实现的显示功能是利用按键来选择要解码和显示的压缩图像文件，在后续的研究中将采用触摸屏来代替按键输入，开发GUI（Graphics User Interface）图形用户界面，对欲解码显示的图像用触摸选择的方式将会更加的灵活和方便。在系统硬件设计中，为了以后功能扩展已经设计有USB接口模块，在进一步的研究中，会对USB软件部分进行实现。另外，本系统现在是在无操作系统模式下运行，在硬件平台上嵌入操作系统，利用操作系统对整个系统的软硬件进行管理，这对系统的性能将会有一个很大的提高，使得系统可以实现更多的功能。

# 参考文献

[1] 何斌.VC++数字图像处理（第二版）[M].北京：人民邮电出版社，2002.

[2] 景小军.图像处理技术及其应用[M]. 北京：国防工业出版社，2005.

[3] 沈庭芳 方子文. 数字图像处理及模式识别[M].北京：北京理工大学出版社，1998.

[4] 沈兰荪 卓力.小波编码与网络视频传输[M].北京：科学出版社，2005.

[5] JEAN J.LABROSSE 著，邵贝贝 译.μC/COSII－源码公开的实时嵌入式操作系统[M].北京：中国电力出版社，2001.

[6] 邹思轶.嵌入式 Linux 设计与应用[M].北京：清华大学出版社，2002.

[7] Ahmed N. Discrete Cosine Transform[J]. IEEE Tran. Computers, 1974, Vol.23(1):90～93.

[8] 技术在线网站新闻.[日经 BP 社 2005 年 10 月 17 日报道][Z]. http://china.nikkeibp.co.jp/china/news/elec/200510/pr_elec200510170120.html

[9] SOHU 网站新闻.[中星微推出 Vinno 芯片系列产品][Z]. http://it.sohu.com/20060328/n242503085.shtml

[10] 上海三意电机驱动有限公司.DM642 图像处理平台[Z]. http://www.dspdsp.com/Article_Show.asp?ArticleID=907.

[11] http://www.qjy168.com/shop/disp_provide.php?id=52094.

[12] 章承科.多核处理器架构的高速 JPEG 解码算法[J].单片机与嵌入式系统应用，2006，No.1:44～47.

[13] 魏璞.JPEG 解码算法在多 CPU 嵌入式系统中的实现及性能优化[D].成都：电子科技大学，2006.

[14] 罗凤武.基于 MCF5275 的 JPEG 解码算法的设计与实现[D].成都：电子科技大学,2003.

[15] 魏忠义 朱磊. 基于 DSP 的 JPEG 图像解码算法的实现[J].现代电子技术，2005, Vol2: 66～68.

[16] 金燕波 朗锐 罗发根. 利用 TMS320C6201 DSP 芯片进行图像压缩[J]. 电子技术应用，2004, Vol.1: 63～66.

[17] 张浩.基于 TMS320DSC21 的嵌入式手持图像显示系统的软件设计[D]. 北京：

中国农业大学，2005.

[18] 刘东.用 VHDL 设计实现 JPEG 基本系统硬件编码器[D].成都:西南交通大学，2003.

[19] 尹伟.基于 FPGA 的编解码芯片的设计[D].大连：大连理工大学，2004.

[20] 汪宇飞.JPEG 高速编码芯片的设计及性能优化[D].西安：西北工业大学,2006.

[21] W.B.Pennebaker J.L.Mitchell, JPEG Still Image Data Compression Standard[M]. New York: Chapman & Hall, 1993.

[22] Tinku Acharya, Ping-Sing Tsai, JPEG2000 STANDARD FOR IMAGE COMPRESSION[M]. New Jersey: John Wiley & Sons. Inc, 2005.

[23] 胡栋.静止图像编码的基本方法与国际标准[M]. 北京：北京邮电大学出版社，2003.

[24] 傅得荣. 多媒体技术及其教育应用[M].北京：高等教育出版社，2003.

[25] 姜楠 王键. 常用多媒体文件格式与压缩标准解析[M].北京：电子工业出版社，2005.

[26] 杨宁. 静态影像压缩编码标准JPEG基本模式研究与FPGA实现[D].成都:电子科技大学，2004.

[27] [日]小野定康 铃木纯司 著.叶明 译. JPEG/MPEG2 技术[M].北京：科学出版社，2003.

[28] 吴腾奇.视频压缩技术的进展[J].中国有线电视.2001,Vol.16:40～44.

