基于嵌入式Linux的实时多媒体信息传输系统的研究与设计

摘 要

随着多媒体信息压缩技术和宽带网络技术的快速发展，通过网络传输高质量的多媒体信息已经成为信息技术领域研究的热点之一。同时嵌入式 Linux 系统的研究和应用也正在成为信息技术研究重点之一。本文设计的多媒体传输系统主要可以分成三部分：一是基于MPEG-2 标准的多媒体信息编解码硬件系统、二是基于嵌入式 Linux 的多媒体信息发送系统、三是基于嵌入式 Linux 或计算机的多媒体信息接收系统。文中多媒体传输系统的框架和组成该系统的各个部分进行了详细的介绍。在整个系统的设计过程中应用了 MPEG-2 多媒体编码技术、TCP/IP 传输技术、RTP 协议、嵌入式 Linux 系统应用技术。文中对这些技术的主要部分都进行了较为详细地介绍。本文设计的基于嵌入式Linux系统的实时多媒体传输系统不仅是对多媒体压缩和网络传输技术的应用和研究，也是对嵌入式 Linux 系统应用技术的一次有意义的尝试，为进一步做研究打下一个良好的基础。

关键词：多媒体，传输系统，MPEG-2，TCP/IP，RTP，嵌入式 Linux

DESIGN AND INVESTIGATION OF A REAL-TIME MULTIMEDIA TRANSMISSION SYSTEM BASED ON EMBEDDED LINUX SYSTEM

ABSTRACT

With the rapidly development of multimedia signal compression and broadband networks technologies, methods of transmitting high-quality multimedia information via networks have become one hottest point of IT technologies. Researches and developments of Embedded Linux system technology are becoming another important IT technology. In this thesis, we present a real-time multimedia transmission system based on embedded Linux technology. There are three parts in the system, a pair of MPEG-2 standard codec, a transmitting system and a receiving system. We describe the whole system skeleton and principle and the design of each parts in detail. During the system design, we take advantages of these technologies, such as MPEG-2, TCP/IP, RTP and embedded Linux technology. These technologies are discussed in different chapters. To design and research a real-time multimedia transmission system based on embedded Linux not only combines multimedia signal compression technology and networks technology, but also explores the application of the embedded Linux technology. The content of this thesis might be helpful to develop the multimedia transmission technology.

KEYWORDS: Multimedia, transmission system, MPEG-2, TCP/IP, RTP, Embedded Linux

第一章 综述

§1.0 概述

本章主要分三部分，第一部分介绍多媒体传输系统的基本概念和组成，描述多媒体数据的特点、压缩编码技术以及对传输网络的要求。第二部分介绍当前国内外的多媒体处理和传输技术。最后介绍本文提出的一种基于嵌入式 Linux 系统的实时多媒体传输系统的设计。在本章的结尾介绍本论文其余各个章节的结构和安排。

§1.1 多媒体传输系统

随着电子、计算机、声像和网络通信技术的发展，多媒体技术前进一大步，特别是由于数字化技术在计算机领域广泛而成功的应用，极大地方便多媒体信息的存储、处理和网络传输，使多媒体系统广泛应用在视频会议、视频点播、远程计算机辅助教学、协同 CAD 等各个领域。[1~3]

§1.1.1 多媒体概念[1~3]

多媒体（multimedia）主要指文字、图形、图像、声音等人的器官能直接感受和理解的多种信息类型，ITU 对它的描述是：是用计算机交互式综合技术和数字通信网技术处理多种表示媒体——文字、图形、图像、声音，使多种信息建立逻辑连接，集成为一个交互系统。从概念上说，多媒体中的“媒体”应该是指一种表达某种信息内容的形式。多媒体信息这个概念来表示包含文本信息、图形信息、图像信息和声音信息等不同信息类型的一种综合信息类型。在通信领域内，无论在信息的采集、存储、处理、传递，还是在信息的显示和控制，广义的多媒体就是指各种信息类型的综合。通常就把包含两种以上信息类型的信息称为多媒体信息。

多媒体数据的特点：

- 数据类型复杂

  多媒体数据由多种不同类型的数据组成，通常包括正文、图形、图像、

声音、视频、动画等不同的数据类型。而且同一数据也可以有不同的表示方式，如编码方式、内部数据结构等。

- 数据信息量巨大

  多媒体数据的信息量是非常巨大的，仅以声音数据为例说明。声音数据进行采样量化时，为满足人耳的感知最高频率为 20KHz 这一特点，通常采样频率为 44.1KHz，为达到较大的动态范围和信噪比，每一样本 16 比特，这样对双声道立体声而言，数据量为 176KB/s 或 10MB/min。

- 高实时性

  多媒体数据中的声音和视频数据都和时间有关的信息，很多场合都要求实时处理，如声音和视频图像信息的实时压缩、传输、解压缩等。

- 数据的分布性

  由于多媒体数据原始采集往往与处理不在同一个地方，使得分布式多媒体数据库技术和多媒体通信技术成为多媒体技术中的关键，

- 数据的交互性

  多媒体技术的特点之一就是交互性，这种交互是一种实时操作，要求整个系统的软硬件都能够实时响应。

§ 1.1.2 多媒体传输方式[3,9]

多媒体信息的传输通常可以分成两类：一类是下载回放方式，下载回放是一个异步过程，并把多媒体信息和其它数据统一看待，差别仅在于数据内容。另一类是同步显示的实时传输方式，也就是流媒体传输方式，所谓流媒体是指视频、声音和数据从源端同时向目的地传输，在目的地连续实时接收这些媒体数据流。实时传输的多媒体信息都是使用流媒体传输，对时间依赖性很高。

流数据从服务器端应用传输后可由客户端应用接收并显示或回放，一般是客户端应用接收到足够的数据并将之存储在缓冲区后便立即将视频显示出来，或将音频回放出来。

流媒体的一个重要特征是对时间的敏感性，这正是实时性要求高的应用所必需的。流媒体的实现主要取决于网络带宽和压缩算法的提高。随着网络协议的改善、网络基础设施和压缩技术的发展，流媒体的实现已经变得越来越容易。

本文中主要讨论的实时多媒体传输系统本质就是一个实时流媒体直播系统。

§ 1.1.3 多媒体传输特性[3,11,24]

多媒体传输对网络的要求可以定性的分成两类：传输误码率和传输延时。而对于不同的应用中这两个要求的重要性是不同的，比如：

对于接收端直接把多媒体内容展现给人的系统中，减小传输延时比减小传输误码率更重要；在接收端是媒体纪录系统的应用却不能容忍过高传输误码率，而传输延时要求相对较为宽松。

通常传输连续多媒体信息对通信网络的要求主要体现在网络带宽、时延、时延抖动等。在分组交换网络中如传输实时视频、音频、动画等连续媒体对网络提出不同于传输普通数据的要求：

- 多媒体数据的带宽要求较高，通常传输 MPEG-1 会要求 1.5Mbps，而具有较高视觉质量的 MEPG-2 要求 6 Mbps。
- 时延要求，典型地在传输实时 MPEG-1 压缩视频要求时延不超过 500ms。
- 严格的时间有效性限制，这类信息在某一段时间之后未能成功传输就会失去价值。
- 能够忍受一定限度的数丢失，根据不同的编码技术，和人的察觉能力，可以允许在传输中丢失部分数据而不影响应用。
- 具有一定的周期性，比如在视频中，每秒都会出现固定数量的帧结构，通过分组网络传输会丢失这种周期性。

在分组交换网络中传输时通常还会造成分组传输顺序被打乱，数据分组丢失等。在设计在分组交换网络中传输连续实时多媒体数据时，必须能对网络传输而引入的问题采取必要的机制，这些机制应该能够改善网络传输质量，使多媒体传输应用的服务质量达到可以接受的程度。在普通非实时数据传输中采用较为熟悉的 TCP 协议，确保数据传输。而在实时多媒体传送中较少采用可靠传输的协议 TCP，因为 TCP 的重传机制和拥塞控制机制在保证数据的完整性同时却破坏数据的实时性。在以后的章节中将介绍基于 RTP/UDP 设计的传输机制来传输实时多媒体数据。

在分组交换网络中传输错误几乎是不可消除的，传输错误分成两类：数据比特错误和数据分组丢失。而在视频和音频数据流中错误，根据不同的编码技术会产生不同的影响。通常在 MPEG 编码中都采用差分编码和运动估计，不仅采用

帧内编码还采用帧间编码，这样在某一帧中的数据错误会造成连续的解码失败。因此要求能够采取一定的错误恢复机制来最大限度的满足实际应用。有三类方法可供选择：自动重传（ARQ）、前向纠错（FEC）和错误隐藏。在实时传输视音频数据时，由于数据本身的容错特性和时间紧迫性导致倾向于采用后两类方法。

§ 1.1.4 多媒体传输技术[2,3,9,16~18]

某些应用中要求网络提供多点发送的能力，如在视频会议中，通常只有一个发言者，其他参与者只是接收数据，采用多播技术可以提高网络资源利用率。多媒体的传输技术主要有三种：点对点(Unicast)、多址广播(Multicast)和广播(Broadcast)。多址广播又称为组播。点对点的特点是流媒体的源和目的地是一一对应的，即流媒体从一个源(服务器端的应用)发送出去后只能到达一个目的地(客户端应用)。组播是一种基于“组”的广播，其源和目的地是一对多的关系，但这种一对多的关系只能在同一个组内建立，流媒体从一个源(服务器端的应用)发送出去后，任何一个已经加入与源同一个组号的目的地(客户端应用)可以接收到，该组以外的其他目的地(客户端应用)接收不到。广播的源和目的地也是一对多的关系，但这种一对多的关系并不局限于组，流媒体从一个源(服务器端的应用)发送出去后，同一网段上的所有目的地(客户端应用)均可以接收到，广播可以看作组播的一个特例。

广播和组播对于流媒体传输来说是很有意义的，因为流媒体的数据量往往都很庞大，需要占用很大的网络带宽。如果采用点对点方式，多个目的地就得传输多份流媒体，所以所需的网络带宽与目的地的数目成正比，如果采用广播或组播方式，那么流媒体在源端只需传输一份，组内或同一网段上的所有客户端应用均可以接收到，这就大大降低网络带宽的占用。

§ 1.1.5 多媒体传输网络[2~4]

本小节介绍常用的多媒体传输网络技术。局域网通常通常是在小范围内将各个独立系统互联实现资源共享的通信网络。目前局域网通常都有 100Mbps 的传输容量，最新的局域网技术中传输率可达 1Gbps。

在 IEEE 802 系列标准中描述两种 LAN 技术——以太网和令牌环网。局域网具有中高数据传输速率、低时延、低误码率的特征，能够满足传输多媒体信息的需要。另外一种 LAN 技术是光纤分布式数据接口(Fiber Distributed Data

Interface)，其实是高速令牌环网的变体。

以太网（**IEEE 802.3**）是目前最流行的局域网技术，价格较低廉，支持多种应用，能够提供较高的通信量和较小的时延。以太网传输利用总线和星形拓扑结构的优点。以太网传输数据的标准速度为 10Mbps，目前最常用的快速以太网可以以 100 Mbps 的速度传输。千兆以太网速度可达 1Gbps。以太网使用带有冲突检测的载波侦听多路存取(Carrier Sense Multiple Access with Collision Detection)控制方法。

对于多数应用而言使用 100Mbps 的网络传输，带宽是足够的。而且这些网络的传输延时一般都低于 1ms。如在 MPEG-2 种码率达到 4Mbps 时的视频质量已经非常好，完全能够在 100M 以太网中传输。100M 快速以太网已经完全可以满足多媒体信息的带宽要求。

令牌环网（**IEEE 802.5**）是 IBM 公司于 70 年代发展的，现今仍然是一种主要的 LAN 技术。在老式的令牌环网中，数据传输速度为 4Mbps 或 16Mbps，新型的快速令牌环网速度可达 100Mbps。令牌环网的传输方法从物理上采用星形拓扑结构，在逻辑上采用的是环形拓扑结构。令牌环网的优点是非常可靠，因此经常用于关键任务中。令牌环网优于以太网在于广播风暴(broadcast storm)和工作站之间的干扰非常少。而在以太网中，当许多计算机或设备试图同时发送帧时，或者当计算机或设备坚持重复地发送信号导致出现广播风暴。

FDDI 标准由 ANSI X3T9.5 标准委员会制订，为繁忙网络上的高容量输入输出提供一种访问方法。FDDI 网络与令牌环网十分相似，按照实时和非实时可以分成两种工作模式：同步 FDDI 和异步 FDDI。同步 FDDI 能提供确定的延时保证，用于要求连续进行的对时间敏感的传输(如音频、视频和多媒体通信)。异步 FDDI 用于不要求连续脉冲串的普通的数据传输。FDDI 环网访问速率为 100M，可达的端到端位率高度依赖于站点性能和高层协议，典型的速率为 50~60Mbps。

ATM（异步传输模式）实质上是一种使用异步时分复用技术的快速分组交换方式，既能提供固定的短时延电路交换特性，又具有动态分配资源的分组交换特性。ATM 网络能够根据上层应用所需的服务质量提供承载服务，适合于传输可变比特率（VBR）的数据。ATM 标准定义的基本访问速率为 155Mbps，最大速率达到 622Mbps。ATM 具有动态分配带宽、接入网络灵活、多个视频信息源共

享信道、可处理突发信息等特点，因此 ATM 网络是一种较为理想的多媒体传输网络。

§1.1.6 多媒体信息传输系统的组成[2,3]

多媒体传输系统由信息获取，模拟数字转换，压缩编码，传输网络，解码显示等部分组成。模拟视频和声音信号经过捕获设备转换成数字形式后，其数据量是非常巨大，如果没有采用压缩技术，那么要实现数字视频和声音的网络传输是不可想象的。目前已发展和正在发展的数字视频和音频压缩技术有很多种，不同的压缩技术有不同的侧重点，适应不同的应用。常用的已经标准化的压缩技术有 MPEG-1、MPEG-2、H.261/H.263、 MPEG-4 等。MPEG-1、MPEG-2 适用于高带宽的能够提供高质量低延迟的视频和音频应用，而 H.261、H.263 以及正在发展 MPEG-4 则适用于低带宽的对图象质量的延时要求不高的应用。

另一方面，数字视频和声音传输对时间的敏感性很强，实时性要求很高，必须采用特别的网络传输协议来满足要求。所以数字视频和声音传输的做法是：在源端先将数字视频和声音信息进行压缩，然后经由诸如 ATM 这样的有服务质量(即 QoS)保证的网络传输到目的地，再在目的地将之进行解压后显示或回放出来。如果需要在诸如 IP 网络这样的没有 QoS 保证的网络上传输，则就必须采用实时传输协议(RTP)进行传输。

图 1-1 多媒体传输系统的组成

通常局域网多媒体系统采用客户机和服务器的模型，系统可以将信息通过网络发送给多个用户。多媒体服务器提供多媒体服务给其它客户机系统，服务器具有多媒体的存储或获取机制，如静止图像服务器、运动视频服务器、扫描服务器、广播电视发送服务器和传真服务器等。点播服务也是网络多媒体传输系统中的重要一类，如 VOD、MOD 系统等。多媒体客户机通过网络接收信息，并能够向服务器系统发出服务请求信息和反馈信息，客户机可以是桌面计算机，视频发送系统等。

§1.2 国内外发展及现状[6~9]

全世界范围内的研究人员对网络多媒体传输技术进行大量的研究，开发许成功的网络流式媒体商用系统，并在远程教育、电子商务以及网上媒体点播系统众多领域中得到成功的应用，最著名如 Real system、Windows Media、QuickTime 等。下面就对这几个系统进行简要介绍。

1. Real System 是由 RealNetworks 公司发布的，被认为是在窄带网上最优秀的流式媒体传输系统，其允许的带宽限制从 28.8Kbps 的拨号上网到 10M 的局域网。Real System 支持 Winows NT/2000、Solaris、Linux、Free BSD 等多种平台，整个系统共分三部分：媒体制作工具（Real Producer）、服务器端软件（Real Server）、客户端播放器（Real Player）。RealMedia 包括 RealAudio、RealVideo 和 RealFlash 三类文件，采用的是基于小波变换的 REAL 专用算法。
2. 微软公司的 Windows Media 是一套能够在 Internet 或企业内部网这样的计算机网络上传送数字音频和视频媒体的组件。Windows Media 的主要工具有：Windows Media 编码器、Windows Media 服务器、Windows Media 播放器、Windows Media 权限管理器、Windows Media 软件开发工具包。Windows Media 以 ASF 为核心采用 MPEG-4 压缩编码技术。

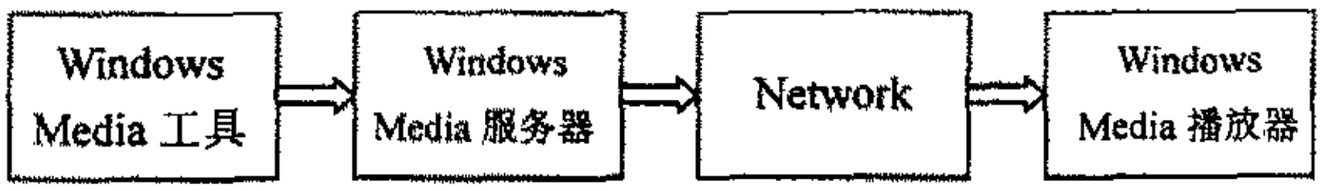

图 1-2 一个使用 Windows Media 组件的基本方案

3. Apple 公司的 QuickTime。主要组件有：QuickTime Streaming Server 服务器、QuickTime Broadcaster 编码器、Quick Time Player 播放器。QuickTime 采用包括 H.263 在内的多种编码，主要以 Sorenson V ideo 为主，目前已经采用 MPEG-4 压缩技术。

以上三种系统采用的编码技术各不相同，但传输技术都采用 RTP、RTCP 等协议，代表网络流媒体技术的发展主流。这些系统结构复杂功能十分强大即能够播放文件媒体，也能够实时直播多媒体信息，而且都有一套完善的媒体制作、管理和开发工具。

§1.3 基于嵌入式 Linux 的实时多媒体信息传输系统[26-29]

前面介绍的几种现有系统技术完善功能强大，但它们共同的缺点是服务器端需要依赖于笨重的计算机系统，并通过软件对视频信号进行压缩处理，把媒体格式转换成适于传输的格式。由于过多的依赖软件处理，实时性难以保证，而且对每一个采集点都要配有专门的计算机，成本较高，因此并不十分适用于地理分散的多点信息采集应用。

本文考虑设计一个基于嵌入式 Linux 的实时多媒体传输系统，多媒体信息的采集和压缩都使用硬件实现，整个系统针对地理分散的多点远程视频监控类应用而设计，并符合 MPEG-2标准具有画面清晰度高，编码时延低的特征，更适合实时传输应用。发送端（服务器）基于嵌入式 Linux 系统设计，来实现通过局域网传送多媒体信息，特点是可靠性高，时延小，运行成本低。接收端即可以使用嵌入式系统接收，也可以使用计算机接收，并分别对应硬件或软件解码处理。和通用计算机系统不同，嵌入式系统针对具体应用设计的专用系统，直接面向产品、面向应用，系统硬件和软件都高效地设计，是在完成目标功能的基础上最小化的系统，系统运行具有稳定高效的特点。而且嵌入式 Linux 具有开放源码，支持多种体系结构和大量硬件设备的特征，使得基于嵌入式开发应用系统的优势更加明显。目前嵌入式 Linux 系统的开发和应用受到国内外广泛关注，比较流行的嵌入式 Linux 系统有，Monta Vista Linux，uCLinux，红旗嵌入式 Linux 等。在 PDA、手机、机顶盒、汽车、微波炉、电梯、安全系统、自动售货机、医疗仪器、立体音响、自动取款机等设备中都有嵌入式系统的身影。

嵌入式 Linux 的实时多媒体信息传输系统是一个基于局域网的主要传输实时视频和音频信息传输系统。主要设计目标是实现把远端多媒体采集系统采集实时的视音频信息经过压缩编码，并利用嵌入式 Linux 系统模块实现 IP 协议和多点发送的方式通过以太网传输，使多个接收端系统能从网络中接收实时视音频并重新展现给用户，并使发送端脱离通用计算机。系统示意图 1-3 如下：

