摘　要

随着社会的进步，工业的发展，人们对通信的要求越来越高，尤其随着网络技术的飞速发展，人们对通信的要求逐渐转向以音视频通信为主的多媒体通信上来，其中 IP 网络视频会议系统是多媒体应用的一个主流方向。在整个系统中多点控制单元（MCU）是会议通信的重要设备， 担负着多点视频会议的控制、管理和处理的任务，其性能优劣直接影响视频会议系统的质量。但是到目前为止，还没有公开的专门用于测试 MCU 性能的方法。针对这个问题，本文讨论并提出了一套能有效测试 MCU 性能的方案。

由于 MCU 性能受多种因素的影响，并且应用传统的方法不容易测试，因此文章提出了一种操作简便的测试方案，同时设计和开发了一个模拟呼叫器设备，用它来模拟多个实终端的呼叫、连接、传送音视频、挂断等功能，并且根据提出的方案实现了 MCU 性能的测试。本文重点设计了模拟呼叫器的软件结构，同时对各功能模块进行了描述和设计；介绍了应用模拟呼叫器测试 MCU 性能的具体过程。

首先，第一章概述中将简要介绍视频会议的概念、发展、标准和任务简介。第二章将扼要介绍视频会议系统的组成和 H.323 协议。第三章介绍 MCU 的实现，包括多点控制和多点处理功能。接下去的第四、五章将介绍测试方案及其实现过程，其中第四章提出测试方案，包括测试需要的设备和具体步骤。第五章是实现过程，包括模拟呼叫器的实现和方案的具体应用及结果。最后在结束语中给出整个任务的总结和展望。

模拟呼叫器开发完成后在实验环境下对其进行了实际检验，符合事先的设计期望，证明了 MCU 性能测试方案的可行性和正确性。而且本论文提出的方案占用资源少，耗费成本低，简单易行，具有很强的实用性。

关键词：视频会议　　多点控制单元（MCU）　　性能　　H.323 协议

Abstract

With the progress of society and development of industry, people have advanced more and more demand for communication. Especially the need of people on communication is converted to multimedia communication because of the fast improvement of the network. IP-Based video conference system is promised to be a trend of future multimedia application. Multimedia Control Unit (MCU) is an important equipment in such systems. It acts as control, management and process of multi-conference. Its performance directly influences the quality of the video conference. But so far, there isn't an open and special method to test the performance of MCU in this field. According to this problem, this paper discusses and puts forward a new scheme that can test the performance of MCU efficiently.

Because the performance of MCU is influenced by many factors and is difficult to test applying to conventional method, a new scheme is designed in this paper. We design and develop an analog caller device that analog many true terminal to test performance of MCU. The emphasis is put on the design the analog caller device's structure, and every module function has been described, and testing detailed steps are introduced.

In the first chapter the concept, the development, the standard of video conference and my task are introduced. In the second chapter I will concisely state the compose and H.323 protocol of video conference. Then the implement of MCU is introduced in Chapter3. Chapter4 and Chapter5 depict the scheme and the implement of the caller. At last, some advices are proposed to give direction to the following work.

The analog caller device in this scheme to test the performance of MCU has passed the test in experiment environment. Practice has proved the feasibility and correctness of the scheme. This scheme is not only simple and feasible, but consumes less money and fewer resources, so this scheme has high practicability.

Key Words: video conference Multipoint Control Unit performance H.323 Protocol

创新性声明

本人声明所呈交的论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果；也不包含为获得西安电子科技大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

本人签名：党薇　　日期：2005.1.17

关于论文使用授权的说明

本人完全了解西安电子科技大学有关保留和使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权单位属于西安电子科技大学。本人保证毕业离校后，发表论文或使用论文工作成果时署名单位仍然为西安电子科技大学。学校有权保留送交论文的复印件，允许查阅和借阅论文；学校可以公布论文的全部或部分内容，可以允许采用影印、缩印或其它复制手段保存论文。

本人签名：党薇　　日期：2005.1.17

导师签名：　　日期：2005.1.17

第一章 视频会议系统概述

在人类通信中，有效性的信息 55%～60%依赖于面对面的视觉效果，33%～38%依赖于说话者的语音，而只有 7%依赖于内容。传统的通信工具，如电话、传真机等，则无法达到面对面或一群人聚集在一起的沟通效果。特别是随着现代通信技术、INTERNET 以及计算机技术的飞速发展，当今信息社会对通信提出了更高的要求，人们已不满足于简单的语音和文字通信，而是希望能够在任何时间（whenever）、任何地点（wherever）与任何个人（whoever）实现图像、文字、声音等信息的实时交互（exchange），充分享受丰富多彩的多媒体通信服务。这些需求促使视频会议系统应运而生。视频会议系统通常又被称为多媒体会议系统，是融计算机技术、通信技术、网络技术为一体的产物。它将各种媒体信息数字化，利用各种网络进行实时传输和交换，使用户享受到真实感强、交互性好的信息服务，充分体现了信息社会的特征—获取信息的高效性和直观性。

随着改革的不断深入，视频会议作为一种现代化的多媒体通信工具，在及时召开重要会议、做出重要决策、发布重要信息、提高工作效率、节约时间和经费开支等方面的重要价值已被人们所认识，并迅速向新闻、教育、航天、政府、医疗等各个领域漫延，其应用领域也已超出“会议”的范畴，在技术培训、生产调度、远程监视、紧急援助、商务谈判、远程教学、远程医疗等方面都具有广泛的应用价值。

1.1 视频会议系统的概念

视频会议亦称视讯会议、会议电视，是利用计算机技术和通信设备通过传输信道在两点或多个地点之间建立可视多媒体通信，实现语音、图像和数据交流的一种会议形式，是集通信、计算机技术于一体的远程异地通信方式。与会者既能看到对方发言人和会场场景，也能听到对方的声音，若辅以电子白板、书写电话、传真机等通信设备，可以实现与对方会场的与会人员进行磋商，在效果上完全可以代替现场会议。它是通过网络通信技术来实现的虚拟会议，使在地理上分散的用户可以共聚一处，通过图形、声音等多种方式交流信息,支持人们远距离进行实时信息交流与共享、开展协同工作的应用系统。视频会议极大地方便了协作成员之间真实、直观的交流，对于远程教学和会议有着举足轻重的作用。

1.2 视频会议系统的发展历程

视频会议系统的历史可追溯到本世纪 60 年代初，发达国家开始进行会议电视的研究。80 年代末、90 年代初，随着微电子、计算机、数字信号处理及图像处理技术的发展，会议电视的理论研究和实用系统研制方面也得到了迅速发展。总的说来，其发展主要经历了模拟视频会议、数字视频会议和国际统一标准的数字视频会议三个阶段。

60 年代至 80 年代是模拟视频会议发展阶段。由于所传送的信号为模拟信号，所需的传输带宽高，使得价格相当昂贵，因此并没有广泛的大规模的应用。 进入 70 年代以来，由于视频会议相关技术领域的长足进步，最主要是数字式传输的出现，传统视频会议系统所用模拟信号的采样或传输方法也得到极大的改善，数字信号处理技术开始走向成熟。从总体看，70 年代视频会议系统的发展处于相对平静的时期，但视频会议系统的研究工作并未中断。进入 80 年代中期，通信科技发展迅猛，编码和信息压缩技术的发展，使得视频会议设备的实用性大为提高。但此时的视频会议系统由于价格和技术的因素，仍只限于高档的会议室视频会议系统的应用，从而限制了视讯会议系统的进一步普及。

90 年代初期至 1995 年是 ISDN 网络上数字视频会议的发展阶段。由于视频压缩技术的进步和 ISDN 技术及标准的成熟，该阶段是视频会议发展较快的一个时期。90 年代初期，第一套国际标准 H.320 获得通过，不同品牌产品之间的兼容性问题得到解决。配合 H.261 视频压缩集成电路技术的开发，视频会议系统也有朝小型化发展的趋势。在 1992-1995 年期间，中小型视频会议系统成为视频会议应用中的主要产品。

1995 年以后，针对不同的网络应用环境，ITU 先后制定了许多视频会议标准。1996 年，随着在数字信号处理领域出台了低比特率的压缩标准（H.263），ITU-T 又通过了适合于极低速率通信网络（PSTN）的 H.324 标准。同时，ITU-TA 也为适用于 ATM 环境（ISDN）的视频会议系统制定了两套标准：H.321 系列标准和 H.310 系列标准。

1996 年 10 月 ITU-T 又针对 Internet 上多媒体视频会议系统制定了 H.323 系列标准，为那些与 Internet 和 Intranet 相连的视频会议系统提供了互通的标准，会议电视的技术和市场都发生了革命性的变化。越来越多的厂家竞相投入 H.323 新产品的开发，越来越多的用户采用 H.323 技术和产品构造他们自己的会议电视系统。这是通信网络技术、计算机技术和多媒体技术飞速发展和日益融合所带来的必然趋势。

1.3 视频会议系统相关技术

视频会议的发展在很大程度上受用户应用需求的影响，但最终起作用的还是相关技术的发展水平。与会议电视相关的技术有宽带网技术、分布式处理技术、芯片技术及多媒体信息处理技术等。

多媒体信息处理技术是会议电视的关键技术，主要针对各种媒体进行信息压缩和处理，而压缩处理后的视频质量的高低是决定多媒体服务质量好坏的主要因素，特别是早期的会议电视产品，各厂商都以编解码算法作为竞争的法宝。

新算法、新技术不断出现，进而推动多媒体信息处理技术的发展。比如 ITU-T 的 H.263+和 ISO/IEC 的 MPEG-4、MPEG-7（多媒体内容描述接口）等。MPEG-4 是继 MPEG-1 和 MPEG-2 之后的标准，该标准将众多的多媒体应用集成于一个完整的框架内，为多媒体通信及应用环境提供标准算法。于 1998 年提出的 MPEG-7 的目标是对各种不同类型的多媒体信息进行标准化描述，以实现对庞大图像、声音信息进行有效的管理和迅速检索。这些标准覆盖了很大的视频速率范围和应用领域，支持不同速率、不同的图像质量要求等条件的视频业务，能够满足包括电视会议、视频电子邮件、可视电话、广播及视频应用等不同要求的服务。

宽带网络技术是影响会议电视发展的另一个重要因素。近几年来，宽带网络技术有了很大的发展，目前正逐渐成为主流以及有潜力成为未来主流的接入技术，它包括 xDSL、混合广纤同轴（HFC）、广纤到户（FTTH）、交互式数字视频系统（SDV）等。ADSL 是目前最令人注目的宽带接入技术，ADSL 业务能力可提供的传输通道为非对称高速传输通道，ADSL 可提供 6～8Mbps 的下行数据流，上行通信速率可以达到 640Kbps，同时可进行 PSTN 上的模拟话音通信。因此 ADSL 是目前宽带接入的主要技术。APON(即 ATM PON 的简称，PON 是指无源光网络)代表了面向 21 世纪的宽带接入技术的最新发展方向。从长远看，可满足丰富多彩的多媒体业务日益增长的需要，是一种结合 ATM 多业务、支持多比特率和透明宽带传送能力的较理想的解决方案。

会议电视不是简单的点对点通信，它是一点对多点、多点对多点的实时同步通信，使用分布式处理技术，便于对不同媒体、不同位置的终端进行收发同步协调，使与会终端数据共享，使系统更接近人类的信息交流方式和处理方式。

高性能的芯片是实现电视会议所必需的基础，近年来 MMX 技术的引入，使 CPU 对多媒体信息的处理有了很大提高。

1.4 视频会议系统的类型

根据视频会议系统所用硬件设备、应用场合和产品价格等因素的不同，可把

视频会议系统分为基于房间的视频会议系统、中档视频会议系统、基于桌面的视频会议系统和可视电话视频会议系统四大类。

1．基于房间的系统在带有环境控制设备的专用会议房间里装置一个或多个大屏幕，系统由屏幕、摄像机、麦克风和辅助设备等组成。这些配置是永久性的，它们不能移到别的房间或大楼内，但可以提供高质量的视频和同步音频。该系统价格比较昂贵，一般适用于政府机构、大型企业的专用会议室。

2．中档视频会议系统对于中小型企业来说，能提供良好的图像质量，系统安装简单，操作方便，价格合理，适合 ISDN 传输，最高工作速率可达到 384kbit/s。如果某个客户只是将视频会议当作应急的必要措施，那么中档视频会议系统将是最有效的方案。该系统可以移动，系统组成包括一个屏幕、一两个摄像机和几个麦克风等。

3．桌面视频会议日益受到青睐，因为它有效地利用了现有的资源。视频会议仅仅是桌面上运行的多种应用之一。桌面机可以同时用于局域网和广域网中的视频会议，还可通过白板功能实现数据共享。一些桌面视频会议系统支持 TCP / IP，这类系统能用在桥接器、路由器和拨号线的衔接处，使用户能够方便、经济地接入视频会议系统，特别是在 ISDN 业务还没有实现的地区。

4．可视电话系统用于点到点通信，它满足了在电话上进行视频会议传输的需求。系统组成包括一个小屏幕、内部摄像机、视频编解码器、音频系统和键盘。

根据通信节点的数量，视频会议系统可分为：点对点视频会议系统和多点视频会议系统。

1．点对点视频会议系统

点对点视频会议系统支持两个通信节点间视频会议通信功能，它的主要业务有可视电话、桌面视频会议系统和会议室型视频会议系统三种。

可视电话是在现有公共电话网上使用的具有双工视频传输功能的电话设备。由于电话网带宽的限制，可视电话只能使用较小的屏幕和较低的视频帧率。

桌面视频会议系统利用用户现有的台式机（如 PC 机）平台以及网络通信设备和远地另一台装备了同样或兼容设备的台式机通过网络进行通信，这种系统仅限于两个用户或两个小组用户使用。Intel 公司的 Proshare Personal Conferencing Video System 200 是这类系统的一个典型示例，这是一种点对点的个人视频会议系统，支持 ISDN 和 LAN 的连接，采用硬件编码压缩，软件解压缩，为了方便协同工作，Proshare 还提供共享笔记本和共享应用程序。

在会议室型视频会议系统的支持下，一群与会者集中在一间特殊装备的会议室中，这种会议室作为视频会议的一个收发中心，与远地的另外一套类似的会议室进行交互通信，完成两点间的视频会议功能。由于会议室与会者较多，因此对视听效果要求较高，一套典型的系统一般应包括：一台或两台大屏幕监视器、高

质量摄像机、高分辨率的专用图形摄像机、复杂的音响设备、控制设备及其他可选设备，以满足不同用户的要求。

以上三种点到点视频会议业务均有各自的限制因素，所以均未得到广泛发展。

2. 多点视频会议系统

多点视频会议系统允许三个或三个以上不同地点的参加者同时参与会议。多点视频会议的一个关键技术是多点控制问题，多点控制单元（MCU）在通信网络上控制各个点的视频、音频、通用数据和控制信号的流向，使与会者可以接收到相应的视频、音频等信息，维持会议正常进行。由于多点视频会议系统允许多个参加者同时参与会议，能很好的模拟真实会议，因此它将成为视频会议系统发展的趋势。

1.5 视频会议系统的国际标准

目前使用最为广泛且发展前景最好的多媒体业务主要是视频会议、远程教育、远程医疗、远程监控、视频点播及可视电话等业务，而实现这一系列业务的核心就是会议电视体系标准。国际电信联盟（ITU）的主要任务是制定通信标准，以便世界范围内通信的相互操作。ITU-T 的 H 系列标准是为多媒体会议而制订的，其中包含了视频编解码、语音编解码、复用、控制信令及传输速率等内容。为推进视频会议的标准化进程，国际标准组织针对不同的网络环境陆续开发出一系列视频会议标准。ITU-T 制定的 H.32x 多媒体通信系列标准主要有：

H.320：在窄带可视电话系统和终端上进行多媒体通信的标准；

H.321：在 B-ISDN 上进行多媒体通信的标准；

H.322：在有 QoS 保证的 LAN 上进行多媒体通信的标准；

H.323：在无 QoS 保证的 PBN 上进行多媒体通信的标准；

H.324：在低比特率通信终端（PSTN 和无线网络）上进行多媒体通信的标准；

上述每个框架型 H.32x 系列标准又都包括了其相应的视频、音频、通信协议、复用/同步等系列标准，数据通信协议采用 ITU-T 第 8 组制定的 T.120 系列标准。详细情况如表 1.1 所示：

表 1.1 视频会议国际标准

标准名称	H.320	H.321	H.322	H.323	H.324
公布年份	1990	1995	1995	1996	1996
适用网络	N-ISDN	B-ISDN ATM	保证 QoS 的 PBN	不保证 QoS 的 PBN	PSTN 无线网络
视频标准	H.261 H.263	H.261 H.263	H.261 H.263	H.261 H.263	H.261 H.263
音频标准	G.711	G.711	G.711	G.711	

	G.722 G.723	G.722 G.723	G.722 G.723	G.722 G.723 G.728 G.729	G.723
复用标准	H.221	H.221	H.221	H.225.0	H.223
数据标准	T.120	T.120	T.120	T.120	T.120
控制标准	H.230 H.242	H.242	H.230 H.242	H.245	H.245

1.5.1 H.320 系列标准

H.320 系列标准是 ITU-T 在 1990 年制定的，是会议系统中应用最早，最为成熟的协议。支持 ISDN，E1，T1，带宽从 64kbps 到 2Mbps，几乎所有会议系统厂家都支持。H.320 规定了视频和语音编解码的标准、复用和控制等一整套协议。其中有语音编解码的标准 G.711、G.722、G.728、G.729 和视频编解码标准 H.261、H.263、H.263+、H.263++、H.26L，其中复用部分采用定义到比特的 H.221、H.230 标准，控制部分采用 H.242 进行终端之间的能力交换。由于 ISDN 协议是基于电路交换的，优点是能保证传输带宽，从而充分保证视频服务的实时性，而且其技术发展已相当成熟，它推动了多媒体视频会议系统的迅猛发展。

1.5.2 H.324 标准

H.324 是 1996 年颁布的基于 PSTN 的视频会议系统，同样也规定了视频和语音编解码的标准、复用和控制等一整套协议。其中语音编解码的标准采用 G.723.1，视频编解码标准采用 H.261、H.263、H.263+，复用采用 H.223，控制部分采用 H.245。因为传输速率很低（33.6Kbps），视频质量比较差，而没有很大的发展。

1.5.3 H.323 标准

H.323 由 H.320 发展而成，是 ITU 在 1996 年制定的，称为“不保证服务质量的局域网上的多媒体视听系统”。该协议提供了在没有传输质量保证的网络上进行多媒体通信的规范。H.323 终端之间可以实时传送语音、数据和视频图像。H.323 中语音编解码的标准有 G.723.1、G.711、G.722、G.728、G.729。视频编解码标准有 H.261、H.263、H.263+，其中复用采用 H.225、H.230、TCP/IP，控制采用 H.245，适用于 LAN 和 Internet 网络。由于 IP 协议是目前网络应用的主流协议，它不仅统治了桌面和通信网络的接入层，而且正在向通信网络的骨干层和汇聚层渗透，这使得 H.323 协议族有了非常旺盛的生命力。

1998 年 ITU 又将 H.323 扩展为“基于分组的多媒体通信系统”，即 ITU-T H.323 V2。这些分组网络主宰了当今基于 TCP/IP、IPX 分组交换的以太网、快速以太网、令牌网和 FDDI。H.323 标准为 LAN、WAN、Internet 和 Intranet 上的多媒体通信提供了应用基础和保障。

1.5.4 其他视频会议标准

除了 H.320、H.323、H.324 外，还有 H.310、H.321、H.322 标准。ITU-T 的 H.321 和 H.310 标准是针对 ATM 网络结构的会议电视标准和规范。H.321 是将窄带视听多媒体终端（H.320）适配到 ATM 环境（B-ISDN）的技术规范。H.310 是 B-ISDN 视听通信业务系统和终端的标准。B-ISDN 网络可用高达 155Mbps 的信息传输速率，能传输高保真音频、视频和 VOD 等。虽然 B-ISDN 有许多优势，但由于技术复杂，设备昂贵，今后很难成为主流。H.322 建议是涉及到“有 QOS 保证的局域网的可视电话系统和终端设备”的一个标准。其实质上是 H.320 终端配置上交互工作适配器及 LAN 的接口单元的会议电视设备。这些标准覆盖了很大的视频速率范围和应用领域，支持不同速率、不同图像质量要求等条件的视频业务，能够满足包括电视会议、视频电子邮件、可视电话、广播级视频应用等不同要求的服务。

随着 IP 网上多媒体系统（H.323）、IP Phone、IP FAX 等标准的制定，以及 ATM 技术、IP 交换技术、千兆以太网技术在网络层都可以统一到 IP 上，基于 IP 的会议电视将成为实时多媒体通信最为理想的发展方向，是今后会议电视发展的主流。