[29] Osterman. Object-oriented Analysis-synthesis Coding Based on the Source Model of Moving Flexible 3D Objects[J]. IEEE Trans, 1994, Vol.3:1128～1136.

[30] 朱秀昌.视频压缩编码的国际标准[J].南京邮电学院学报,1995, Vol.15(2):16～25.

[31] 朱磊.基于 DSP 的 JPEG 图像编解码器[D].西安：西安工程科技学院，2005.

[32] 刘涛. 网络时代的图像数据压缩研究—JPEG 和 JP2000 格式的研究[D].武汉：武汉理工大学，2003.

[33] 苏艳玲. 基于DSP的图像编码与实现[D].北京：北方工业大学，2005.

[34] Cheng-Tie Chen. Video compression:Standards and Applications[J]. Journsl Communication and Image entation, 1993, Vol.4(2):103-111.

[35] David Katz，Rick Gentile. JPEG(baselint)压缩综述[J]. 电子与电脑,2007，每月特刊(1－2)：23～27.

[36] 汲清波. 基于JPEG的图像压缩系统的设计及实现[D].哈尔滨：哈尔滨工程大学，2004.

[37] Samsung Electronics User' s Manual: S3C44BOX 32BitRISC Microprocessor[Z]. 2003.

[38] 贾智平 张瑞华. 嵌入式系统原理与接口技术[M]. 北京:清华大学出版社，2005.

[39] Sharp Microelectronic User' s Manual LM7M632 Passive Matrix LCD Module[Z]. 1998.

[40] Philips Semiconductors. PDIUSB12_data. pdf[Z]. Http://www. philips. com.

[41] 吴明晖等.基于ARM的嵌入式系统开发与应用[M].北京:人民邮电出版社,2004.

[42] 孙昊 曹玉强 杜秀芳.ARM处理器启动代码的分析与编程[J].工业控制计算机，2005,Vol.18(11)：54～55.

[43] 李岩,荣盘祥.基于S3C44BOX嵌入式μCLinux系统原理及应用[M].北京：清华大学出版社，2005.

[44] 蔡士杰.连续色调静止图像的压缩与编码(第一版)[M].南京:南京大学出版社，1995.

[45] 魏鸿铭.频域视觉景深量测与自走车路径规划之分析设计[D].台湾：朝阳科技大学资讯工程系，2003.

[46] 姚庆栋，毕厚杰，王兆华.图像编码基础[M].杭州:浙江大学出版社，1993.

[47] W.H. Chen，C. H. Smith and S. C. Fralick. A fast computational algorithm for thediscrete cosine transform[J].IEEE Transactions on Communications，1977,vol.COM－25(9):1004-1009.

[48] B.G Lee. A new algorithm to compute the discrete cosine transform[J].IEEE Transactions on Algorithms Speech and Signal Processing. 1984，vol.ASSP-32(6):1341-1344.

[49] 陈禾 毛志刚 叶以正.DCT快速算法及其VLSI实现[J].信号处理，1998，Vol.14:62～70.

[50] Fernando Rereira. MPEG-4:why，what，how，and when[J].Signal Processing:

Image Communication, 2000 No.15: 271～279.

[51] Tran T D. The BinDCT: Fast Multiplierless Approximation of the DCT[J]. IEEE Signal Processing Letters, 2000, 7(6): 141～144.

[52] 赵耀，季文铎,袁保宗. 一种基于矩阵分解的DCT快速算法[J].北方交通大学学报，1994. 18(2)：182～189.

[53] P. Duhamel, C. Guillemot. Polynomial transform computation of the 2-D DCT[J]. Acoustics, Speech, and Signal Processing, 1990, Vol.3:1515～1518.

[54] 魏本杰 刘明业 章晓莉.适用于嵌入式系统的二维DCT算法[J].计算机应用，2005, Vol.25(4):772～774.

[55] 杜相文 陈贺新 赵岩.基于查表的无乘法DCT快速算法[J].计算机工程，2004，Vol.30(20):159～160.

[56] Arai Y, Agui T, Nakajima M. A fast DCT-SQ scheme for images[J]. Transations of the IEICE, 1988, Vol.71(11): 1095～1097.

[57] 金丽 包志华 陈海进.嵌入式系统的C程序优化设计方法[J].南通大学学报，2006，Vol.5(3):61～63.

[58] 费浙平.基于ARM 的嵌入式系统程序开发要点(六)[J].单片机与嵌入式系统应用，2004，Vol.1:84～86.

[59] 杜春雷.ARM体系结构与编程[M].北京：清华大学出版社，2003.