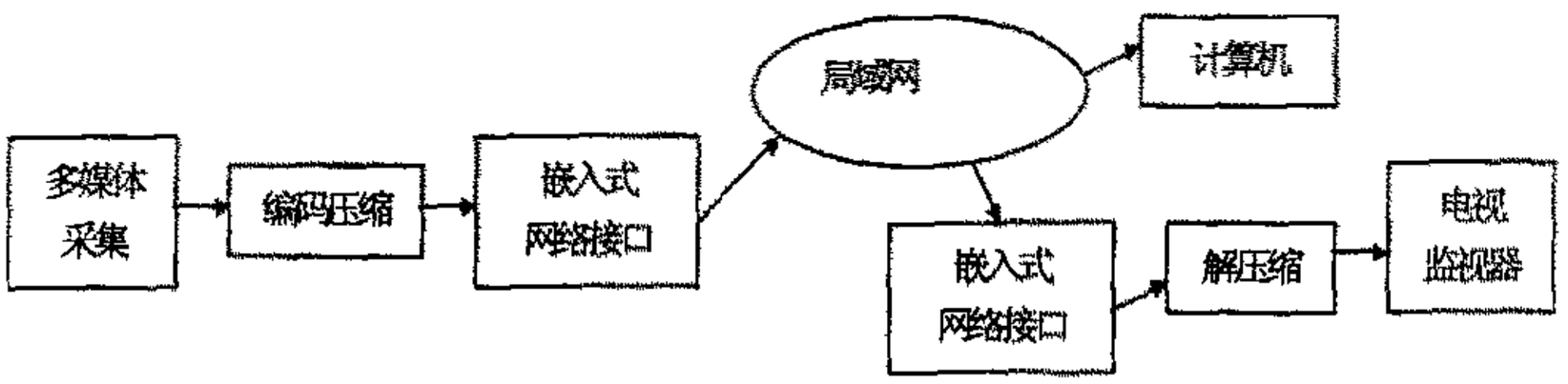

图 1-3 系统示意图

其中多媒体信息采集系统用摄像机和麦克风采集实时视频和音频信息，经过模数（A/D）转换以后采用 MPEG-2 编码器硬件压缩输出 PS 流或 TS 流。然后通过一个基于嵌入式系统设计的以太网接口，把实时的压缩媒体数据流打包形成以太网数据包，再按照 UDP 报文的方式通过局域网传输。在接收端有两种实现方案，一是使用带有网络适配器的计算机接收媒体数据报文，利用软件解开网络数据报文、并对压缩多媒体数据进行解码显示；第二种使用嵌入式以太网接口接收网络数据报文，并把其中的视音频数据送给的硬件解码器解码，解码器输出的数字视音频信号经过数模转换后送给电视或监视器显示。

系统中的主要组成部分和组要功能：

媒体信息采集设备：实现对实时多媒体信息进行采集，输出模拟的视音频信号，主要包括实时视频和音频信息。这类设备包括摄像机，麦克风等输入设备。

编码压缩模块：首先把模拟视音频信号进行数字化处理，然后采用 MPEG-2 标准进行压缩，输出是适合于传输应用的 TS 数据。编码压缩模块是利用富士通公司的 MB86390 编码芯片设计的硬件编码器可以完全独立于计算机工作，能够实现对非压缩数字视音频的按照 MPEG-2 MP@ML 标准压缩，生成高质量的压缩视音频。该 MPEG-2 编码器能够输出多种格式的压缩数据，编码速率 2~20Mbps 可调，输出编码速率为 4Mbps 时的 D1 格式的压缩视频已经具有良好的视觉效果，画面中的快速运动物体也较为流畅。编码模块使用 8 位并行的方式输出 TS 数据给嵌入式以太网接口实现网络打包。

嵌入式网络接口：基于嵌入式 Linux 系统设计的网络接口。发送端网络接口模块通过 8 位并行口从编码器读取视频数据，并对数据以 RTP/UDP 协议进行分组打包，再通过多播或广播的方式发送到局域网中；在接收端是先从网络中接收数据，解包，然后输出数据到解码器。网络接口还要实现对网络报文发送输率进行控制，传输状况进行统计等。网络接口模块的设计可以分成硬件设计和软件设计两大部分，硬件上网络接口是基于 Motorola 公司的 MCF5307 处理器设计的，能够运行嵌入式 Linux 操作系统，支持 10M 以太网接口和 8 位的并行数据传输接口，以及一些必要的控制信号；软件基于一个嵌入式 Linux 系统——uCLinux 进行设计，主要可以分成两大部分：从编码器的读取速据和把数据按照一定的速

率和传输机制发送到网络中。

在接收端有两种接收方案，基于硬件解码器的接收方案和基于计算机的接收方案。

基于硬件解压模块的接收端：首先该方案中也需要一个基于嵌入式系统的网络接口对数据接收，并对数据进行缓冲和错误恢复处理然后输出到解码器。在解压缩模块种对压缩的视音频数据进行解码，生成数字非压缩视音频，并对其进行数字模拟转换，最终输出 PAL 制式的模拟电视信号，能够用电视机和监视器，进行显示。解码器也是按 MPEG-2标准设计的，硬件实现时编码核心采用富士通的 MB87L2250，可对任何标准 MPEG-2 编码器输出的 MPEG-2 TS 流解码。

计算机接收端：该部分是独立利用计算机来实现整个接收端的功能，包括网络数据报文接收、数据包分解、数据缓冲、数据恢复、压缩视音频解码、终端显示几个部分。计算机直接利用现有的网卡功能从以太网中接收带有压缩多媒体数据的报文，进行缓存解包从中取得压缩多媒体数据，然后把压缩数据使用解码软件解码在计算机显示器中显示。

在本设计中注重设计一种能够使两种接收方案都能接收的发送端和它们之间的一种传输机制的设计。这两种类型接收端互相独立，而且基于两种不同的系统设计，如硬件方案中传输机制的实现基于 uCLinux 操作系统/Motorola Coldfire 硬件平台环境实现，而在基于多媒体计算机的接收端是在 Windows（或 Linux）操作系统/Intel 硬件平台环境下实现，在软硬件上都有很大的不同，而两者之间使用以太网连接，只要支持相同的传输协议就实现。压缩视频在不可靠的网络中进行实时传输的总会在一定程度上对视频质量产生影响，如何在保证实时传输的同时尽可能的改善视频质量是目前广泛研究的一个课题。

§1.4 论文安排

在本章中已经对本篇论文的主要工作做一些介绍，本设计牵涉的内容非常丰富，不但和嵌入式 Linux 系统、网络协议、MPEG-2 等相关，还与很多低层的硬件有密切的联系。因此论文中安排适当的篇幅对嵌入式系统和一些主要的标准协议进行介绍。

第一章中已经对多媒体传输系统的一些相关知识和国内外的相关技术的发

展状况和存在的一些问题分别做介绍。然后在本章中提出本文设计的一种基于嵌入式 Linux 的实时视频传输系统，对整个系统的各个主要组成部分进行介绍。

第二章首先对本设计相关的视音频编码国际标准和进行介绍；然后在这一章节中，对 TCP/IP 协议以及 RTP 协议进行分析说明如何利用这些协议来传输实时 MPEG-2 编码数据流，并对几种不同的传输方式进行比较，分析那种协议更适合传输实时的多媒体数据流（主要是 MPEG-2 编码数据）。

第三章描述一种符合 MPEG-2 标准的编码器的设计和硬件实现，并对相关的软件设计和调试中所遇到的一些具体问题与以探讨。

第四章首先对 Linux 系统进行简单的介绍，然后对嵌入式 Linux 系统结构和特点进行介绍，以及这种环境下的开发工具的使用，如 gcc、gdb 等。在这一章中重点讨论本设计中使用的嵌入式 Linux 系统（uCLinux）和开发应用程序设备驱动程序的一些概念。

第五章对多媒体信息传输系统的设计的系统框架和系统中的各个模块设计进行介绍，包括 MPEG-2 编码器和嵌入式系统板的接口模块，MPEG-2 编码器驱动程序，发送端和接收端应用软件以及嵌入式开发系统的建立等。

第六章总结本文设计的工作成果和系统中存在的不足以及如何开展进一步的工作。

在文章最后列出本文的参考文献，并对完成本文过程中给予指导和帮助的老师和同学等人员表达由衷的谢意。

§1.5 小结

本章节是论文的第一章，在这里简要介绍多媒体传输系统有关基本概念、它的组成以及一些相关的技术，如编码技术、网络技术等。然后列举当前国内外的已经采用的和研究的多媒体处理和传输技术。最后对该篇论文的主要工作和论文结构的安排进行介绍。

第二章 多媒体编码传输技术

§2.0 概述

本章主要介绍多媒体信息编码技术和网络传输协议，分析如何在分组网络中传输实时多媒体数据。§2.1 节论述 MPEG-2 压缩编码标准的系统、视频和音频三部分，并介绍 MPEG-2 视频部分的关键技术。在 §2.2 中对 TCP/IP 协议进行分析，指出哪种传输协议更适合承载实时数据。§2.3 节中介绍实时传输协议，并对如何再 RTP 中承载 MPEG-2 进行讨论。

§2.1 MPEG-2 多媒体压缩标准[9,12~15]

MPEG 是活动图像专家组(Moving Picture Exports Group)的缩写，于 1988 年成立。到目前为止 MPEG 已颁布三个活动图像及声音编码的正式国际标准，分别称为 MPEG-1、MPEG-2 和 MPEG-4。MPEG-2 标准制定于 1994 年，设计目标是高级工业标准的图象质量以及更高的传输率。MPEG-2 所能提供的传输率在 2-100Mbits/sec 间，支持 PAL、NTSC、SECAM 等多种制式。

MPEG-2 标准具有以下几个突出特点：所支持的图像分辨率高，包括符合 CCIR601 格式的标准分辨率的数字电视、更高分辨率的 HDTV 和 CD 级的音质。支持包括高速体育运动在内的活动图像。所支持的应用最为广泛，既包括存储媒体中的 DVD，广播电视中的数字广播电视和 HDTV，还可应用于交互式的点播视频(VOD)和准点播视频(NVOD)，此外，还能够适配于 ATM 宽带通信网。

MPEG-2 能够提供广播级的视频质量。MPEG-2 的音频编码可提供左右中及两个环绕声道，以及一个加重低音声道，和多达 7个伴音声道。由于 MPEG-2 在设计时的巧妙处理，使得大多数厂商的 MPEG-2 解码芯片也能支持 MPEG-1 格式的数据，即能够解压 VCD 格式的压缩视频。MPEG-2 特别适用于广播级的数字电视的编码和传送，被认定为 SDTV 和 HDTV 的编码标准。MPEG-2 还可用于为广播，有线电视网，电缆网络以及卫星直播(Direct Broadcast Satellite)提供广播级的数字视频。MPEG-2 还专门规定多路节目的复用分接方式。MPEG-2 提

供一个较广的范围改变压缩比，以适应不同画面质量，存储容量和带宽的要求。使得 MPEG-2 标准的编辑码器应用场合十分广泛。

MPEG-2 标准目前分为 9 个部分，统称为 ISO/IEC13818 国际标准。各部分的内容描述如下：

第一部分-ISO/IEC13818-1，System：系统，描述多个视频，音频和数据基本码流合成传输码流和节目码流的方式。

第二部分-ISO/IEC13818-2，Video：视频，描述视频编码方法。

第三部分-ISO/IEC13818-3，Audio：音频，描述与 MPEG-1 音频标准反向兼容的音频编码方法。

第四部分-ISO/IEC13818-4，Compliance：符合测试，描述测试一个编码码流是否符合 MPEG-2 码流的方法。

第五部分-ISO/IEC13818-5，Software：软件，描述 MPEG-2 标准的第一、二、三部分的软件实现方法。

第六部分-ISO/IEC13818-6，DSM-CC：数字存储媒体-命令与控制，描述交互式多媒体网络中服务器与用户间的会话信令集。

第七部分规定不与 MPEG-1 音频反向兼容的多通道音频编码；

第八部 10 比特视频；

第九部分规定传送码流的实时接口。

1990 年成立的 ATM 视频编码专家组与 MPEG 在 ISO/IEC13818 标准的第一和第二两个部分进行合作，因此上述两个部分也成为 ITU-T 的标准，分别为：ITU-TRec.H.220 系统和 ITU-TRec.H.262 视频。

§2.1.1 系统部分[12,15]

系统编码有两种方法：传输流（TS）和程序流（PS）；都是面向分组的多路复用流。

PES 分组是由按照 ITU-T REC.H.262|ISO/IEC 12818-2 和 ISO/IEC 13818-3 标准对视频和音频信号进行压缩编码后的原始流组合成。

程序流是一个或多个具有相同时间基点的数据流 PES 分组合成的单个流，用来传送和保存一道程序的编码数据或其它数据。针对错误相对较少的环境设计，如多媒体软件处理。程序流分组长度是可变的且相对较长。

程序流由系统层和压缩层构成。程序流解码器的输入流有一个包含压缩层的系统层。音频、视频解码器的输入流只有一个压缩层。

传输流是将有多个独立时间基点的多道程序合成一个单独的数据流，其中属于同一道程序的各个原始数据流的 PES 分组具有相同的时间基点。传输流针对易发生错误的环境设计，如高噪声或易损的媒体中存储和传送。传输流分组长度固定为 188 字节。

传输流支持的操作：

- 从传输流中的一道程序中恢复被编码数据，解码并显示。
- 从传输流中的一道程序抽取分组生成新传输流。
- 从多个传输流中提取一道或多道程序的分组生成新的传输流。
- 从传输流中提取一道程序生成程序流。
- 程序流和传输流相互转换以适于传输处理。

传输流由系统层和压缩层构成。传输流解码器的输入流有一个包含压缩层的系统层，视频和音频解码器的输入流只有压缩层。

传输流的系统层分为两个子层：传输流分组层，PES 分组层。

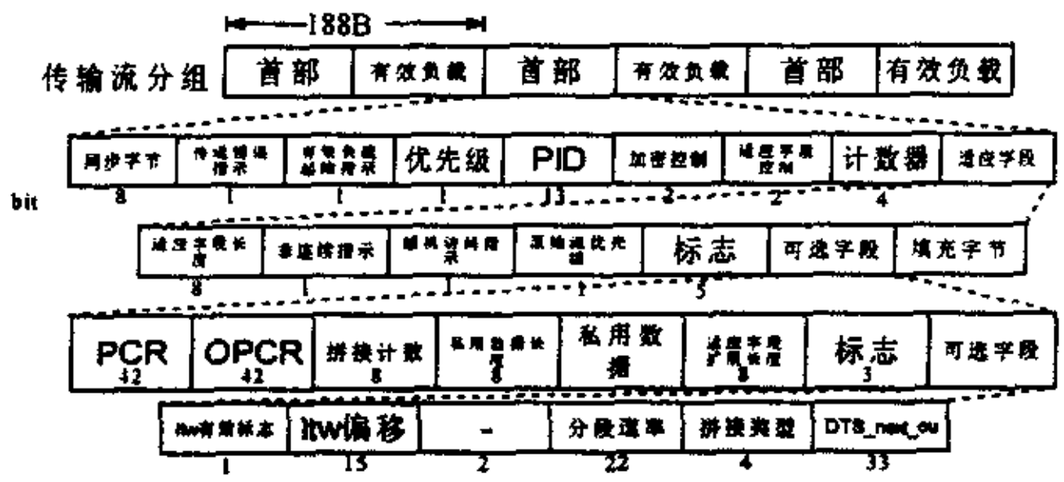

图 2-1 传输流分组语法图

分组的原始数据流 PES

PES 分组的长度比传输流分组大得多。一个原始流的具有相同流号的连续 PES 分组可以用来构造 PES 流。PES 流数据保持原始流中的顺序。PES 流中不包含系统信息：如组头、系统头、程序流映射、程序流目录、程序映射表及传输流分组语法的信息。

§ 2.1.2 视频部分[9,12~15]

MPEG-2 视频编码标准是一个分等级的系列，按编码图像的分辨率分成 4 个“级(Levels)”；按所使用的编码工具的集合分成 5 个“类(Profiles)”。“级”与“类”

的若干组合构成 MPEG-2 视频编码标准在某种特定应用下的子集。如下表：

表 2-1 MPEG-2 类等级组合

等级\类	简单类	主类	SNR 可分级类	空间可分级类	高类
低级（352×288）		MP/LL	SNR/LL		
主级（720×576）	SP/ML	MP/ML	SNR/ML		HP/ML
高 1440 级（1440×1152）		MP/H1440	SSP@H1440		HP@H1440
高级（1920×1152）		MP/HL			HP/HL

MPEG-2 视频编码原理：

MPEG-2 图像压缩的原理是利用图像中的两种特性：空间相关性和时间相关性。一帧图像内的任何一个场景都是由若干像素点构成的，一个像素通常与它周围的某些像素在亮度和色度上存在空间相关性；节目中的情节常常由若干帧连续图像组成的图像序列构成，图像序列中前后帧图像间也存在时间相关性。这两种相关性使得图像中存在大量的冗余信息。将这些冗余信息去除，只保留少量非相关信息进行传输，就可以节省传输频带提高效率。利用保留的非相关信息，按照一定的解码算法，可以在保证图像质量的前提下恢复原始图像。MPEG-2 就是用 DCT 加熵编码去除图像内的空间相关性称为帧内编码，并利用运动补偿除去图像序列中的时间相关性称为帧间编码，只保留少量非相关信息。

在帧内编码的情况下，编码图像仅经过 DCT，量化器和比特流编码器即生成编码比特流，而不经过预测环处理。DCT 直接应用于原始的图像数据。

在帧间编码的情况下，原始图像首先与帧存储器中的预测图像进行比较，计算出运动矢量，由此运动矢量和参考帧生成原始图像的预测图像。而后，将原始图像与预测像素差值所生成的差分图像数据进行 DCT 变换，再经过量化器和比特流编码器生成输出的编码比特流。帧内编码与帧间编码流程的区别在于是否经过预测环的处理。

MPEG-2 中编码图像的三类帧：I 帧，P 帧和 B 帧。

I 帧图像采用帧内编码方式，即只利用单帧图像内的空间相关性，而没有利用时间相关性。I 帧主要用于接收机的初始化和信道的获取，以及节目的切换和插入，I 帧图像的压缩倍数相对较低。I 帧图像是周期性出现在图像序列中的，

出现频率可由编码器选择。

P 帧和 B 帧图像采用帧间编码方式，即同时利用空间和时间上的相关性。P 帧图像只采用前向时间预测，可以提高压缩效率和图像质量。P 帧图像中可以包含帧内编码的部分，即 P 帧中的每一个宏块可以是前向预测，也可以是帧内编码。

B 帧图像采用双向时间预测，可以大大提高压缩倍数。值得注意的是，由于 B 帧图像采用未来帧作为参考，因此 MPEG-2 编码码流中图像帧的传输顺序和显示顺序是不同的，如下图：

编码器输入端

1 2 3 4 5 6 7 8 9 10 11 12 13

I B B P B B P B B I B B P

编码器输出端、编码码流中、解码器输入端

1 4 2 3 7 5 6 10 8 9 13 11 12

I P B B P B B I B B P B B

解码器输出端

1 2 3 4 5 6 7 8 9 10 11 12 13

I B B P B B P B B I B B P

图 2-2 传输顺序和显示顺序

MPEG-2 的编码码流分为六个层次。从上至下依次为：视频序列层(Sequence)，图像组层(GOP： GroupofPicture)，图像层(Picture)，像条层(Slice)，宏块层(MacroBlock)和像块层(Block)。除宏块层和像块层外，上面四层中都有相应的起始码(SC：StartCode)，可用于因误码或其它原因收发两端失步时，解码器重新捕捉同步。

序列指构成某路节目的图像序列，序列起始码后的序列头中包含图像尺寸，宽高比，图像速率等信息。

图像组层由相互间有预测和生成关系的一组 I、P、B 图像构成，但头一帧图像总是 I 帧。GOP 头中包含时间信息。

图像层分为 I、P、B 三类。PIC 头中包含图像编码的类型和时间参考信息。

像条层包括一定数量的宏块，其顺序与扫描顺序一致。

宏块层。MPEG-2 中定义三种宏块结构：4:2:0 宏块 4:2:2 宏块和 4:4:4 宏块，分别代表构成一个宏块的亮度像块和色差像块的数量关系。

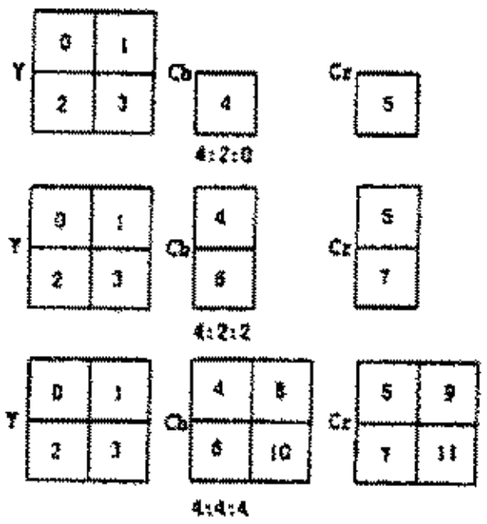

图 2-3 宏块结构

像块层是 MPEG-2 码流的最底层，是 DCT 变换的基本单元。MP@ML 中一个像块由 8×8 个抽样值构成，同一像块内的抽样值必须全部是 Y 信号样值，或全部是 Cb 信号样值，或全部是 Cr 信号样值。另外，像块也用于表示 8×8 个抽样值经 DCT 变换后所生成的 8×8 个 DCT 系数。。

§2.1.3 音频部分[9,12-15]

MPEG-2 音频标准是建立在 MPEG-1 基础上发展起来的多声道编码系统。在 MPEG-1 中，对音频压缩规定三种模式，即层 I、层 II（即 MUSICAM，又称 MP2），层 III（又称 MP3）。由于在制订标准时对许多压缩技术进行认真的考察，并充分考虑实际应用条件和算法的可实现性（复杂度），因而三种模式都得到广泛的应用。VCD 中使用的音频压缩方案就是 MPEG-1 层 I；而 MUSICAM 由于其适当的复杂程度和优秀声音质量，在数字演播室、DAB、DVB 等数字节目的制作、交换、存储、传送中得到广泛应用；MP3 是在综合 MUSICAM（Masking Pattern Universal Subband Intergrated Coding And Multiplexing）和 ASPEC（Audio Spectral Perceptual Entropy Coding）的优点的基础上提出的混合压缩技术， MP3 在低码率条件下高水准的声音质量，使得广泛用于软解压及网络广播。MPEG-2 音频标准的第 I，II 层称为 MUSICAM 环绕声。在 MPEG-2 音频第一层中，多声道扩展信息被分成 3 部分，在连续 3 帧 MPEG-2 的辅助数据部分中传输。而在第 II，III 层中，多声道扩展在帧 MPEG-1 辅助数据中传输。见下图：