1.6 任务简介

根据以上分析介绍可知，ITU-T 在 1996 年制定的 H.323 标准是当前视频会议系统的主流标准，本论文的课题项目就是基于 H.323 标准展开的。在视频会议系统结构中，多点控制单元 MCU 是一个必不可少的重要部件，它的性能优劣将直接影响到视频会议系统的质量。如果 MCU 的性能不能满足所要召开会议的要求及规模，那么各终端之间不能正常通信，以至于会议无法进行。因此召集会议前，如果知道了会议模拟环境下的 MCU 性能，就可以控制会议的规模以保证会议顺利进行。同时它也可以作为衡量视频会议系统性能的重要参数，以便更好地完善该系统。至于如何测试 MCU 的性能，目前并没有公开的专门的测试方法，因此很有必要提出一种测试 MCU 性能的方案。并且应用所提方案，就 MCU 处理运动程度不同的图像的性能进行分析。该性能是 MCU 的重要性能之一。

本论文的主要研究开发工作有：

1. 简要介绍了视频会议系统的概念、发展历程及国际视频会议系统的各个标准；深入研究了 H.323 协议体系、基于 H.323 的多点会议的通信过程和主要协议以及 H.323 系统的重要组成部件一多点控制单元 MCU 的多点控制和多点处理功能。

2. 提出了一套支持 H.323 标准并能有效测试 MCU 性能的方案，同时设计开发了一个测试设备——模拟呼叫器。

3. 编写代码，实现了一个可以模拟多个终端的模拟呼叫器应用程序。该应用程序经过测试检验，能够实现其预先设计的功能，并能测试 MCU 性能。

4. 根据提出的方案，应用模拟呼叫器，测试 MCU 的性能并加以分析。

第二章 基于 H.323 的视频会议系统

H.323 是 ITU-T 多媒体通信系列标准 H.32x 的一部分，它定义了无 QOS（服务质量）保证的分组网络 PBN（Packet Based Networks）上的多媒体通信标准。这些分组网络主宰了当今的桌面和网络系统，包括基于 TCP/IP、IPX 分组交换的以太网、快速以太网、令牌网、FDDI 网。因此，H.323 标准为 LAN、WAN、Intranet、Internet 上的多媒体通信应用提供了技术基础和保障。本章将着重从 H.323 系统组成和协议族两方面来介绍 H.323 视频会议系统。

2.1 H.323 系统的组成

H.323 系统结构如图 2.1 所示[4]，图中还示出和其它系统终端的互通连接。

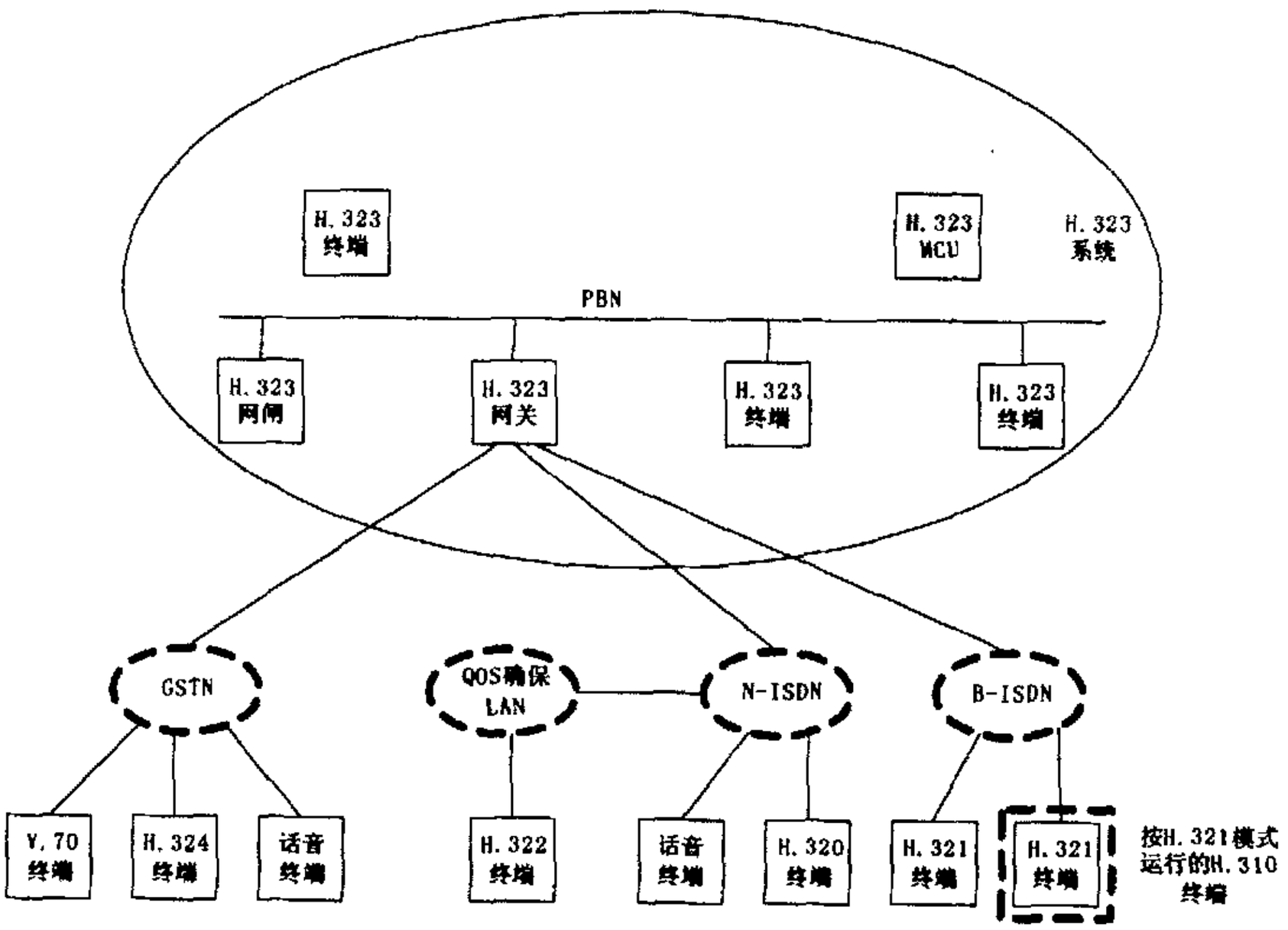

图 2.1 H.323 系统结构

H.323 系统的组成部件称为 H.323 实体（entity），从图中可看到它包括终端、网关、网闸、多点控制器（MC）、多点处理器（MP）和多点控制单元（MCU—Multipoint Control Unit）。其中，终端、网关和 MCU 统称为端点，端点可以发起呼叫，也可以接受呼叫，媒体信息流就在端点生成或终接。网闸、MC 和 MP 则不可呼叫，但是网闸参与呼叫的控制，具有运输层地址，是可寻址的 H.323 实体；MC 和 MP 执行多点呼叫信息流的处理和控制，是系统的功能实体，物理上总是位于

某个端点之中，因此没有独立的运输层地址，是既不可呼叫又不可寻址的 H.323 实体。下面，将一一介绍组成 H.323 系统的各个部件。

2.1.1 H.323 终端

H.323 终端是分组网络的一个终端用户设备，可以与其他 H.323 终端进行实时的双向的音频、视频或者数据通信。该终端也能与 H.323 网关和一个多点控制单元（MCU）进行通信。尽管列举了支持音频、视频和数据通信的能力，但终端并不需要配置成能支持所有的业务，而 H.323 也不需要终端具有支持如此多业务的能力。

一个典型的 H.323 终端包括以下部分：用户设备接口、视频编码解码器、音频编码解码器、数据应用处理设备、H.225.0 层功能、系统控制单元与分组交换网络的接口。虽然网络接口不在 H.323 建议范围内，但它必须提供 H.225.0 建议所要求的两种服务：用于支持建立 H.245 控制信道、数据流信道和呼叫控制信道的可靠的点对点服务（如 TCP 及 SPX），和用于支持建立 RAS 控制信道、音频流信道和视频流信道的不可靠的点对点服务（如 UDP 及 IPX）。

图 2.2 描述了 H.323 终端的组成[4]。

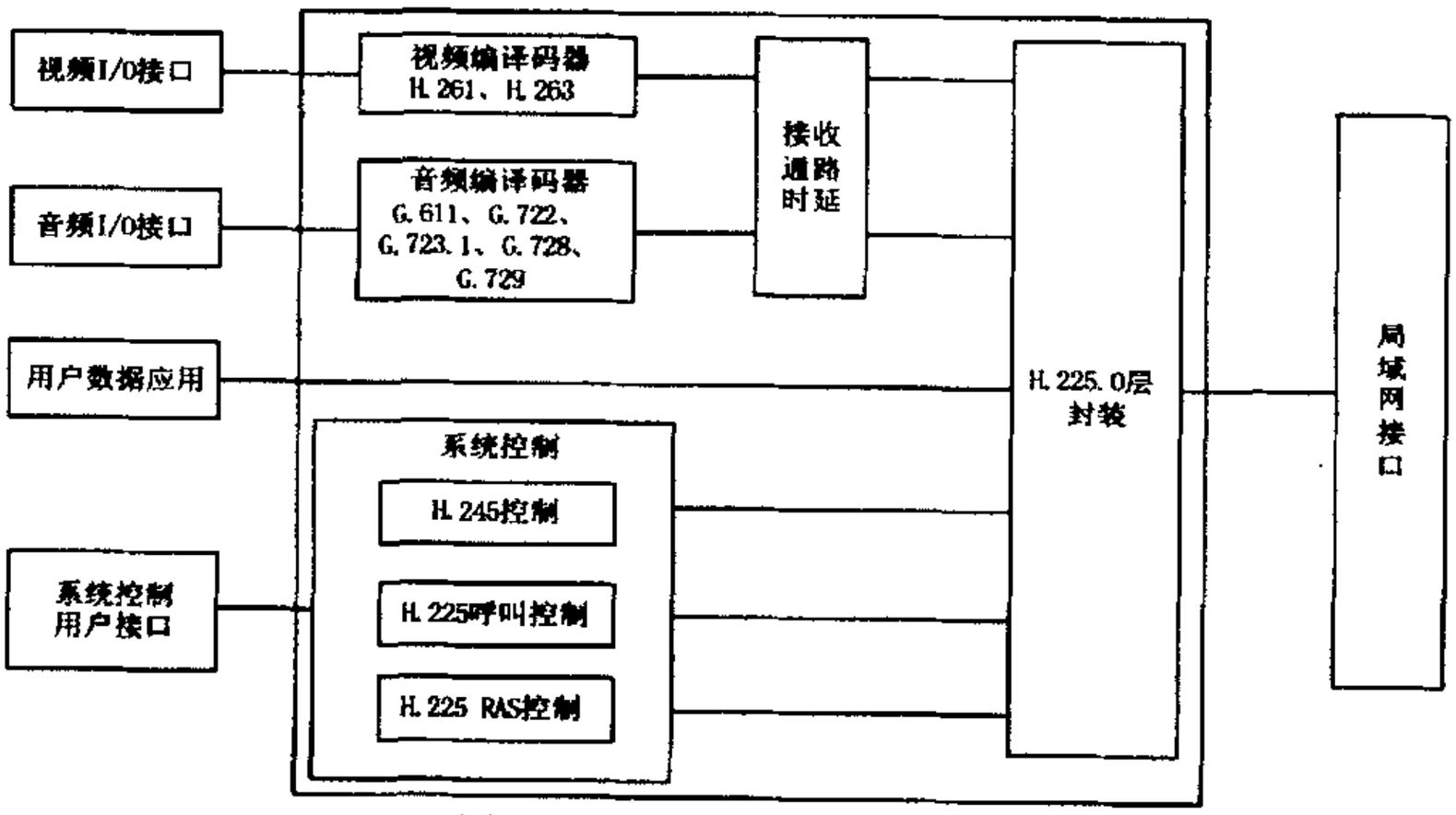

图 2.2 H.323 终端功能结构

视频编解码器：是一个可选项，图像格式为 H.261 的 CIF、QCIF 或 H.263 的 CIF、4CIF、16CIF、QCIF、SQCIF。视频不进行 BCH 纠错。允许以不对称的比特率、帧率和图像分辨率运行。

音频编解码器：所有 H.323 终端都应具有音频编解码单元，应根据 G.711 标准编解码，能够发送和接收 A 律和μ律信号。可选择地根据 G.722、G.728、G.729、MPEG1 音频等标准进行编解码，并且可以不对称。

用户数据功能：此项可选，根据需要选择一条或多条数据通道，数据通道可以是单向的，也可以是双向的。

H.245 控制：传送端到端的控制消息，控制 H.323 实体的操作，包括能力交换、

打开和关闭逻辑通道、模式选择请求、流量控制消息、命令与指示等。H.245 控制信令信道可以在两个端点之间或端点和网闸之间建立，每个控制信令信道处理一个呼叫。

H.225 呼叫控制：用于在两个端点之间建立连接。H.225 信道的建立先于 H.245 信道和其他任何逻辑信道。

RAS 控制：用于在端点和网闸之间实现登记、管理、带宽改变、状态查询等功能。系统没有网闸时不建立 RAS 信道，系统包含网闸时 RAS 信道是最早建立的信道。

H.225 层：用于将音频、视频、数据和控制信息流打包后送往网络接口。

2.1.2 H.323 网关

网关是分组网络的一个端点，可以与 H.323 终端、其他 ITU-T 终端或其他网关之间进行实时双向通信。网关用于连接两个异构的网络，H.323 网关用于连接 H.323 网络和非 H.323 网络，负责不同网络之间的信令和控制信息的转换以及媒体信息变换和复用。为了实现这种异构网络的互联，网关必须进行呼叫建立和释放的协议转换、不同网络间的媒体格式的转换等工作。所有 H.323 终端通过网关实现了与其他类型终端的互通。图 2.3 给出了 H.323/H.320 网关。

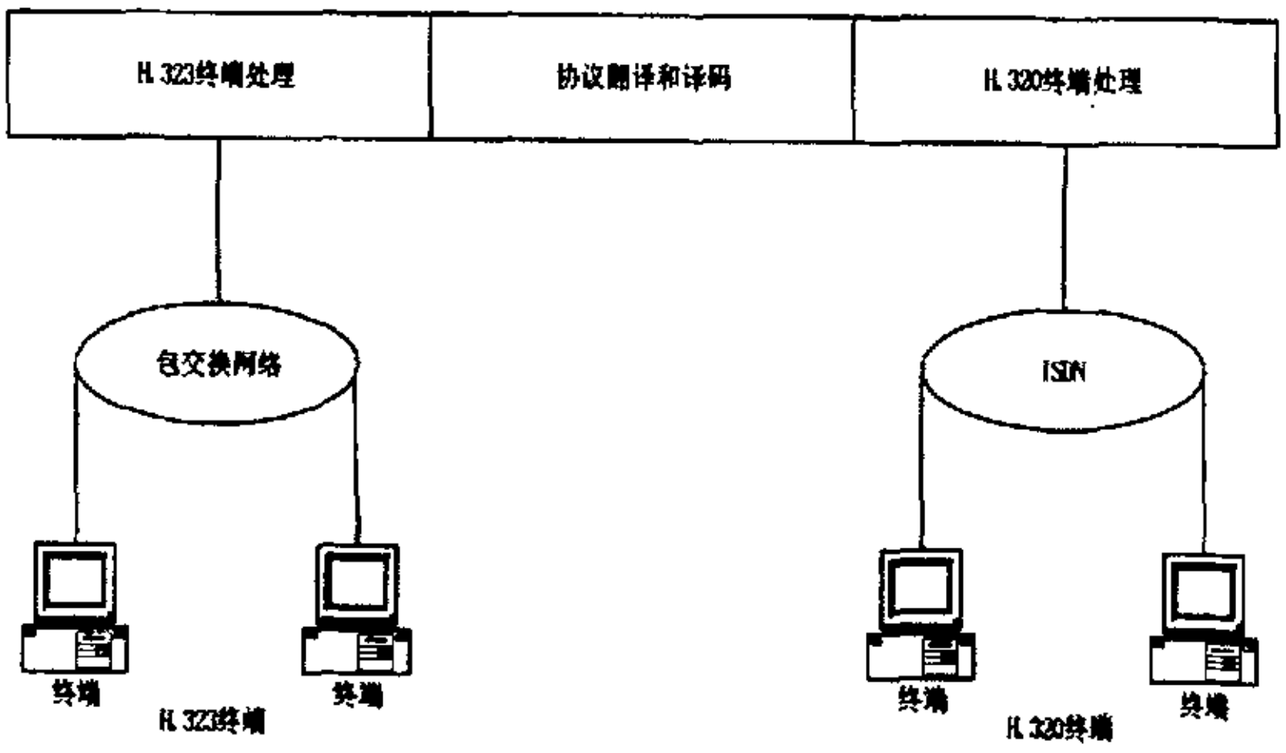

图 2.3 H.323/H.320 网关

当然，H.323 系统并不是一定要有网关，如果仅在 H.323 网络内的端点间进行通信，则不需要任何网关。

2.1.3 H.323 网闸

网闸提供对 H.323 端点和呼叫的管理功能，是网络的管理点，被称为 H.323 网络的“大脑”，因为它能够提供中央管理和控制功能。在一个 H.323 系统中可以没有网闸，也可以有一个或多个网闸，网闸之间可以相互通信。系统中如果有网闸，它提供鉴权、选路、呼叫和带宽管理。它的功能包括（1）地址翻译 根据翻译表将别名地址翻译为传输层地址；（2）接入控制 应用 ARQ/ACF/ARJ H.225.0

消息，根据呼叫授权、带宽等条件确定是否允许用户发起该呼叫；（3）带宽控制 应用 BRQ/BCF/BRJ 消息，根据带宽情况进行带宽管理；（4）域管理 （5）呼叫控制信令 （6）呼叫授权 利用 H.225.0 信令，网闸可以允许或拒绝某个终端的呼叫；（7）带宽管理 控制同时接入网络的 H.323 终端的数量，利用 H.225.0 信令，当网闸判定带宽不足时，可拒绝某个终端的呼叫。当某个已建立的呼叫要求增加带宽时，也可来判定。（8）呼叫管理 管理正在进行的呼叫，可显示被叫终端忙闲。上述功能中，（1）～（4）为网闸的基本功能。

2.1.4 H.323 多点控制单元（MCU）

MCU 支持在三个或三个以上的终端和网关之间的多点会议，一个两端点的点对点的会议可以扩充到一个多点会议。为了使三方以上的端点间能相互通信，MCU 必须包含多点控制器（MC）和零个或多个多点处理器（MP），MCU 可以是独立的设备，也可以集成在终端、网关或网闸中。MCU 采用 H.245 协议过程实现其控制功能。MC 和 MP 只是功能实体，并非物理实体，都没有单独的地址。

一．多点控制器（MC）

在集中型会议中，所有参会终端（包括网关）都要和 MCU 中的 MC 建立 H.245 控制信道的点对点联系，该 MC 执行对整个会议的集中控制。它和参加会议的每个端点执行“能力交换”过程，指示信息可发送的操作模式，当有终端加入或离开此会议时，MC 可能会调整向各终端发送的能力集信息。MC 据此确定会议的“选定通信模式”（SCM——Selected Communication Mode）。一般说来，SCM 对于所有参会终端是相同的，但也有可能某些终端的 SCM 和其它终端不同，此时 MP 要进行通信模式的转换，以保证各终端之间的正常通信。除此之外，在视频混合中，选择哪些源和多少源图像进行混合由 MC 决定。MC 的控制功能通过 H.245 协议完成，为此参会各端点首先要建立至 MC 的 H.245 控制信道。

二．多点处理器（MP）

集中型会议各终端的音频、视频和数据信道和 MCU 中的 MP 相连。MP 对各终端送来的信号进行音频混合、视频交换或混合和 T.120 数据分配，然后将处理后的音视频流和数据流再送回各终端。因此，MP 必须能执行各种媒体信息的编解码算法。

对于视频信号，MP 必须能进行视频交换或视频混合。视频交换就是选定某一源的信号发往各终端，源选择可通过发言者变换（检测话音信号电平确定）检测实现或由 H.245 控制实现。视频混合是将多个视频源信号组合成一个信号后传给各终端，其典型应用是将 4 幅源图像组合成一个 2×2 的多画面图像。如果终端之间的 SCM 不同，MP 则要进行通信模式的转换以保证各终端之间的正常通信。

对于音频信号，MP 能够通过交换、混合或二者的组合将 M 个音频输入经处理后生成 N 个输出。需要注意的是，音频混合需要将每个输入音频首先解码为线

性 PCM 信号或模拟信号，然后将各路信号合成，最后再将合成后的信号重新编码成相应格式。混合时，MP 可以去除或衰减某些输入信号，以降低噪声和其他不希望的信号。终端应保证自己的话音信号不在自己的终端输出。

2.2 H.323 系统通信协议栈

H.323 属于 ITU 多媒体通信协议系列 H.32x 的一种，提供基于分组网络的话音、视频、数据和控制等协议。H.323 作为一种协议框架，提供了系统及组成部分描述、呼叫方式描述和呼叫信令程序。H.323 相关的协议包括 H.261、H.263 视频编码标准，G.711、G.722、G.723、G.728 音频编码标准，T.120 多点数据会议系列标准、H.225.0 分组和同步标准，H.245 系统控制标准。H.323 系统中的通信可以看成是视频、音频、数据和控制信息的混合，其协议栈结构如图 2.4 所示[4]：