[60] 田泽.嵌入式系统开发与应用教程[M].北京：北京航空航天大学出版社，2005.

[61] 李红蕾 凌捷 徐少强.关于图象质量评价指标PSNR的注记[J].广东工业大学学报，2004，Vol.213：74～78.

# 攻读学位期间发表的学术论文

[1] 王战盟 谷爱昱.基于S3C44B0X的启动代码U－boot的研究与移植.福建电脑，2007.2:144～145.

[2] 王战盟 谷爱昱. 基于ARM S3C44B0X启动代码的分析与研究.第18届电工理论学术年会，2006.7：239～244.

[3] 张敬春 谷爱昱 王战盟.基于盲分离的电机故障诊断.电力系统及其自动化学报，2006.8:67～70.

# 致谢

本文的研究工作是在导师谷爱昱副教授的悉心指导下完成的。三年来，不论是在课程的学习和论文的撰写，还是平常的学习生活中，导师都给了我无微不至的关怀和爱护。导师严谨的治学态度，勤奋求实的工作精神和高尚的品德给我留下了深刻的印象，产生了很大影响，使我终身受益。值此论文完成之际，谨向辛勤培育我成长的谷老师致以崇高的敬意和深切的谢意！谢谢您，谷老师！

在论文的撰写过程中，兄长魏胜利在生活上给了我莫大的帮助，在此向他表示深深的谢意！我不会忘记这段时间你和我一起走过的路。感谢我的兄弟周军华、陈锋、李燕宁、刘学刚，感谢你们在我的论文撰写过程中给予我的关心支持和帮助。

在论文的研究和撰写过程中，张敬春师兄、肖琳君同学、梁伟中同学、董巍同学给了我很大的帮助，提出了很多宝贵的意见，同时得到了师妹刘梦亭、张春娟、师弟王成群的关心，在此对他们表示感谢。还要感谢所有在攻读研究生期间给予我关心和帮助的同学和朋友，感谢他们使我的生活充满精彩。

特别感谢我的父母和亲人，感谢他们多年来对我的无私奉献和全力支持！

# 附录

表 1 JPEG 常用的标记码

| 标记码 | 标记码值 | 含义 |
|---|---|---|
| SOI : Start Of Image | 0xFFD8 | 图象开始标记，可作 JPEG 格式的判据 |
| $APP_0$ | 0xFFE0 | JFIF 应用数据块，JFIF 将文件的相关信息定义在此标记中 |
| $APP_n$ (n=1～15) | 0xFFE1～0xFFEF | reserved for application use，其他应用数据块 |
| DQT: Define Quantization Table | 0xFFDB | 量化表开始 |
| SOF: Start Of Frame | 0xFFC0 | 帧开始 |
| DHT: Define Huffman Table | 0xFFC4 | Huffman 表开始 |
| SOS: Start Of Scan | 0xFFDA | 扫描线开始 |
| EOI: End Of Image | 0xFFD9 | 图象数据结束 |

表 2 Huffman 编码分组表

| 数值范围 | DC 差份值分组（SSSS） | AC 系数分组（SSSS） |
|---|---|---|
| -1,1 | 1 | 1 |
| -3,-2,2,3 | 2 | 2 |
| -7,…,-4,4,…,7 | 3 | 3 |
| -15,…,-8,8,…,15 | 4 | 4 |
| -31,…,-16,16，…,31 | 5 | 5 |
| -63,…,-32,32,…,63 | 6 | 6 |
| -127,…,-64,64,…,127 | 7 | 7 |
| -255,…,-128,128,…,255 | 8 | 8 |
| -511,…,-256,256,…,511 | 9 | 9 |
| -1023,…,-512,512,…,1023 | 10 | 10 |
| -2047,…,-1024,1024,…,2047 | 11 | 11 |
| -4095,…,-2048,2048,…,4095 | 12 | 12 |
| -8191,…,-4096,4096,…,8191 | 13 | 13 |
| -16383,…,-8192,8192,…,16383 | 14 | 14 |
| -32767,…,-16384,13684,…,32767 | 15 | 15 |

表 3　直流 DC 差分值 Huffman 编码表（部分）

| SSSS | 编码位数 | Huffman 编码 |
|---|---|---|
| 0 | 2 | 00 |
| 1 | 3 | 010 |
| 2 | 3 | 011 |
| 3 | 3 | 100 |
| 4 | 3 | 101 |
| 5 | 3 | 110 |
| 6 | 4 | 1110 |
| 7 | 5 | 11110 |
| 8 | 6 | 111110 |
| 9 | 7 | 1111110 |
| 10 | 8 | 11111110 |
| 11 | 9 | 111111110 |