MPEG-1 帧头	MPEG-1 CRC	MPEG-1 声音数据	MPEG-2 多声道扩展数据	MPEG-1 AUX

MPEG-2 声音的数据帧结构

MC 帧头	MC CRC	MC 组合状态信息	MC 声音数据	ML 声音数据

MPEG-2 多声道扩展数据帧结构

图 2-4 MPEG-2 音频数据帧结构

MUSICAM 环绕声能够确保与 MPEG-1 的第 I—III 层比特流的向前和向后兼容。

§2.1.4 使用 MPEG-2 的理由

- 编码延时低，适于实时应用。
- 压缩视频质量好，支持多种视频格式。
- 支持的图像分辨率高，支持高速体育运动的活动图像。
- 广泛应用于存储媒体 DVD，广播电视中的 DVB 和 HDTV，交互式的点播视频(VOD)和准点播视频(NVOD)，还能够适配于 ATM 宽带通信网。

§2.2 网络协议 TCP/IP [17~19,23~25]

TCP/IP不是一个简单的协议，而是由一系列专业化协议，包括TCP、IP、UDP、ARP 、ICMP 以及其他的一些被称为子协议组成的协议族。大部分网络管理员将整组协议称为TCP/IP ，有时也简称为IP协议 。TCP/IP的前身是由美国国防部在2 0世纪6 0年代末期为其远景研究规划署网络（ARPAnet）而开发的。由于低成本以及在多个不同平台间通信的可靠性，TCP/IP迅速发展并开始流行。它实际上是一个关于因特网的标准，迅速成为局域网的首选协议。

§2.2.1 TCP/IP 协议栈的五层参考模型[17~19,25]

TCP/IP 协议遵守一个五层的模型概念：应用层、传输层、互联（Internet）层和网络接口层以及硬件（物理层）。第五层是一个硬件层，在上面建立其它四个软件层：

- 网络接口：模型的基层是网络接口层。负责数据帧的发送和接收，帧是独立的网络信息传输单元。网络接口层将数据帧发送到网络中，或从网络中接收数据帧。
- 互联（Internet）层 ：互联协议将数据包封装成 Internet 数据报，并运行必要的路由算法。这一层上有四个互联协议：
  - 网际协议 IP：负责在主机和网络之间寻址和路由数据包。
  - 地址解析协议 ARP：获得同一物理网络中的硬件主机地址。
  - 网际控制消息协议 ICMP：发送消息报告有关数据包的传送错误。

◆ 互联组管理协议 IGMP：主机用来向本地多路广播路由器报告主机组成员。

- 传输层：传输协议在计算机之间提供通信会话。传输协议的选择根据数据传输方式而定。该层的两个传输协议：
  - ◆ 传输控制协议 TCP：为应用程序提供可靠的有连接通信。适合于一次传输大批数据的情况，并适用于要求得到响应的应用程序。
  - ◆ 用户数据报协议 UDP：提供面向无连接的通信，且不对传送包进行可靠的保证，可靠性则由应用层来负责。
- 应用层：应用程序通过这一层访问 TCP/IP 网络。

下图是对应于 OSI 七层参考模型的 TCP/IP 互联网络协议的参考模型。

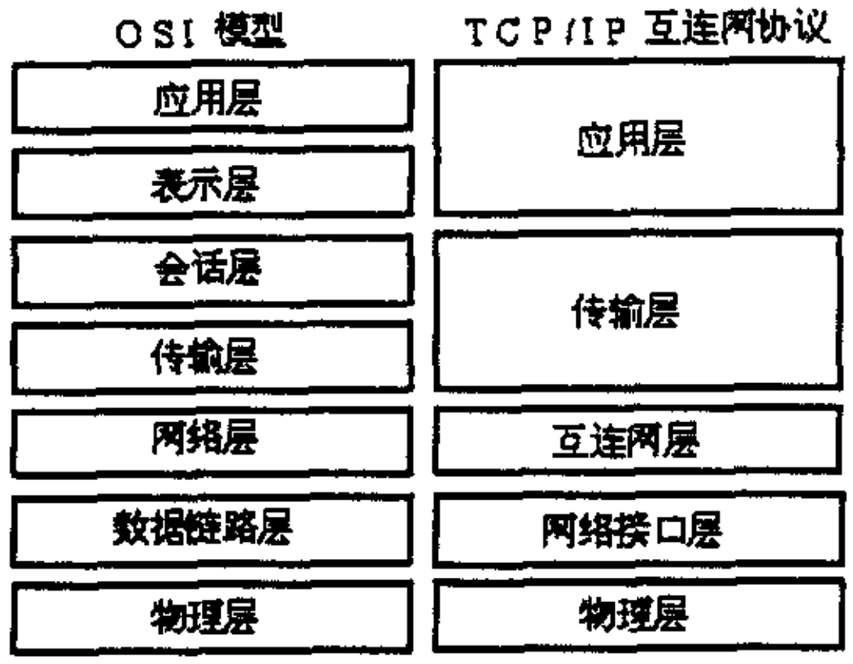

图 2-5　TCP/IP 与 OSI 模型的比较

下面只对 IP、TCP 和 UDP 进行介绍，并在它们之间作一些比较，在哪种协议层次上传输实时多媒体数据更合适。

§2.2.2 网际协议 IP[17~19]

IP 是一个面向无连接的协议，主要功能是：无连接数据报的传送、数据包路由选择和差错控制。在交换数据前它并不建立会话，不保证正确传递，另一方面，数据在收到时无论正确与否都不回送确认信息，IP 也不保证分组的正确顺序是不可靠的传输协议，可靠性只能由上层协议保证。

下图是 IP 数据报格式，各字段的含义可参考有关文献。

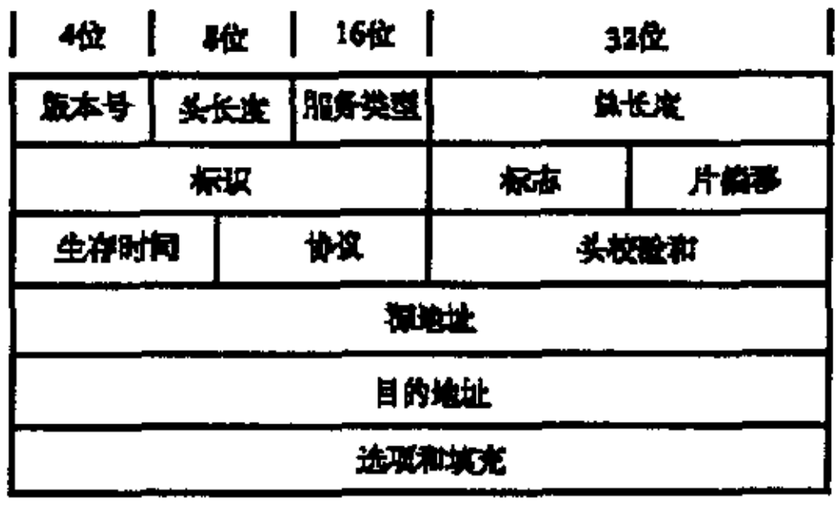

图 2-6　IP 数据报格式

IP 协议是一个数据链路（网络接口）层的协议，提供最基本的网络数据报传送功能，在它之上承载 TCP 和 UDP 协议。

§2.2.3 传输控制协议 TCP[17~19,22~24]

TCP 是一种可靠的面向连接的传送服务，它提供可靠的报文流传输和对上层应用的连接服务。它在传送数据时是分段进行的，主机交换数据必须建立一个会话。它采用传输数据比特流的方式通信，即数据被作为无结构的字节流。在传输数据之前 TCP 协议的双方通过三次握手建立连接，使数据段的发送和接收同步，告诉其它主机其一次可接收的数据量。

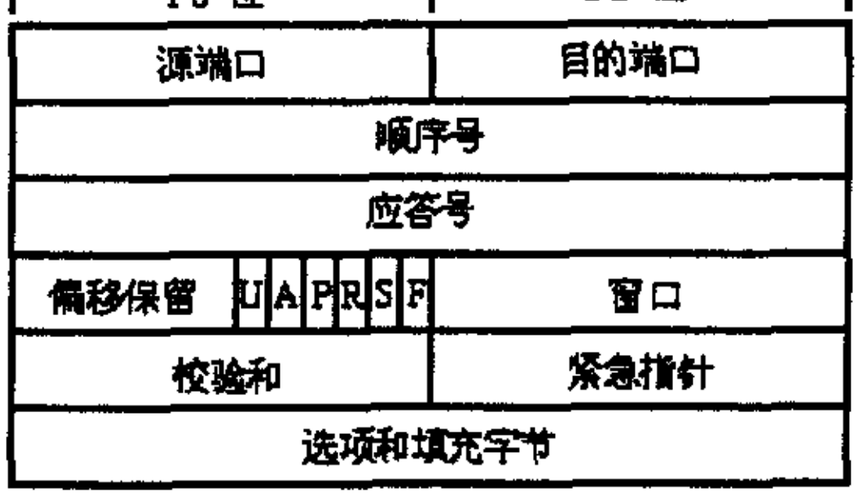

图 2-7　TCP 报头

三次握手的过程：

- 初始化主机通过一个同步标志置位的数据段发出会话请求。
- 接收主机通过发回具有以下项目的数据段表示回复：同步标志置位、即将发送的数据段的起始字节的顺序号、应答并带有希望收到的下一个数据段的顺序号。
- 请求主机再回送一个数据段，并带有希望收到的下一个顺序号和确认号。

在数据传输过程中通过对每个 TCP 传输的字段指定顺序号，以保证数据传输的可靠性。如果一个分段被分解成几个小段，接收主机会知道是否所有小段都已收到。通过发送应答，用以确认别的主机收到数据。对于发送的每一个小段，接

收主机必须在一个指定的时间返回一个确认。如果发送者未收到确认，数据会被重新发送；如果收到的数据包损坏，接收主机会舍弃它，因为确认未被发送，发送者会重新发送分段。数据传输完成后，拆除连接。

在 TCP 协议中采用滑动窗口算法来提供以下功能：

- 保证数据报的可靠传递；
- 确保数据的有序传输；
- 增强发送法和接收方的流量控制。

TCP 采用适应性重传来保证数据传输的可靠性，即如果在一定时间内发送方没有收到确认信号，就重传刚才发送的数据段。通常 TCP 把这个超时设为 RTT（往返时间 Round Trip Time）的一个函数。

拥塞是在一个或多个交换节点中数据报负载过重而出现的严重现象。此时在端点表象为时延的急剧增加，从而使 TCP 的重传机制反映为发出更多的重传数据报，进一步加重拥塞。TCP 采用拥塞控制机制来减缓拥塞现象，简单的原理是，在发生拥塞时把数据流量限制为小于接收方的缓冲区的大小。通常采用如下两种机制：

- 避免拥塞的加速递减策略：一旦发现丢失报文段，立即将拥塞的窗口大小减半（最小值为 1）。对于保留在发送窗口中的报文段重传定时器实现加倍。
- 恢复传输的慢启动策略：在启动新的连接或在拥塞之后增加通信量时，仅以一个报文段作为拥塞窗口的初始值，在每收到一个确认之后拥塞窗口加倍。

以上这些机制对于保证数据传输的可靠性是非常有效的，但是在传输实时数据尤其是实时多媒体数据的应用中，TCP 协议显得力不从心，这些用来保证数据传输可靠性的机制反而限制实时数据的及时传输。更为严重的是 TCP 的拥塞控制算法使用户可分配带宽随着网络的数据传输量而不断发生变化，结果造成实时数据的传输变得更不确定。

§2.2.4 用户数据报协议 UDP[17~19,22~24]

UDP 提供无连接的、不可靠的、无流量控制、不排序的简单协议，只是充当数据报的发送者和接收者。一般 UDP 只用于传输数据量较少的交互式应用，可

靠性完全由高层的应用程序来提供，包括处理丢失的报文，重复、时延、乱序、连接失效等。与直接使用 IP 报文传递的应用相比，使用 UDP 可以增加在给定主机上的多个目的传递的能力。通过在 IP 层上承载 UDP 数据报，这样就使用 IP 地址加端口号的方式来投递报文。

16 位	32 位
源端口	目的端口
报文长度	校验和
数据	

图 2-8　UDP 数据报文格式

其中应注意的是校验和的覆盖内容超出 UDP 数据报本身，包括 IP 报文的地址字段和协议字段。在实现中引入 UDP 伪首部（pseudo-header）的概念来计算 UDP 校验和。

UDP 协议是简单的传递用户数据，在本协议层中不提供任何应答，不能够保证用户数据是否到达目的地址；同时 UDP 协议不提供任何流量控制机制，网络拥塞时发送端仍可能发出大量的 UDP 数据报。从这两点上看 UDP 传递数据时非常不可靠的，难以用来传输可靠性要求很高的数据。但对于实时多媒体数据而言，及时把数据送达目的地为第一要素，可靠性相对不太重要，因此 UDP 就有可能用来承载这种数据传递的应用。而且 UDP 面向无连接，无应答和多点数据传递方式非常适于多播和广播应用。

由于 UDP 无法保证可靠性，仅仅使用 UDP 是不现实的，在传递实时数据时需要有更高层的传输协议来保证，那就是实时传输协议。

§ 2.2.5 UDP 比 TCP 更适于实时应用的理由[17~19,22~24]

- TCP 在保证数据传输可靠性同时削弱实时性：
  - TCP 的重传机制会尝试重复发送已经失效的数据而造成更多的实时数据失效。
  - TCP 的拥塞控制机制使用户可分配带宽随着网络的数据传输量而不断发生变化，结果造成实时数据的传输变得更不确定 。
- UDP 在本协议层中不提供任何应答重传机制，也不提供任何流量控制机制。虽然不可靠，但能较好的保证实时性，不会造成后继数据丢失时间有效性。

- 对实时多媒体数据而言，及时把数据送达目的地为第一要素，可靠性相对不太重要 。

  UDP 面向无连接，无应答和多点数据传递方式适于多播和广播应用。

§2.3 实时传输协议 RTP[10,16,19-24,39-46]

从前面协议介绍中可以看出，实时多媒体视频和音频的传输应该在 UDP 上传输，而 UDP 设计时并没有考虑实时业务传送的特殊要求，如媒体流的同步等。因此在 UDP 上传送实时业务时，要对 UDP 进行扩充，为此 IETF 制定实时传输协议（RTP）。

§2.3.1 RTP 传输协议[10,16,19-24]

实时传输协议（RTP）是为支持实时多媒体传输而设计的传输层协议。RTP 为应用进程之间提供实时通信能力，可应用与多媒体会议、连续数据存储、交互式仿真和控制测量等。RTP 协议族包括两个协议：实时传输协议（RTP）和实时传输控制协议（RTCP）。

- RTP 用来承载实时信息的传送，提供净荷类型指示、数据分组号、数据发送时间戳和数据源标识。
- RTCP 则用传送实时信号的质量参数、提供 QoS 监视机制；同时还可以传输通信中参与者的信息，向没有显示的成员控制和呼叫建立“松弛型”的通信控制机制。

RTP 自身不提供任何机制担保及时传送和服务质量保证，而是依赖底层的网络，也不能保证传送数据的顺序，而且也不认为下层网络传输时可靠的。

RTP 依赖下层网络传输 RTP 数据流和 RTCP 控制信息。RTP 作为传输层协议但是并不直接封装在 IP 数据报文中，而是由 UDP 报文来传递 RTP 数据。通常 RTP 使用一个偶数号的 UDP 端口，而相对应的 RTCP 则使用下一个奇数号 UDP 端口。RTP 数据包没有长度字段和其他边界，只能由下层网络协议来提供一个长度指示。用 UDP 作为下层协议的数据封装：

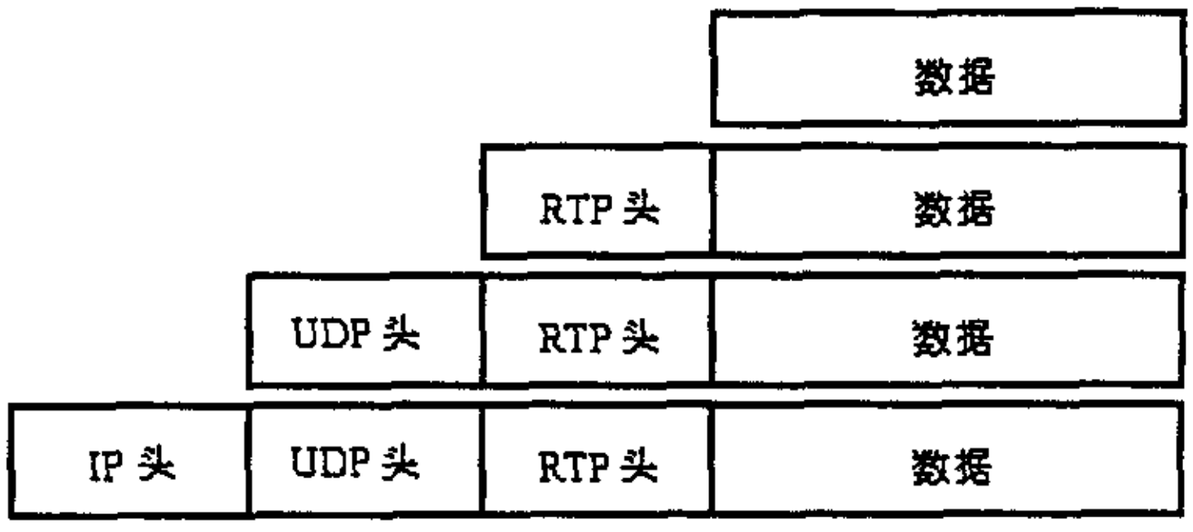

图 2-9　RTP 数据的封装

RTP 头部的格式，最开始的 12 字节总是有的，参与源表示符仅在某些情况下出现，之后都时可选的扩展头部。

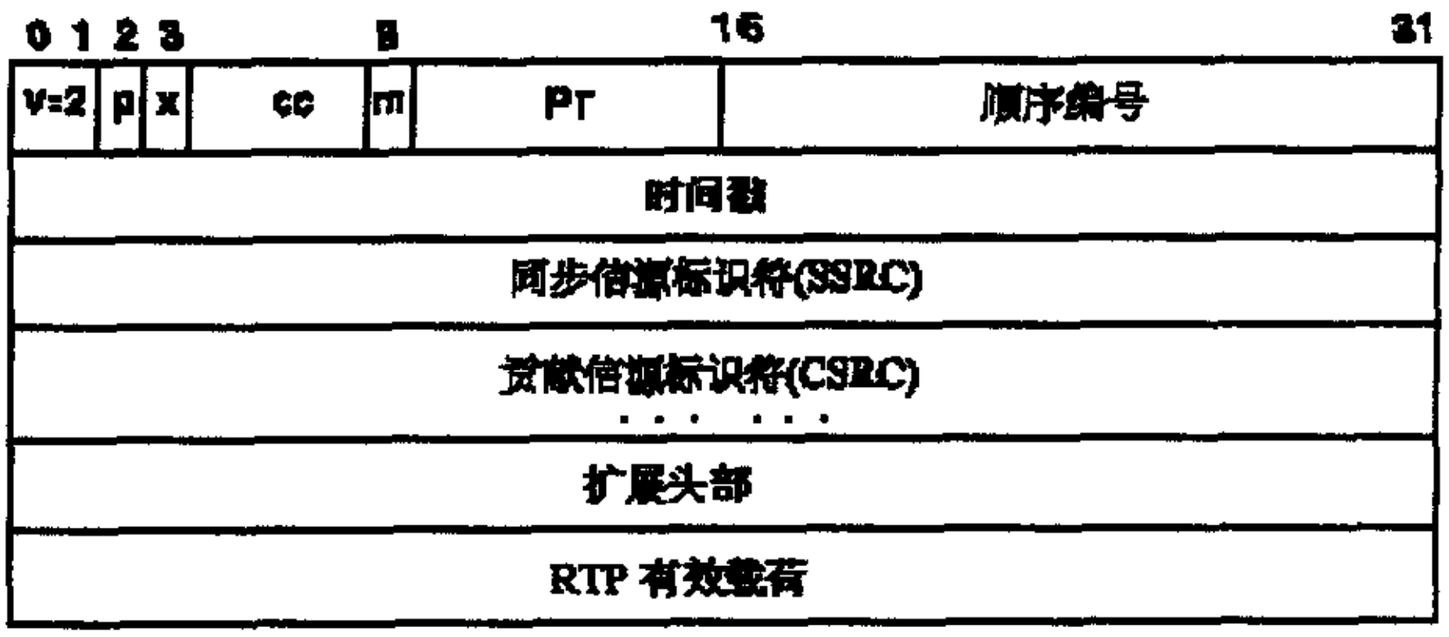

图 2-10　RTP 头部格式

- 版本（V，2比特）：这个域标识RTP 的版本当前版本是2。
- 填充（P，1比特）：如果设置填充比特位，包的实际数据内容比包本身要小。如果P = 1 ，填充位后面的一个字节是应该忽略掉的字节数。某些固定块大小的加密算法可能需要使用填充位，一个低层协议数据单元携带几个RTP 包也需要使用填充位。
- 扩展（X ，1比特）：如果设置扩展比特位，那么固定报头之后是一个报头扩展。
- CSRC源计数（CC，4比特）：CSRC值是固定报头后CSRC标识符的数目。这个数值用来支持被称为“混合”的能力。混合器是一个设备，它从多个信源收集多媒体包，把它们合并成一个包，然后转发到目的地。
- 标记（M，1比特）：标记的解释是由一个配置来定义的。它可以用来在包流中标记重要事件，如帧边界。配置可以定义另外的标记位，或者通过改变有效载荷类型域中的比特数来规定不使用标记位。