<table>
<tr><td colspan="2">声像应用</td><td colspan="4">终端控制和管理</td><td>数据应用</td></tr>
<tr><td>G.7XX</td><td>H.26X</td><td rowspan="3">RTCP</td><td rowspan="3">H.225.0
终端至
网守信
令
(RAS)</td><td rowspan="3">H.225.0
呼叫
信令</td><td rowspan="3">H.245
媒体信
道控制</td><td rowspan="3">T.120
系列</td></tr>
<tr><td colspan="2">加 密</td></tr>
<tr><td colspan="2">RTP</td></tr>
<tr><td colspan="3">不可靠传送协议</td><td colspan="3">可靠传送协议</td></tr>
<tr><td colspan="6">网 络 层</td></tr>
<tr><td colspan="6">链 路 层</td></tr>
<tr><td colspan="6">物 理 层</td></tr>
</table>

图 2.4 H.323 协议栈结构

- 网络层和传输层协议

在 H.323 协议栈中，IP 和 TCP 协作，共同完成面向连接的传输。可靠的传输保证了数据包传输时的流量控制、连续性以及正确性，但这样会增加传输时延、时延抖动和占用的带宽。H.323 将可靠的 TCP 用于 H.245 控制信道，T.120 数据信道和呼叫信令信道。由于音频和视频信息允许一定的数据包传输差错，但对时延及抖动有较高要求，所以采用不可靠的、非连接的用户数据报协议（UDP）传输。虽然无法提供很好的服务质量，但其传输时延较 TCP 小。

- 系统控制

系统控制功能是 H.323 终端的核心，它提供了 H.323 终端正确操作的信令。这些功能包括终端认证与注册、呼叫过程控制、能力交换、用于媒体信息控制的各种命令和指示信令以及用于建立和描述各种逻辑信道内容的报文等。整个系统的控制由 H.245 媒体控制信道、H.225.0 呼叫信令信道、以及 RAS 信道提供。

H.225.0 的呼叫信令 Q.931 是端点和端点之间的信令协议，用于在两个端点之间建立/拆除呼叫。当 H.323 网络中没有网闸时，H.225.0 信令直接在端点间传递，

当有网闸存在时，呼叫信令则会通过网闸进行路由。

H.225.0 的 RAS（注册，认证和状态）是端点和网闸之间的信令协议，定义了登记（Registration）、认证（Admission）和状态检测（Status）等过程。只有当网闸存在时，才会使用 H.225.0 RAS。H.225.0 RAS 通信包括以下各方面：寻找网闸；端点登记，所有端点必须向网闸登记，因为网闸需要知道自己管区内的所有端点的别名和传输地址以进行地址解析；端点定位，网闸采用该功能寻找具有某一传输地址的端点。

H.225.0标准描述了在无QoS保证的分组网上媒体流的打包分组与同步传输机制，它对传输的视频、音频、数据以及控制流进行格式化，以便输出到网络接口。另外，它还完成逻辑成帧、顺序编号、纠错与检错功能。

H.225.0 建立了一个呼叫模型，在这个模型中，呼叫建立和性能协商没有使用 RTP 传输地址，呼叫建立之后才建立若干个 RTP/RTCP 连接。呼叫建立之前，终端可以向某个网闸注册。在任何呼叫开始之前，首先必需在端点之间建立呼叫联系，同时建立 H.245 控制信道。，开始一个呼叫一般必须首先发送一个认可请求信息，接着发送一个初始建立信息，这个过程以收到连接消息为结束。

H.245 协议定义了主从判别功能，当在一个呼叫中的两个终端同时初始化一个相同的事件时，就产生了冲突。为了解决这个问题，终端必须判断谁是主终端，谁是从终端，主从判别过程用来判定哪个终端是主终端，哪个是从终端。终端的状态一旦确定，在整个呼叫过程期间都不会改变。

H.245 协议还定义了性能交换过程，用来保证传输的媒体信号是能够被接收端接收的。这要求每一个终端的接收和解码能力必须被对方终端知道。终端不需具备所有的能力，对于不能理解的要求可以不予理睬。终端通过发送它的性能集合使对方知道自己的接收和解码能力，接收方可以从中选择某种方式。

- 分组与同步：

H.225.0协议描述了在无QoS保证的分组网上媒体流的打包分组与同步传输机制，它对传输的视频、音频、数据以及控制流进行格式化，以便输出到网络接口。另外，H.225.0 还完成逻辑成帧、顺序编号、纠错与检错功能。

- 音频编解码协议：

音频编解码协议完成对输入的音频信号进行编码、压缩传输，并在接收端解码输出的功能。H.323 支持的音频编解码算法协议都是 ITU 颁布的。H.323 协议规定一个 H.323 终端必须支持 G.711 协议，传送和接受 A 律和μ律压缩音频信号。支持其他音频编解码协议（如 G.722、G.723、G.728、G.729 等）是可选的。H.323 终端之间每次通信使用的音频编解码算法必须在能力交换期间由 H.245 控制消息确定。H.323 终端应能对本身所具有的音频编解码能力进行非对称的操作。例如：以 G.711 编码发送而以 G.728 解码接收。

● 视频编解码协议：

视频编解码协议完成在视频源处将输入的视频信号进行编码、压缩传输，并在接收端解码显示的功能。H.323 协议规定，支持视频功能是可选的。但任何支持视频功能的 H.323 终端都必须支持 H.261 QCIF 格式，支持 H.261 的其他格式和 H.263 格式是可选的。

● 数据通信协议：

数据通信协议用于支持诸如静态图像、二进制文件等的传输以及实现电子白板等。H.323 协议规定支持数据通信功能是可选的。H.323 系统中的数据通信采用 T.120 系列协议。T.120 系列协议是一个完整的数据通信协议，它规定了支持数据通信的网络传输规程、支持数据标准应用的规程，还给出了多点数据通信控制规范。其中的 T.126 对应多点电子白板，T.127 对应多点文件传输，T.128 对应多点应用共享。

● 实时传输协议（RTP）和实时传输控制协议（RTCP）

RTP 即实时传输协议（Realtime Transport Protocol）,是 H.323v2 规定的在传输时延较低的 UDP 上实现实时传输的协议,其目的是提供时间信息和实现流同步；RTCP 即实时传输控制协议（Realtime Transport Control Protocol），提供数据传输和服务质量的端到端监控，用于管理传输质量和提供 QoS 信息，增强 RTP 的功能。当应用程序开始一个 RTP 会话时将使用两个端口：一个给 RTP,一个给 RTCP。RTP 本身不能为按顺序传送数据包提供可靠的传送机制、流量控制和拥塞控制，只能通过 RTCP 来提供。

RTP 相当于 OSI 七层模型中的会话层协议，完成的主要工作有：

——在媒体流数据包头加上时间标签；

——提供包内数据类型的标志；

——对数据包进行排序，包序号可以用来在接收端重建正确的包顺序。

RTP 协议从上层接收媒体信息码流，组装成 RTP 数据包，然后发送给下层 UDP 协议打包传输。RTCP 主要监视时延、抖动和带宽。一旦所传送的多媒体流的带宽发生变化或时延、抖动等参数发生明显改变，接收端则通知发送端，改变符号化识别码和编码参数。

2.3 H.323 通信过程的主要协议

H.323 端点之间建立通信关系一般要经过三个控制过程，现在介绍这三个过程所要执行的三个协议：RAS 协议，H.225.0 呼叫信令协议，H.245 呼叫控制协议。

2.4.1 RAS 协议

RAS 是端点（终端或网关）和网闸间执行的协议，基本上是管理功能。它包

含网闸搜寻、端点登记、端点定位、呼叫接纳、呼叫退出、带宽管理、状态查询和网关资源指示等几个过程，每个过程都包含一组 RAS 消息。各过程所用的 RAS 消息如表 2.1 所示。

表2.1 RAS消息一览表

过 程	消 息	消 息 名
网闸搜寻	GRQ GCF GRJ	网闸搜寻请求 网闸搜寻证实 网闸搜寻拒绝
端点登记	RRQ RCF RRJ URQ UCF URJ	登记请求 登记证实 登记拒绝 去除登记请求 去除登记证实 去除登记拒绝
端点定位	LRQ LCF LRJ	端点定位请求 端点定位证实 端点定位拒绝
呼叫接纳	ARQ ACF ARJ	呼叫接纳请求 呼叫接纳证实 呼叫接纳拒绝
呼叫退出	DRQ DCF DRJ	呼叫退出请求 呼叫退出证实 呼叫退出拒绝
带宽管理	BRQ BCF BRJ	带宽请求 带宽证实 带宽拒绝
状态查询	IRQ IRR	信息请求 信息响应
网关资源指示	RAI RAC	网关资源可用性指示 网关资源可用性证实

● 网闸搜寻：

网闸搜寻用于端点搜寻其归属网闸，采用多播机制完成。其后所有 RAS 消息均限定在端点和其归属网闸之间传送。

网闸搜寻有两种方式：人工方式和自动方式。人工方式通过端点配置完成，将其归属网闸的运输层地址预置入配置文件或初始化文件。自动方式允许端点和归属网闸的关系可以随时间而改变，当原有网闸出故障时可以自动切换到替换网闸上去，当所属网闸改变后无需在每个终端上都修改其配置文件。

在自动方式中，端点采用搜寻多播地址发送 GRQ 消息，询问“谁是我的网闸?”。可以有一个或多个网闸回送 GCF 消息，表示“我可以是你的网闸”，并在消息中告之其 RAS 地址。不愿意该端点在其上登记的网闸则返回 GRJ 消息。如果有多个网闸回送 GCF 消息，端点可在其中任选一个作为其归属网闸。如果超时仍未收到

网闸的响应，端点可重发 GRQ 消息，但两次相邻 GRQ 之间的时间间隔不能小于 5 秒。如果仍未收到响应，则改用人工搜寻。当然如果事先知道网闸的 IP 地址，则可以直接使用人工搜寻。

端点发送的 GRQ 消息包含端点类型、端点自身的 RAS 地址、希望在其上登记的网闸标识等参数。若未含网闸标识参数，就表示端点愿意在任何一个网闸上登记。网闸返回的 GCF 消息除了包含该网闸标识和 RAS 地址外，还可包含“替换网闸”序列参数，他按优先级顺序指定，以后如果至该网闸的请求未予响应或被拒绝又未给出转向信息时，可转至那些网闸重新提出请求。GRJ 消息也包含此参数，另外还给出拒绝原因。

- 端点登记

端点登记用于端点向网闸登记其自身信息，主要是别名和呼叫控制信道运输层地址。包括去除登记过程。端点必须在搜寻过程中确定的网闸上登记，必须在登记后才能发起和接受呼叫，登记表明该端点加入了管理区。

端点需向网闸的 RAS 地址发送 RRQ 消息，消息包含的最重要的两个参数就是端点别名及其呼叫信令运输层地址。别名可为 E.164 地址或 H.323 标识。E.164 标识可由接入码和电话号码组成，该接入码可以用来标识网关；H.323 标识为字符串形式，可以是用户名、E-mail 名或其它标识名。一个端点可以有多个别名，所有别名都应在 RRQ 消息中送往网闸，它们将翻译为同一个运输层地址。

在一般情况下，网闸将返回 RCF 消息，告之网闸的呼叫信令运输层地址，并将端点别名和地址登记入翻译表。如果网闸收到 RRQ，其别名和以前 RRQ 消息相同，但运输层地址不同，则回送 RRJ 消息，并指明拒绝原因是“重复登记”。如果收到 RRQ，其运输层地址和以前相同，但别名不同，则作为更新登记消息修改其翻译表。

如果端点在 RRQ 消息中没有包含别名，网闸可以赋予一个，然后将此别名由 RCF 消息回送给终端。如果端点想改变其别名和地址的对应关系，可以先将以前的登记信息删除，即向网闸发送 URQ 消息，网闸回送 UCF 消息。如果网闸发现该端点原来未在它上面登记，则回送 URJ 消息。网闸也可以向端点发送 URQ 消息，表示要删除该端点的登记信息，端点须回送 UCF 消息。其后，端点须重新登记才能发起呼叫，一般要在新的网闸上登记。

未在网闸上登记的端点称为未登记端点，这类端点不能请求网闸的接纳控制、带宽控制、地址翻译等服务。

- 端点定位

用于端点或网闸向相应的网闸询问某一端点呼叫控制信道的运输层地址。当端点或网闸已知某一端点的别名，需要知道其呼叫信令信道运输层地址时，可向相应的网闸发送 LRQ 消息，发送地址是该网闸的 RAS 地址。如不知道该端点的

归属网闸，也可用搜寻多播地址发送 LRQ 消息。该端点在其上登记的网闸收到此消息后应回送 LCF 消息，消息中包含该端点的呼叫信令信道运输层地址，或者是该网闸的呼叫信令信道运输层地址，究竟传送哪一个地址取决于呼叫信令是采用直接选路方式还是网闸选路方式。

对于该端点未在其上登记的网闸，如果是在 RAS 信道上收到 LRQ 消息，则须回送 LRJ 消息；如果是在搜寻多播地址上收到 LRQ 消息，则不需要作任何响应。

- 呼叫接纳和退出

呼叫接纳是起呼时的第一步操作，询问网闸是否允许该呼叫发起。呼叫退出是呼叫结束后通知网闸，该端点已退出呼叫。

ARQ/ACF 和 DRQ/DCF 是整个呼叫控制过程第一对和最后一对消息，分别标志呼叫的开始和结束。在 ARQ 中，端点给出目的地信息（如其 E.164 地址或 H.323 标识）以及所要求的带宽。网闸如果同意接纳此呼叫，则回送 ACF，其中包括的两项主要参数是允许分配的带宽和翻译后所得的目的地呼叫信令传输层地址或者是网闸本身的呼叫信令运输层地址。网闸在 ACF 消息中给定的带宽可以低于端点在 ARQ 中请求的带宽。

不但主叫发呼叫时需要请求网闸接纳，被叫收到入呼叫建立信令时也要向网闸发送 ARQ 请求。

- 带宽管理

用于呼叫中途改变呼叫接纳时确定的带宽，改变请求可由端点或网闸发起。网闸也可主动向端点发送 BRQ，请求改变带宽。如果是降低带宽请求，端点必须服从，降低其总比特率且回送 BCF。同时向对端端点发送 H.245 控制消息，通知其信道带宽已变，然后由对端端点再通知其网闸。如果是增加带宽请求，端点根据需要决定是否增加比特率。

- 状态和资源

状态查询主要用于网闸询问终端的开机/关机状态。网关资源指示用于向网闸通告该网关的可用资源。

2.4.2 H.225.0 呼叫信令协议

呼叫信令消息的传送有直接选路和网闸选路两种方式，H.323 定义了不同网络环境下采用两种消息传送方式时的信令过程，主要有两端点均未在网闸登记、两端点在同一网闸上登记、仅主叫端点有网闸、仅被叫端点有网闸、主被叫在不同网闸登记等多种情况。鉴于本论文提出的方案属于两端点在同一网闸上登记，下面将具体介绍这种方式的信令过程。

- 两端点在同一网闸上登记的直接选路方式

其信令过程如图 2.5 所示。简要说明如下：

①端点 1（主叫）在 RAS 信道上向其网闸发送 ARQ 消息，请求发起至端点 2 的呼叫。

②网闸同意接纳此呼叫，并翻译得出端点 2 的呼叫信令信道运输层地址（IP 地址＋TCP 端口号），由 ACF 消息回送端点 1。

③端点 1 建立至端点 2 的呼叫信令信道，在此信道上发送 Setup 消息。如果 ARQ 中已带呼叫引用值 CRV，则 Setup 及其后信令消息中的 CRV 应取此相同值。

④端点 2 回送 Call Proceeding 消息，指示呼叫已抵达，正在处理之中。对于两个 H.323 终端之间的呼叫，除了 UUIE 外，消息一般不必带其它信息单元。如果是 H.323 终端和 ISDN 终端之间的通信，即端点 2 是网关，则端点 2 将把自 SCN 侧收到的信息单元，如承载能力和进展表示语透明回传给端点 1。若端点 1 是 H.323 终端，则不予解释；若端点 1 也是网关，则需将这些信息单元继续向前传给 SCN 侧的主叫。

⑤端点 2 愿意接受此呼叫，经 RAS 信道向网闸发送 ARQ，请求接受此呼叫。

⑥网闸同意接纳，回送 ACF；

⑦端点 2 向端点 1 回送 Alerting 消息，等待用户应答。

⑧用户应答，端点 2 向端点 1 发送 Connect 消息，消息中带有端点 2 的 H.245 控制信道 TCP 端口号。至此，呼叫建立完成。

如果网闸不同意端点 2 接受此呼叫，则回送 ARJ，此时端点 2 将向端点 1 发送 Release Compelete 消息。

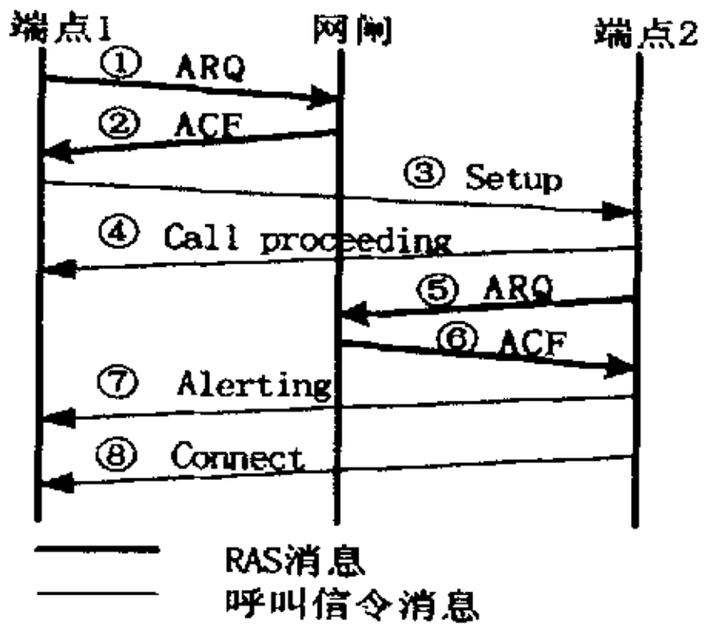

图 2.5 公共网闸（直接选路）信令过程

● 两端点在同一网闸上登记的网闸选路方式

其信令过程如图 2.6 所示。其差别在于：

①网闸向端点 1 回送 ACF 消息中包含的不是端点 2 的呼叫信令信道运输层地址，而是网闸自身的呼叫信令信道运输层地址。同时，网闸建立至端点 2 的呼叫信令信道。

②其后，端点 1 的呼叫信令消息只能发往网闸，再由网闸将其转发给端点 2。由于端点 2 只和网闸建有信令信道，因此其信令消息也只能发往网闸，再由网闸

转发给端点 1。

③呼叫建立成功时，端点 2 仍经 Connect 消息告之其 H.245 控制信道运输层地址，但网闸向端点 1 发送的 Connect 消息所含信息取决于 H.245 控制消息的传送方式。如果网闸决定采用直接传送方式传送媒体控制消息，则消息中包含的是端点 2 的 H.245 控制信道地址；如果采用转接方式，则消息中包含的是网闸的 H.245 控制信道地址，此时，网闸中一般包含 MC 功能。

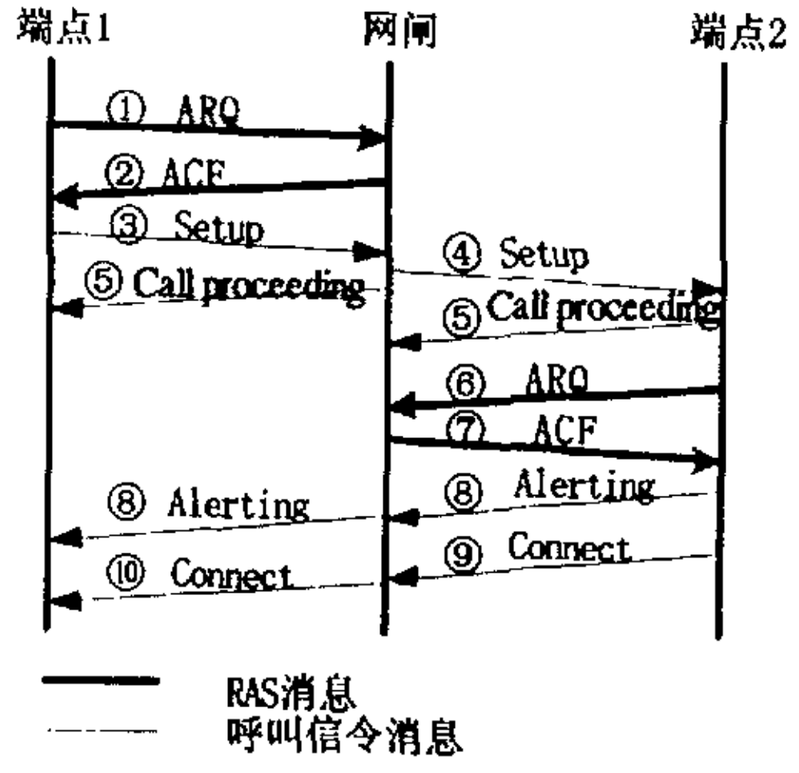

图 2.6 公共网闸（直接选路）信令过程

2.4.3 H.245 媒体通信控制协议

2.4.3.1 能力交换过程

能力交换过程是 H.225.0 呼叫建立成功后首先要执行的一个过程，它使通信双方了解对方发送和接收信号的能力。接收能力限定了对端发送信号的模式；发送能力给定了对端请求接收信号希望模式的协商范围。