表 4　直流 AC 系数 Huffman 编码表（部分）

| Run/SSSS | 码长 | 码子 |
|---|---|---|
| 0/0(EOB) | 4 | 1010 |
| 0/1 | 2 | 00 |
| ... | ... | ... |
| 0/A | 16 | 1111111110000011 |
| 1/1 | 4 | 1100 |
| 1/2 | 5 | 11011 |
| ... | ... | ... |
| 1/A | 16 | 1111111110001000 |
| 2/1 | 5 | 11100 |
| 2/2 | 8 | 11111001 |
| ... | ... | ... |

# 基于ARM的嵌入式静态图像显示系统的研究与实现



作者： 王战盟
学位授予单位： 广东工业大学

## 相似文献(8条)

1.学位论文 王爽 可重配置系统研究及其在图像处理系统中的应用 2007

近年来，随着微电子技术、计算机技术的发展，尤其是大规模高性能的可编程器件的出现，以及软硬件设计方式和设计工具上的改进，动态可重配置技术逐渐成为国际上计算系统研究中的一个新热点。动态可重配置技术的出现使过去传统意义上硬件和软件的界限变得模糊，让硬件系统软件化。基于此技术设计的可重构系统在高速数字滤波器、图像压缩、硬件演化计算、定制计算嵌入式系统等方面，都有着广泛的应用前景。

本文将可重配置技术应用到图像处理系统领域，并使用Top-Down设计方式进行设计。与传统图像处理系统相比，本文开发的系统具有速度快，并行性高，可扩展性强等优点。本文所设计的图像处理系统由图像采集和图像压缩两部分组成，其中图像采集部分负责接收原始的图像数据；图像压缩部分负责将接收的图像数据进行压缩。

本文以高性能、低功耗的FPGA作为核心部件，利用FPGA的可编程以及控制逻辑实现方式灵活等特点，设计出图像处理系统。当给系统上电后，FPGA芯片会对视频芯片(SAA7113)进行初始化配置。随后，开始工作的视频芯片会将接收到的模拟信号转换成数字信号并发送给FPGA，接收到的数据会被保存到FPGA内部的RAM中，并通过串口将接收完全的一张图像的数据传送给上端的PC。最后，PC会对图像进行JPEG压缩处理，形成最终的图片。

论文给出了整个系统的详细设计方法，通过使用几个典型的测试案例，对系统进行了验证，结果表明该系统工作时性能稳定，采集到的图像数据精准，无失真。最后，本文对该系统进行了总结，并对其未来的发展和应用进行了展望。

2.学位论文 盖志强 基于MMS的油田无线报警系统研究 2007

在国际油价居高不下的背景下，随着计算机技术和无线通讯技术的发展，实现一种新型实用的油田报警系统成为石油行业的迫切需要。本文设计了一种基于多媒体短消息服务(MMS)的远程无线报警系统，系统以ARM作为核心处理器，通过高性能CMOS图像传感器完成对模拟视频信号的数字化采集，对图像信息进行软件压缩处理并通过多媒体短消息服务发送报警信息，监控中心通过邮件客户端软件接收报警照片。

本文构建了一种基于公共网络的图像监控系统架构，确定了图像采集、压缩以及传输的实现方案。根据油田报警系统的实际需求，在不便铺设有线通信网络以及无需传输连续报警图像的情况下，采用了多媒体短消息方式对报警图像信息进行传输，并对多媒体短消息服务的工作原理和系统的工作策略做了一些简要地介绍与分析。系统结合了MMS服务、GPRS网络和因特网各自的优点，在空间范围和大数据量传输上取得了突破。

本文对硬件的重要组成部分如视频信号的采集、报警触发单元、短距离无线传输和远距离GPRS图像传输等进行了重点讨论。采用Philips公司ARM7系列嵌入式处理器LPC2214为核心进行事务控制和软压缩算法；采用了CMOS集成视频处理芯片MT9V111完成视频信号采集；采用被动红外器件感应报警事件；为了降低成本，在油井分布密集区域采用无线射频芯片CC1020传输图像数据包，然后通过数据集中器接入GPRS，在油井孤立分布区域直接用GPRS模块传输数据。