- 有效载荷类型（PT，7比特）：这个域标识RTP 有效载荷类型的格式，并且决定应用对它的解释。一旦会话建立起来，发送者不能在会话期间使用不同的RTP 有效载荷类型，所以不可能在同一个包内多路复用不同的介质流。
- 顺序编号（Sequence number ，16 比特）：每发送一个RTP 数据包，顺序编号就会增加1 ，它可被接收者用来检测包丢失和恢复包顺序。包重新排序很难实现，而且耗费音频和视频质量，因为实际上无法预测一个包是真的丢失还是迟到。由于不可预测的顺序编号的简单属性，使得在使用报头压缩时不得不排除掉它。
- 时间戳（Ti me stamp，32比特）：时间戳是在RTP 数据包的第一个字节采样时刻。采样时刻必须随时间单调线性增加，以允许同步和抖动计算。时钟分辨率必须满足要求的同步精确度，以及对包到达抖动的测量，并且可以分接成端到端延迟的分辨率。时钟频率与作为载荷携带的数据格式无关。对于周期性发送RTP 包而言，采样时刻是由采样时钟确定的，如果一个应用读入的数据块占据N个采样周期，时间戳对于这样的数据块会增加N。
- 同步源SSRC（32比特）：SSRC标识同步信源。同步信源指发送者。SSRC的数值是随机选取的，因此在同一个RTP段内没有两个发送者具有相同的SSRC标识。
- 参与源CSRC列表（0～15 项，每项3 2 比特）：CSRC列表标识提供这个包中包含的有效载荷的所有信源。标识符的数量是由CC域确定的。CSRC标识符仅在通过混合器时插入。

RTCP控制协议为多媒体应用程序提供一个与数据流相联系的控制流，它采用和数据分组同样的配送机制向RTP会话中的所有与会者周期性的传送控制分组，从而提供数据传送QoS的检测手段。它的主要功能是：

- 提供数据传送质量的反馈信息。这是RTCP的最基本的控制功能，反馈信息可以直接用于控制自适应编码，在IP多播中对于诊断数据分配故障也十分有用。

- 传送RTP源运输层永久性标识。该标识被称之为规范名（CNAME），由于SSRC标识在发生冲突或程序重启时是要改变的，因此接收方需要固定的CNAME来跟踪每个与会者。
- 确定RTCP分组的发送速率。由于RTCP分组需要定期发送，为防止控制分组过多必须根据可用带宽和回话规模来确定RTCP分组的发送速率。
- 传送少量会话控制信息。如传送与会者的标识，可在用户界面中显示。

RTCP分组类型：

- SR（sender report）：发送者报告。由数据发送者发出的发送和接收统计数据。

V	P	RC	PT	长度
发送方SSRC				
NTP时间戳（高字位）				
NTP时间戳（低字位）				
RTP时间戳				
发送方RTP分组计数值				
发送方RTP字节计数值				
SSRC_N（第N个同步源的SSRC）				
丢失率	累计丢失分组数			
扩展的已接受最高序号				
到达时延抖动				
最后SR时间戳（LSR）				
最后SR时延（DLSR）				
SSRC_N+1（第N+1个同步源的SSRC）				
……				
应用文档特定的扩展部分				

V：版本号，2比特。含义同RTP头部。

P：填充指示，1比特。通常填充在最后一个分组进行，因此只有最后一个分组的P为可能为1。

RC：接收报告计数，5比特。指示本分组包含的接收报告块的个数，可为零。

PT：净荷类型，8比特。SR的净荷类型值为200。

长度：16比特。其值加1为本RTCP分组的长度，长度单位为32比特字，包含头部和填充字节。

NTP时间戳：64比特。发出SR的绝对时间。NTP（Network Time Protocol）时间戳表示相对于1900.1.1时界零时的时间差值，单位为秒。

RTP时间戳：32比特。就是NTP时间戳的时间，只是时间单位及初始值和数据分组中的RTP时间戳相同。

SSRC_N：32比特。本报告块信息所属信源的SSRC表识。最多可有31个。

- RR（receiver report）：接受者报告。由非数据发送者发出的接受统计数据。RR分组格式与SR相同，差别在于：没有20字节的发送者信息，及PT=201。
- SDES（source description）：源描述项。包括CNAME。

V	P	RC	PT	长度
SSRC/CSRC_1				
SDES描述项 ……				
SSRC/CSRC_2 ……				

其中PT=202。

描述项类型编码表：

编码	英文名	中文名	描述项示例
1	CNAME	规范名	zhang@zjut.edu.cn
2	NAME	用户名	Zhang ang ran
3	EMAIL	电子邮件	zhangangran@sohu.com
4	PHONE	电话号码	+86 571 88320044
5	LOC	用户地理位置	Hang zhou， Zhe jiang
6	TOOL	应用工具名	Videotool 1.2

7	NOTE	状态通知	On the phone
8	PRIV	专用扩展描述	Prefix+value
（0）		描述项列表结束	

- BYE：指示会话退出。

V	P	RC	PT	长度
SSRC/CSRC				
… …				
长度	退出理由			

其中PT=203。

- APP：应用特定功能，提供新的应用或新的功能试验使用。

每个RTCP分组都有固定头部和若干个可变长度的数据单元组成。分组长度为32的整数倍，不同于RTP它在头部中有长度字段，因此多个RTCP可组装为一个RTCP复合分组。

§2.3.2 利用 RTP 传输 MPEG-2 [39-46]

本节描述一种用于 MPEG 视频和音频流的分组方案。本方案可以用于在支持 RTP 的传输协议上传输视频或音频流。在 MPEG-2 系统规范中定义两种系统流格式：MPEG-2 传输流（TS）和 MPEG-2 节目流（PS）。

为传输 MPEG 在 RFC2250 中定义的四种终端系统：

1．发送交互单元（TIU）

从本地 MTS 系统中接收 MPEG 信息，并使用本地基于 RTP 系统层通过分组网络发送（如 IP 网络）。比如实时编码器、MTS 卫星链路到 Internet、采用 MTS 编码节目的视频服务器。

2．接收交互单元（RIU）

从基于 RTP 网络中实时接收 MPEG 信息，并转发到本地 MTS 环境中。如：基于 Internet 的视频服务器到基于 MTS 的有线分配设备。

3．发送网际端系统（TAES）

发送或传输自身产生或存储或从基于 Internet 的计算机网络中接收的 MPEG 信息的 Internet 端系统。如：视频服务器。

4．接收网际端系统（RAES）

通过一个基于 RTP 的 Internet 接收 MPEG 信息在 Internet 端系统中显示或转发到传统的计算机网络。如：浏览训练视频的桌面 PC 或工作站。

根据应用中使用的不同 MPEG-2 码流，定义几种各式的 RTP 载荷类型来支持完整的 MPEG-2 系统语法以提供在四种端系统之间的基本互操作。

RTP 中的 MPEG-2 载荷封装

- 基于 **MPEG-2 TS** 的封装：

  适合于处理硬件编码器输出的 TS 操作，只要存在 TS 数据就可以使用这种方法。该方法使用 MPEG-2 时间模型，包括 PCR、DTS、PTS，而 RTP 的时间字段不用于解码，只用来估计和减少收发者之间的网络抖动和时间偏移。分组方案为：每个 RTP 分组载荷包含整数个 MPEG-2 传输流分组。由于 MPEG-2 TS 流具有同步字节的定长（188 字节）分组格式，而且这种格式的设计本身是适合于易发生错误的环境下传输。通常为提高端系统效率，把多个 TS 分组中的数据被合并到一个大的 RTP 分组中。封装在一个 RTP 分组中的传输流分组数等于 RTP 载荷长度除以 TS 分组长度（188 字节）。

  在 RTP 头部的 PT 字段用类型 MP2T=33 表示 MPEG-2 传输流，其中包括视频和音频。

RTP 报头	TS 分组 1	.....	TS 分组 n

图 2-11　TS 在 RTP 中的封装

- 基于 **MPEG-2 PS** 的封装：

  同样适合于处理硬件编码器输出的 PS 操作，只要存在 PS 数据就可以使用这种方法。而且没有严格的分组限制，这种格式只当作字节流来处理没有任何额外的结构。只能适于链路条件非常好的情况。

- 基于 **MPEG-2** 视音频 **ES** 独立分组的封装：

  为 MPEG-2 视频（ISO/IEC 13818-2）和 MPEG-2 音频（ISO/IEC 13818-3）两种载荷分别指定唯一的载荷类型。MPEG-2 视频的类型为 MPV=31，MPEG-2 音频类型为 MPA=14。

  MPEG-2 ES 比特流语法是针对相对不容易出错的环境设计的，在比特流

中存在大量的相关性（时间和空间上），丢失某些数据会造成其它数据失效。因此在 RTP 比特流描述首部添加额外的信息来增加某种恢复机制。另外 MPEG 图像可能非常的大，通常尺寸应小于典型 LAN/WAN MTU 的分组。使用以下分组规则：

1. 如果包含 MPEG Video_Sequence_Header，总是出现在 RTP 载荷起始的位置。
2. MPEG GOP_Header 总是出现在 RTP 载荷首部或者之后跟随有 Video_ Sequence_Header。
3. MPEG Picture_Header 总是出现在 RTP 载荷首部或之后跟随有 GOP_ Header。
4. 每一个首部都必须完全包含在同一个 RTP 分组中。

在 MPEG 中，每一幅图像都由一个或多个“像条”组成，一个“像条”被当作发生丢失或破坏的数据的恢复单元。一个“像条”的起始必须是分组的第一个数据，或者是在同一分组的整数个像条之后。这样保证丢失一个分组之后接受者不需要对分组内容进行扫描就可找到下一个“像条”的首部。只要满足上述条件，“像条”也可以被分到几个分组中传输。

还应注意的是：这种 RTP 载荷类型中应该在 MPEG-2 比特流中周期性的重复 Video_Sequence_Header 和 GOP_Header，用这种方法可以既可以实现增强数据流的容错能力，也实现频道切换和允许在连续重放的 MPEG 比特媒体流的任意点上接收并播放。在每一个 RTP 分组中重复 Picture_Header 也可以使解码器减小分组丢失的影响。

该方案中的时间机制使用网络时间协议（NTP），收发双方的时间都由 NTP 提供，RTP 时间的作用类似于 TS 封装。

- **基于 MPEG-2 视音频 ES 联合分组（MPEG-2 捆绑格式）的封装：**

捆绑联合分组方案中音频和视频统一分组传输在一些重要的应用中比其他的分组方案有较大的优势：

与独立分组格式相比联合分组的优点：

1. 只需要一对收发端口就可以实现传输视音频信号，减少独立数据流数量。
2. 提供隐式的视音频同步，不需要占用额外的带宽传输同步信息。

3. 视音频捆绑格式可以减少信息包首部信息，减少冗余信息提高传输效率。
4. 可以减少接收端的缓冲区尺寸，独立分组格式会由于唇同步的要求而增加额外的缓冲区。
5. 应用程序对总带宽的控制更容易。

与 **MPEG-2-TS** 分组格式相比联合分组的优点：

1. 减少冗余，由于捆绑格式中不需在每个 RTP 分组中都传输达相同的系统信息，据相关资料统计可以节省 2-3%的带宽。
2. 易于应用各种差错控制方案。TS 的设计中不能够容忍太多的数据分组的丢失，捆绑格式的分组结构符合 ALF 原理更容易实现错误隐藏或错误恢复。

§ 2.3.3 采用 MPEG-2 TS

尽管捆绑格式有较大的技术优势，但本设计中采用 MPEG-2 TS 分组格式。主要原因：

- ◆ 视音频同时传送，减少独立数据流。
- ◆ 无需额外同步数据流，节约带宽。
- ◆ 局域网中出错可能性并非难以忍受，TS 能提供有限容错能力。
- ◆ 捆绑格式需要更多的软件处理，对处理器要求高。
- ◆ 可以直接从硬件编码器输出 TS 分组。

§ 2.4 小结

本章已对 MPEG-2 视音频编码压缩标准、TCP/IP 协议以及 RTP 协议进行的详细地探讨，并给出如何在 RTP 中传输 MPEG-2 的几种方法。

第三章 MPEG-2 编码器的设计和硬件实现

§ 3.0 概述[2,3,9]

当多媒体数据数字化后会产生大量的数字信号，给存储和传输都带来巨大困难。尽管网络容量也在飞速发展，百兆、千兆 LAN 相继出现，但对于网络中众多用户的大容量数据对象还是需要较长的时间传输，因而多媒体数据需要经过压缩来提高网络传输的效率。

目前各种多媒体压缩技术众多，各有优点。MPEG-2 标准因以下突出特点而得到广泛应用：压缩视频质量好支持多种视频格式，支持的图像分辨率高，支持包括高速体育运动在内的活动图像，应用最为广泛，既包括存储媒体 DVD，广播电视中的 DVB 和 HDTV，还可应用于交互式的点播视频(VOD)和准点播视频(NVOD)，还能够适配于 ATM 宽带通信网。

多媒体编码器作为多媒体传输系统中的重要一环，本章将主要讨论以下内容：基于 MPEG-2 标准的编码器的设计原理，以及编码器实现中的一些问题。

§ 3.1 MPEG-2 编码器设计[26]

本节介绍一个通用 MPEG-2编码器的设计原 理，并对组成编码器的几个主要部分作详细地介绍。

§ 3.1.1 设计目标

本设计的目标是实现针对远程多媒体传输应用的 MP@ML 级别的 MPEG-2 实时硬件编码器，编码器输出的数据流要适合于通过 IP 网络传输。在设计中选用以 FUJITSU 公司的 MPEG-2 编码芯片 MB86390 作为核心，采用可编程逻辑器件和单片机来实现一个分辨率、数据格式、输出码率可调的通用多媒体编码器。

§ 3.1.2 系统结构

该编码器系统主要由编码芯片、视频解码电路、音频 A/D 电路、控制电路、输出码流变换电路、接口电路等构成。如下图所示：

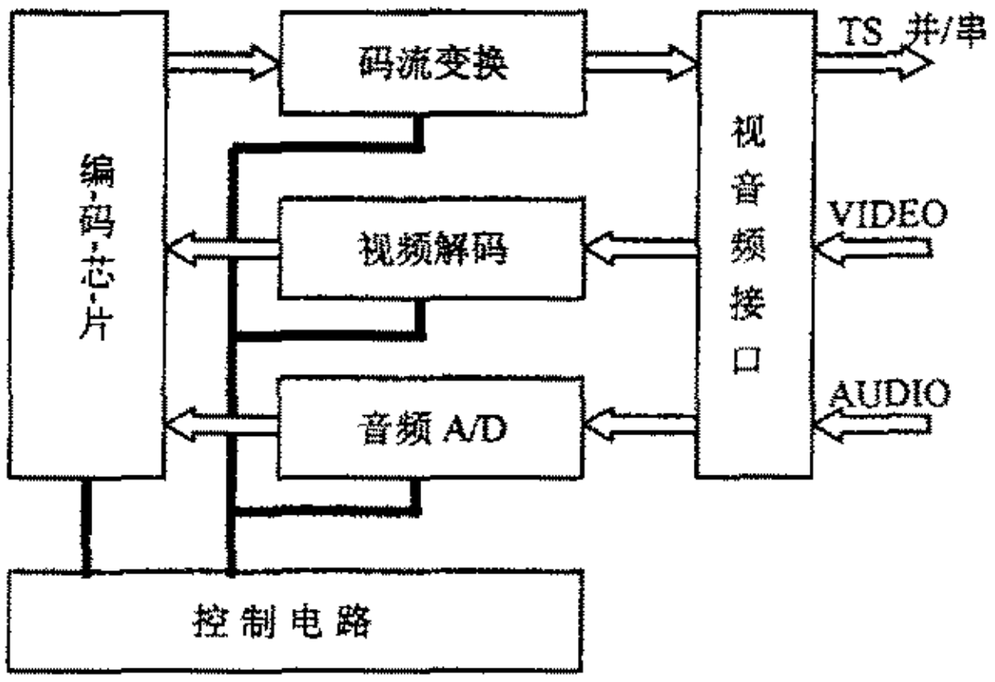

图 3-1　系统框图

编码芯片在工作时需要两片 64Mbits 的 SDRAM 来运行编码程序和作为视频编码存储区。编码器实现对原始数字视频进行 MPEG-2压缩并和压缩音频进行复用生成 PS 或 TS 流输出的功能。

视频解码电路采用 PHILIPS 公司的 SAA7113 视频解码器来实现模拟视频到数字视频的转换。SAA7113 是一种能够支持多种制式（PAL、NTSC、SECAM 等）并带有 4 路输入信号选择的 9 位视频信号处理器。

音频 A/D 主要是先把通过麦克风采集的模拟音频进行数字化，输出符合 MPEG-2 编码器格式的数字音频。

控制电路主要实现 MB86390 运行程序的写入，MB86390 运行参数的设置及运行控制，视频解码电路及音频 A/D 电路初始化。控制电路所控制器件包括所有的可编程控制的芯片。单片机作为控制主机向 MB86390 发出指令控制外围其它芯片初始化和运行时控制，CPLD 设计的信号转换接口，来实现 MB86390 和外部芯片的连接。这样整个编码器的控制只要通过单片机编程就能实现不同编码方式、速率、画面尺寸、视频输入选择的控制。

输出码流变换电路实现把并行输出的TS流数据转换成串行输出的TS流数据（这部分提供其它设备的接口不在本文讨论）。由于数据时钟提高 8 倍并向 MB86390 提供同步时钟，而且要根据不同的应用有不同的输出数据格式，该电路设计也采用基于可编程逻辑器件 CPLD 的设计。传输中利用 TS 流内部的同步字节来避免传输额外的同步信息，而在解码器接收端只要利用 TS 同步字节来捕获和跟踪就能保证 TS 流的同步接收。

系统输出接口提供 8 位并行输出接口，一对 LVDS 接口输出串行码流，模拟

视频、音频输入接口及相关匹配电路。

此外由于编码器数据输出时钟频率要求高于设定的编码速率，因此多数情况下数据流中存在 TS 空分组填充，编码传输效率低下。解决的基本方法可以设置编码器的数据输出时钟尽量低，只要能满足系统码率的要求即可，这可以保证 TS 分组出现有效负荷的概率大大增加，系统效率得到提高。

§ 3.1.3 编码器有关参数：[26]

- 视频编码
  - MPEG-1 编码 （ISO/IEC1 1172-2）
  - MPEG-2 MP@ML 编码 （ISO/IEC13818 -2）
  - 支持 NTSC 和 PAL 视频
  - 屏幕尺寸：32m×32n（NTSC 最大 720×480，PAL 最大 720×576）
  - 最大码率 15Mbps
  - 视频输入接口：D1 8bit 并行，YCrCb 8bit 并行
- 音频编码
  - MPEG-1 音频 layer-1/2 编码（ISO/IEC11172-3）
  - 最大码率 448Kbps
  - 音频通道：2 个（单声道，立体声，双声道，联合立体声）
- 系统复用
  - 输出格式：PS，TS，PES 和 ES
  - CBR 或 VBR
  - 最大码率 20Mbps

§ 3.2 编码器有关软件设计[26]

§ 3.2.1 编码器控制软件设计

编码器的控制程序需要对 MB86390、视频解码器、音频 A/D 等进行控制，在初始化过程中要对输入视频的制式、音频数据流格式、码流输出格式、系统编码方式、图像质量等等进行控制。视频解码器中的许多与视频模数转换有关的参数进行设置。而 MB86390 本身也需要设置很多与外围电路相同的参数，并且该芯片本身能够提供并行或串行通信方法。基于这一点可以使用单片机对 MB86390 的参数设置进行控制，达到对整个编码器进行初始化的目的。