对终端接收和/或发送能力的描述包含在终端能力集消息之中，不但给出终端可支持的各种媒体信号的操作模式，而且还给出终端同时处理多种媒体信号的可能的组合操作模式。消息中包含一个能力表，该表列出了终端所有允许的操作模式，如 G.723 音频、G.728 音频、H.263 视频等。每种模式对应能力表中的一个表项,赋予相应的序号（能力号）。

若干个能力号可以构成一个“可选能力集”（Alternative Capability Set）数据结构,表示该终端可以按其中的任一种方式工作。例如，可选能力集{G.711,G.723,G.728}表示终端可以采用其中任何一种音频编码方式，但是不能同时使用其中的两种或两种以上的模式。实际上可选能力集描述了终端的一个媒体信道的能力。

若干个可选能力集又构成一个“同时能力”（Simultaneous Capabilities Set）数据结构，表示该终端可以同时使用一组能力进行工作。例如，由可选能力集{H.261,H.263}和{G.711,G.723,G.728}组成的同时能力结构，它表示终端可以同

时工作于一个视频信道和一个音频信道，视频信道可以采用 H.261 或 H.263 编解码；音频信道可以采用 G.711、G.723 或 G.728 编解码。

最后，若干个同时能力又可构成一个“能力描述语集”（Capability Descriptors）数据结构,它包括一组能力描述语，每个描述语由一个同时能力和一个能力描述语序号组成。能力描述语集给出了终端的总体能力。之所以要定义多个描述语，是因为终端可能选用多种组合方式工作，在不同的组合方式下，各个信道允许的操作模式可能不同。

终端能力集消息至少应包含一个能力描述语。在通信过程中，允许终端发送新增或修改的能力描述语，以动态增加或删除终端能力。

本论文不涉及能力交换消息问题。

2.4.3.2 主从确定过程

主从确定过程用于会议通信之中，解决了两个均含 MC 功能的端点的 MC 冲突问题；也用于解决两个端点同时打开双向信道时的冲突问题。在建立信道连接之前必须解决端点之间的主从关系。

在执行此过程时，每个端点需生成一个随机数，称为“状态确定号”，其取值范围为 0～224－1，每个端点对于每个呼叫只能选定一个随机数。为了确定主从关系，任一个端点可向对方发送一个主从确定消息，该消息包含两个参数：状态确定号和终端类型。终端类型亦为一个整型数，其值按表 2.3 的规定确定。由于终端不可能有 MP 功能，MCU 不可能没有 MC，网闸没有 MC 功能就无需参加主从确定过程，因此表中相应元素无需考虑。

对方收到主从确定消息后执行确定计算过程：首先比较终端类型，终端类型级别高者为主机；若终端类型相同则比较状态确定号，大者为主机；若仍相同则返回确定拒绝消息，指示“数字相同”，此时本端重新生成一个状态确定号，再次启动主从确定过程。

表 2.2 用于主从确定过程的 H.323 终端类型取值

实体功能	H.323实体			
	终端	网关	网闸	MCU
实体不含MC功能	50	60	/	/
实体含MC，但不含MP功能	70	80	120	160
实体含MC和数据MP功能	/	90	130	170
实体含MC和数据、音频MP功能	/	100	140	180
实体含MC和数据、音频、视频MP	/	110	150	190

由表 2.2 可见，若两个实体为同样类型，则功能强的实体将为主机。如果两个实体为不同类型，有 MC 功能的高于无 MC 功能的实体；若均无 MC，则网关高于终端；若均有 MC，则优先级别自高到低顺序为：MCU、网闸、网关、普通终端。

如果一个 H.323 实体参与多个呼叫，则需根据其在呼叫中的功能确定其在该呼叫中的终端类型值，每个呼叫中的类型值可能会不相同。如果某个实体可以与表中的多个表元素匹配，则取最高值作为其类型值。

在会议通信中，如果某个 MC 成为主 MC，则其终端类型值立即上升为 240。也就是说，一旦某个 MC 在某次主从确定过程中赢得主机地位，它将在此会议中始终保持其主 MC 的地位。

能力交换和主从确定是 H.245 协议的两个初始过程，只有在这两个过程成功完成后，才能进行后续的信道建立过程。如果任一过程失败，必须至少重试两次，才能放弃连接建立并释放呼叫。

2.4.3.3 逻辑信道信令过程

在能力交换过程完成后，端点就可以根据对方的接收能力发起媒体信道建立过程。

- 单向信道打开过程

信道打开恒由发送方启动。它向接收方发送打开逻辑信道消息，消息包含前向逻辑信道号及信道参数。其中，信道号必须由发送方赋值，证实消息返回此值，以和请求消息匹配。信道参数包括数据类型、媒体信息是否需要确保传送、是否执行静音抑制、目的地终端标记等。如果该信道用来传送 RTP 封装的实时媒体信息（如音频或视频），则信道参数还应包括：

①会话标识：即 RTP 会话标识。

②关联会话标识：若为可视电话应用，则会话标识对应音频 RTP 流，关联会话标识对应视频 RTP 流，二者应保持唇同步。

③媒体控制信道：反向 RTCP 信道的传输层地址。

接收方收到打开逻辑信道消息并确认后，给发送方返回打开逻辑信道确认消息，消息包含前向逻辑信道号和前向复用证实参数。如果传送的是 RTP 流，则证实参数中应包含：

④媒体信道：前向信道的 RTP 传输层地址。

⑤媒体控制信道：前向 RTCP 信道的传输层地址。

这样，经由此过程，发送方和接收方之间建立起了前向 RTP 信道和双向 RTCP 信道。“前向”指的是打开逻辑信道消息发送方至接收方的方向。需要注意的是，在打开单向信道的过程中，消息不含反向信道参数。

如果某 RTP 流的前向信道已经建立，接着又由对方发起建立反向信道，则应保证本次建立过程中交换的 RTCP 信道的传输层地址和前向信道建立时交换的地址相同。通过 RTP 会话标识可以判定这两个信道和同一个 RTP 流相关。如果双方同时发起对同一 RTP 会话的信道建立过程，则主机方应拒绝对方的请求。其后从主机方在重新发起反方向的信道建立。

● 双向信道打开过程

打开双向信道的过程和打开单向信道的过程基本相同，其主要差别在于消息中还包含反向信道参数，因此一次消息交换同时建立两个方向的信道。此外，请求方收到对端的证实消息后，还需发回一个确认消息，表示反向信道建立成功，可以开始传送信息。

如果对端不能支持本端要求的反向信道，则回送拒绝消息，然后立即启动双向信道打开过程，其反向信道参数等于本端发送的前向信道参数，其前向信道参数为双方都能支持的参数值。如果发生双方同时启动同一个双向信道打开过程，则执行主机方发起的请求。冲突检测方法是双方建立信道的参数相同，或者虽然参数不尽相同，但可判断为用于同一目的。

2.4.3.4 呼叫释放过程

通信的任何一方都可以发起呼叫中止过程，其信令过程如图 2.11 所示。为简单计，假设呼叫信令和 H.245 控制信令均采用直接选路方式，其释放步骤为：

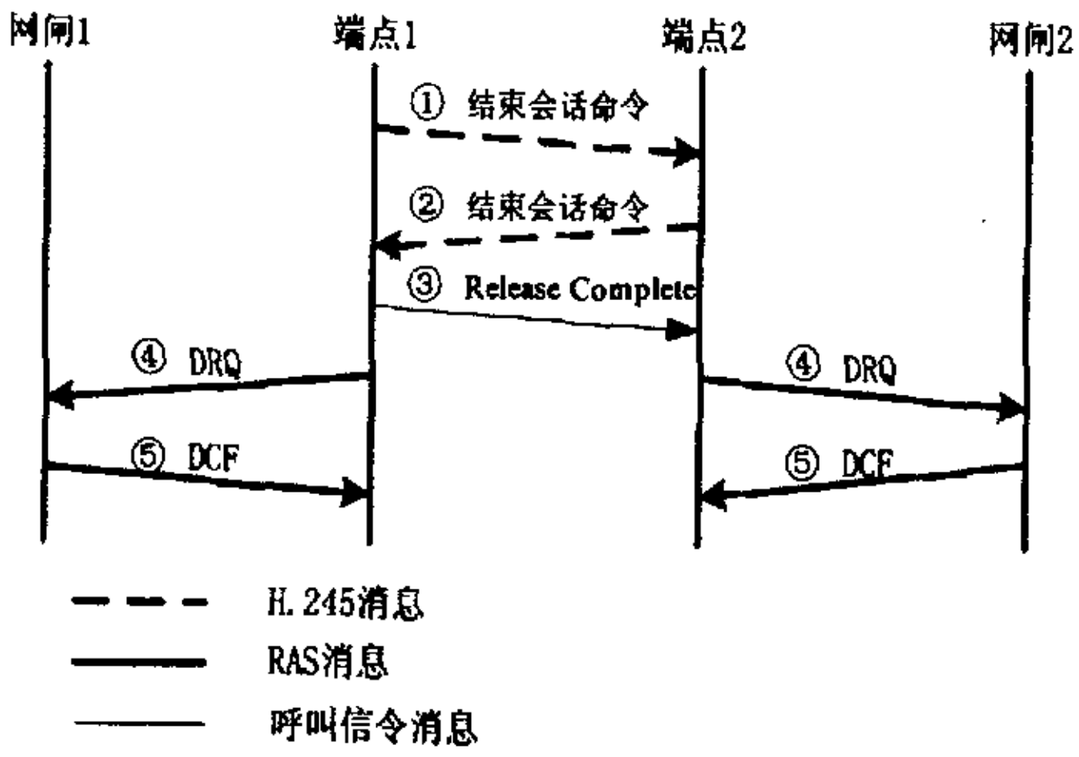

图 2.7 端点发起的呼叫释放过程

①端点 1 停止在逻辑信道上传送信息，关闭所有逻辑信道。然后在 H.245 控制信道上向端点 2 发送“结束会话”命令消息，告诉对方要结束该呼叫，其后停止发送 H.245 控制消息。

②端点 2 收到上述消息后，关闭所有逻辑信道，向端点 1 回送“结束会话”命令消息。至此，H.245 控制信道关闭。

③如果呼叫信令信道尚未关闭，端点 1 向端点 2 发送 H.225.0Release Complete 消息，关闭此信道。至此，呼叫已释放。

④端点 1 和端点 2 分别向其网闸发送 RAS 消息 DRQ，告之该呼叫占用的带宽资源可予释放。

⑤网闸向端点回送 DCF。至此，完成全部释放过程。

如果是会议呼叫，则首先用 H.245“退出会议”消息明确中止会议，然后各端

点等待 MC 启动呼叫释放过程。

呼叫释放过程也可由网闸发起，其释放过程和步骤不再赘述。

2.4 H.323 系统部件之间的通信

H.323 系统部件之间的通信通过各种消息流来进行，这些消息流可归为以下五类：呼叫控制信令、通信控制信令、视频流、音频流和数据流。

呼叫控制信令用于呼叫的建立、呼叫的拆除以及其他一些控制功能。

通信控制信令主要用于能力协商、打开及关闭逻辑信道、模式控制及其他一些通信控制功能。

视频流包含编码后的视频数据及相应的视频控制信号。

音频流包含编码后的音频数据及相应的音频控制信号。

数据流包括静止图像、传真、文档和计算机文件等。

主席控制模式下 H.323 系统各部件即终端、网闸和多点控制单元中的多点控制器和多点处理器的通信过程如图 2.8 所示。

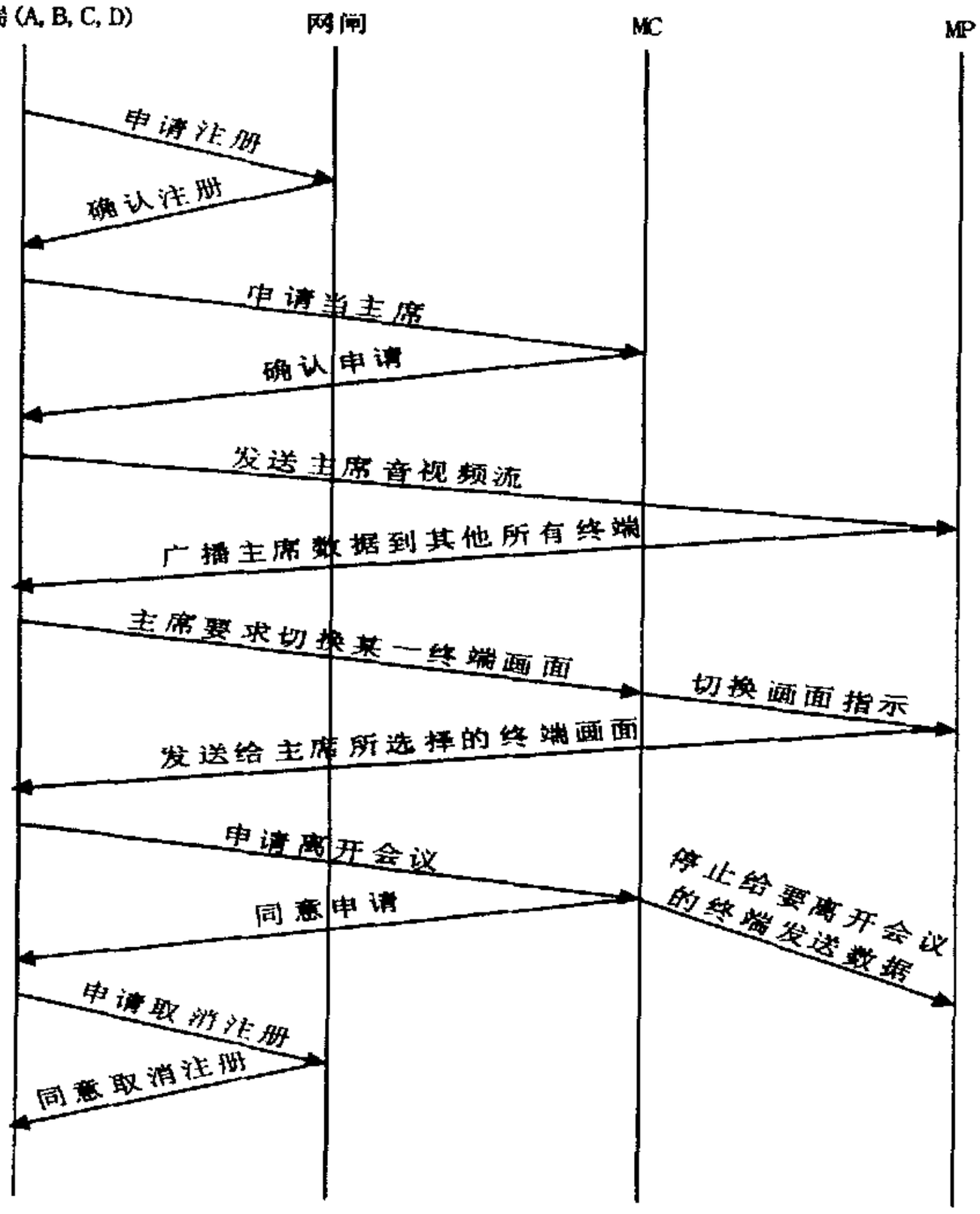

图 2.8 主席控制模式下各组件的通信过程

第三章 H.323 系统的多点视频会议及 MCU

多点视频会议是三个以上的会议终端之间的会议，与会者从终端屏幕上可以看到其他与会者，身在不同地方的人们可以互相传输声音和图像信号，进行“面对面”会议，并能互相查询文件、传送重要文件、图表，讨论问题并加以修改，在终端上建立一个虚拟会议环境。

要在 H.323 端点之间实现多点会议，必须有多点控制器（MC）和多点处理器（MP）的支持，根据多点会议的类型不同，MC 和 MP 所处的位置也不同，它们既可以处在多点会议控制单元（MCU）中，也可分布在其他 H.323 端点（终端、网闸）中。而本论文的多点控制器和多点处理器同处在多点控制单元中。MCU 已经在第二章做过介绍，在此不再赘述。本章首先介绍基于 H.323 标准的多点会议的类型，再分析会议控制的一般机理，最后介绍多点控制单元 MCU 的实现。

3.1 多点会议系统的类型

多点会议系统按媒体通信方式和形成方式，大致可分为：集中型多点会议系统、分散型多点会议系统、混合多点会议系统，其中混合多点会议系统又分为集中音频的混合多点会议系统和集中视频的混合多点会议系统。

1．集中型多点会议

在该类会议中，所有参会终端（包括网关）和 MCU 中的 MC 建立 H.245 控制信道的点到点联系，该 MC 执行对整个会议的集中控制。

各终端的音频、视频和数据信道和 MCU 中的 MP 相连。该 MP 对各终端送来的信号进行音频混合、视频交换或混合和 T.120 数据分配，然后将处理所得的音频、视频和数据流再送回各终端。MP 可以选择转送哪一个终端或哪几个终端的信号，也可以对不同的音频、视频和数据格式和比特率进行转换，使得各个参加会议的终端可以使用不同的通信模式。视频信号还可以采用多播方式分发至各终端。

支持该类会议的 MC 所在实体 MCU 发送 H.245 终端能力集消息时，其复用能力单元的 MC 能力字段应指示具有集中式会议 MC 功能。参加该类会议的终端和网关，其终端能力集/复用能力/接收多点能力/媒体分配能力应设定为：集中式控制、集中型音频、集中型视频、集中型数据。在 H.323 系统中，所有终端都应具有集中型多点会议能力。

图 3.1 为集中型多点会议结构示意。其中，虚线表示控制信道，传送 H.245 控制消息；实线表示逻辑信道，传送媒体信息。

本论文是基于集中型多点会议形式的。

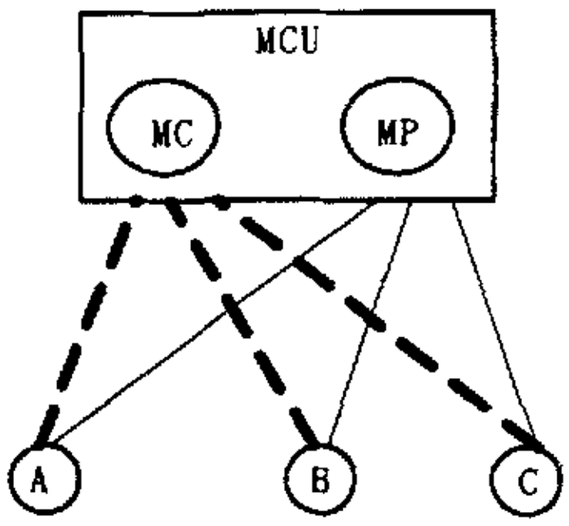

图 3.1 集中型多点会议

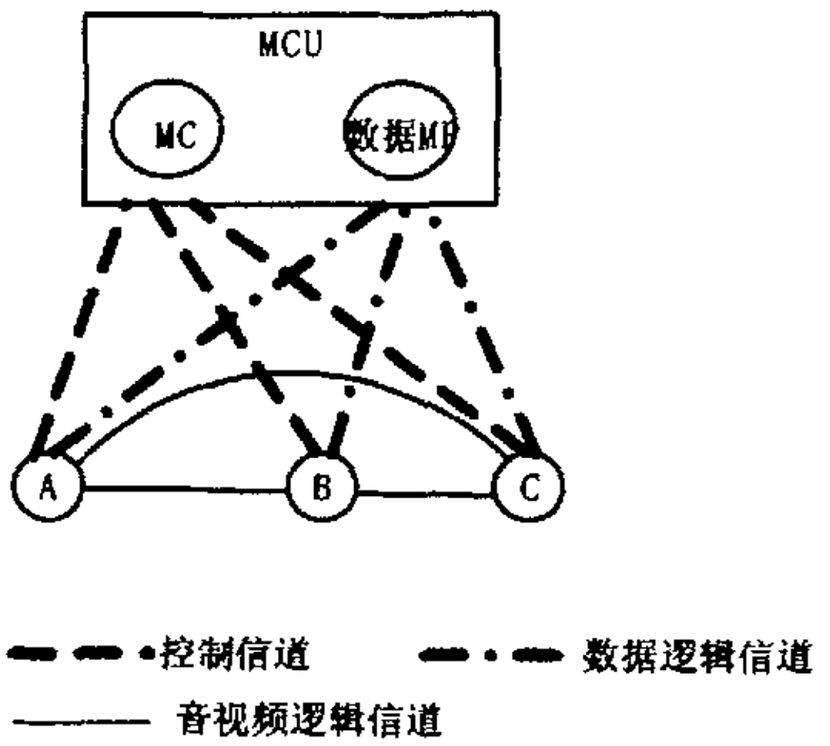

图 3.2 分散型多点会议

2. 分散型多点会议

该类会议的终端仍然以点到点的方式和 MC 建立 H.245 通信信道连接，MC 可位于 MCU、网关、网闸或某个终端中。各终端的数据信号一般仍通过 MP 集中分发，但音频和视频信号则直接发送至其它终端，其发送方法有两种：多播方式和多重单播方式。所谓多重单播指的是每两个终端之间都建有单播信道。这种管理方式没有集中控制和集中管理的设备，MCU 的功能以 MC 和 MP 功能模块形式分别存在于系统的其它设备（如终端、网关和网闸）中。MC 仍然具有对会议的控制功能。图 3.2 为分散型多点会议结构示意。