在软件设计方面，采用了模块化思想设计终端软件的总体流程。图像软压缩采用了压缩比较高的JPEG标准，并将压缩后的图像构建成多媒体短消息格式发送到监控中心的邮箱或责任人的手机，对短距离无线通讯进行了详细分析和设计。完成了系统的调试。

论文对监控系统后台软件进行了分析设计，完成了报警图像的接收、监控软件和图像终端的短信息指令收发等功能，最终构成了完整的油田远程无线图像报警系统设计。

3.学位论文 李宏圆 Linux嵌入式单兵信息系统音、视频采集和压缩的研究与实现 2006

现代化的战争是一场信息战、数字战，谁掌握了第一手的信息谁就占据了战场的主动权。将单兵数字助理装备于士兵能极大的提高战斗力。数字化单兵侦察信息系统是由侦察、传输、终端三个分系统组成。本文围绕着单兵信息系统的特点，研究并开发出基于Linux操作系统的集数据采集、压缩、传输的音频和视频软件。

本文首先介绍系统的结构和功能，详细的介绍本系统所使用的硬件和软件的开发环境，以及各个分系统所实现的功能；然后，具体的介绍声音和图像数据的数据采集模块。数据的采集调用Linux操作系统的Api；介绍了JPEG编解码技术，并且运用JPEG标准完成对图像压缩和解压缩的软件设计。在本文中，开发了在局域网下的音频和视频数据的同步通信软件。最后，研究在GPRS公网条件下，对于音频和视频数据压缩的策略以适合战场的需要。

4.学位论文 邢霄飞 基于C6000系列DSP的JPEG图像压缩技术研究 2006

本文中讨论的JPEG图像压缩程序是应用在川大图形图像研究所车牌识别系统中的一个功能模块，其运行环境是一个典型的嵌入式系统环境，要求压缩模块有实时运算的运行效率，并且要有合适的图像压缩比和良好的图象质量，且应易于将来扩展。基于这些考虑，压缩模块采用了JPEG压缩技术。

JPEG标准是成熟的图像压缩技术，其压缩比例高，失真小，运行速度快，易于实现，适用范围广泛，作为国际标准十几年来，已得到了广泛的应用和支持。压缩程序的运行平台核心是TI(德州仪器)公司型号为TMS320DM642的DSP处理器，拥有极高的运行速度和丰富的片上外设接口，是一款面向多媒体数字应用的高性能DSP。在这个平台上我们实现了JPEG压缩程序，本文是对这一工作的总结。

首先，本文介绍了JPEG标准在实时系统中优化实现的意义，以及JPEG标准和其运行平台DM642的概况。

然后对JPEG标准展开论述，在概要介绍了JPEG压缩标准的工作模式之后，对在实际系统中应用到的顺序编码模式(Sequential Encoding Mode)按其流程做了详细论述，并讨论了JPEG采用的色彩空间和文件格式。

接着介绍了程序的运行平台TMS320DM642的CPU结构和对程序性能影响很大的指令流水线。并总结了在此平台上开发和优化程序需要注意的事项。

之后本文根据实际工作中的经验，详细论述了如何在DM642平台上实现JPEG算法以及在平台上优化程序所使用的各种方法。这些方法包括：改变代码结构、充分利用优化编译器、使用TI提供的优化库函数、使用SIMD(SingleInstmction Multiple Data，单指令多数据)指令、人工影响流水线和充分利用片上资源等等。在对程序作了大量优化工作之后，我们配合系统中的其它模块对程序进行了大量测试，并根据测试数据分析程序的运行性能和压缩质量，证实程序运行性能良好，可以满足实时运行要求。

5.学位论文 雷志同 电能计量的无线采集与监视系统的研究 2005

根据近年来电力系统自动化抄表技术、嵌入式系统技术、GSM/GPRS数据通信技术以及防盗报警技术的发展趋势，本文针对电力系统抄表难、窃电现象日益严重的现状，提出并实现了基于GSM网络通信的、低成本快速51单片机的电能计量的无线采集及监视系统。用市场上价格低廉的手机作为系统的显示和控制终端，降低了系统的开发成本，提高了电能表监视的可靠性和及时性。同时根据实际的需要，在JPEG标准的基础上实现了字符的可视嵌入以及入侵人员的红外探测。