控制主机和 MB86390 的串行口的通信协议流程如下。

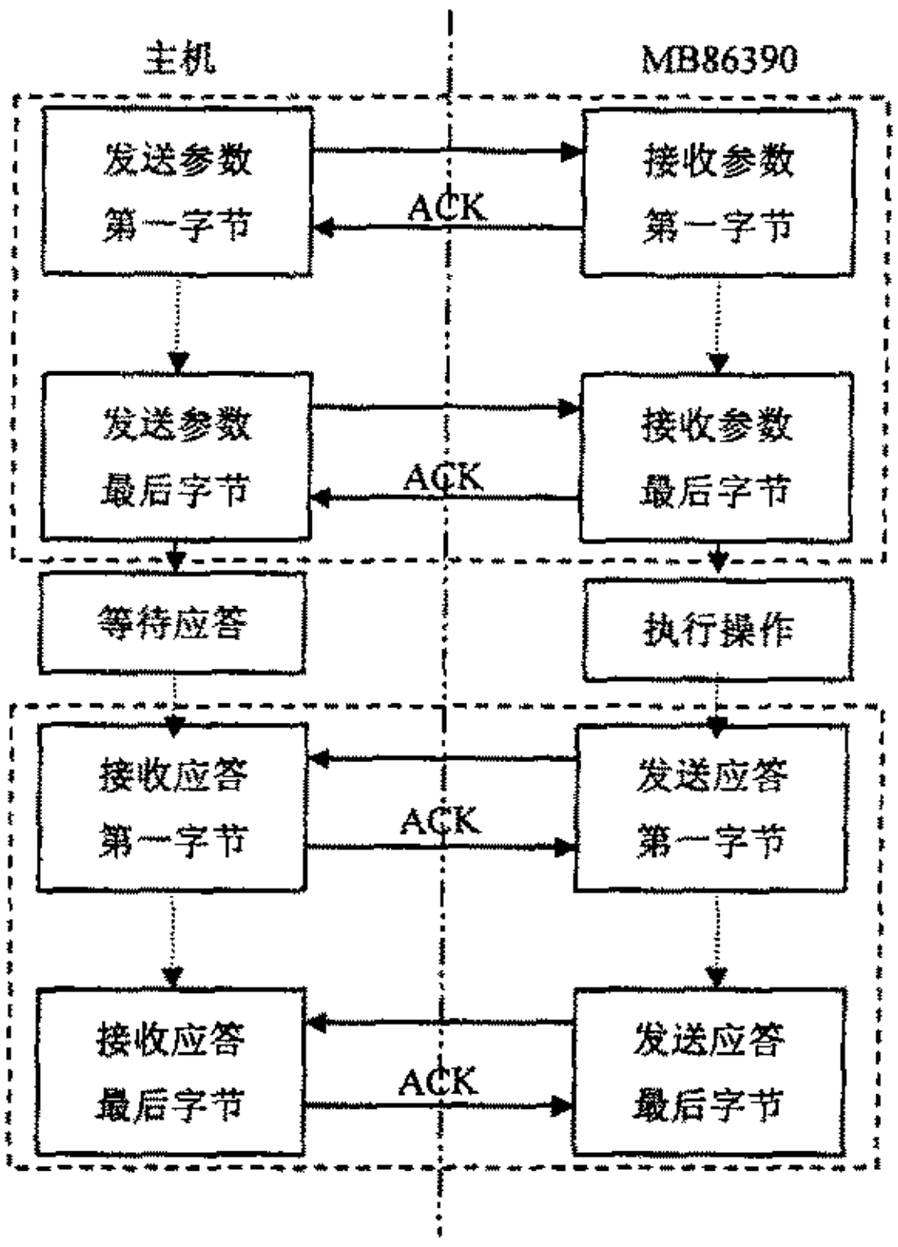

图 3-2 编码器通信协议流程

控制软件的主要功能可以分成如下层次：

- 通信方式设置。设置与编码器通信的数据传输率，校验方式，数据传输格式。
- 对各个命令参数按字节进行 8-6 比特格式调整。
- 对编码器的通信协议的支持模块。保证提供与编码器内核处理器通信的正确性，确保参数正确传输。
- 编码器参数设置流程控制。由于编码器设置相互关联，必须更具一定的规则来实现，否则编码器不能正常工作。

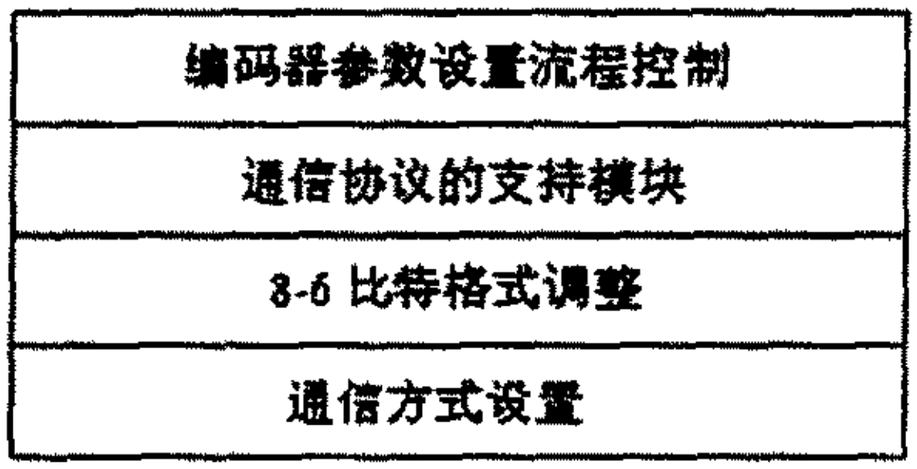

图 3-3 控制软件功能层次

§ 3.2.2 编码器输出数据码流[26]

编码器输出的 TS 流是 8 位并行数据，除数据输出同步时钟之外还有输出请求信号、输出使能信号和 TS 同步信号。这些信号都是和编码器直接相关的。除

数据之外，其中的输出时钟和 TS 同步信号必须传给解码器，使解码器同步工作。

下面是用于 IP 传输应用的并行格式的时序图：

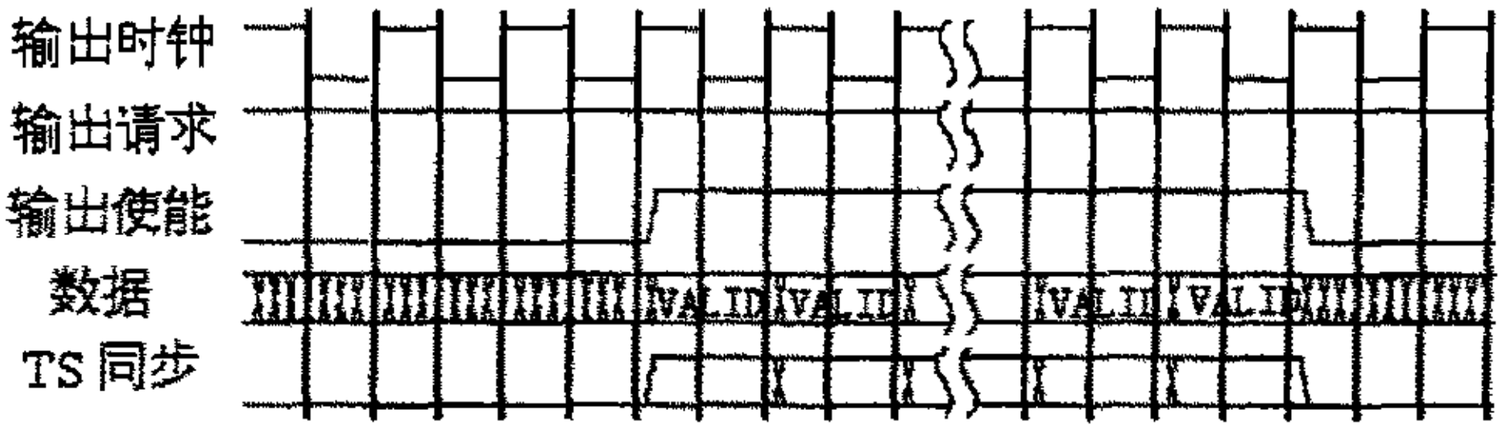

图 3-4 编码器输出数据的并行格式

§3.3 编码器的调试

对 MPEG-2 编码器的调试可以分成电路调试、控制程序调试、数据转换调试以及系统调试等几个阶段。

- 电路调试：首先完成对编码器电路的检查，确保电路各部分正确连接。出于可靠性的考虑在 PCB 设计时布线比较紧凑，采用的元件大多使用表贴封装，因此在电路焊接时的仔细操作可以大大节省电路调试的时间。在这一阶段中重点检查电源电路部分是否有错误连接，应该在上电前排除所有的人为电路错误。
- 控制程序调试：编码器系统上电后都会进行初始化且编码器本身支持使用串行口与主机通信，利用这一点来编写测试数据调试控制程序，能够从中掌握许多有用的信息以补充资料的差错和不足。在调试中利用单片机仿真器向编码器发送测试数据，通过对返回数据的分析，逐步掌握通过串行口设置编码器参数的协议流程，逐步完成控制程序的设计与调试。
- 数据转换调试：需要完成从编码器中读取并行 TS 流转成串行输出的 TS 流数据，这样串行 TS 流输出时钟是编码器读取时钟的 8 倍。而在解码器的接收端对串行 TS 流数据进行反变换，通过对 TS 流的同步码的收索、跟踪、伪同步码剔除确保 TS 流的正确恢复。

调试中需要借助示波器来观察和记录各个信息，检查时钟和数据信号判断错误的性质和来源，对相关部分进行调整和修改。完成上述几个调试过程之后就可进行系统调试。通常在通过以上几个步骤的调试之后，编码器可以正常工作。

§3.4 小结

本章首先对 MPEG-2 编码器的主要设计原理进行论述，并对编码器的相关软硬件进行详细的探讨。本章最后部分介绍软硬件调试过程。本文设计的 MPEG-2 编码器可以满足 2~20Mbps 系统码流的高清晰度视音频传输/存储应用。在视频传输系统设计中 MPEG-2 编码器的设计与实现是不可或缺的一部分。

第四章 嵌入式 Linux 系统

§4.0 概述

目前嵌入式 Linux 系统广泛用于各类计算应用，包括手持设备（PDA 和蜂窝电话）、因特网装置、瘦客户机、防火墙、工业机器人和电话基础设施设备。Linux 本身对网络协议的良好支持，使它能够在网络应用中大显身手。

本章就 Linux 以及嵌入式 Linux 系统的概念、特点进行介绍，并重点对嵌入式 Linux 系统中的应用开发工具、设备驱动程序和网络支持进行讨论。最后对本设计中使用的嵌入式系统模块 SOM5307A 进行简单的介绍。

§4.1 Linux 系统简介[29,31-33]

在如今的操作系统市场上，微软公司以其 Windows 系列场品 强劲攻势横扫全球市场，Linux 是为数不多的能够与其相抗衡操作系统。本节就对 Linux 系统的背景，历史和特点简单介绍。

§4.1.1 什么是 Linux?

Linux 是一个遵循 POSIX（Portable Operating System Interface,标准操作系统界面）标准的免费操作系统，具有 BSD 和 SYS V 的扩展特性，其外表和性能上兼容的 UNIX，但所有系统内核代码全部重新编写。Linux 由芬兰人 Linus Benedict Torvalds 于 1991 年 8 月正式发布，便宣布属于 GNU 软件，遵循 GPL 声明（GNU General Public License）。声明可以向任何人免费拷贝，自由使用，但不可涉及商业行为并要求软件发布者必须提供源代码。1994 年的 3 月 14 日发布 Linux 的第一个正式版本 1.0 版。到 2002 年 8 月最新稳定内核版本为 2.4.18。Linux 是目前最为流行的免费操作系统，并由无数的计算机专业人士和经验丰富的黑客不断的维护该系统，使得 Linux 系统稳定而又高效。Linux 的强大的网络功能和稳定高效的系统特性使得因特网以及局域网中的许多服务器都采用 Linux 系统。与同类型的 Windows NT 服务器相比，Linux 对硬件的要求降低一到两个档次，用普通 PC 机就能够实现复杂的网络任务。而且 Linux 下基于 TCP/IP 协议的软件非常丰

富，可以实现强大的网络功能。Linux 的开放源代码的特性使得系统本身利于裁减使得 Linux 被认为是一种理想的嵌入式操作系统。

§4.1 .2 Linux 的主要特点

1. 多任务系统：Linux 系统可以同时运行多个程序、多个进程，而且可以指定各个进程的优先级，达到合理分配资源的效果。
2. 多用户系统：Linux 系统支持多个用户通过各自的联机终端同时使用一台计算机，并响应多个用户的不同请求。
3. Linux 系统采用保护内存方式执行程序，所以个别程序失控时，不会引起整个系统的崩溃。
4. Linux 系统在磁盘上只读取实际用到的部分（动态链接 Dynamic Linking），大大提高系统响应速度，增强系统实时性能。
5. Linux 符合 POSIX（Portable Operating System Interface），源代码与 System V、一部分的 BSD 及 SVR4 完全兼容，软件移植是工作量低。
6. Linux 系统的大部分源代码都可以免费获得，包括所有内核、驱动程序、开发工具等。
7. Linux 提供一个良好网络接口，可以在其下开发网络功能非常强大的应用程序，如 FTP、Telnet、WWW 等。对基于 TCP/IP 的网络应用，Linux 具有非常优秀的性能。

§4.2 嵌入式系统和嵌入式 Linux[28~38]

本节介绍嵌入式系统和嵌入式 Linux 的一些概念。

§4.2.1 嵌入式系统概念

嵌入式系统(Embedded Operating System)是指操作系统和功能软件集成于计算机硬件系统之中。简单的说就是系统的应用软件与系统的硬件一体化，类似与 BIOS 的工作方式。具有软件代码小，高度自动化，响应速度快等特点。

高实时性是嵌入式系统的基本要求，其次，还要求代码尽可能小，运行速度尽可能快，可靠性尽可能高。最简单的嵌入式系统根本就没有操作系统,只有一个控制环。复杂的嵌入式系统有一个完整的操作系统能够完成资源管理和任务调度功能。

- 嵌入式系统特征

嵌入式系统是面向用户、面向产品、面向应用的。和通用计算机不同，嵌入式系统是针对具体应用的专用系统，具有成本敏感性，它的硬件和软件都必须高效率地设计，力争在同样的硅片面积上实现更高的性能。优秀的嵌入式系统是完成目标功能的最小系统。

- 对软件的要求
  - 固态化存储；
  - 代码高质量、高可靠性；
  - 系统软件具有实时处理能力；
  - 多任务操作系统。
- 嵌入式系统开发工具和环境

嵌入式系统本身不具备自举开发能力，必须依靠开发工具和环境才能进行开发。对于嵌入式系统，应用程序可以没有操作系统直接在芯片上运行，但是实现多任务处理，有效的利用系统资源、系统函数、专家库函数接口，必须选配 EOS(嵌入式操作系统)开发平台，这样才能保证程序执行的实时性、可靠性，并减少开发时间，保障软件质量。

- 嵌入式操作系统分类：

1、专用的实时操作系统（RTOS），如 QNX、VxWorks、pSos、lynx、Windows CE 等。但这些专用操作系统都是商业化产品，价格高昂，源代码封闭。具有模块化、实时性好、稳定性好的优点；但是也有难以得到源代码、开发和维护成本高、支持的硬件数量有限等缺点。

2、另外一大类就是嵌入式 Linux，它是对 Linux 经过小型化裁剪后，能够固化在容量只有几百 K 字节或几兆节的存储器芯片或单片机中，应用于特定嵌入式场合的专用 Linux 操作系统。最大的特点是遵循 GPL 协议源代码公开，在近两年以来成为研究热点。代表系统有：**uClinux** —它是一种没有 MMU （内存管理单元）的系统上运行的 Linux。uClinux 系统支持 Motorola 68K、ColdFire 微处理器。**LEM** — 运行在 386 上的小型(<8 MB)多用户、网络 Linux 版本。**LOAF** — "Linux On A Floppy"分发版，运行在 386 上。**ETLinux** — 设计用于在小型工业计算机上运行的 Linux 的完全分发版。**uLinux** — 在 386 上运行的 tiny Linux 分发版。

§ 4.2.2 嵌入式 Linux 优点：[28,29]

1. 遵从 GPL 源代码公开，可以按照需要任意修改；无须为每例应用交纳许可证费方便开发。
2. Linux 具备一整套工具链，容易自行建立嵌入式系统的开发环境和交叉运行环境，并且可以跨越嵌入式系统开发中仿真工具的障碍。如著名的 GNU 编译器 GCC 和调试器 GDB，功能强大且可靠。
3. 软件的开发和维护成本低。只需懂 Unix/Linux 和 C 语言。
4. 强大的网络支持功能。Linux 支持所有标准因特网协议，可以利用 Linux TCP/IP 网络协议栈。
5. 系统运行稳定。linux 本身具备的优点。
6. 内核精悍，运行时占用的资源少，十分适合资源紧张的嵌入式应用。
7. 支持的硬件数量众多，如 Intel X86 芯片家族、 Motorola 公司的 68K 系列 CPU 和 IBM、Apple、Motorola PowerPC、Strong ARM CPU 等处理器系统等。

在嵌入式领域 Linux 获得飞速发展，目前正在开发的嵌入式系统中，49%的项目选择 Linux 作为嵌入式操作系统。Linux 之所以能在嵌入式系统市场上取得如此迅速的发展，与它自身的优良特性有着不可分割的关系。Linux 现已成为嵌入式操作系统的理想选择。

§ 4.2.3 嵌入式 Linux 的构架：[36]

典型的嵌入式 Linux 系统构架如下图：

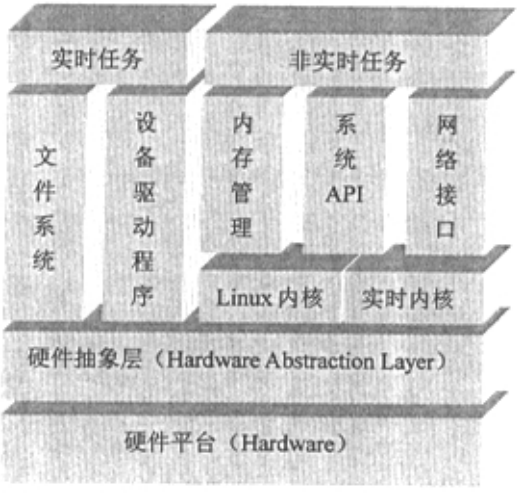

图 4-1 嵌入式 Linux 系统结构

硬件抽象层（HAL）包含所有和硬件平台相关的代码，如上下文切换和 I/O 寄存器访问等。它存在于嵌入式 Linux 的最底层，直接访问和控制硬件，对上层的硬件提供访问和控制服务。

内核是用来为大多数程序乃至网络、文件系统、驱动程序构建一系列在抽象的文件上工作的抽象机，使用户程序及上层组件对系统设备透明。

实时内核提供对事件优先级的调度和抢占支持，增加系统实时调度的能力。

应用程序函数（API）接口，为用户定制网络、图形、视频等应用程序接口。

设备驱动程序，为用户提供系统操作到物理设备的映射，控制管理物理设备。

文件系统，在物理介质上对数据访问的组织方法。

§ 4.2.4 uCLinux[28,29,38]

uClinux 就是 Micro-Control-Linux，字面上的理解就是针对微控制领域而设计的 Linux。uClinux 的内核的可执行映象烧写到 Flash 上，系统启动时从 Flash 的某个地址开始逐句执行。

特点

- 从 Linux 2.0 内核发展而来，面向 MMU-less 的微处理器。
- 内核代码被重新改写，代码更加紧凑精悍，生成内核小于 512Kb。
- 保持 Linux 的多任务和高稳定性的特色，兼容 Linux API。
- 继承 Linux 优良的网络功能，完整的 TCP/IP 协议栈。
- 支持众多的文件系统，ROMFS，Fat16/32，MS-DOS，NFS，EXT2。

根（root）文件系统

uClinux 系统采用 romfs 文件系统，这种文件系统相对于一般的 ext2 文件系统要求更少的空间。空间的节约来自于两个方面，首先内核支持 romfs 文件系统比支持 ext2 文件系统需要更少的代码，其次 romfs 文件系统相对简单，在建立文件系统超级块（superblock）需要更少的存储空间。Romfs 文件系统不支持动态擦写保存，对于系统需要动态保存的数据采用虚拟 ram 盘的方法进行处理（ram 盘将采用 ext2 文件系统）。

应用程序库

uClinux 小型化的另一个做法是重写应用程序库，相对于越来越大且越来越

全的 glibc 库，uClibc 对 libc 做精简。

uClinux 对用户程序采用静态连接的形式，这种做法会使应用程序变大，但是基于内存管理的问题，这种做法也更接近于通常嵌入式系统的做法。

开发环境

Gnu 开发套件作为通用的 Linux 开放套件，包括一系列的开发调试工具。主要组件：

- Gcc：编译器，可以做成交叉编译的形式，即在宿主机上开发编译目标上可运行的二进制文件。
- Binutils：一些辅助工具，包括 objdump（可以反编译二进制文件），as（汇编编译器），ld（连接器）等等。
- Gdb：调试器，可使用多种交叉调试方式，gdb-bdm（背景调试工具），gdbserver（使用以太网络调试）。

打印终端

通常情况下，uClinux 的默认终端是串口，内核在启动时所有的信息都打印到串口终端（使用 printk 函数打印），同时也可以通过串口终端与系统交互。在 uClinux 系统中没有控制台输出，所有的输出信息都是从串行口打印。

uClinux 在启动时启动 telnetd（远程登录服务），操作者可以远程登录上系统，从而控制系统的运行。

§4.3 嵌入式 Linux 开发工具[29-31]

在嵌入式 Linux 系统开发中，经常会遇到脚本编程、编写 Makefile、gcc、gdb 等工具，因此本节就这些开发工具做一个简单的介绍。

§4.3.1 Bash 脚本

在应用嵌入式 Linux 系统时，经常需要改动系统的运行脚本或编写自己的脚本程序，如在利用 gcc 编译 c 代码时需要开发自定义的 Makefile，而在 Makefile 编写过程中会用到大量的 Bash 语法。Bash 是 Linux 系统中的一种命令语言解释器，它是由 Linux 系统提供的多种解释语言中的一种，在语法上与其他 shell 解释器有一定的相似之处，但 Bash 的处理能力更强。