3. 混合型多点会议（集中型音频）

该类会议的终端和 MC 之间仍然是点到点的 H.245 控制连接。音频和数据信道连接至 MP，由 MP 进行音频混合，并能为每个终端发送不同的音频组合信号。视频信号则经由多播或多重单播方式在终端之间直接传送。终端的媒体分配能力设定为：集中式控制、集中型音频、分散型视频、集中型数据。

4. 混合型多点会议（集中型视频）

该类会型和第 3 种会议类似，只是音频信号采用直接传送方式，视频信号经由 MP 集中分发，可以根据需要为每个终端发送不同的视频流，也可以按多播方式统一发送，以降低网络带宽消耗。终端的媒体分配能力设定为：集中式控制、分散型音频、集中型视频、集中型数据。

5. 特殊型多点会议（Ad Hoc Multipoint Conference）

这类会议开始时是一个点到点会议，以后在呼叫过程中通过邀请他人或他人主动加入的方式扩展成为多点会议。它要求在初始的点到点呼叫中必须有一个 MC。其可能的情况包括：其中一个终端含有或两个终端都含有 MC；呼叫控制通过网闸转接，该网闸具有 MC 功能虽然只是两方呼叫，但也是通过 MCU 作为多点呼叫来处理的。

该类会议的扩展控制有其特殊性，H.323 专门对其信令过程作了详尽的规定。

H.323 标准规定，H.323 终端和 MCU 必须支持集中式多点会议，而对分布式和混合式多点会议的支持是可选的。表 3.1 列出了各种多点会议系统的特征。

表 3.1 四种多点会议的特征

会议类型	集中型	分散型	集中音频的混合	集中视频的混合
支持类型	强制	可选	可选	可选
音频通信模式	点对点	多播	点对点	多播
视频通信模式	点对点	多播	多播	点对点
数据通信模式	点对点	点对点	点对点	点对点
MC 所在位置	MCU	终端 网闸 网关	MCU	MCU
MP 所在位置	MCU	终端	音频：MCU 视频：终端 数据：所有端点	音频：终端 视频：MCU 数据：所有端点

3.2 多点会议系统的组网方案

在局域网上实现集中式多点会议，星形组网方案是我们的首先方案。星形局域网组网方案将所有终端通过集线器或交换机连接到多点控制单元，由多点控制单元完成信息流的广播或转发。在本系统中，多点控制单元是一个高性能服务器，它能完成多点控制器和多点处理器功能，因此又把多点控制单元称为多点控制服务器。这种组网方式具有以下特点：任一终端只和多点控制服务器通信，会议的控制命令只能通过多点控制服务器器发送给各组件；网络中只包含一个多点控制器。这种集中控制和管理机制，能简化网络结构和系统程序设计。多点控制服务器完成信息的广播、组播、或转发，另外还完成码流同步、视频画面合成和音频混合等操作。但后三种操作一般必须对接收到的信息码流进行重新编码后才能够完成。图 3.3 为多点视频会议系统的网络结构。

3.3 会议控制一般机理

由上可知，各种型式的集中控制式多点会议的共同特点是，会议各类媒体信道的连接控制都集中归于 MC。MC 利用 H.245 通用控制消息和有关会议控制的一下特定消息完成对各终端和会议功能的控制。下面概要说明控制机理的要点。

1．呼叫建立

每一端点应与 MC 完成呼叫建立，建立至 MC 的 H.245 控制信道。由于 MC 本身并无独立的地址，无法作为呼叫方，因此一般 MC 是位于 MCU 中的，但也可能位于网闸或终端中。

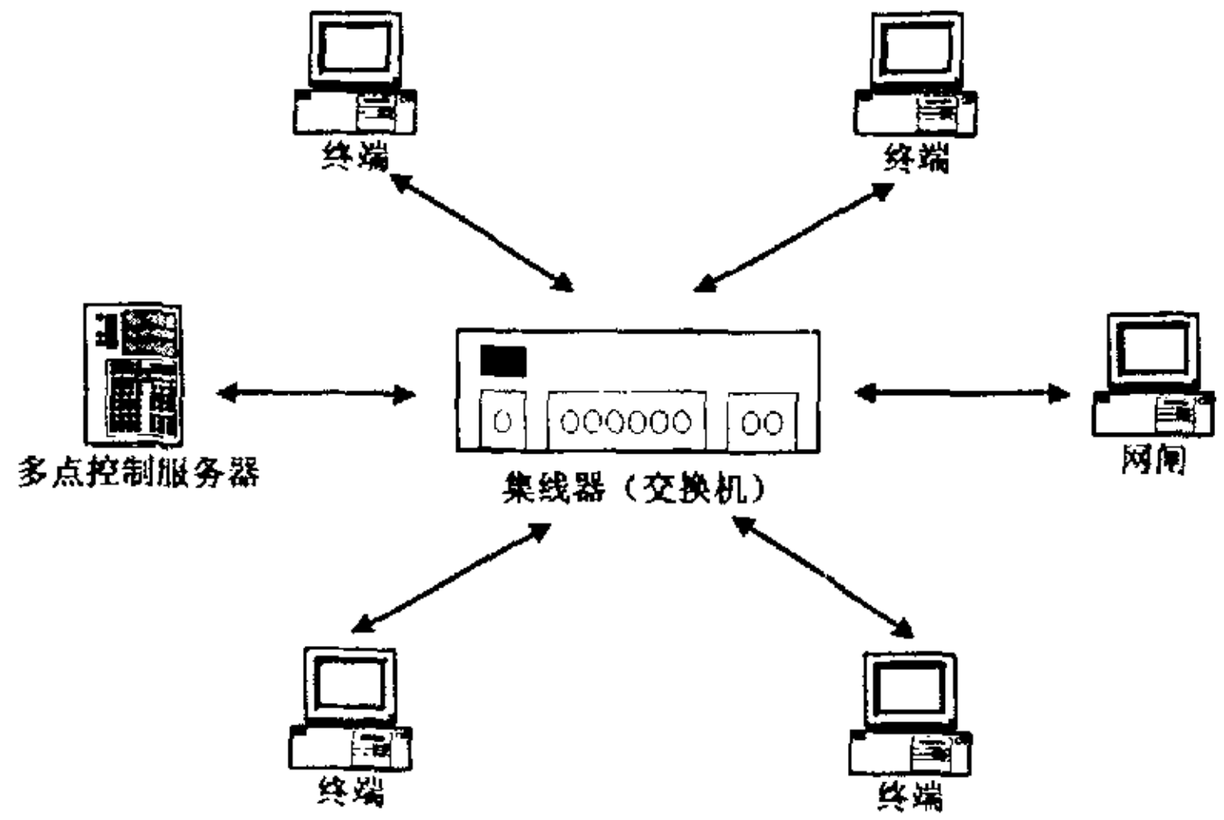

图 3.3　多点视频会议系统的网络结构

．会议标识（CID）：唯一标识会议的全局号码，关联该会议所属所有呼叫的信令消息，由会议发起端点创建。CID 由 16 个八位位组组成：

CID 八位位组	15~10	9~8	7	6	5~4	3~0
八位位组名	N5：N0	C1：C0	H1	AV	M1：M0	L3：L0

其中，0 为最低八位位组。

N5：N0 为 48 位 LAN 地址。如无此地址，则以随机数充填。

C1：C0 为 16 比特计数器，每次会议加 1。

H1、A、M1：M0、L3：L0 为 100ns 时钟计时值的最低 60 比特，时钟起始时间是当地时区 1582 年 10 月 15 日。比特排列顺序为：

H1		AV		M1	M0	L3	L2	L1	L0
59	52	51	48	47	32	31			0

V 为 4 比特的版本号，位于 CID 字节 6 的低 4 比特。由此可见，这样构成的 CID 在整个 LAN 范围内必定是唯一的。

．会议目的：指示建立该呼叫的目的。可取值为：创建（创建新会议）、加入（加入已有会议）、邀请（邀请新端点加入已有会议）。

和 MC 建立 H.245 控制连接后，利用主从确定过程确定主 MC，即经由 MC 位置指示消息通告主 MC 的地址，然后利用能力交换过程选定会议型式（集中型或分散型），会议型式的选择受限于端点和 MC 的能力。对于每个新加入的端点，MC 用“终端号分配”消息赋予终端号，用“终端加入会议”消息通知其它端点新成员的加入，新端点也可用“终端清单请求”消息请求其它端点的名单。

2．通信模式的确定

该过程就是确定各个逻辑信道的媒体类型、编码方式、媒体信道和媒体信道运输层地址等参数。

在单播情况下，端点通过“打开逻辑信道”和“打开逻辑信道证实”消息建立至 MCU 或另一端点的逻辑信道。

在多重单播情况下，端点必须逐一打开至所有其他端点的逻辑信道。由于端点只和 MC 建有 H.245 控制信道，因此“打开逻辑信道”消息必须送往 MC，消息中带有逻辑信道对端的终端号，然后由 MC 转送给对端。端点收到回送的“打开逻辑信道证实”消息，可根据消息中的“前向逻辑信道号”和原请求消息匹配。

多播情况下，则由 MC 分配多播地址并决定通信模式，经由“通信模式信令”消息告知每个端点。然后，由各端点向 MC 发送“打开逻辑信道”消息，消息中带有该多播地址；MC 根据多播地址将此消息转发给每个接收端点。

3. 逻辑信道和 RTP 流的关联

多播情况下，接收端点会从同一端口收到来自不同源的 RTP 流，在处理时需要予以识别。为此，H.245 规定端点在发送“打开逻辑信道”消息时，应将 MC 分配给它的终端标记置入逻辑信道参数的源字段；目的地字段空缺，表示该信道的媒体流是多播流。同时，约定源端点发送的 RTP 流的源标记 SSRC 的最低字节取为该端点终端标记的最低字节。这样，目的端点收到 RTP 流后，通过比较 SSRC 和逻辑信道参数源字段的最低比特就可判定该媒体流的所属逻辑信道。

4. 速率匹配

在多点会议中，各终端可能会以不同比特率发送信号。为了使发送能力和接收能力匹配，并使各个终端地位平等，MC 可以通过发送“流量控制命令”消息来限定各终端的发送比特率。如有新的终端加入会议，MC 将向其发送“多点会议”指示消息，要求它遵从比特率均等的安排。

5. 加密

在集中型多点会议中，MP 被认为是可信任实体。它可以将来自各端点的信息流解密，进行必要的处理，然后将处理后的复合信息重新加密后送往各端点。

6. 会议控制

H.245 专门定义了一组会议请求及响应消息和一组会议命令消息，供 MC 在通信过程中对会议进行控制。例如，在会议中，终端可以请求参会终端清单、充当主席、退出会议等，MC 可以要求终端输入口令、终端标识，命令结束会议等。

7. 同步

在集中式或混合式多点会议中，为了保证音频和视频的同步，提供音频混合的 MP 应该修改音频流和视频流的时间戳，实现多点会议中的多点音频同步。当 MP 把进入 MP 的音频流和/或视频流经过处理产生新的媒体流时，它还应该在产生的音频或视频流包上打上自己的序列号。

3.4 MCU 的实现

MCU 提供在以太网上三个以上的终端参与多点会议的能力，它通常由一个 MC 和零个或多个 MP 组成。最典型的集中型多点会议的 MCU 由一个 MC、一个音频多点处理器、一个视频多点处理器、和一个数据多点处理器组成；典型的分散型多点会议的 MCU 由一个 MC 和一个数据多点处理器组成，这是由于该会议的音视频信号都是由各终端通过多播方式直接播发。

第二章已经介绍过 MC 和 MP，在此不再作介绍。

3.4.1 H.323 终端的注册登记过程

H.323 标准框架中的 H..225.0 定义了初始的呼叫控制和终端对网闸的登记、申请等消息。RAS 信令功能使用 H.225.0 消息在终端和 GateKeeper 之间完成登录、许可权认证、带宽改变、状态和脱离过程。在没有 GateKeeper 的网络环境下，RAS 信令信道将不被使用。在包含一个 GateKeeper 的网络环境下，RAS 信令信道在终端和 GateKeeper 之间打开，并且 RAS 信令信道的打开在 H.323 终端的任何其他信道的建立之前进行。当 H..225.0 消息建立了初始呼叫之后，H.245 规定了一系列握手信号来完成通道控制。

H.323 标准的呼叫连接是终端和 GateKeeper 之间的通信过程。终端 A 或终端 B 在参与某一会议之前，必须要在 GateKeeper 上注册登记。具体过程如下：

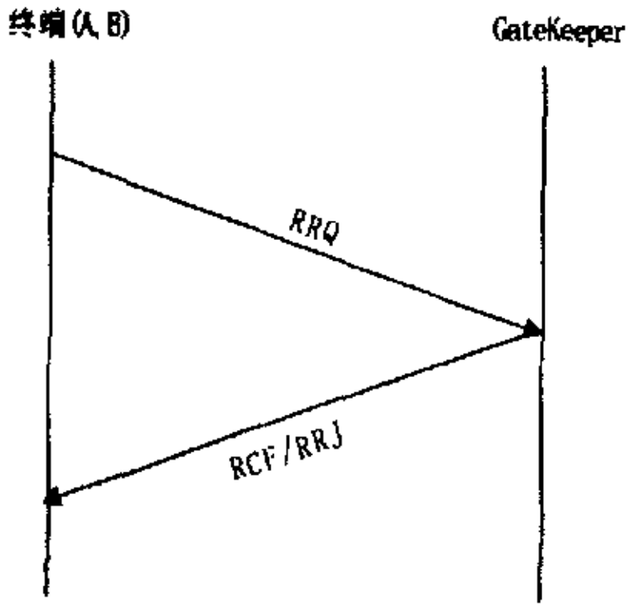

图 3.4 终端申请注册的通信过程

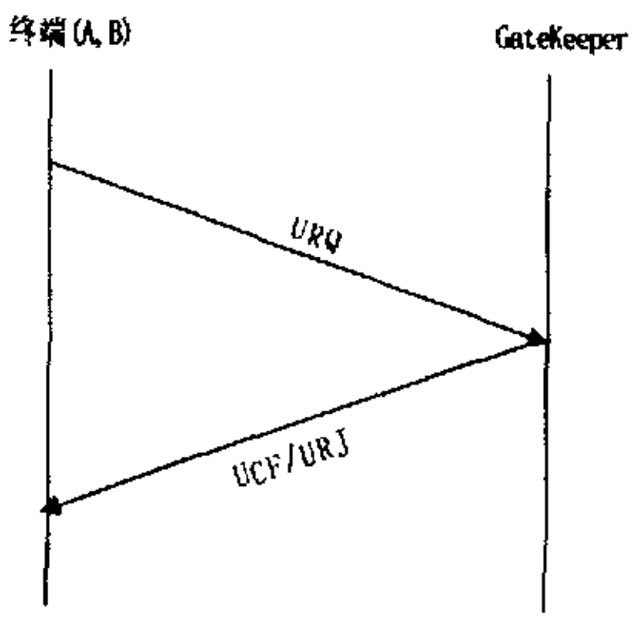

图 3.5 终端取消注册的通信过程

若终端要参加会议，它必须首先在 GateKeeper 上注册登记，GateKeeper 把申请加入会议的终端列入与会者名单，通信过程如图 3.4 所示。当终端在 GateKeeper 上登记时，先发送请求注册登记的命令（RRQ），同时将需要登记的信息（别名、地址等）随之传送过去。网闸接受登记则回发 RCF 作为注册确认，至此登记完成。若终端要取消注册，则发送取消登记注册请求（URQ）给网闸，网闸从与会者名单中删除此终端并回复 UCF 作为确认或回复 URJ 作为拒绝取消注册。通信过程如图 3.5 所示。

3.4.2 基于 H.323 标准的多点控制功能

有多个终端参与的多点会议的控制功能由MCU中的MC来完成。会议开始时，终端与 MC 通过 H.245 控制信息相连，H.245 控制功能使用 H.245 信道来携带端到端的控制信息，这些控制信息用来管理 H.323 实体的操作，包括能力交换、打开和关闭逻辑信道、模式选项要求和一般的指示和命令。H.245 信道在两个终端、一个终端和 MC 或者一个终端和一个网闸之间建立。

多点控制器还为会议选定会议模式。在会议的进行过程中，可以动态的指定会议模式为主席模式或者非主席模式。在主席模式下，终端有三种可能的身份：主席、选定听众以及一般听众，除主席外，所有终端接收的视频源均为主席的，即它们收看主席的图像，而主席则收看选定听众的图像。在非主席模式下，采用语音激励方式进行数据的转发，这时，所有会议成员（除发言者外）均接收发言者的信号源，而发言者则接收上一发言者的信号源。

3.4.3 基于 H.323 标准的多点处理功能

多点处理功能由多点处理器（MP）来完成，它主要包括音频信号和视频信号的处理。对音频信号的处理主要是指音频混合；而对于视频来说，则分为主席模式和非主席模式两种情况下的视频交换和视频合成。软件设计中，每一个与会终端在 MP 中都要有与之对应的一对线程：接收线程和发送线程。同时在 MCU 上分别设计了音频和视频缓冲区来接收、处理及发送音频流和视频流。

3.4.3.1 音频信号的处理

在一个会议组中，为便于进行语音混合，我们为每个与 MCU 建立呼叫连接的终端均设计了 M 个音频缓冲区，其中，M=（与 MCU 建立呼叫连接的总终端数 - 1）。即假设一个会议组中有 N 个终端与 MCU 建立了呼叫连接，则 MCU 为每一个终端均准备了 M=（N-1）个缓冲区，这(N-1)个缓冲区分别用来存放从除本终端外其它终端接收到的音频数据，MCU 对它们进行音频混合后再回送给这一个对应的终端。据此推断，MCU 中共计有音频缓冲区 N*(N-1)个。每增加一个建立连接的新终端，我们都要新开辟多个缓冲区，首先要为其它终端都开辟一个缓冲区用于存放从这个新终端接收到的音频数据，然后还要为这个新增终端开辟多个缓冲区用于存放从其它终端接收到的音频数据。图 3.6 给出了仅有三个终端与 MCU 建立连接的情况下，缓冲区的分配情况。

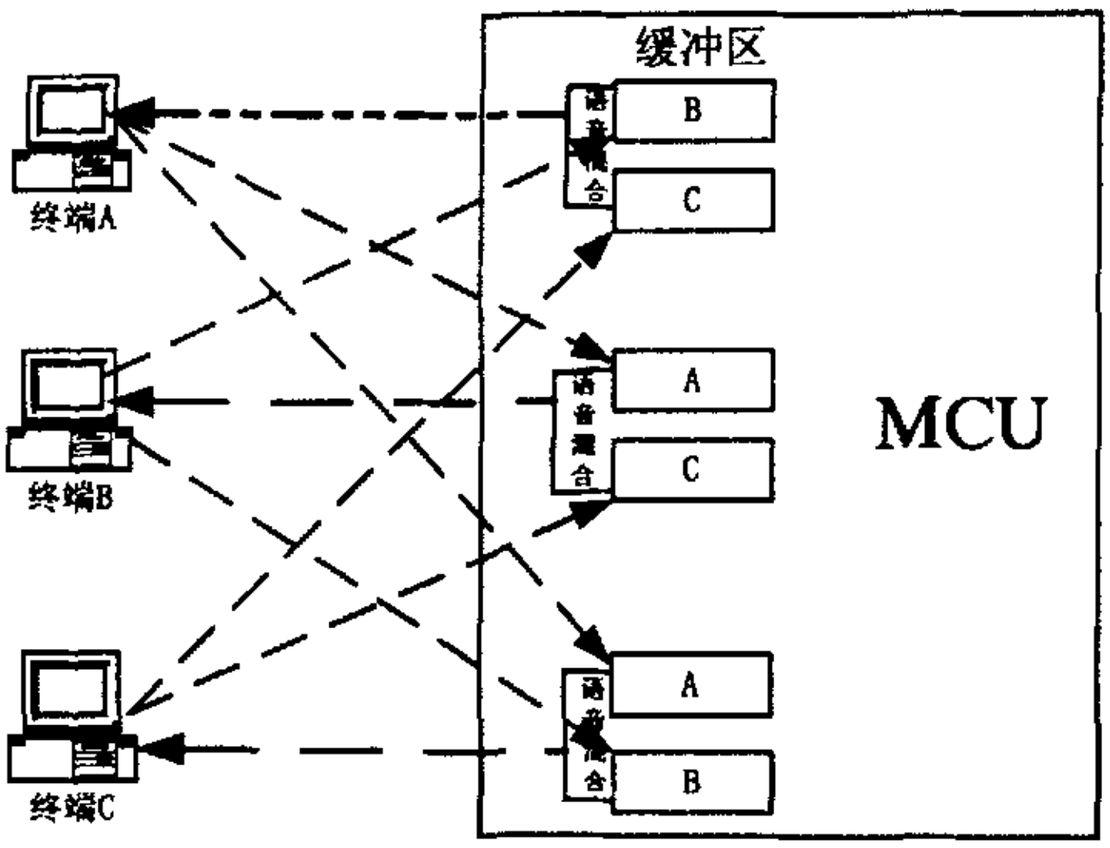

图 3.6　MCU 中音频缓冲区的分配

对于 MCU 的音频处理方式，采用了事件驱动的多线程机制。对于终端与 MCU 建立的每个呼叫连接都对应有两个线程：接收线程负责将从网络上接收到的音频数据送去解码器进行解码，并将得到的解码流存放入用于存放这个终端的音频信号的缓冲区；而发送线程负责将已混合的音频数据送去给编码器进行编码，并将得到的编码流发送出去。MCU 将与本连接对应的多个音频缓冲区中的数据进行音频混合的过程并不在这两个线程当中。这两个线程的公用资源便是应用程序中的这多个缓冲区。