1) 分析系统现状，设计并调试出第一台样机，同时针对在应用中的缺点，设计制作出第二台样机。

2) 在熟悉ABB通信协议的基础上，实现数据的抄读。

3) 深入分析JPEG标准以及可视数字水印技术，实现了文字的嵌入。

4) 深入分析热释电红外传感器原理，实现了人员的入侵检测。

最后，针对计量表计安装现场存在强电场和磁场干扰这一特点，从硬件设计和软件设计两个方面采取了有效的抗干扰措施，确保整个系统能够在现场环境中稳定工作。

本文所研究的抄表及监视系统不仅满足电能计量应用的需求，而且它还可以用在其它的远程无线监视系统中，因此具有广阔的应用前景，会带来较好的经济和社会效益。

6.学位论文 文凤霞 视频图像GPRS传输技术的研究与实现 2004

视频监控系统是安全防范系统的组成部分,它是一种防范能力较强的综合系统.视频监控以其直观、方便、信息内容丰富而广泛应用于许多场合.近年来,随着计算机、网络以及图像处理、传输技术的飞速发展,视频监控技术也有长足的发展.但是,无论是模拟视频监控系统还是数字视频监控系统均采用

有线的方式传输视频数据,视频数据在网络上传输,那么网络就成了视频监控系统中的首要问题.有线网络以其自身的特点不容易扩展,而无线网络恰恰弥补了有线的缺点.正是基于此考虑,才使得对视频数据的无线传输的研究更加迫切,而且具有理论意义和实践意义.首先,本文分析了国内外视频监控系统的现状和发展趋势,并对通用分组无线业务(GPRS)网络终端及其网络现状进行了详细的讨论,另外对比分析了静态图像专家组(JPEG)标准、H.261以及动态图像专家组(MPEG)标准的优缺点,并针对本系统的要求提出使用JPEG标准的理由.其次,对系统的总体功能进行了规划,并将系统划分成两个功能单元,进行讨论.这两个功能单元分别是图像压缩模块和无线传输单元.每个功能单元按其功能的需求选用合适的芯片构造出硬件电路,并使用C语言实现其软件功能.系统的总体协调功能由高级精简指令集(ARM)处理器来完成,构造出了高效、便捷、灵活的嵌入式系统.最后,本文讨论了服务器端功能的实现,在服务器端主要是接收系统传送的数据,并将压缩过的图像还原出来.

7.会议论文 蒙智明.屈百达 基于TMS320DM275嵌入式系统的JPEG解码实现 2007

研究了一个高性能的JPEG解压缩处理系统,该系统以美国TI公司的双核处理器TMS320DM275(ARM7和DSP:TMS320C5409)为处理中心,提供了一种低成本的JPEG解压缩方案.该系统从SDRAM中获取压缩数据,然后经过熵解码、逆量化、IDCT,在TMS320DM275上实现了JPEG的快速解码.实验表明该方法具有灵活高效的特点,可以应用到实际中.

8.学位论文 崔巍 基于DaVinci技术的图像压缩系统的研究 2008

CCD图像包含的信息量大，内容更加丰富，对军事、航空航天、天文等领域的作用更是不言而喻，但是海量的数据传输负担成了高分辨率CCD航空相机广泛应用的瓶颈，因而研究高效的图像压缩系统成为必然。

JPEG标准是成熟的图像压缩技术，其压缩比例高，失真小，运行速度快，易于实现，适用范围广泛，作为国际标准十几年来，已得到了广泛的应用和支持。压缩程序的运行平台核心是TI(德州仪器)公司型号为TMS320DM6446的DSP处理器。DaVinci技术的硬件产品TMS320DM6446在一个芯片封装内集成了ARM嵌入式处理器内核与C64x+数字信号处理器内核，提高了系统集成度、降低了系统板级成本，双处理器的协同运作效率也有很大提高。

本文在深入研究TI公司最新的DaVinci技术后，针对双核处理器的特点，提出了图像压缩系统的设计。主要研究内容涉及DaVinci技术的硬件系统架构、软件系统架构及Linux下的应用程序开发等。

硬件系统主要工作涉及芯片选型、DDR2动态存储器接口设计、Flash存储器接口设计、ATA硬盘接口设计、以太网接口设计、USB接口设计、UART接口设计及系统电源设计，本文还对系统的各个组成部分的功能及特点作了详细的介绍。系统充分应用TMS320DM6446的处理优势，具有处理速度快、板级成本较低等特点。

嵌入式软件系统的开发首先建立了宿主机端针对DM6446的ARM端的交叉编译环境。接着根据DM6446的引导启动机制，设计了适合的Bootloader代码和内核。最后，在此操作系统平台上实现JPEG图像压缩算法。

关键词：图像压缩；JPEG；达芬奇技术；TMS320DM6446；嵌入式系统