Bash 作为一种脚本语言编程只有两个简单的步骤：创建脚本和运行脚本。在

Linux 环境中通常使用 Emacs、gEdit、Vi 等编辑器就可以编写 Bash 脚本。在嵌入式 Linux 系统的开发过程中，经常要利用 Bash 脚本程序控制 Linux 系统的启动和运行，因此熟悉 Bash 是非常重要的。

§4.3.2 Make 管理项目

嵌入是系统开发中编程是必不可少的，通常使用的编译工具都是 GNU 的 gcc 多数情况下都要使用 make 管理项目。Make 管理项目通过把编译命令保存在 Makefile 中简化编译工作，而且还在数据库中维护当前开发工程中的各个文件的依赖关系，在编译前就能确定所需文件是否存在，make 用来维护程序十分方便有效。

Makefile 是一个数据库文件，以文本的形式存储，其规则指明 make 如何编译文件和编译何种文件。一条规则包含三个部分的内容：

- 要创建的目标文件 target
- 依赖文件列表 dependencies list
- 命令组 commands group

通常 make 默认的 Makefile 文件名为 GNUmake、Makefile、makefile 也可以指定其他名字。通常都使用 Makefile 作文件名。一个简单的 Makefile 规则可用如下代码表示：

```
targe: dependency file1 dependency file2 [...]
    command1
    command2
    [...]
```

在基于嵌入式 Linux 系统的开发中必须为应用程序编写 Makefile 才能保证用户程序被系统正确的编译。

§4.3.3 Gcc 编译器

Gcc 编译器是一个在 Unix/Linux 系统上运行的功能强大的编译器，主要用于对 c/c++等语言的编译。随着 Linux 的流行，gcc 不断的完善和发展使得许多商业编译器都相形见拙。

结合使用 gcc 和 make，可以对程序编译过程进行深层次的控制。Gcc 编译器将编译过程分为预处理、编译、汇编和链接四个阶段，并能够在任意的一个阶段

暂停并检查输出信息，还能在生成二进制文件式输出调试信息。

Gcc 是一个交叉平台编译器，能够在所处的 CPU 平台上为异构体系硬件系统开发应用程序。如在 Intel 的机器上可以编译出能够在 Motorola 的 CPU 平台上运行的应用程序，这一点在嵌入式系统开发中尤为重要。因为多数嵌入式系统本身没有开发工具和开发环境，必须借助一个开发系统环境（如 redhat）然后利用交叉编译器生成可在目标系统中运行的程序。

§4.3.4 调试工具 GDB

调试是开发过程中必不可少的环节。嵌入式操作系统与通用的桌面操作系统在调试环境上存在明显的差别：桌面操作系统中，调试器与被调试的程序往往是运行在同一台机器、相同的操作系统上的两个进程，调试器进程通过操作系统专门提供的调用接口控制、访问被调试进程。嵌入式系统一般采用远程调试，调试器运行于通用桌面操作系统，被调试的程序则运行于目标系统。在目标操作系统和调试器内分别加入某些功能模块，二者互通信息来进行调试，如图所示。

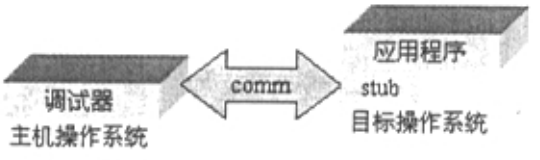

图 4-2 远程调试

Linux 包含一个称为的 GDB 的 GNU 调试程序可以调试各种程序，包括 C、C++、JAVA、PASCAL、FORAN 和 GNU 所支持的所有微处理器的汇编语言。利用 GDB 可以在程序运行时观察程序的内部结构和内存的使用情况：

- 监视程序中变量的值。
- 设置断点使程序在指定的代码行上停止执行。
- 单步跟踪执行代码。

运行 GDB 的开发平台通过串行端口连接到目标系统，GDB 就可以对应用程序进行远程调试。GDB 通过串口单步调试程序代码、设置断点、检验内存，同目标系统交换信息，并把调试信息在主机终端上显示。

§4.4 嵌入式 Linux 设备驱动程序[29~31,51]

§4.4.1 设备驱动程序和内核接口

Linux 内核必须能够用标准的方式和设备驱动程序通信。字符、块和网络设备驱动程序都提供通用的接口供内核在需要时调用。这些通用的接口意味着内核可以完全相同地看待不同的物理设备和对应的设备驱动程序。例如，SCSI 和 IDE 磁盘的行为非常不同，但是 Linux 内核对它们使用相同的接口。

§4.4.2 字符设备

字符设备是 Linux 最简单的设备。应用程序使用标准系统调用打开、读取、写和关闭，把字符设备当作一个文件访问。当字符设备初始化的时候，它的设备驱动程序向 Linux 内核登记，在 chrdevs 向量表增加一个 device struct 数据结构条目。这个设备的主设备号（如 tty 设备是 4），用作这个向量表的索引。一个设备的主设备号是固定的。Chrdevs 向量表中的每一个条目（device struc 数据结构），包括两个元素：一个登记的设备驱动程序的名称的指针和一个指向一组文件操作函数的指针。这组文件操作函数位于这个设备的字符设备驱动程序中，每一个都处理特定文件操作，比如打开、读、写和关闭。/proc/devices 中字符设备的内容来自 chrdevs 向量表。

§4.4.3 块设备

块设备也支持像文件一样被访问和字符设备的十分相似。Linux 用 blkdevs 向量表维护已经登记的块设备文件，像 chrevs 向量表一样，使用设备的主设备号作为索引。它的条目也是 device_struc 数据结构。和字符设备不同，块设备进行分类，分为 SCSI 和 IDE。块设备驱动程序向 Linux 内核登记并向内核提供文件操作。一种块设备类的设备驱动程序向这种类提供和类相关的接口。

每一个块设备驱动程序必须提供普通的文件操作接口和对于 buffer cache 的接口。块设备驱动程序填充 blk_dev 向量表中它的 blk_dev_struct 数据结构。这个向量表的索引还是设备的主设备号。这个 blk_dev_struct 数据结构包括一个请求例程的地址和一个指针，指向一个 request 数据结构的列表，每一个都表达 buffer cache 向设备读写一块数据的一个请求。

§4.4.4 网络设备

网络设备是与 Linux 的网络子系统有关的发送和接收数据包的物理设备，如

以太网卡。但是也有一些网络设备是纯软件的，如 loopback 设备，用于本机会环发送数据。每一个网络设备都用 Device 数据结构表示。网络设备驱动程序在内核启动网络初始化的时候向 Linux 登记它控制的设备。Device 数据结构包括这个设备的信息，以及允许大量 Linux 支持的网络协议使用这个设备的服务的函数的地址，这些函数多数和使用这个网络设备传输数据有关。设备使用标准的网络支持机制，向适当的协议传输接收的数据。传输和接收的所有的网络数据包（packets）都用 sk buff 数据结构表示，这是一种灵活的数据结构，允许网络协议头很容易地增加删除。

网络设备驱动程序像其他 Linux 设备驱动程序一样，可以加载到 Linux 内核中。每一个可能的网络设备都用 dev base 列表指针指向的网络设备列表中的一个 device 数据结构表示。如果需要设备相关的操作，网络层调用网络设备服务例程（放在 device 数据结构中）其中的一个。

§4.5 嵌入式 Linux 系统对网络协议的支持

嵌入式 Linux 从具有优良的网络功能的 Linux 系统发展而来，继承 Linux 的全部优势包括对从多网络协议的支持，也随着 Linux 的发展而不断发展。

Linux 支持 TCP/IP，IPX，X.25，AppleTalk 等的协议，各种具体协议实现的源码都可以免费获得。Internet 网络的核心协议是 TCP/IP，Linux 支持全部工业标准的 TCP/IP 协议栈：

- 控制数据的协议：TCP，UDP
- 数据路由协议：IP，ICMP，RIP，OSPF，ARP，RARP
- 用户服务：BOOTP，FTP，TELNET，EGP，GGP，IGP
- 其它协议：NFS，NIS，RPC，SMTP，SNMP

更重要的是 Linux 提供所有 TCP/IP 协议的源代码，并且支持 BSD 风格的套接字，有助于网络应用程序编写。因此在基于 IP 网络的多媒体传输系统的设计中，采用嵌入式 Linux 的设计充分可以利用系统对底层网络协议的支持，增强应用系统的可靠性和执行效率。

§4.6 嵌入式系统模块[28,29]

本设计过程中需要借助于万禾公司开发的嵌入式系统模块 SOM5307A 是一个基于 Motorola Coldfire MCF5307 CPU 的 32 位微处理器系统模块(System on Module)，包含运行嵌入式操作系统所需要的核心硬件如 CPU、SDRAM、Flash 以及 10M 以太网接口等，具有体积小、耗电低、处理能力强、网络功能强大的特点，能够装载和运行嵌入式 Linux 操作系统。如图 4-3 所示。

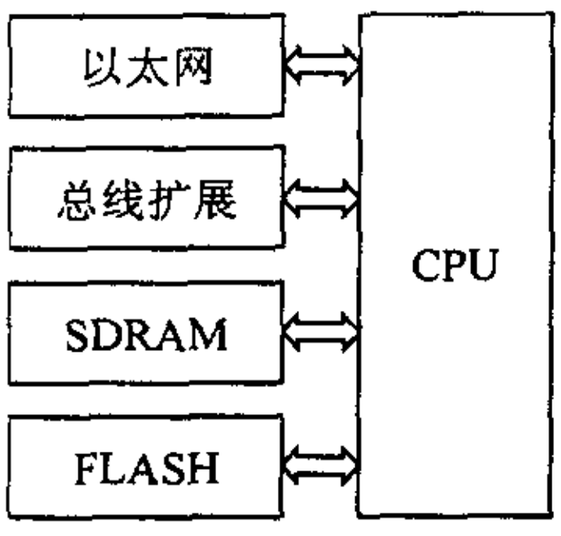

图 4-3 SOM5307A 框图

SOM5307A 的核心部件是 Motorola Coldfire MCF5307 CPU，这是一个专为嵌入式应用设计的 32 位 CPU，工作频率 90MHz，处理能力 75 Dhrystone 2.1，功耗 910 毫瓦；16M 字节 SDRAM，速度 70ns，通过 32 位数据总线与 CPU 交换数据，用于运行嵌入式 Linux 系统和用户程序；2M 字节快闪存储器（FLASH MEMORY），速度 70ns，通过 16 位数据总线与 CPU 交换数据，用于存储嵌入式 Linux 操作系统和用户程序的可执行影象文件；以太网接口是基于 Cirrus Logic 公司 CS8900A 芯片的 10BaseT 以太网接口。

§4.7 小结

本章首先介绍 Linux 系统的发展和特点和对嵌入式系统的概念、特点，说明 Linux 是理想的嵌入式操作系统。然后介绍本设计使用的 uClinux 统以及嵌入式 Linux 系统应用开发的主要工具进行介绍，讨论嵌入式 Linux 系统驱动程序设计，嵌入式 Linux 系统对 TCP/IP 网络协议的有力支持，说明嵌入式 Linux 是开发网络应用的理想选择。最后介绍嵌入式系统模块 SOM5307A。

第五章 嵌入式实时多媒体信息传输系统

§5.0 概述

本章就实时多媒体信息压缩传输系统框架和它的工作原理，以及各个模块设计进行说明。在本章中涉及的编码器、嵌入式系统、传输协议等都在前面章节有较具体介绍，在本章设计过程中应用这些技术。在此章节中只对几个主要的软硬件模块子系统功能设计加以说明，如编码器接口、编码器驱动程序、发送端和接收端程序 、建立嵌入式开发平台等。

§5.1 系统

在论文第一章已经对这个基于嵌入式 Linux 系统设计的实时多媒体信息压缩传输系统概况进行简单的介绍，在随后的章节中已经先后说明本设计涉及的相关知识和 MPEG-2视音频编码器模块的设计，下面就对这个多媒体信息传输系统设计的具体内容作具体介绍。

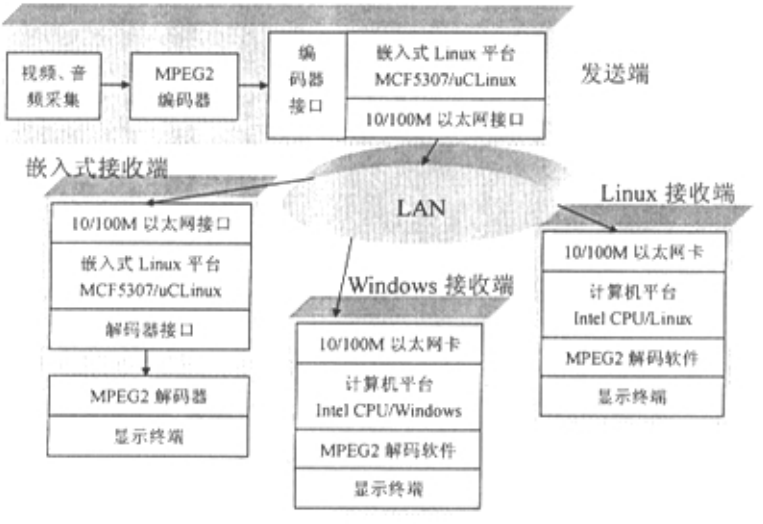

图 5-1 系统框图

在图 5-1 所示的实时传输系统中，实时视音频信息通过摄像机和麦克风采集，经过模拟数字变换后送到 MPEG-2 编码器压缩编码，输出 4Mbps 的 TS 分组数据

流；然后 MPEG-2 编码器输出的压缩码流通过一个接口送到嵌入式发送系统中，在这个接口中利用可编程器件实现对 MPEG-2编码器输出的连续压缩数据流进行缓冲、以及实现和嵌入式系统平台 CPU 的读写时序功能；在嵌入式系统中运行的操作系统是基于 2.0.38 版内核的 uCLinux 系统，基于该系统设计一个驱动程序来实现从 MPEG-2编码器中读取数据，把数据按照一定长度打包向局域网中广播发送。在发送软件中保证每个数据包中包含整数个 TS 分组，采用 RTP 协议传输实时数据。

发送系统指基于嵌入式 Linux 平台的子系统，完成读取 MPEG-TS 对数据打包封装多点发送功能。

- 基于 Motorola Coldfire5307 系统模块
- 10/100M 以太网接口
- 基于 2.0.38 内核的 uCLinux 操作系统
- 基于 UDP 的发送端软件

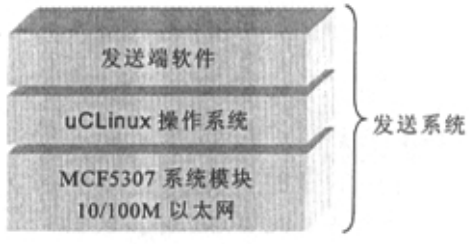

图 5-2 发送系统

接收端负责从网络中接收数据流、缓冲、解码并回放，结构模式类似发送端分成两大模块：负责网络传输功能模块和 MPEG-2压缩数据流解码 模块。在本设计中的接收端按照不同的解压方式可以有两种类型，并且分别以不同的计算机平台来实现。图 5-1 中显示这三种不同的接收端，其中嵌入式接收端是基于嵌入式 Linux 系统平台＋硬件 MPEG-2 解码的系统，其实现方式非常类似于发送端，其中需要嵌入式的接收系统和 MPEG-2解码器，显示终端可以采用电视机或监视器；Windows 接收端则是装有网卡和 Windows 系统的计算机系统，并用软件实现 MPEG-2 TS 解码，并把电脑显示器作为视频显示终端，音频从声卡输出。Linux 接收端则是装有网卡和 Linux 系统的计算机，同样使用软件实现 MPEG-2 TS 解码和显示。

- 基于硬件解码的接收系统

- ◆ 基于 uCLinux 和 MPEG-2 硬件解码
- ◆ 软硬件上与发送系统类似

■ 基于软件解码的接收系统

- ◆ 基于 Windows、Linux 系统
- ◆ 以 IBM PC 作为硬件平台
- ◆ 采用 VLC 实现解码软件

在本章的以后章节如不加以说明都用以下称呼：

发送端：基于嵌入式 Linux 系统设计的 MPEG-2 发送端。

接收端：对嵌入式接收端，Windows 接收端，Linux 接收端的统称。

嵌入式接收端：基于嵌入式 Linux 系统设计的 MPEG-2 接收端。

Windows 接收端：Windows 系统计算机平台接收 MPEG-2 数据解码并显示。

Linux 接收端：Linux 系统计算机平台接收 MPEG-2 数据解码并显示。

§5.2 MPEG-2 编码器接口

本节介绍嵌入式发送系统和 MPEG-2 编码器的硬件接口，嵌入式系统通过该接口实现从编码器的读取压缩数据和输出数据时钟、复位信号等控制信号。设计该接口的主要目的就是减少嵌入式系统被外设中断次数，以提高系统运行效率。

§5.2.1 工作原理

发送系统和编码器的接口工作原理如图 5-3 所示，按照功能可以划分成 MPEG-2 编码器接口、先进先出缓冲、5307 接口和控制单元。

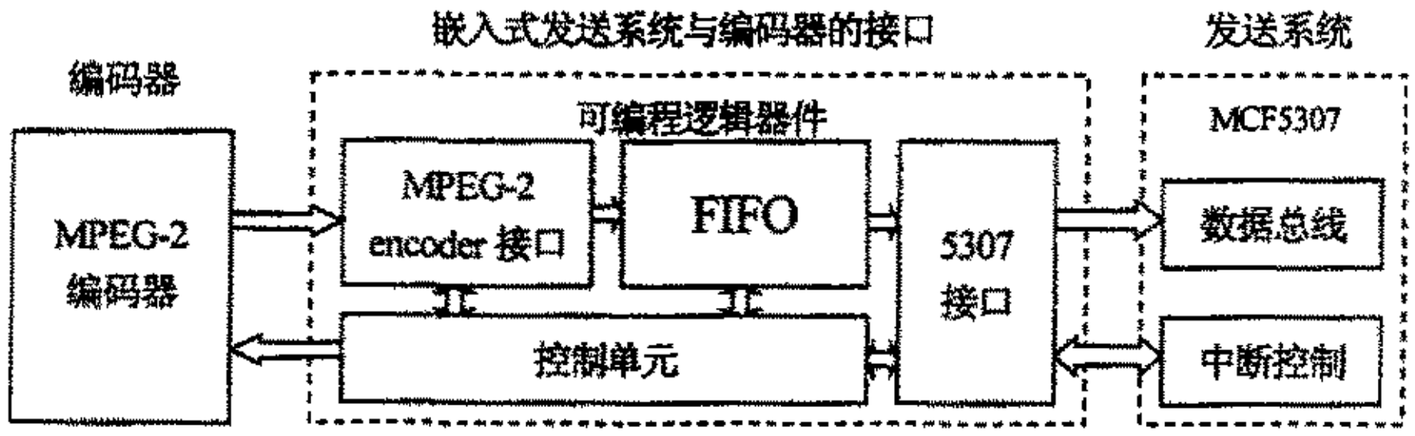

图 5-3 接口原理框图

首先接口开始工作时内部寄存器和进行先进先出缓冲初始化，如清除中断寄存器、数据有效信号等；然后接口开始检测和编码器的接口，测试是否有有效的 MPEG-2 TS 数据分组输出，如果未能检测到有效地数据则向上层系统发送错误

信号；检测到有效数据后，MPEG-2 编码器接口输出 TS 数据同步时钟并读入 TS 数据分组。读入的数据流在控制单元的调度下按照 TS 分组的方式送入 FIFO 缓冲区。控制单元对缓冲区的状态进行监控，FIFO 中存储的数据达到一定范围时向嵌入式系统发出中断信号通知系统读取数据。发送系统得到中断信号通知后迅速作出反映，通过 MCF5307 的数据总线接口读取 FIFO 中缓冲的数据。

§5.2.2 缓冲区和控制单元

在接口的几个组成部分中，编码器接口和 5307 接口可以跟据有关技术文档进行设计，在有关编码器设计一章中已经对编码器输出数据格式做过较多的讨论，可以作为参考。本小节只对缓冲区和控制单元设计中的一些问题作探讨。

缓冲区和控制单元作为嵌入式系统平台和编码器接口的主要部分，需要注意几个问题：需要确定 FIFO 缓冲区的尺寸、读写方式、溢出处理等。控制单元主要负责控制缓冲区读写，缓冲区内数据达到设定值后产生外部中断。

确定 FIFO 缓冲区尺寸：编码器以 TS 分组的方式输出数据，每个数据分组的长度都是以固定 188 字节（1504bit）的速率传输。不论何种情况下 TS 分组的完整性必须保证，所以缓冲区一个读写单元的大小必须是以 TS 分组长度 (188 字节)为单位。由于在嵌入式系统中运行的操作系统是 uCLinux，系统对外部中断的响应，调用中断服务程序，以及在中断服务程序中对数据的读取都需要有一定的时间，而在此同时编码器也以固定速率向缓冲区输出数据。在设计中缓冲区速率的读取速率大于写入速率，所以在开始读取数据后不会有写入指针追上读取指针的情况（溢出）。缓冲区尺寸计算公式：

$$SIZE = V_w T_i + D$$

$SIZE$ ：缓冲区尺寸
V_w ：缓冲区写入速率
T_i ：中断响应时间
D ：每次中断读出数据量

在以上公式中，每次中断读出数据 D 通常应该取 188 字节的整数倍，在器件容量允许的条件下应该取尽可能大的值；中断响应时间 T_i 微秒级。缓冲区读取和写入速率根据实际情况决定。缓冲区尺寸 $SIZE$ 应该至少大于 188 字节。接近一个以太网最大传输单元的 D 是比较理想的选择。