由于两个线程之间存在有共享资源，为避免由于资源共享而引起的竞争、死锁等错误，MCU 软件采用了事件和互斥方法来达到线程之间的同步。事件 1 是由接收线程将解码流送入对应缓冲区，缓冲区被更新；事件 2 是读取缓冲区中的数据并对其进行语音混合。事件与线程之间的关系如下：接收线程挂起直到事件 2 发生才继续执行，发送线程被挂起直到事件 1 发生才继续执行，这样，就可以保证线程间安全的同步了。

3.4.3.2　视频信号的处理

对于视频来说，我们为每个与 MCU 建立连接的终端仅仅准备了一个缓冲区，它既是接收缓冲区也是发送缓冲区。在主席模式下，对接收到的视频数据，MCU 需先判断是否是来自主席或选定听众的，若是主席的，则将它们分别存放入与其它所有终端相对应的缓冲区中，若是选定听众的，则将其存放入为主席准备的缓冲区中，若是来自其它终端的，则 MCU 不予理会。发送时，只需将本缓冲区中的数据发往与本缓冲区对应的呼叫连接即可。在非主席模式下，MCU 在接收到来自各终端音频数据时就要进行判断，选其音频最高的作为当前发言者，对于视频来说，MCU 需将当前发言者的视频数据存放入与其它所有终端相对应的缓冲区中，而与当前发言者相对应的缓冲区中则存放从上一个发言者接收到视频数据。图 3.7

给出了主席模式下缓冲区的分配情况，而非主席模式下的情形类似。

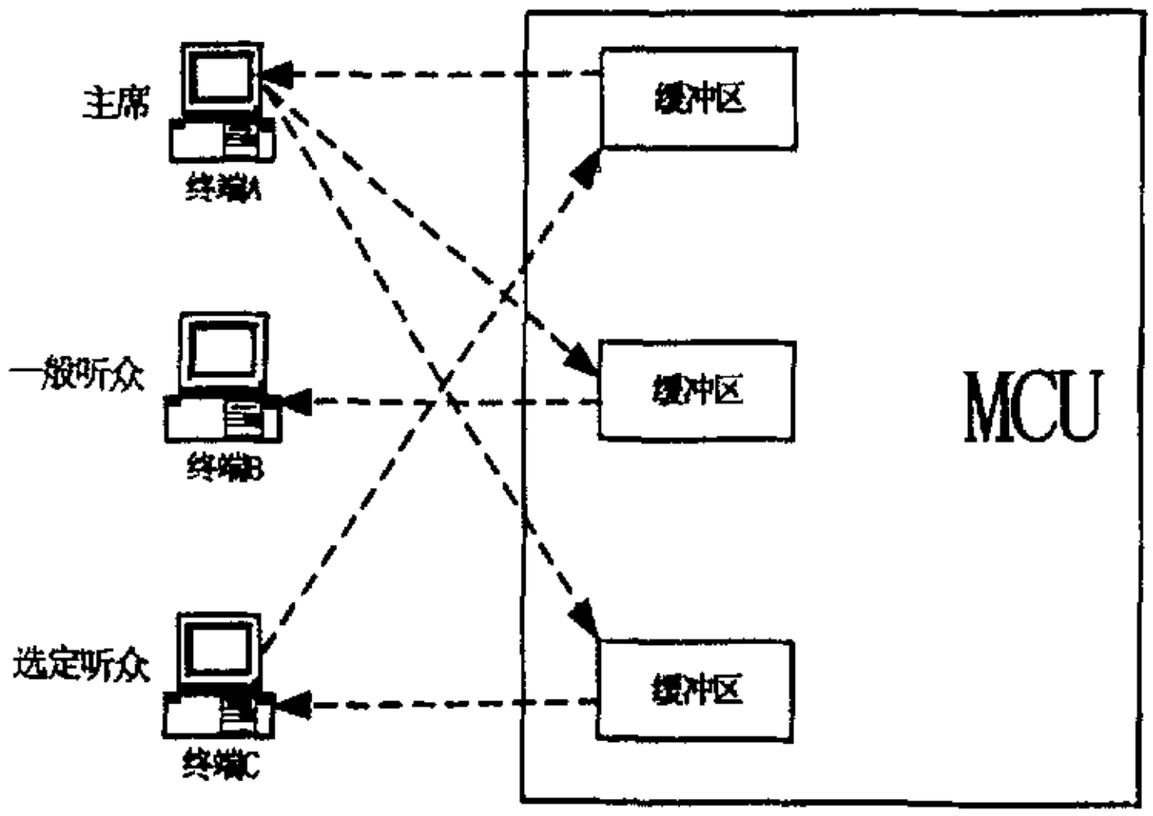

图 3.7 主席模式下 MCU 的视频缓冲区的分配

若 MCU 要进行四画面合成，则情形又要稍复杂一些。需要合成的四个源终端可由用户操作 MCU 的图形界面确定，MCU 将为这四个源终端另外创建四个缓冲区，分别用来存放这四个源终端的视频数据，MCU 将这四路视频数据合成后再送到与各终端相对应的缓冲区中，发送情况和单画面时相同。

对于视频数据的处理方式，采用和音频数据相似的事件驱动的多线程机制。

第四章 MCU 性能测试方案

随着网络技术与多媒体技术的发展，计算机网络已逐渐演变成丰富多彩的多媒体信息网，多媒体应用异军突起，尤其是实时多媒体会议，它能广泛用于社会各行各业，具有广阔的市场前景，因而是当前研究和开发的热点。西安邮电学院通信技术研究所开发出了基于 H.323 的视频会议系统，其中多点控制单元（MCU）是会议通信的重要设备，它相当于一个交换机（服务器），是视频会议系统的控制和数据交换中心，不仅要管理和控制会议，而且还要对各个终端送来的信号进行音频混合、视频交换或混合和 T.120 数据分配，然后将处理后的音视频流和数据流再送回参加会议的终端。目前，大部分视频会议采用的是集中式多点会议，所有终端以点对点方式与 MCU 建立连接并通信，MCU 的负载势必过重，网络带宽的利用率降低，从而参加会议的终端数目受到限制，所以系统对 MCU 的性能要求比较高，MCU 的性能好坏决定了整个系统的优劣程度，从而涉及到 MCU 性能（接入量、时延抖动、处理不同运动程度图像的能力等）测试的问题。

在本论文方案提出以前，业界内并无公开的专门的测试方法，因此很有必要提出一种测试 MCU 性能的方法。

本章将首先提出了MCU性能测试问题并加以分析，接着提出了一种测试MCU性能的方案，最后分析了本方案的优点及不足之处。

4.1 MCU 性能测试问题的提出和分析

IP 网络视频会议系统是多媒体应用的一个主流方向，也是用户迫切需要的一种多媒体服务。在整个系统中多点控制单元（MCU）是会议通信的关键设备，其性能的好坏直接影响着整个系统的优劣。近几年来以 H.323 标准为基础的 IP 网络会议系统得到迅速发展，随着高速网络建设步伐的加快和会议规模的扩大，采用集中控制的 H.323 系统中执行会议控制功能的多点控制单元（MCU）可能成为系统瓶颈。而在包括音频、视频和数据的视频会议呼叫中，视频部分通常占用整个呼叫可用带宽的绝大部分，因此很有必要就 MCU 的性能进行分析，特别是 MCU 对运动程度不同（缓慢、一般、剧烈）的图像的处理性能，即传输这三种图像时，在保证图像质量的前提下 MCU 能接入的最大终端数目和最小通信带宽，同时分析评价 MCU 性能的几个性能指标。

4.1.1 提出 MCU 性能测试问题

一般的测试 MCU 性能的方法是接入几台终端直接对 MCU 发起呼叫，再接入相同数目的终端用于显示发送和接收的音视频信号，用来测试 MCU 是否能接收这

些终端对它的呼叫，或是否能成功的呼叫这些终端，或在其和这些终端建立呼叫连接（connection）后，MCU 是否能正确地处理所接收到的音视频信号并将其回送给相应的终端，以及其他性能。该方法理论上是合理的，但当需要接入的终端的数量比较大（如 16）时，这种方法其实是不太现实的，也就是说传统的测试方法不能用于测试 MCU 的性能，我们必须另行设计一种方法。因此，我们的任务是设计出一种测试设备，它可以模拟多个终端的行为，如可以对 MCU 发起多路呼叫；可以接收 MCU 对其发起的多路呼叫；在与 MCU 建立了多路呼叫连接后，每一路呼叫连接都可以打开一组媒体通信信道，并可以在这一组媒体信道上传送和接收属于本路呼叫连接的一组媒体信息。即使我们能设计出这样一种设备，它可以实现对 MCU 进行 n 次呼叫，即可以模拟 n 个终端接收 MCU 对其发出的 n 次呼叫，并且能与 MCU 建立 n 个呼叫连接。但由于任何一个设备都不可能具有无限多个 IP 地址，那么 MCU 如何能区分这 n 个呼叫连接，从而给每个呼叫连接的发起方指定一个与会身份，并根据不同的与会身份，对所接收到的来自不同的与会身份的呼叫连接的音视频数据进行不同的处理并回送呢？还有，所设计的设备能否像真正的终端一样产生音视频信号？

基于上述的分析，本文主要致力于设计并实现一个能够模拟多个终端行为的测试软件，而后提出一套完整的利用该测试软件来测试 MCU 性能的方案，最终实现该方案。

4.1.2 影响 MCU 性能的参数分析[15]

影响 MCU 性能的因素有很多，如可用的网络带宽资源的多少，视频传输速率高低，MCU 软件本身的性能优劣以及运行 MCU 软件程序的计算机的内存大小和 CPU 的速度快慢等。

首先来看可用的网络带宽资源的多少和视频传输速率高低这两个因素对 MCU 性能的影响。在视频会议中，MCU 要对音视频信号进行接收、处理和转发，而视频信号占用的带宽资源较多，一般视频的传输速率为 P×64kbit/s（P=1, 2, 3, ...）。当视频信号的传输速率越高时，其所占用的带宽资源就越多。所以，可用的网络带宽资源就成了 MCU 性能的限制因素。在可用的网络带宽资源一定并且保证音视频信号的质量的情况下，视频的传输速率越低，MCU 的性能越好；而在视频传输速率一定的情况下，可用的网络带宽资源越多，MCU 的性能越好。

接着看运行 MCU 软件程序所用计算机的内存大小和 CPU 速度快慢对 MCU 性能的影响。一般的 MCU 设计都需要从内存中开辟一定的缓冲区域用于存放接收到的音视频流和欲发送的音视频流，并且由于音视频流的实时性要求 MCU 具有较高的处理速度，能很快地将接收到的音视频流进行处理后回送给各终端，因此，视频会议系统对运行 MCU 程序的计算机的内存大小和 CPU 速度具有很高的要求。若内存太小以致没有足够的空间用于存放接收到的音视频信号，那么就会有一部

分音视频数据丢失（称为丢包），音频丢包会使得接收方听到的声音是断断续续的，而视频丢包则会引起接收方的图像出现马赛克情况。若 CPU 速度太低以致不能及时的处理音视频信号，则会引起图像和声音的较大失真。在这两种情况下，MCU 只能减少其接入的终端数目以满足音视频信号的质量要求，从而 MCU 能接入的最大终端数目减少了，MCU 的性能就不太好。

最后，看 MCU 软件本身的性能优劣这个影响因子。这是 MCU 性能好坏的决定性因素。在其他外部条件一定的情况下，MCU 的性能决定于 MCU 软件本身。在集中型视频会议中，当有一个终端呼叫 MCU 并与之建立连接时，在此终端和 MCU 之间将相应打开一条 H.245 控制信道和一组媒体信道，这一组媒体信道应包括音频逻辑信道和视频逻辑信道，若需进行数据通信的话，还应包含数据逻辑信道。当有两个终端呼叫 MCU 并与 MCU 建立连接时，在每个终端和 MCU 之间都将相应打开一条 H.245 控制信道和一组媒体信道，则两个终端共计打开两条 H.245 控制信道和两组媒体信道，而这两个终端各自的控制信道和媒体信道都是相互独立的。这样，MCU 就需要接收并处理分别来自这两个终端的两组媒体流。依此类推，当有 n 个终端呼叫 MCU 并与 MCU 建立连接后，MCU 就需要接收并处理来自这 n 个终端的 n 组媒体流。

当 MCU 接入的终端数目较少时，MCU 能很快的处理它所接收到的音视频信号并将其回送给各参会终端。这时，它所提供的音视频信号质量能满足与会人员的要求。当 MCU 接入的终端越来越多，MCU 所需要处理的音视频信号也越来越多，当 MCU 接入的终端多达一定数目时，以 MCU 的处理能力已无法迅速处理并回送其所接收到的音视频信号，以致于其输出的音视频信号质量开始恶化（包括时延增大，有较多的马赛克情况出现等）且不能满足与会人员的要求或者 MCU 无法维持视频会议系统的正常工作时。

由以上分析可知，MCU 的性能受很多因素的影响。其中，可用的网络带宽资源的多少以及运行 MCU 软件程序的计算机的内存大小和 CPU 的速度快慢是外部限制因素，而 MCU 本身的性能优劣是决定性因素。

4.2 MCU 性能测试方案的设计

4.2.1 虚终端

虚终端即模拟终端，在此将它定义为具有 H.323 协议中所规定的终端的一部分功能，如能发起呼叫，接收呼叫，在呼叫连接建立后能传送音视频信号等，具备动态分配的运输层地址，但却不具备独一无二的 IP 地址的功能单元。即虚终端只能模拟终端行为，不能像普通终端一样根据 IP 地址寻址，所以不是真正意义上的终端。

4.2.2 测试设备的选用

MCU 具有能同时发起多路呼叫并且能同时接收多路呼叫的特性，故可利用 MCU 的这一特性，在其基础上设计一个测试设备即模拟呼叫器，可以模拟多个终端，然后使其呼叫被测 MCU n 次，从而在被测 MCU 和模拟呼叫器之间建立多个视频通道。但是，由于模拟呼叫器只具有有限个 IP 地址，因此，还需要一种辅助测试设备用来识别模拟呼叫器所模拟的 n 个虚终端。又由于 MCU 可以利用终端的别名对其进行呼叫或接收其对它的呼叫，而且 MCU 可以在 GateKeeper（网闸）上注册多个别名，所以模拟呼叫器也可在 GateKeeper 上注册 n 个别名以区分模拟的 n 个虚终端。又系统中 GateKeeper 的性能优于 MCU，所以 GateKeeper 不会成为网络的瓶颈，因此加入 GateKeeper 后不会影响测试的结果，也能识别模拟呼叫器所模拟的 n 个虚终端。最后由于模拟呼叫器不具备播放音视频流的功能，所以再加一台实终端 A，用来接收和播放音视频信号。测试在西安邮电学院通信技术研究所的局域网环境（10Mbps）下进行。测试环境如图 4.1 所示。

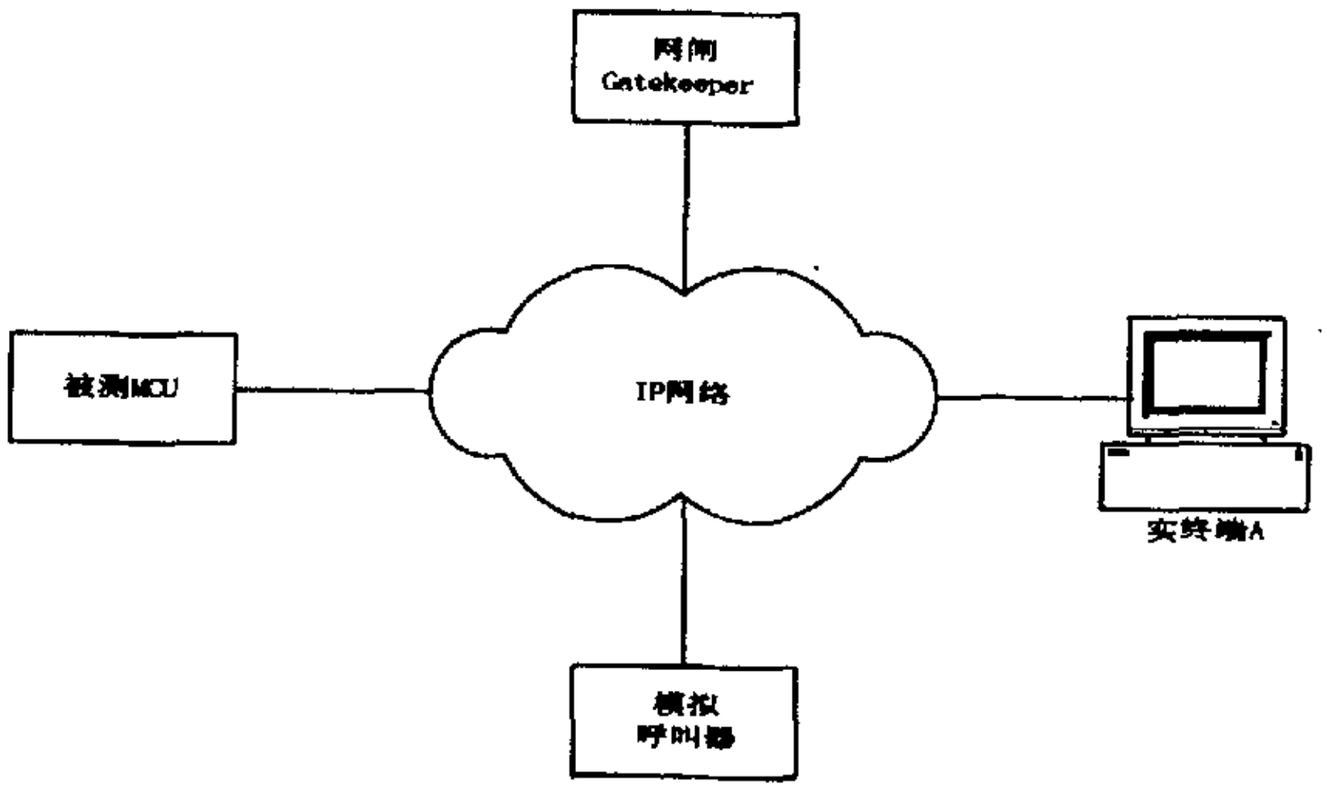

图 4.1 MCU 性能测试环境图

4.2.3 MCU 性能测试原理

所设计的模拟呼叫器要能模拟多个终端的行为，但并不是真正意义上的终端，不能产生音视频信号。因此我们可以在运行模拟呼叫器的 PC 机上预存入各种类型的音视频序列，当模拟呼叫器和 MCU 之间进行呼叫并建立多个连接后打开相应的音视频通道，然后启动读音视频流线程，最先建立连接的虚终端开始读音视频序列，然后将所读的音视频数据作多份复制，分别转发至模拟呼叫器的其余虚终端的发送缓冲区中，并发送音视频数据到 MCU，经过 MCU 处理后，再回送给各个虚终端。实终端 A 用来显示各个虚终端的图像，并且观察其图像质量。图 4.2 描述了其测试原理。

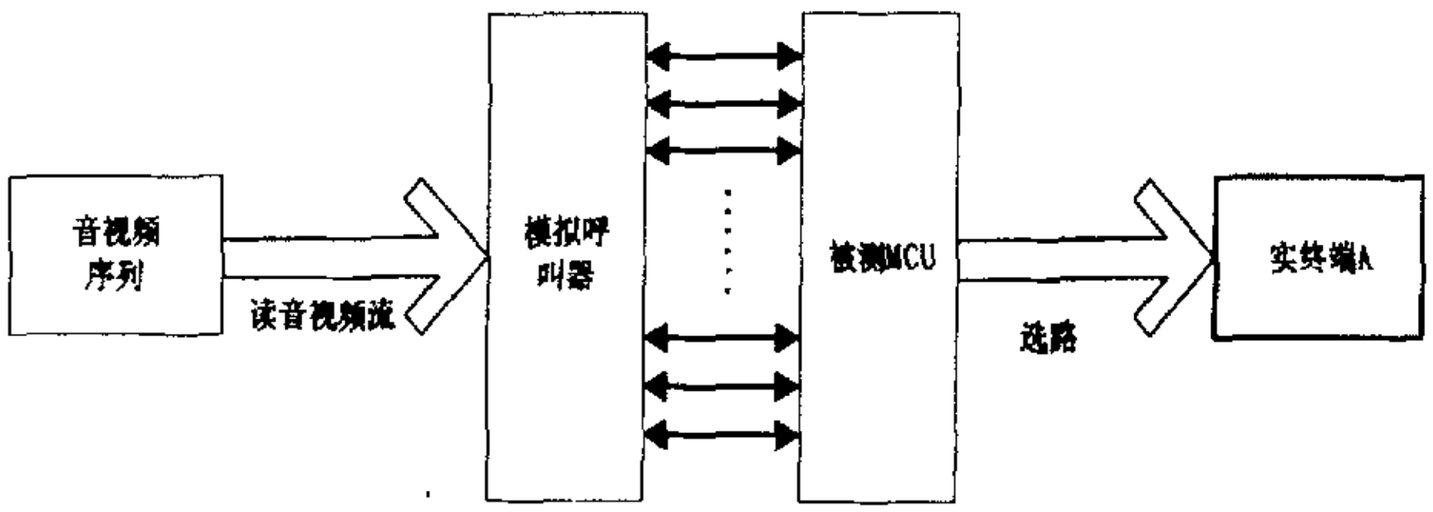

图 4.2 MCU 测试原理

4.2.4 测试前的工作

由于模拟呼叫器不是真正的 H.323 终端，它只能模拟 H.323 终端的呼叫和接收呼叫功能，不能产生音视频信号，而传送音视频信号是模拟终端所必须的功能，因此在测试之前，可以为其预存好音视频序列，连接建立后直接从预存的音视频序列中读取音视频信号。具体步骤：（1）H.323 终端对模拟呼叫器发起呼叫，模拟呼叫器应答并建立连接。(2)H.323 终端以普通形式向模拟呼叫器发送音视频数据。（3）模拟呼叫器接收音视频数据并按接收到的包顺序保存成音视频序列，不对其进行编解码操作。结构如图 4.3 所示。

音视频信号是按帧发送的，每一帧有多个数据包，所以在设计存储所接收的音视频信号时，按数据包顺序存储。当模拟呼叫器接收到一个数据包时，首先判断是音频包还是视频包，并给该包增加一个头部，存放该包的一些信息，如是否音频包、是否每一帧的最后一个包、距离前一个接收到的包的时间间隔 T 等，然后将该包存到音视频序列中。同样，读取音视频数据时，也是按照存储数据包的顺序来读，并解析出存储数据包时的包与包之间的时间间隔参数 T，等待 T 时间后将包发送出去。这样就保证了发送速度与原媒体流同步。

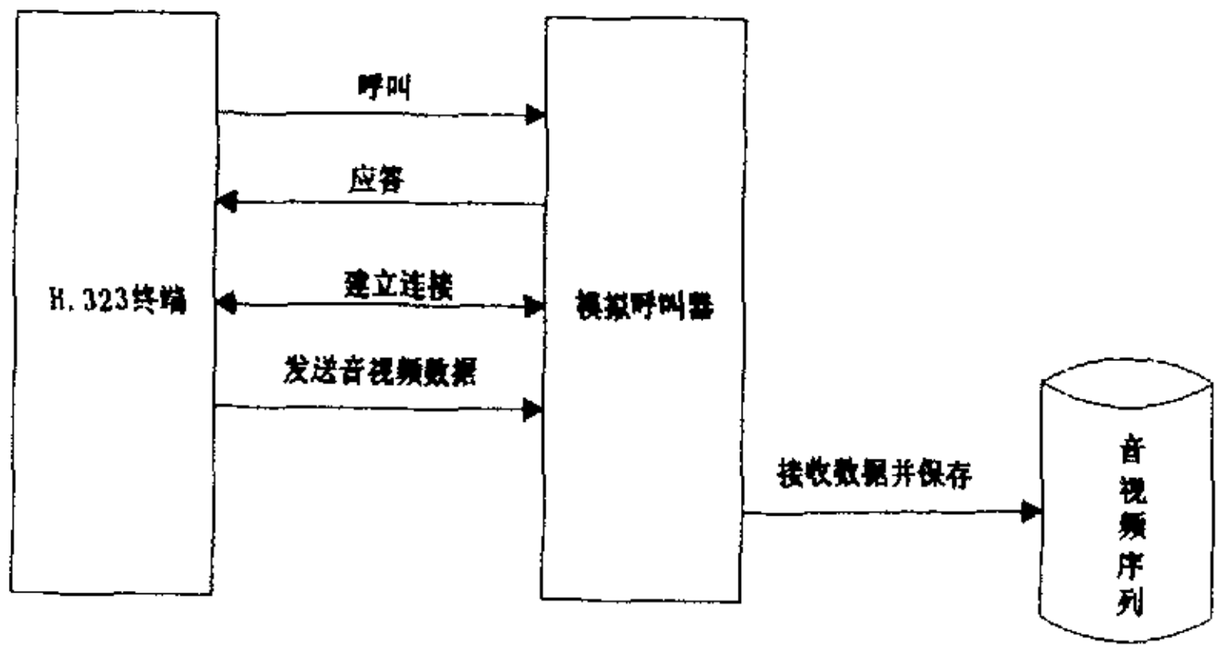

图 4.3 存储文件的结构图

4.2.5 具体方案

具体的测试过程主要由组建会议，建立呼叫连接，传送音视频信号以及观察

图像质量并分析数据等几个关键步骤组成。

（1）组建会议

测试 IP 网络视频会议系统的 MCU 性能时，首先要组建一个会议，这个会议不是真正意义上的会议，而是一个模拟的会议，该会议由一个实终端和若干虚终端组成。

首先模拟呼叫器、被测 MCU 以及实终端 A 均以人工方式搜索网闸，由于此网闸为测试中所提供的辅助测试设备，其 IP 地址是已知的，所以，以人工方式很容易搜索到此网闸。接着，模拟呼叫器通过 RRQ 消息到网闸（GK）上注册 N+1 个别名，记为$T_0,T_1,T_2,\cdots,T_N$，此时模拟呼叫器相当于 N+1 个虚终端，而且每个虚终端的别名互不相同，这样可以用别名来区分不同的虚终端；同样，被测 MCU 也通过 RRQ 消息在 GK 上注册 N+1 个别名，记为$M_0,M_1,M_2,\cdots,M_N$。实终端同样也以别名 A 在 GK 上注册。在所有设备都收到网闸返回的 GCF 消息即均注册成功后，把模拟呼叫器的 N+1 个别名和实终端 A 添加为被测 MCU 的同一组会议成员，被测 MCU 的 N+1 个别名添加为模拟呼叫器的同一组会议成员。这样实终端 A、N+1 个虚终端和被测 MCU 组成了一组模拟会议。

（2）呼叫连接

呼叫时，模拟呼叫器既可以作为主叫方（模拟呼叫器呼叫被测 MCU）也可以作为被叫方（被测 MCU 呼叫模拟呼叫器），而在本论文中选择模拟呼叫器作为主叫方，即它的 N+1 个虚终端分别以其在 GK 上注册的别名依次呼叫相应的被测 MCU 的别名，即T_0呼叫M_0，T_1呼叫M_1，…，T_{N-1}呼叫M_{N-1}，T_N呼叫M_N。网闸会根据别名正确地翻译出呼叫信令信道运输层地址， 这样模拟呼叫器模拟的 N+1 个虚终端和与被测 MCU 之间就建立了多个连接。在这个步骤里有一个关键的问题，就是在模拟呼叫器进行 N+1 次呼叫时，每发起一次呼叫，仅仅只给被测 MCU 发送一个独一无二的别名，并且每次发送的别名都互不相同。这样， 模拟呼叫器的每个别名与每次呼叫所建立的呼叫连接就对应起来了，并且还能将这些别名与相对应的呼叫连接所打开的逻辑信道对应起来。这样，模拟呼叫器就成功的模拟了 N+1 个虚终端，同时被测 MCU 也可以根据别名来区分这些虚终端了。

此时，模拟呼叫器与被测 MCU 就建立了 N+1 个呼叫连接，将虚终端T_0与被测 MCU 建立的通信连接记为C_0，虚终端T_1与被测 MCU 建立的通信连接记为C_1，虚终端T_2与被测 MCU 建立的通信连接记为C_2，……，以此类推，虚终端T_N与被测 MCU 建立的通信连接记为C_N。同时被测 MCU 呼叫实终端 A 并建立连接。

（3）传送音视频流

在所有连接都成功建立后，各个连接会相继打开相应的逻辑信道，包括 H.245 控制信道和音视频媒体信道，这时虚终端 T_0 从音视频序列开始读数据，然后虚终端 T_0 将这些音视频数据作 N 份复制，分别转发至模拟呼叫器的其余 N 个虚终端的

发送缓冲区中，N+1 个虚终端分别通过通信连接$C_0,C_1,\cdots,C_N$的音视频通道将这 N+1 个音视频信号传送至被测 MCU。（或者连接$C_0,C_1,\cdots,C_N$开始从不同的音视频序列中读音视频数据，分别读到模拟呼叫器的 N+1 个虚终端的发送缓冲区中。这 N+1 个虚终端则分别通过通信连接$C_0,C_1,\cdots,C_N$的音视频通道将这 N+1 个音视频流传送至被测 MCU。）由于 MCU 具有将某选定源的信号发往其余终端的能力，则被测 MCU 可根据要求将虚终端$T_0,T_1,T_2,\cdots,T_N$传送过来的音视频信号经处理后再送回各个终端。由于实终端 A 是被测 MCU 的会议成员，这样，被测 MCU 可选择将某一虚终端传送过来的音视频流传送至实终端 A 上并由实终端 A 显示出来。又由于被测 MCU 可以设定主席模式，而在主席模式下选定听众的图像要转发给主席，主席的图像要转发给所有终端，则可将实终端 A 选为主席，将某一虚终端 T1 选为选定听众，或者将某一虚终端 T1 选为主席，A 选为一般听众或选定听众，这样，实终端 A 显示的是 T1 传过来的图像。若 MCU 根据要求选择将虚终端$T_0,T_1,T_2,\cdots,T_N$传送过来的某四路音视频信号组合成一个信号后传送至实终端 A 上并由实终端 A 显示出来，那么，实终端 A 上显示的将是四画面。

（4）数据分析

根据实终端 A 上的音视频质量分析判断被测 MCU 的性能。同时，我们还引入了视频会议系统的几个性能指标（MCU 最大接入终端数、图像时延、延迟抖动），用来协助判断被测 MCU 的性能。

MCU 最大接入终端数定义为保证视频会议系统正常工作的情况下，被测 MCU 能连接的最大终端数目。接入终端数越多，说明 MCU 的性能越好。它是衡量 MCU 性能好坏的一个重要参数。

时延抖动是由于网络负荷量的变化和传输路由的不同，话音分组的时延会发生变化。在时延过大的情况下，话音分组会发生丢失，造成话音失真。视频会议规定的延迟抖动约为 400ms。

传送图像最主要的性能指标是点对点的图像时延。点对点时延是指从某一终端发送一帧图像开始到另一终端接收到同一帧图像的时间差，这可以通过在两个终端上对同一 RTP 流加盖时间戳来实现，即计算出两个时间的差值。然而两台设备的系统时间可能存在差异，所以可能会引起测不准问题。但是终端的网络带宽使用远小于其网络吞吐量，使得数据包在终端处没有拥塞，时延主要集中在 MCU 的处理部分，即 MCU 的处理时延。因此点到点图像时延问题可以转化为 MCU 的处理时延问题，从而避免了因为系统时间不同而引起的测不准问题。MCU 处理时延可以通过先测试 MCU 接收到一个 RTP 流的时间及将同一个 RTP 流转发往终端的时间，再计算这两个时间的差值就可得出 MCU 的处理时延。这可通过在 MCU 接收和发送同一 RTP 流时加盖时戳并计算两个时戳差值得到。视频会议规定点对点的图像时延不超过 150ms。

被测 MCU 对运动程度不同的视频图像的处理能力有所不同，在其余参数都一定的前提下，需要使用的带宽有所不同，因而，传送运动程度不同的视频图像时，当传输带宽一定，在接收到的视频质量良好的情况下，MCU 的最大接入终端数有一定差别；当接入数一定，在实终端 A 接收到的视频质量良好的情况下，MCU 的传输带宽可能也有差别。在保证视频质量的前提下并且带宽一定时，若传送的是缓慢运动的图像，则 MCU 的最大终端接入数将比传送剧烈运动的图像时要大一些。同样接入数一定时，传送缓慢运动图像的带宽要比传送剧烈运动图像的带宽要少一些。如果视频带宽不足，画面细节部分就不够清晰，分辨率低，严重的出现色彩失真或丢失。同时分析其他一些性能指标，如图像时延和延迟抖动，看他们是否满足视频会议所规定的值。如果图像质量良好而且图像时延和延迟抖动均达到了规定的要求，则可以增加接入终端数量或降低带宽再进行测试。

4.3 MCU 性能测试方案分析

本章详细具体地介绍了 MCU 的性能测试方法，它不仅适用于单画面的情况，也适用于四画面情况。所提出的方案很好地解决了 MCU 的性能测试问题。下面主要讨论本方案的优点以及实际应用。

4.3.1 本方案的创新点

MCU 和模拟呼叫器均不具备从外界摄取图像和声音的功能，所以开始的设计方案是接入两个实终端 A、B，用来发送和接收音视频信号。在所有连接都已经成功建立并保持以后和在各个连接的音视频媒体信道均打开后，实终端 A 开始向虚终端 T^0 传送音视频信号，然后虚终端 T^0 将这些音视频信号做 N 个复制，分别转发至模拟呼叫器的其他 N 个虚终端（从 1，2...，直至 N）的发送缓冲区中。而本方案在选用测试需要的设备时，只选用了一个实终端用来接收音视频信号，改进为将连接后要传送的音视频信号以文件格式预先保存在运行模拟呼叫器的计算机里，相当于模拟呼叫器具有音视频获取能力。因此文件的保存和读取是本方案的难点所在，也是能否完成测试的关键。

4.3.2 本方案的优点

在本论文方案提出之前，没有公开的专门用于测试 MCU 性能的方法，不像网闸的性能测试，信息产业部专门为网闸性能测试制订了一套测试标准《IP 电话网闸设备技术要求和测试方法》。

传统的 MCU 性能测试方法虽能测试出 MCU 的基本功能（不同于性能），但是当用这种传统的测试方法用来测试 MCU 性能时，在需要接入的终端数量较多时，该方法不仅不容易实现，而且占用大量资源，成本较高，因此可行性很低。

本论文所提出的方案具有以下优点：

1. 完全针对 MCU 性能测试所提出，具有很强的针对性，很好的解决了 MCU 性能测试的问题。

2. 除了能测试 MCU 处理不同运动程度图像的处理性能外，该方案也能用于测试 MCU 的其他性能，比如 H.323 协议的内容，以及它所支持的音视频编解码格式。

3. 只需要一台所设计的模拟呼叫器设备、一台终端设备和一台网闸设备，就可完成整个过程的测试，占用资源少，耗费经济成本低，降低了 MCU 性能测试的难度。

4. 简单易行，操作方便。

5. 模拟呼叫器设备是基于 H.323 标准设计的，可以和任何 H.323 系统组件通信，因此可以用来测试其他厂家的 MCU 性能，实用性很强。

4.3.3 本方案的实际应用

此方案可用于一切 MCU 性能的测试如 MCU 容量、图像处理时延等。也可推广应用于 4 画面时 MCU 性能的测试以及 H.323 协议内容的性能测试，比如可以使用的音视频标准。

所谓 4 画面时 MCU 性能的测试，是指 MCU 对已建立的 n 路呼叫进行视频混合，将 4 个指定的视频源信号组合成一个 2×2 的多画面图像送给各终端时对 MCU 的性能进行的测试。这种测试只需在本方案实现音视频流传送的阶段使被测 MCU 根据要求选择将虚终端 1，2…n 传送过来的任 4 路音视频流传送至实终端 A 上并由实终端 A 显示出来即可实现。

H.323 协议内容的性能测试是指 MCU 支持的音频/视频编解码算法，即视频是否支持 H.261CIF、H.261QCIF、H.263 等，音频是否支持 G.711U、G.711A、G.723.1、G.722 等。测试时，根据前面的方案设计，事先在模拟呼叫器设备上保存经过各种编解码算法的音频和视频序列，具体测试步骤与前面介绍的相同，只是传送的音频频序列的类型根据我们所要测试的具体内容而定。

4.3.4 本方案的不足

虽然本方案具有很多优点，但也有它的不足之处。模拟呼叫器虽然能模拟终端的行为，能发起和接收呼叫，也能发送视频信号和接收媒体通道上的视频信号，但是却无法实时显示该信号，测试方案中通过被测 MCU 的选路功能利用实终端 A 来显示虚终端的视频图像，终究每次只能显示选定的某一个虚终端的图像。如果解决了该问题，测试时只需要模拟呼叫器就能完成任一个厂家的支持 H.323 标准的 MCU 的性能测试，这样模拟呼叫器的实用性就更强。

第五章 MCU 性能测试方案的实现

第四章提出了一种测试 MCU 性能的有效方法，首先设计了一个测试设备模拟呼叫器，接着详细介绍了应用模拟呼叫器和一台终端、一台网闸用来测试 MCU 性能的具体过程。因此，MCU 性能测试过程不仅仅是测试过程的实现，而且还包括模拟呼叫器的设计与实现。而模拟呼叫器的实现是测试方案能否实现，测试工作能否进行的关键。

本章将首先介绍模拟呼叫器软件开发平台，然后介绍模拟呼叫器各个功能模块的设计，并给出程序实现中所涉及的最重要的类和函数，同时针对其设计不完善的地方给出了改进方案，最后给出了 MCU 性能测试方案的具体应用及测试结果。

5.1 模拟呼叫器软件开发平台

模拟呼叫器的应用软件设计与实现是个很庞大的工程。所以，设计并实现它时，将针对它的每个功能模块进行设计。为便于应用软件实现，我们采用了 C++语言及 VC 开发环境，并在 Windows2000 的工作平台上设计。采用 C++语言是由于 C++语言简洁、高效，是结构化、模块化的程序设计语言，具有较高的移植性和很强的数据处理能力。而 Windows 漂亮、统一、友好的用户界面及面向对象的程序设计和消息驱动的程序结构成为我们首选的工作平台。

开发模拟呼叫器软件是在 MCU 软件的基础上改进的，而西安邮电学院通信技术研究所所实现的大型 MCU 软件使用的协议栈是澳大利亚公司 Equivalence Pty Ltd 发布的 OpenH323 协议栈，它是一个开放源代码的 H.323 协议栈，已基本实现了 H.323 协议框架。而 OpenH323 类库是该协议栈中实现了 H.323 协议内容和协议过程的核心类库，且该类库完全符合 H.323 协议，能和任何符合该协议的软件进行音视频通信。

模拟呼叫器软件程序是基于 OPENH323 类库编写的上层应用程序。本文将详细介绍模拟呼叫器应用程序的设计与实现，而 OPENH323 类库所实现的协议内容和过程不属于本文的介绍范畴。

5.2 模拟呼叫器的设计

5.2.1 模拟呼叫器功能描述

由第四章的讨论可知，设计模拟呼叫器的目的是模拟 H.323 终端的行为，它

必须能发起和接收多路呼叫、产生和传送音视频信号。因此，模拟呼叫器所要实现的主要功能有：

• 模拟多个虚终端，每个虚终端在网闸上登记注册不同的别名。

• 发起多路呼叫，每路呼叫互不影响，且发起每路呼叫时均只发送本端的一个独一无二的别名，每路呼叫所发送的别名互不相同。

• 接收多路呼叫，所接收的每路呼叫间互不影响，接收的每路呼叫都能识别呼叫发起端的一个别名。

• 挂断呼叫连接，包括挂断至某个用户的呼叫连接和挂断所有呼叫连接。

• 别名设置：用户可以为模拟呼叫器设置任意多个别名。

• 网闸注册：包括自动搜索网闸并注册及人工搜索网闸并注册，注册包括在网闸上登记自身的别名。

• 音视频存储：将接收到的音视频流存成视频序列。

• 读取音视频：并且将其复制后传送至其他虚终端的发送缓冲区。

其中，音视频存储及读取是本论文工作的难点。

5.2.2 模拟呼叫器的模块设计

由测试方案设计可知，由于 MCU 能同时发起多路呼叫并且能同时接收多路呼叫，根据 MCU 的这一特点，在 MCU 程序的基础上修改并添加一些功能，就能符合模拟呼叫器的要求。这里主要介绍模拟呼叫器作为测试设备时所应具备的相关功能模块。

1. 初始化模块

初试化模块的主要任务是设置参数。参数包括呼叫参数和接收发送参数，呼叫参数包括对方终端的别名（包括 IP 地址和号码）、设置网络参数，需要网关支持的话还需要设置网关等。接收发送参数包括输入音视频格式、收发送图像格式、以及采用的音视频编解码协议等。

2. 呼叫连接和释放模块

呼叫连接和释放模块的主要功能是根据初始化所设置的参数，与目的终端建立呼叫连接，并在多媒体信号收发过程结束之后释放呼叫。它又分为呼叫连接子模块和呼叫释放子模块。

两个终端在能够进行音视频收发之前，必须先建立连接，这就是呼叫连接子模块的功能。由第三章的介绍可知，这一过程由 H.225.0 协议及 H.245 协议完成。它又可以分为呼叫控制和连接控制两个子过程。其中，呼叫控制子过程由 H.225.0 协议执行。它先根据公认的端口号建立起 TCP 连接，即可靠的呼叫信道。然后在此呼叫信道上发送 H.225.0 呼叫信令消息，直至建立起另外一条 TCP 信道—H.245 控制信道，呼叫控制过程结束，连接控制过程开始。连接控制过程由 H.245 协议完成。此间最重要的过程为能力交换过程。