缓冲区的读写：编码器输出的数据流是连续的字节流，系统上电或重启后进入缓冲区的第一个字节的数据应该是 TS 同步字节。发送系统从缓冲区读取的数据是 188 字节的整数倍，并且第一个字节也必须是 TS 同步字节。

溢出处理：当发送系统长时间不能够把缓冲区内的数据读出，而编码器不停的输出数据就会使缓冲区产生溢出。在发生溢出时，先进先出缓冲区应该首先抛弃最先输入的数据。被丢弃的数据尺寸应该是以 188 字节为单位，这是因为一个 TS 分组中只要有一个字节的数据被丢弃就可使整个数据包无效。发生缓冲区溢出时，修改缓冲区读取指针时期向前移 188 字节。发送系统一次读取数据的量是固定的，每次读取都由中断通知不会发生下溢出。

控制单元监视缓冲区状态，判断缓冲区内的数据是否达到 D 字节以产生中断通知外部系统，缓冲区溢出处理。同时还要监视编码其输入的数据是否有效，防止无效数据进入缓冲区。把上层系统返回反馈信息传给编码器等。

在接收端中使用的接收系统于解码器的接口设计原理与本节的描述十分类似，不再另立章节讨论。

§5.3 构建嵌入式系统开发平台[28-29]

建立嵌入式系统开发平台需要使用万禾网络公司开发的嵌入式 Linux 系统开发包 Wanhe-uCLinux-coldfire 和交叉编译工具包 m68k-elf-tool，在开发主机中安装工具包后，即可编译 uCLinux 内核、建立文件系统、对应用程序进行开发和调试。本节就开发平台的安装和设置的过程作一介绍。

§5.3.1 安装开发系统

Wanhe-uCLinux-coldfire 以 uCLinux 为基础，由 www.uCLinux.org 发布的一个支持 Motorola 公司嵌入式 CPU——ColdFire5307 的版本改良而来，其 Linux 核心版本为 2.0.38。在本例设计中建立的开发平台的模型如下图所示。

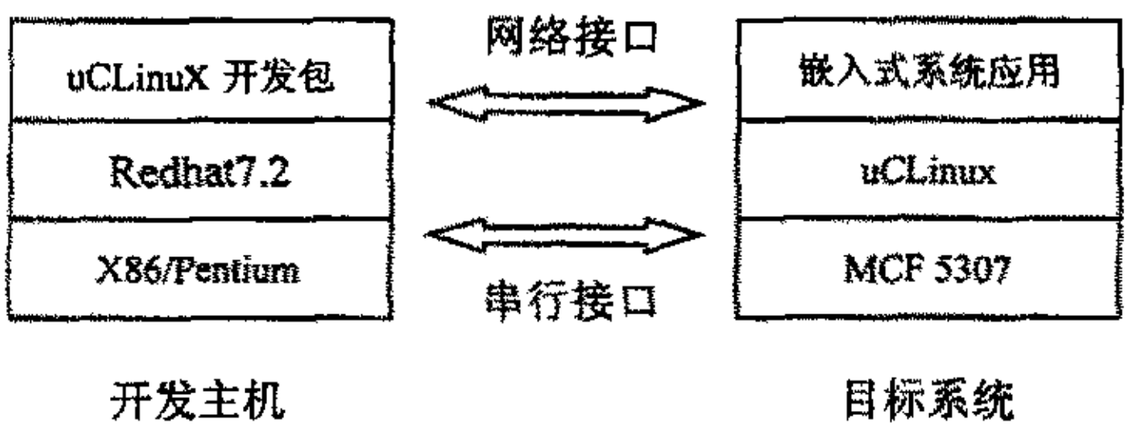

图 5-4 开发平台的安装模型

在以上的安装模型中，X86 主机上的操作系统是 RedHat7.2，uCLinux 开发包和交叉编译工具都在安装在 RedHat7.2 中运行。目标系统的底层硬件以 MCF5307 为基础，目标系统中运行的 uCLinux 系统是在主机中用交叉编译工具对 uCLinux 源代码进行编译通过串行接口或网络接口下载到目标系统中。嵌入式系统应用指高层的应用软件。

安装步骤：

1. 以超级用户登录到主机上。
2. 运行 mount 命令装载 CDROM

   # mount -t iso9660 /dev/cdrom /mnt/cdrom
3. 在/home/目录下安装 Wanhe-uCLinux-coldfire 源代码

   #cd/home

   # tar zxvf /mnt/cdrom/wanhe-Linux/software/Wanhe-uCLinux-coldfire-1.0.tgz
4. 在/usr/local/bin/目录中安装 m68k-elf-tool 工具包

   #cd /;tar zxvf /mnt/cdrom/wanhe-Linux/software/m68k-elf-tools-20010228.tar.gz
5. 运行 umount 卸载 CDROM

   #umount /mnt/cdrom

安装完成后，uCLinux 源代码在/home/uCLinux-coldfire/目录下，编译 uClinux 系统所需要的编译工具都在/usr/local/bin/目录下。

§5.3.2 设置开发系统

因为开发系统本身不是一个集成环境，所以许多设置都要进行手工设置，使用文本编译器对一些配置文件进行修改。需要设置底参数包括目标系统和主机系统的 IP 地址，串行接口波特率设置等。下面是一些参数设置的过程：

1. 以超级用户登录主机，设置主机 IP 地址。

   #ifconfig eth0 192.168.168.100 netmask 255.255.255.0 up
2. 设置串行口设备访问权限，允许普通用户访问串行接口。

   #chmod 0666 /dev/ttyS0
3. 设置目标系统与主机串口。

   $vi /home/uCLinux-coldfire/common.mk

   CONSOLE_BAUD_RATE = 38400

COMM_PORT=/dev/ttyS0

Esc:wq

4. 设置目标系统 IP 地址。

方法一：通过串行口设置目标系统 IP 地址。把目标系统和主机用串行接口连接后运行下列命令：

$make slink

在系统显示 ">"提示符后输入

>ifconfig eth0 192.168.168.123 netmask 255.255.255.0 up

>route add –net 192.168.168.0 eth0

方法二：修改配置文件和启动脚本。

$vi /home/uCLinux-coldfire/common.mk

TARGET_IP=192.168.168.123

Esc:wq

$vi /home/uCLinux-coldire/venders/generic/big/rc

ifconfig eth0 192.168.168.123 netmask 255.255.255.0 up

route add –net 192.168.168.0 eth0

Esc:wq

§5.3.3 配置和编译 uCLinux 内核及应用程序

要使 uCLinux 系统和应用程序能够在目标系统中运行实现预定的功能，必须对 uCLinux-Coldfire 内核进行配置，然后使用编译工具对整个内核系统和应用程序进行编译连接，生成一个能够在目标系统中运行的二进制影象文件。最后使用 sf 工具把这个可执行的二进制影象文件通过以太网口或者串行口下载到目标系统中的 SDRAM 中运行或写入到 FLASH memory 中。

配置内核就是要决定把那些必要的功能编译进内核，忽略其不必要的功能。Linux 内核可以灵活配置，这一特性对提高嵌入式系统性能非常有益。嵌入式系统的功能针对性强，内存和处理能力有限，通过适当配置内核把需要的功能编译进内核，有效减小内核占用内存的大小提高微处理器的效率。在本设计中，对 Linux 内核配置时只选择 MCF5307、FLASH、SDRAM、网络设备、TCP/IP、调试工具等，以及本设计的驱动程序和应用软件。配置使用的命令如下：

$cd /home/uCLinux-coldfire

$make config

编译内核命令：

$make clean

$make dep

$make

编译程序将按照配置文件对内核和应用程序源代码编译，生成可执行二进制影象文件image.bin和imagez.bin，其中imagez.bin是经过gizp程序压缩的影象文件。完成编译后把目标系统和主机连上，可以主机中运行make sload通过串行口下载到目标系统，或者使用make netload通过网络下载到目标系统。然后可以在登陆目标系统后运行flashw /ramdisk1/imagez.bin把影象文件写入存储器。重新启动目标系统就可运行新编译的操作系统和应用程序。

至此一个嵌入式系统开发平台建立完成，已经可以在这一平台上开发调试应用程序。

§5.3.4 调试应用程序

在第四章中对调试工具 GDB 和远程调试原理做过简单介绍，本小节给出说明如何在主机和目标系统之间建立调试。在 uCLinux-coldfire 系统调试中使用的调试器分为两部分：在目标系统上运行的服务器 gdbserver 和在主机中运行的客户机程序 gdb。在主机上运行的 gdb 版本是 4.18 版本程序名称为 m68k-gdb-418。

调试时首先要把主机和目标系统用串行口和以太网连接。比如要调试目标系统中的应用程序/bin/tssever 程序，需要使用 gdbsever 来运行该程序并为它指定一个未被占用的端口。在目标系统中运行如下命令：

>gdbserver :3000 /bin/tssever

此时目标系统的调试服务器已经就绪，等待主机上的调试器的调试命令。然后在主机应用程序 tsserver 源程序目录下运行调试器：

$/home/uCLinux-coldfire/tools/bin/ m68k-gdb-418　tssever.gdb

出现(gdb)提示符时输入：tartget romte 192.168.168.123：3000

即可完成主机和目标系统调试器的连接，接下去就可以对应用程序进行调试。

§5.4 编码器设备驱动程序[29,30,47~51]

在 Linux 系统中绝大多数的硬件设备，如打印机，调制解调器都是作为一个特殊的设备文件来使用和管理的。本节就以创建编码器设备驱动程序的过程加以介绍。

§5.4.1 驱动程序初始化过程

在 Linux 系统中一个启动时进行初始化的设备驱动程序的初始化过程如下图所示，首先 Linux 系统转入核心之后，会创建一个 init 进程通过系统调用 sys_setup 初始化内存，调用 device_setup 初始化系统内核支持的字符设备、块设备、以及实际存在的网络设备。本设计中的编码器设备就在系统进行字符设备初始化时调用 mpegdev_init 函数进行初始化的。

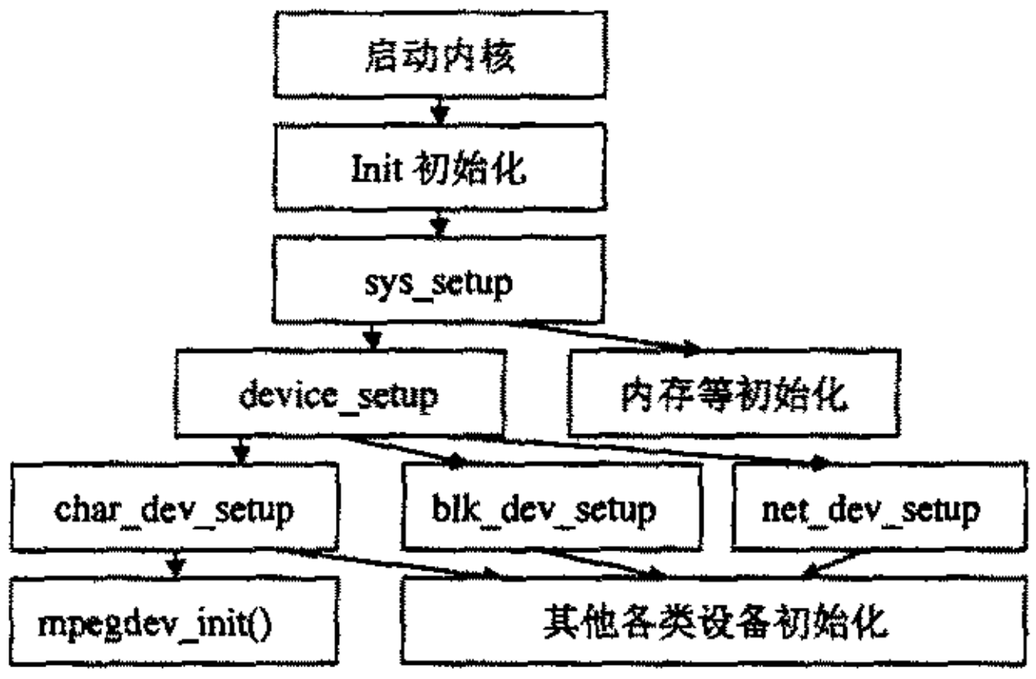

图 5-5 系统启动时的设备初始化过程

在 uClinux 环境下设计驱动程序基本上和普通 Linux，如 RedHat 系统环境下的驱动程序设计没有大的区别。在 uClinux 系统中也把驱动程序分成三类：字符设备驱动程序、块设备驱动程序和网络设备驱动程序。在设计编码器设备的驱动程序时，由于编码器设备只是顺序输出字节流不需要进行随机存取，因此只把把它当作一个字符设备来考虑。

§5.4.2 uCLinux 系统下驱动程序设计

1. 在合适的设备驱动程序目录下面建立驱动程序文件，以 mpegdev 作为设备名并在 uCLinux 目录/home/uCLinux-coldfire/linux/drivers/char 下建立相关的设备驱动程序源文件，mpegdev.h 和 mpegdev.c 等。
2. 修改 uCLinux 的设备配置文件，保证在系统编译配置时会出现相关的选项，在这里需要在文件/home/uCLinux-coldfire/linux/arch/m68knommu/config.in 中

适核心配置部分加入 bool 'ColdFire mpegdev support' CONFIG_MPEGDEV。

3. 在 uCLinux 的设备 Makefile 中加入编译控制代码，使新的设备驱动程序可以正确地编译。本例中需要在/home/uCLinux-coldfire/linux/drivers/char/Makefile 中加入编译控制语句：

```
ifeq($(CONFIG_MPEGDEV),Y)
L_OBJS+= mpegdev.o
endif
```

4. 在系统设备初始化函数中注册设备驱动程序，使系统在启动时能够正确的对新的设备进行初始化工作。在新的设备驱动程序源文件中，如果初始化函数名为 mpegdev_init()，为使该函数能被系统调用则需要在目标系统的设备初始化控制函数/home/uCLinux-coldfire/linux/drivers/char/mem.c 中加入语句：

```
#ifdef CONFIG_MPEGDEV
mpegdev_init();
#endif
```

5. 为新的设备建立设备文件以供应用程序访问。这一步在 uCLinux 环境中和普通 Linux 中有较大的区别，通常在 Linux 环境下要用 mknod 命令来建立一个设备文件，而在 uCLinux 环境下只要在设备文件目录下建立一个特殊命名的普通文件即可，该文件的命名规则为“@文件名，设备类型，主设备号，次设备号”。在系统重新编译后，编译控制程序会生成二进制文件时，把该文件拷贝到目标系统的/dev 目录下作为新设备的驱动程序。如在/home/uCLinux-coldfire/romfs/dev 目录下建立的文件“@mpegdev,c,38,0”在系统编译后，并下载到目标系统中可以发现在设备驱动文件目录/dev 下有设备文件 mpegdev。

§5.4.3 驱动程序中的主要函数

在一个 uCLinux 的设备驱动程序包含初始化函数、设备打开操作函数、设备关闭操作函数、读操作函数、写操作函数、设备控制函数等，如果需要中断操作还需向系统注册中断处理函数。初始化函数需要在 mem.c 文件中向系统注册初始化函数，其他操作函数以函数指针的形式保存在一个称为 file_operations 的结构中，在初始化函数中通过调用 register_chrdev()函数向系统注册，系统内核根

据这个 file_operations 结构跳转表索引调用驱动程序中的正确函数。用户应用程序中就是通过系统调用来实现对驱动操作函数进行操作。

file_operations 结构：

```
struct file_operations{
    int (* lseek)(struct inode * inode, struct file * filp, off_t off, int pos);//设备定位
    int (* read)(struct inode * inode, struct file * filp, char * buf, int count);//设备读
    int (* write)(struct inode * inode, struct file * filp, char * buf, int count);//设备写
    int (* readdir)(struct inode * inode, struct file * filp, struct dirent * dirent, int count);
    int (* select)(struct inode * inode, struct file * filp, int sel_type, select_table * wait);
    int (* ioctl)(struct inode * inode, struct file * filp, unsigned int cmd, unsigned int arg);
    int (* mmap)(void);
    int (* open)(struct inode * inode, struct file * filp);//设备打开
    int (* release)(struct inode * inode, struct file * filp);//设备关闭
    int (* fsync)(struct inode * inode, struct file * filp);
};
```

在这个结构中定义设备的定位、读、写、IO 控制、打开、关闭等相关函数调用的跳转地址，在本编码器设备中的定义为：

```
struct file_operations mpeg_fops = {
    NULL,
    mpeg_read,
    NULL,
    NULL,
    NULL,
    mpeg_ioctl,
    NULL,
    mpeg_open,
    mpeg_release,
                //nothing more, fill with NULLs
};
```

在编码器设备驱动程序中的函数：

- 初始化函数（mpegdev_init()）

在该函数中调用子函数检测设备是否存在，对设备进行初始化设置，调用系统函数 request_region() 向系统注册设备 IO，调用系统函数

register_chrdev()向系统注册新设备。该函数在<linux/fs.h>中申明：

```
int register_chrdev(unsigned int major,//主设备号
                    const char *name, //设备名称
                    struct file_operations *fops);//驱动程序操作函数跳转表
```

可以在初始化函数中向系统申请中断，但是在一个系统的中断资源非常有限，而只在设备打开时向系统申请中断并注册中断处理函数。在编码器驱动程序中，编码器中断注册在 mpeg_open()函数中进行。

- 设备打开操作函数

```
int mpeg_open (struct inode *, struct file *);
```

这个操作是在设备节点上的第一个操作，打开设备使编码器设备驱动程序开始工作，通常在这里向系统注册中断函数申请内存空间等。如果不使用该函数，即向系统注册时在 file_operations 中对应字段填 NULL，使设备打开总是成功的，但系统不会通知驱动程序。

- 设备关闭操作函数

```
int mpeg_release (struct inode *, struct file *);
```

当设备节点被关闭时，调用这个操作函数，可以在此实现释放中断和存储空间、关闭设备等操作。和设备打开操作函数一样也可以是 NULL。

- 读操作函数

```
int mpeg_read (struct inode *, struct file *, char *, int);
```

用来从外部编码器设备中读取数据。如果没有这个函数，当函数返回-EINAL（非法参数）表示操作出错。用户程序对设备文件 mpegdev 进行读操作时系统调用该函数。

- 写操作函数

```
int (*write) (struct inode *, struct file *, const char *, int);
```

系统不需向编码器设备发送数据，未定义。当函数返回-EINAL（非法参数）表示操作出错。用户对设备文件写操作时系统调用该函数。

- 设备控制函数

```
int (*ioctl) (struct inode *, struct file *, unsigned int, unsigned long);
```

系统调用 ioctl 时提供的一种设备相关命令的方法。在使用系统调用函数

ioctl 时系统调用该函数来实现对设备的控制功能。比如发送端程序根据接收端反馈的信息对编码器的输出格式、码率、分辨率等进行反向控制等。

- 中断处理函数

void mpeg_interrupt (int irq, void * dev_id, struct pt_regs * regs);

中断产生时系统调用该函数来对中断事件进行处理。如对设备进行读取或写入数据等。在耗时较多的中断处理程序中还要调用下半部（bottom half）来保证下一次中断不会被忽略。因为系统中断资源非常有限，所以必须在设备驱动程序被调用之前向系统注册中断处理函数，而在设备被关闭前注销。注册和注销中断分别使用<linux/sched.h>中申明的函数：

```
int request_irq ( unsigned int irq, //Linux 中断号，传递给处理函数
                  void (*handler) (int, void*, struct pt_regs *), //中断处理函数
                  unsigned long flags, //选项掩码，可写 0
                  const char *device, //中断设备名，字符串
                  void *dev_id); //传递给处理程序的可自由使用的指针
void free_irq (unsigned int irq, void *dev_id);
```

在 uCLinux 系统中不支持模块化安装设备驱动程序，因此在设备驱动程序中不需要要有 init_module()和 cleanup_module()函数。

§5.4.4 编码器设备的使用

应用程序对编码器设备进行操作是必须通过文件系统的调用来实现。如果编码器设备文件为/dev/mpegdev，则在应用程序中使用编码器时，首先通过文件操作打开设备文件，然后使用读文件命令从编码器设备文件中读取数据。如打开设备文件 mpegdev 可以使调用文件打开函数 fp=fopen("/dev/mpegdev", "r")等，关闭设备文件可使用 fclose(fp)，用 fgets，fread 等函数从设备中读取数据，用 fputs，fwrite 等函数向设备写入数据。

§5.5 软件设计

本节描述和讨论实时视频传输系统的发送端和接收端应用软件设计。在发送端，嵌入式接收端和 Windows 接收端软件中有关 MPEG-2 编码器驱动程序的设计部分已经单独在 5.4 节做非常详尽的介绍，不在本节讨论。

§5.5.1 发送端软件

发送端软件运行于uCLinux平台实现从MPEG-2编码器驱动程序中读取实时TS 分组数据，按照一定格式封装成固定长度的数据分组，用单播、多播或广播的方式发送到局域网中。发送端软件在嵌入式系统启动后立即开始运行。发送端软件基本流程如图 5-6 所示。

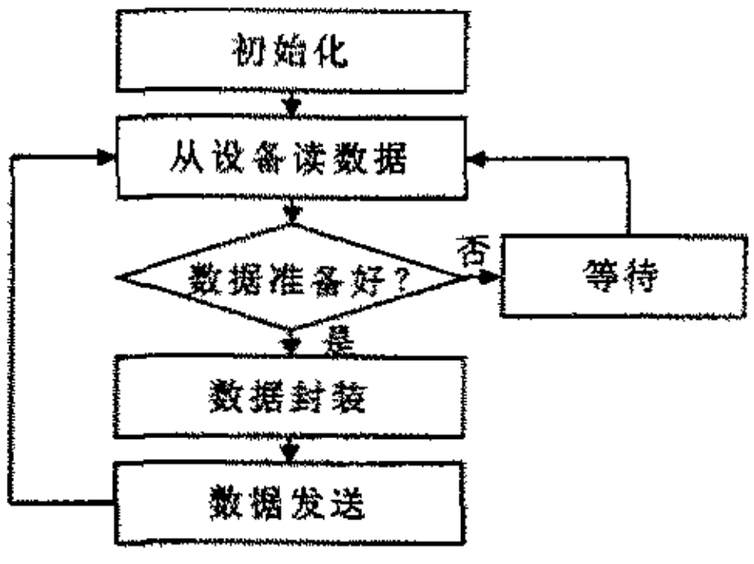

图 5-6 发送端软件基本流程

发送端软件开始工作时首先按照预先设定的目的 IP 地址和端口号建立一个用于数据发送数据报（UDP）套接字，并产生一个随机数作为第一个 RTP 分组的顺序号。然后打开编码器驱动程序设备文件读取数据存入发送缓冲区。当缓冲区中的 TS 分组数据达到一定长度后（188×7 字节），把数据填入 RTP 分组中调用 UDP 发送程序通过套接字发送到网络中。

实时编码器送出的 TS 分组流是按照实际编码的速率输出，因此在发送端软件中不需要重新对 TS 分组的发送进行定时，直接按照从编码器接收数据的输率发送数据包到网络中。只要及时从编码器设备中取出数据不造成缓冲区溢出，就可以保证发送端发送 TS 分组数据是实时的。

§5.5.2 嵌入式接收端软件

嵌入式接收端种运行的软件功能上完全和发送端相反，它从网络中接收数据分组，并验证确定数据是否从有效发送端发送的数据。然后把有效的数据通过解码器驱动程序写入 MPEG-2 解码器。嵌入式接收端软件基本流程如图 5-7 所示。

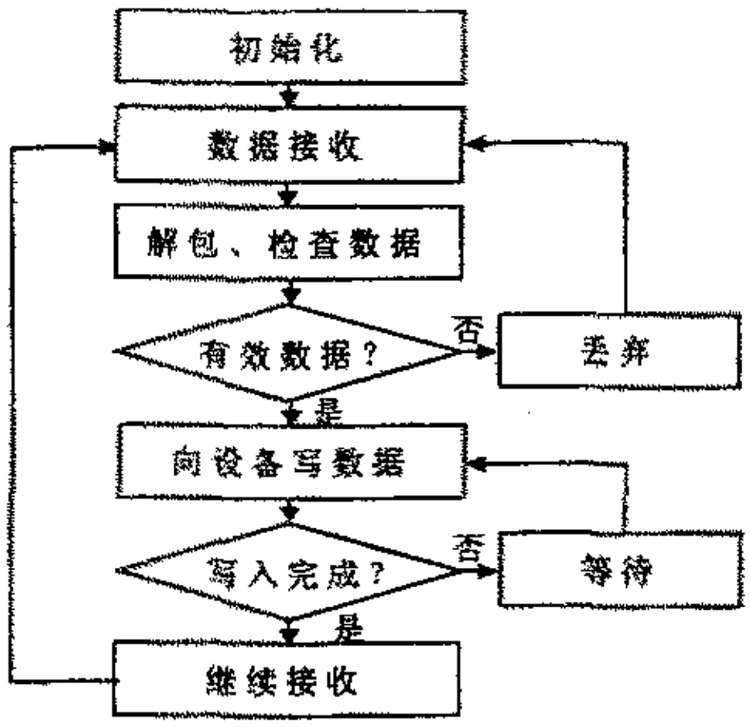

图 5-7 嵌入式接收端软件基本流程

嵌入式接收端软件在嵌入式系统启动后立即开始运行，首先初始化打开解码器设备，建立一个数据报套接字监听是否有数据从特定的端口输入。当收到数据后对数据报文进行有效性检查，若是有效数据报文则取出有效 TS 分组数据向解码器驱动程序写入数据。无效数据则被丢弃。和发送端一样嵌入式接收端的软件也是一个无限循环的程序。

§ 5.5.3 Windows 和 Linux 接收端软件[52]

Windows 和 Linux 接收端软件分别运行于 Windows 和 Linux 系统平台，能够实现从局域网中接收并播放单播、广播、多播形式 MPEG-2 TS 数据分组。在本文的设计中采用一个遵循 GPL 声明的客户端播放软件——VLC 作为 Windows 和 Linux 接收端的播放软件，它具有跨平台运行的特点，可以运行在 Linux，Windows，Unix 等系统平台上，可移植性非常好。该软件的模块结构如图 5-8 所示。

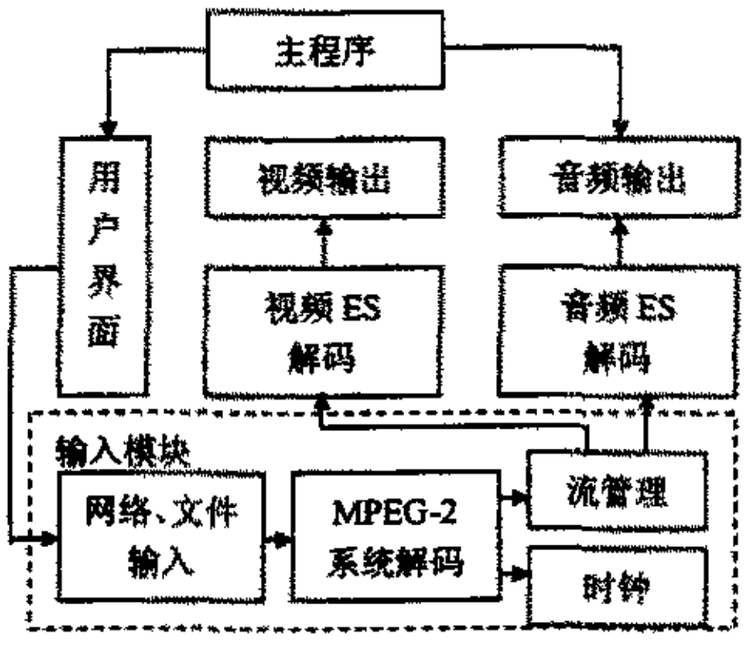

图 5-8 VLC 软件模块结构

主程序初始化后显示用户界面，在用户界面上选择网络输入方式，设置好网

络连接方式和监听端口号，如果此时有正确的 MPEG-2 数据流输入，MPEG-2 系统解码器立即开始工作把复合的 TS 数据流解复用生成视频 ES 流和音频 ES 流。然后在流管理模块的控制下视频 ES 和音频 ES 分别输入对应的 ES 流解码器。视频输出模块把解码后的视频数据转换成 RGB 格式输出显现示。音频输出模块负责把解码后的音频输出到声卡。

VLC 软件能够独立接收并解码 MPEG-2 TS 流广播，但还不能够满足本设计的需要：它缺少对非视频数据分组的过滤能力，对相同地址输入的无效数据分组完全不加识别。无效数据很快会使 VLC 内置接收缓冲溢出，解码失败。而且 VLC 的接收模块本身没有提供对 RTP 的支持，对数据源的识别只依赖于网络地址功能有限。因此为实现实时多媒体数据流的安全传输，需要在外部网络接口和 VLC 解码软件之间设计一层分组过滤模块用来滤除无效数据分组，以及对 RTP 协议的支持。该层软件结构如图 5-9 所示。

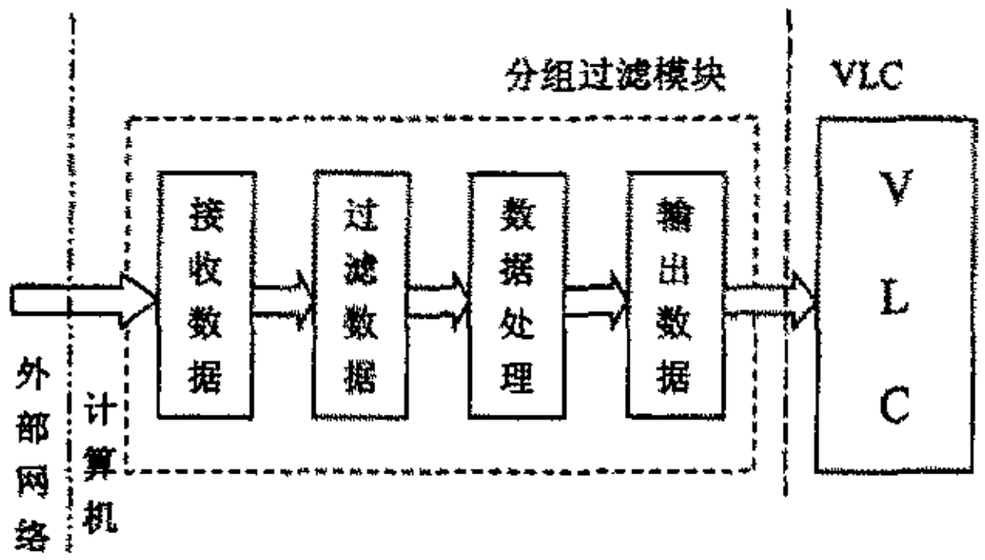

图 5-9 分组过滤模块

分组过滤模块原理：建立两个网络连接套接字，一个用于监听设定的端口接收外部网络数据分组；另一个用于向本机回环地址的新端口发送过滤后的数据分组。过滤程序从外部网络中接收数据分组后，判断数据分组是否来自设定发送端，以及是否包含有效 MPEG-2 TS 数据，丢弃不合要求的数据。然后对数据分组进行进一步处理，比如增加对 RTP 协议支持。最后向本机回环地址发送经过处理的 TS 分组数据。

在增加分组过滤模块之后，VLC 须监听分组过滤模块重定向后的端口实现来实现网络 MPEG-2 TS 的播放。

§5.6 小结

本章描述了基于嵌入式 Linux 的实时多媒体信息传输系统的框架，介绍了其中几个主要软硬件模块子系统的功能设计，如编码器接口、编码器驱动程序、发送端和接收端程序 、建立嵌入式开发平台等。基于 Linux 系统接收部分的基本功能已经实现，能够以单播，多播，广播方式传输 MPEG-2 TS 进行接收和播放。一个用软件模拟的信源也已经完成几乎可以不需更改代码就可很快的移植到嵌入式 Linux 平台之上。由于实际中面临的一些因素，比如嵌入式开发平台的网络接口功能限制，部分功能由于时间因素将有待于将来去完成。

第六章 总结和展望

§6.1 总结

本文应用 MPEG-2 编码技术、IP 传输技术、嵌入式 Linux 等技术设计一个基于嵌入式 Linux 系统的实时多媒体传输系统框架，并对组成该系统的重要模块设计进行探讨，其中的 MPEG-2 编/解码器、基于嵌入式 Linux 的发送端软件、基于 Linux 系统的接收软件都已经实现其功能，为以后开发出完善的应用系统作初步设计。本设计不仅是对多媒体传输技术，也是对嵌入式 Linux 技术应用的有益尝试。

由于设计中采用了一些新的技术，使得这个系统将成为一个具有一定性能价格优势的实际系统，例如系统中采用的 MPEG-2技术在压缩图像的清晰度，编解码延时，实时性保证等方面具有压倒性的优势；嵌入式 Linux 技术的采用使得系统的可靠性，可扩展性大大增强，而且实际系统成本更低。这个实时多媒体传输系统经过进一步完善后将可以在很多场合得到应用，如远程视音频监控，远程教学，商务电视转桌面应用等，具有实际应用价值。

§6.2 展望

本文设计的基于嵌入式 Linux 的实时多媒体信息传输系统框架已经基本成型，但在系统设计中还有许多不足之处有待进一步完善。

1. 多媒传输系统的安全性问题。尽管对数据分组来源的有效性进行识别，但这仅仅是简单的判断数据源标识和数据类型，无法完全阻止网络中的意外误传的数据分组和恶意攻击。另外本系统对接收用户群没有进行有效控制，任何能够取得数据传输源/目的地址的用户都能够接收多媒体广播。下一步应该考虑采用加密技术和用户认证技术。
2. 对网络数据分组丢失的处理。在 IP 网络中这个问题始终存在，本设计中并没有对网络数据分组的丢失进行处理，只是简单的交给解码器解决。尽管在局域网中分组丢失率很小，忽略分组丢失问题不会对视频质量造

成太大的影响，但对一个可靠的多媒体视音频传输系统来说，不能作为一个最终的解决方案。许多参考文献中都提出各种各样的解决方案，如采用信源信道联合编码，多层次描述编码，错误隐藏技术等。将来可以考虑采用一些纠错或错误隐藏技术来增强该系统的容错能力。

3. 对 RTP 协议的支持。本系统中尽管对 RTP 协议作一些简单的支持，如提供分组的顺序号，同步源 SSRC，定期发布发送者报告等，但并没有完全实现 RTP 协议的所有功能，也并没有完全遵循该协议。
4. 在第二章就指出传输 TS 分组不是最优解决方案，本设计中采用该方案是考虑当前采用的嵌入式处理器的处理能力作出的，而且在局域网中传输 TS 分组效果良好。但更好的传输方法是采用纠错和错误隐藏技术的捆绑 MPEG-2 视音频 ES，或采用下一代编码技术。
5. 对编码速率的反馈控制问题。在局域网中带宽能够得到保证，该问题不突出，本文中未加以考虑，但进一步的设计中应该考虑信道状态对信源输出速率的控制问题。

参考文献

[1] Prabhat K. Andleigh & Kiran Thakrar 著，徐光佑，史元春 译，多媒体系统设计，第一版，电子工业出版社出版，1998.9

[2] 朱秀昌，宋建新 编著，多媒体网络通信技术及应用，第一版，电子工业出版社出版，1998.2

[3] 张明敏主编，网络多媒体技术与应用，第一版，清华大学出版社，1998.7

[4] Uyless Black 著，现代通信最新技术，第二版，清华大学出版社，1998.4

[5] 林福宗 编著，多媒体技术基础，第一版，清华大学出版社，2000.8

[6] Real system资料，http://www.realnetworks.com/resources/documentation/index.html

[7] Windows Media资料，http://www.microsoft.com/china/windows/windowsmedia/overview/introwmt7-1.asp

[8] QuickTime资料，http://www.apple.com/quicktime/products/

[9] 流媒体技术资料,流媒体论坛 http://www.liumeiti.com/forum/myforum.asp

[10]叶华 谢玮 梁勇 崔进水 编著，IP电话/传真技术，第一版，人民邮电出版社，2000.6

[11]沈兰荪 著，图像编码与异步传输，第一版，人民邮电出版社，1998.5

[12]钟玉琢 乔秉新 祁卫 译，运动图象及其伴音通用编码国际标准—MPEG-2，第一版，清华大学出版社，1997.6

[13]胡国荣 编著，数字视频压缩及其标准，第一版，北京广播学院出版社，1999.12

[14]张琦 杨盈昀 张远 林正豹 编著，数字电视中心技术，第一版，北京广播学院出版社，2001.3

[15]ITU-T Rec.H.262|ISO/IEC 13818-2: 1994 Information technology-Coding of moving pictures and associated audio-Part 2:video, Telecommunication Standardization Sector of ITU, 1995.7

[16]巴继东 编著，IP技术与综合宽带网，第一版，北京邮电大学出版社，2000.3

[17]胡道元 编著，计算机局域网，第二版，清华大学出版社，1996.12

[18]张公忠 主编，现代网络技术教程，第一版，电子工业出版社，2000.1

[19]Larry L. Peterson Bruce S. Davie 著，叶新铭 贾波 吴承勇 等译，计算机网络，第一版，机械工业出版社，2001.6

[20]罗明宇 陶孜谨 卢锡城，RTP在网络视频传输中的实现研究，计算机工程，2000.9

[21]严峻 马小骏 顾冠群，RTP协议的研究与实现，计算机工程与应用，2000.9

[22]张占军 韩承德 杨学良，多媒体实时传输协议RTP，计算机工程与应用，2001.4

[23]张锦 史小宏，视频会议系统地RTP/RTCP实时反馈机制研究，海南大学学报自然科学版

[24]谢洪胜，毛迪林，黄晓霖，高传善，RTP和TCP在实时传输中的比较，微型电脑应用，2000.7

[25]顾尚杰 薛质 编著，计算机通信网络基础，第一版，电子工业出版社，2000.9

[26]MPEG2 1chip Audio/Video Encoder MB86390A Product Specification, Rev1.4, FUJITSU LIMITED, 2000.9

[27]MCF5307 Integrated ColdFire Microprocessor User's Manual, Rev.2.0, Motorola Inc, 2000.8

[28]嵌入式技术资料，嵌入式开发网,http://www.embed.com.cn/

[29]嵌入式技术资料,嵌入式Linux中文社区 ,http://www.pocketix.com/

[30]王学龙 编著，嵌入式Linux系统设计与应用，第一版，清华大学出版社，2001.8

[31]怀石工作室 编著，Linux上的C编程，第二版，中国电力出版社，2001.5

[32]李卓桓 瞿华 等编著，Linux网络编程，第一版，机械工业出版社，2000.1

[33]吴绍伟 编著，Linux系统管理，第一版，人民邮电出版社，2002.1

[34]金西 黄汪，Linux操作系统是嵌入式系统新的选择，微计算机信息，2000.6

[35]范质坚，Linux在嵌入式系统中的应用，计算机与现代化，2000.6

[36]陈莉君，Linux内核的分析与应用，西安邮电学院报，2001.3

[37]金西 黄汪，嵌入式Linux技术及应用，计算机应用，2000.7

[38]uCLinux技术资料 http://www.uclinux.org/description/

[39]Chris Jesshope and Yong Qiu Liu, High Quality Video Delivery over Local Area Networks With Application to Teaching at a Distance, Institute of Information Sciences and technology Massey University.

[40]Chunlei Liu, Multimedia Over IP: RSVP, RTP, RTCP,RTSP, http://citeseer.nj.nec.com/

[41]A. Basso, G.L.Cash, M. R. Civanlar, Transmission of MPEG-2 Streams over Non-Guaranteed Quality of Service Networks, AT&T Labs – Research, 1997.

[42]D.Hoffman, G.Fernando, V.Goyal and M.R.Civanlar, "RTP Payload Format for MPEG1/ MPEG2 Video" draft-ietf-avt-mpeg-new-01, Internet draft, June 1997.

[43]M. Reha Civanlar, Glenn L. Cash, Barry G.Haskell, "RTP Payload Format for Bundled MPEG," draft-civanlar-bmpeg-01, Internet draft, February 1997.

[44]Thomas J. Kostas, Michael S. Borella, Ikhlaq Sidhu, Guido M. Schuster, Jacek Grabiec, and Jerry Mahler, Real-Time Voice Over Packet-Switched Networks, 3COM, 1998

[45]H. Schulzrinne, GMD Fokus, S. Casner, R. Frederick, V. Jacobson, RTP: A Transport Protocol for Real-Time Applications（RFC1889）, Network Working Group, 1996.1

[46]D. Hoffman, G. Fernando, V. Goyal, M. Civanlar, RTP Payload Format for MPEG1/MPEG2 Video（RFC2250）, Network Working Group, 1998.1

[47]赖松林，Linux系统设备驱动程序设计一例，福州大学学报（自然科学版），2001.6

[48]何刚，吴志美，IP组播和INTERNET上的视频广播，计算机研究与发展，1999.10

[49]郭敏，基于Linux的实时音频与视频传输，计算机应用，2000.3

[50]陈健 陈文智，一个应用于嵌入式Linux浏览器的视频播放器，2001.6

[51]Alessandro Rubini 著, Lisoleg译, Linux设备驱动程序, 第一版, 中国电力出版社, 2000.4

[52]VLC技术资料, http://www.videolan.org/vlc/doc.html

攻读学位期间发表的学术论文目录

[1] 张盎然, 陈明华, 杨扬, 基于准同步算法的谐波分析方法, 电测与仪表, 2002年第 1 期

[2] 张盎然, 陈明华, 杨扬,石旭刚,一种 MPEG-2编码器的 设计实现与调试,电视技术,录用待发

致 谢

本文是在导师杨扬教授和石旭刚老师指导下完成的。两年多来，两位老师始终重视对本人各方面能力的培养，不仅向我传授了知识，为我提供了难得的实践机会，更加指导了我获取知识的方法及做人的道理；两位老师渊博的知识，严谨的治学态度是我不断追求的目标；导师的宽厚待人也给我留下了深刻的印象，在此我要向他们表示最衷心的感谢。

感谢光纤通信重点实验室的各位老师对我的教诲和帮助。感谢我的同学和室友在学习和日常生活中给予的帮助。感谢中威公司的各位员工在我实践期间提供的帮助。

衷心感谢所有关心和帮助我的人们！