呼叫释放子模块的功能是释放一个呼叫。此过程可以由任一方终端发起。首先发起终端停止在逻辑信道上传送信息，关闭所有逻辑信道。然后通过 H.245 控制信道向对方终端发送“结束会话”命令。对方终端接到上述消息之后，关闭所有逻辑信道，向发起终端回送“结束会话”命令消息。至此，整个过程结束。

在本模块设计中，对别名的处理特别重要，它是完成本方案的关键之一。

3．存储音视频流模块

存储视频流模块是整个软件的核心。模拟呼叫器在存储视频流时不对其进行解码操作。由于音视频信号是按帧发送的，每一帧有多个数据包。所以在设计存储所接收的音视频信号时，也是按数据包存储的。当模拟呼叫器在非指定呼叫连接上接收到一个数据包时，首先判断是否是视频包，如果是，则给该包增加一个头部用来存放该包的一些额外信息，如是否视频包、是否每一帧的最后一个包、距离前一个接收到的包的时间间隔 T 等，然后将该包写入文件。音视频存储相对简单一些，只要控制好写入操作的互斥，保证同一时刻只能有一个写操作就可以了。视频信号存储如图 5.1 所示。

4．读音视频流模块

读音视频流模块也是整个软件的核心。在第一个连接建立后，就启动读文件线程，从存储音视频流模块中的保存文件中读取视频信号，并将音视频信号放入该连接相对应的音视频发送缓冲区。读操作相对复杂一些，因为是从一个文件读，而且读文件的速度应该不能太快也不能太慢，要保证与原媒体流同步，而且音视频同时从文件读取也要控制好读的顺序，如连接建立以后，想要获取音频包，而此时文件头是视频包，这就需要等待，一直等到通信线程获取视频包后，再读下一包，如果还没有音频包，则还需等待，直到等到音频包后才读取返回。读取一个数据包后，并获取存储时添加的一包与前一包之间的间隔时间 T，然后等待 T 时间后发送出去。图 5.2 示出了该模块流图。

5．音视频信号复制模块

所设计的模拟呼叫器并不像实际的终端能从外界获取所要传送的音视频信号，针对这个问题提出了两种解决方法：一是第一个连接建立后，该连接首先从

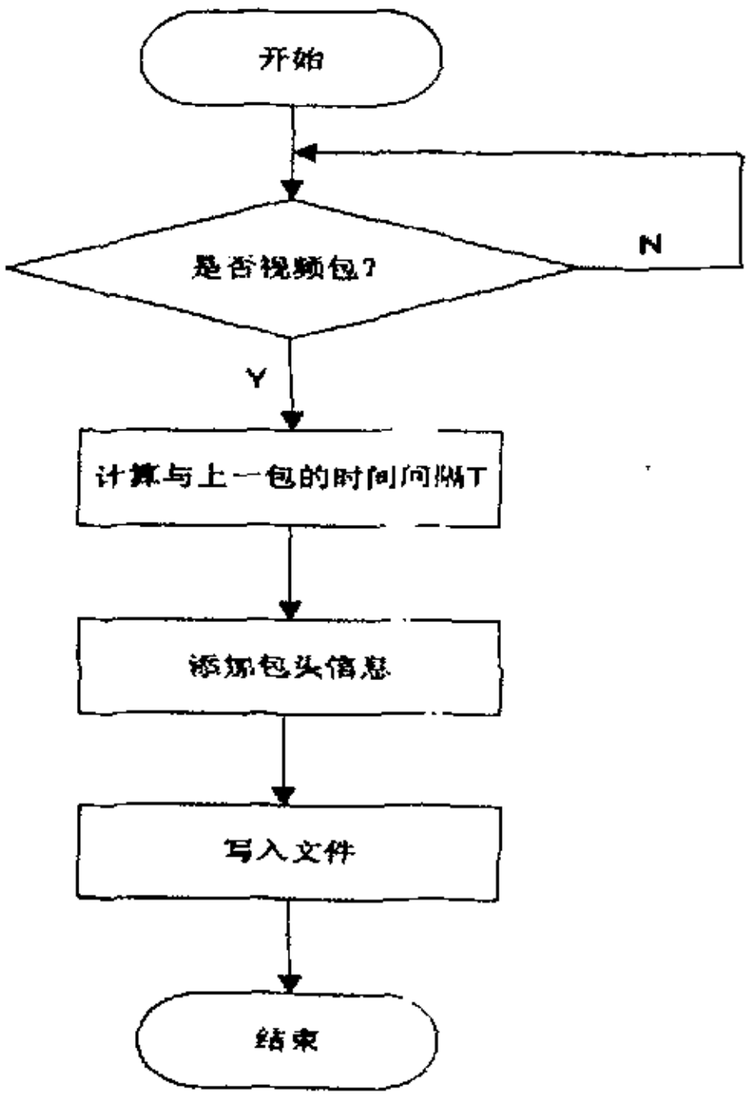

图 5.1 存储视频流程图

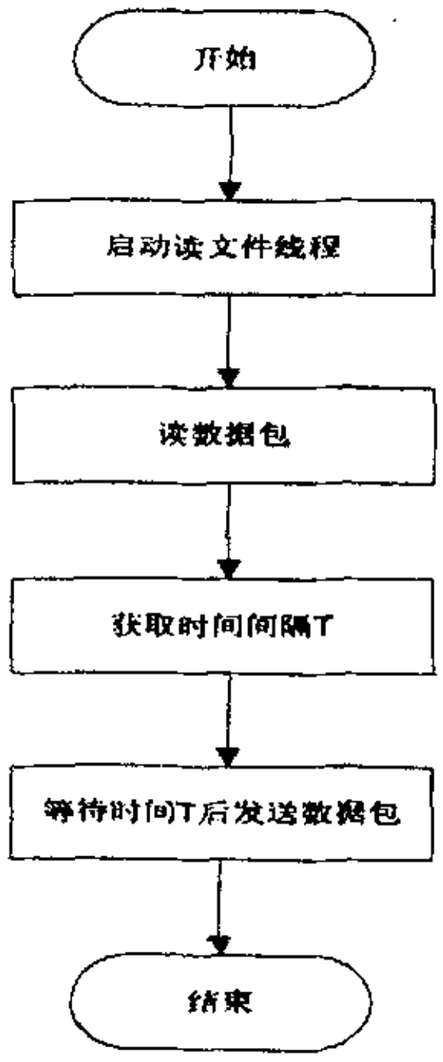

图 5.2 读文件流程图

预先保存的文件中读取音视频数据，并将其复制 N 份后存到其余虚终端的音视频发送缓冲区中；二是所有连接建立后，不同的连接开始从不同的预存文件中读取音视频信号，并把读取的信号放到各个连接所对应的虚终端的发送缓冲区中。在本论文中，采用第一种方法，因为它比较容易实现。

5.2.3 模拟呼叫器的原理

模拟呼叫器系统原理如图 5.3 所示。Endpoint 对应模拟呼叫器的多个虚终端，每个 Connection 对应与一个虚终端的 H.323 连接。Video Process 模块处理接收和发送的音视频信号，接收 Incoming 的音视频数据时，按照接收顺序，并加上适当的延迟之后写入文件；Outgoing 发送数据时，根据写入时的延迟控制播放速度。Write File or Read File 模块定义一个音视频文件，完成基本的文件读写操作。

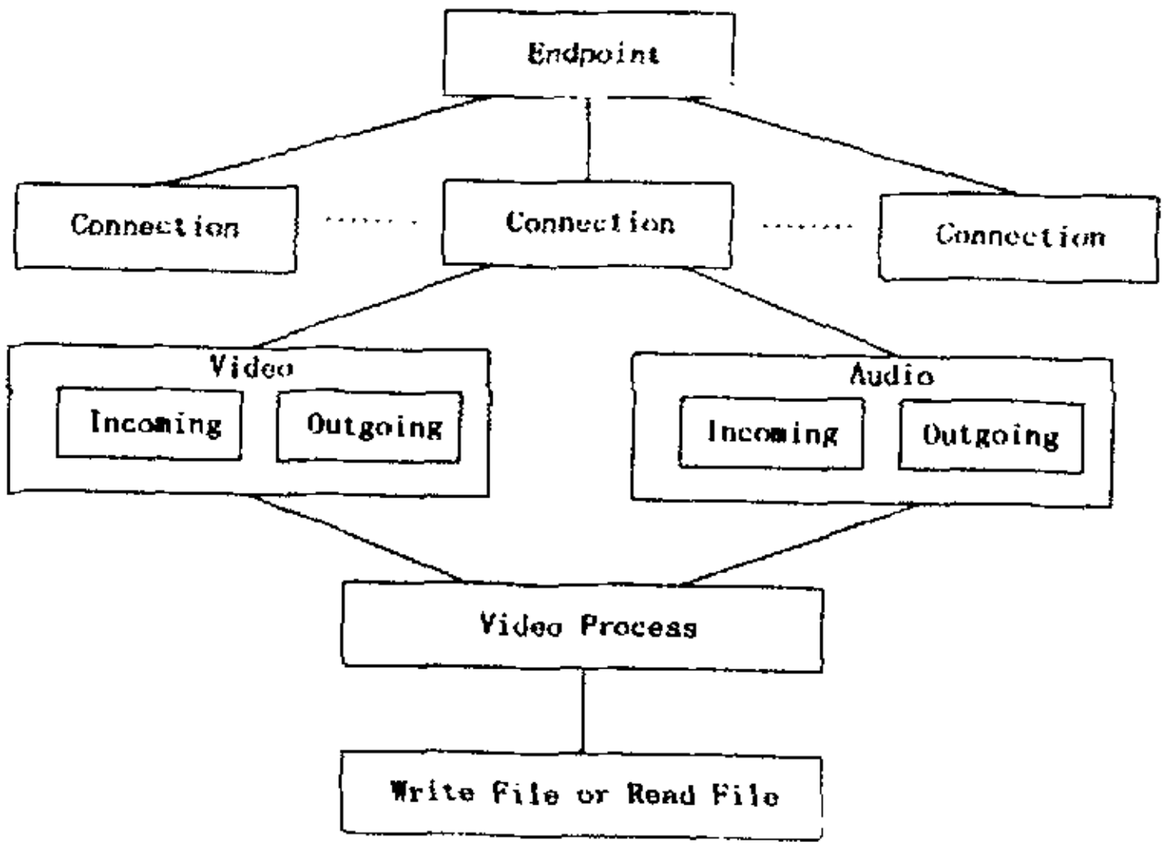

图 5.3 模拟呼叫器原理图

5.3 模拟呼叫器的实现

模拟呼叫器的设计是基于 H.323 协议的，由 5.1 可知 H.323 信令和控制消息的协议数据单元（PDU）和一些通用的标准 H.323 信令操作过程由协议栈 OpenH323 类库封装提供。而模拟呼叫器是对 MCU 的程序改进而得到的，下面首先介绍这两个应用程序共用的核心类及其函数，接着介绍模拟呼叫器实现新增功能的新类及其函数。

5.3.1 核心类介绍

MCU 和模拟呼叫器程序实现时，继承了 OpenH323 类库中的许多有用的类，这些类包括：

1. H.323EndPoint 类：OpenH323 的体系结构中主要的对象就是 H323EndPoint 类，通常情况下应用程序都会有一个该类的继承类的实例。应用程序所定义的子类将会建立一些缺省的 H323 参数（比如超时时限等），其中最重要的是能力表，该表定义了应用程序能够处理的 codec 和信道类型。类 H.323EndPoint 管理 H.323 端点，一个 H.323 端点可以有 0 个或多个侦听端口来接受别的端点请求发起的 H.323 连接；它也可以通过调用具有呼叫功能的 MakeCall()函数来和别的端点建立多个 H.323 连接。一旦一个 H.323 连接建立了，这个连接就将由 H.323Endpoint 这个类的实例来管理。

2. H.323Connection 类：用来描述两个端点之间的 H.323 连接。当第一个 PDU 通过使用 Q.931 和 H.225 协议的 H323Transport 到达的时候，有一个呼叫引用用于识别所建立的连接，这些连接是通过 H323Connection 类来体现的，它包含了 H323 端点之间连接的所有状态信息。应用程序通常是让系统来创建一个 H323Connection 类的子类实例，而不是类本身。这是因为这样可以重载很多虚函数。这些虚函数是一些库的回调函数使得应用程序既能获取信息，也能够修改协议协商不同阶段的行为。

3. H.323Capability 类：能力集描述类。这个类是 H.323 能力的描述集合。由这个类还派生了许多有用的类，包括视频编解码能力描述类和音频编解码能力描述类等。

4. H.323Codec 类：是各种音频，视频，数据的编解码器描述类。由这个类还派生了视频描述类 H323VideoCodec 和音频描述类 H323AudioCodec。而由这两个视频和音频描述类又派生出了基于各种编解码算法如 H.261、H263、G.711、G.722 等的音视频描述类。

5. H.323Channel 类：用来描述两个端点之间的逻辑信道。该类将使用 H323Capability（协议协商时传送过来的）来创建一个 H323Codec。

6. H245Negotiator 类：用于描述 H.245 协议过程，维护 H.245 协议中定义的

每一个命令和变量的功能以及状态。

7. H323Listener 类：用于描述一个 H.323 连接上的守听方。由其派生的类 H323ListenerTCP 用于描述一个 H.323 连接上的守听方的 TCP 传输通道。应用程序要创建的 H323Endpoint 可能是 H323Listener 类的一个或多个子类实例。对于每一个所要支持的协议都存在一个该类的子类。比如 H323ListenerIP 可能就是用于 IP 的。每个 listener 都产生一个线程来监控它的协议，当一个新的呼叫到达的时候，创建一个 H323Transport 类的子类。就 H323Listener 而言，每一个协议都有一个 H323Transport 的子类，比如 H323TransportIP。

8. H225_RAS：这个类用于描述端点和网闸之间的 RAS 协议过程。

9. 和协议数据单元（PDU）有关的类：主要指类 H323RasPDU、类 H323SignalPDU 及类 H323ControlPDU。

H323RasPDU 描述的是 RAS 连接上端点和网闸交换的 RAS 消息；

H323SignalPDU 描述的是 H323 连接上两个端点交换的 H.225.0 信令消息；

H323ControlPDU 描述的是 H323 连接上两个端点交换的 H.245 控制消息。

以上简单介绍了一下模拟呼叫器程序所继承的 OpenH323 类库中的一些重要的类，当然还有其他一些重要类，鉴于篇幅有限在此就不一一介绍了。

5.3.2 核心函数介绍[5]

上面介绍了几个核心类，下来就几个核心类的主要成员函数作以介绍。

5.3.2.1 类 H.323EndPoint 的成员函数：

1. 终端能力集的操作函数

· void AddCapability(H323Capability * capability) ：向能力表中添加编解码算法。

· PINDEX SetCapability(PINDEX descriptorNum, PINDEX simultaneous, H323Capability * cap)：设置能力描述集。

·PINDEX AddAllCapabilities(PINDEX descriptorNum, PINDEX simultaneous, const PString & name)：向能力描述集中添加所有匹配的终端能力。

· void AddAllUserInputCapabilities(PINDEX descriptorNum, PINDEX simultaneous) ：把所有用户的接收能力添加到终端的能力表中。

· void RemoveCapabilities(const PStringArray & codecNames) ：删除能力表中的能力项。

· void ReorderCapabilities(const PStringArray & preferenceOrder)：对能力表中的能力集重新排序。

· H323Capability* FindCapability(const H245_Capability & cap)

H323Capability* FindCapability(const H245_DataType & dataType)

H323Capability* FindCapability(H323Capability::MainTypes mainType,

unsigned subType)

查找能力表中是否有已经注册的能力。

2. 管理网闸的函数

·BOOL SetGatekeeper(const PString & address, H323Transport * transport = NULL)：选择一个外部网闸并在其上登记。

·BOOL LocateGatekeeper(const PString & identifier, H323Transport * transport = NULL)：定位并选择一个网闸。

· BOOL DiscoverGatekeeper(H323Transport * transport = NULL)：搜索并选择一个网闸。

· virtual H323Gatekeeper* CreateGatekeeper(H323Transport * transport)：创建一个 H.323GateKeeper 类的实例。

· H323Gatekeeper* GetGatekeeper()：获得已登记的网闸。

· BOOL RemoveGatekeeper(int reason = -1)：去除登记并删除该网闸。

· void SetGatekeeperPassword(const PString & password)：为网闸设置 H235 密码。这个函数不常用。

3. 管理连接的函数

· BOOL StartListener(H323Listener * listener)

BOOL StartListener(const H323TransportAddress & iface)为端点添加侦听端口。

· BOOL RemoveListener(H323Listener * listener)：删除侦听端口。

· H323Connection* MakeCall(const PString & remoteParty, PString & token, void * userData = NULL)

H323Connection* MakeCall(const PString & remoteParty, H323Transport * transport, PString & token, void * userData = NULL)呼叫远端。

·void ParsePartyName(const PString & party,PString & alias, H323TransportAddress & address) const ：将一方地址解析成别名和运输层地址。

· virtual BOOL ClearCall(const PString & token,

H323Connection::CallEndReason reason =

H323Connection::EndedByLocalUser)：清除当前呼叫。

· virtual void ClearAllCalls(H323Connection::CallEndReason reason = H323Connection::EndedByLocalUser, BOOL wait = TRUE)：清除所有当前呼叫。

· H323Connection * FindConnectionWithLock(const PString & token)：查找特定 token 值的连接。

· virtual BOOL OnIncomingCall(H323Connection & connection, const H323SignalPDU & setupPDU, H323SignalPDU & alertingPDU)：响应入呼叫。

· virtual void OnConnectionEstablished(H323Connection & connection, const PString & token)：连接建立时的响应函数。

· virtual BOOL IsConnectionEstablished(const PString & token)：确定是否建立了一个连接。

· virtual void OnConnectionCleared(H323Connection & connection, const PString & token)：连接中断时的响应函数。

·H323Connection* OnIncomingConnection(H323Transport * transport, H323SignalPDU & setupPDU)：处理新的呼叫连接。

· virtual H323Connection* CreateConnection(unsigned callReference, void * userData, H323Transport * transport, H323SignalPDU * setupPDU)：创建一个使用特定呼叫引用值（Call Reference）的连接。

· void CleanUpConnections()：清除连接。

4．管理逻辑信道的函数

· virtual BOOL OnStartLogicalChannel(H323Connection & connection, H323Channel & channel)：打开逻辑信道的响应函数。

· virtual void OnClosedLogicalChannel(H323Connection & connection, const H323Channel & channel)：关闭逻辑信道的响应函数。

· virtual BOOL OpenAudioChannel(H323Connection & connection, BOOL isEncoding, unsigned bufferSize, H323AudioCodec & codec)：打开音频信道。

· virtual BOOL OpenVideoChannel(H323Connection & connection, BOOL isEncoding, H323VideoCodec & codec)：打开视频信道。

5．其他函数

·void SetLocalUserName(const PString & name) ：设置本端用户名，可用于本端的所有连接。

·const PString & GetLocalUserName( ) const { return localAliasNames[0]; } ：获得本端用于任何 connection 的用户名。

· BOOL AddAliasName(const PString & name) ：为本端添加别名。

· BOOL RemoveAliasName(const PString & name) ：删除本端的一个别名。

·const PStringList & GetAliasNames( ) const {return localAliasNames;}：得到本端别名。

5.3.2.2 类 H.323Connection 的成员函数：

·BOOL Lock() ：锁定连接

· void Unlock() ：解除连接的锁定

·virtual void OnEstablished()：连接建立时的调用函数

· virtual void OnCleared()：清除连接时的调用函数

· virtual BOOL ClearCall(CallEndReason reason = EndedByLocalUser) ：清除当前呼叫

· virtual void CleanUpOnCallEnd() ： 当与当前连接相关联的呼叫被清除时，作一些清理工作

·void AttachSignalChannel(H323Transport * channel , BOOL answeringCall) ：绑定连接的传输信道作为呼叫信令信道

· BOOL WriteSignalPDU(H323SignalPDU & pdu) ：向呼叫信令信道中写协议数据单元（PDU）

·virtual BOOL OnIncomingCall(const H323SignalPDU & setupPDU, H323SignalPDU & alertingPDU) ： 处理入呼叫

·virtual AnswerCallResponse OnAnswerCall(const PString & callerName, const H323SignalPDU & setupPDU, H323SignalPDU & connectPDU) ：控制对入呼叫的应答

· void AnsweringCall(AnswerCallResponse response) ： 指示应答入呼叫的结果

· virtual CallEndReason SendSignalSetup(const PString & alias, const H323TransportAddress & address) ： 发送最初的 PDU 消息

· virtual BOOL StartControlChannel()

virtual BOOL StartControlChannel(const H225_TransportAddress & h245Address) ： 打开一个独立的 H.245 控制信道

· BOOL WriteControlPDU(const H323ControlPDU & pdu) ： 向控制信道中写 PDU

· BOOL StartControlNegotiations() ： 启动控制信道协商过程

· virtual void HandleControlChannel() ： 在控制信道上读取数据

· virtual BOOL HandleControlData(PPER_Stream & strm) ： 处理读取的数据

· virtual BOOL HandleControlPDU(const H323ControlPDU & pdu) ：处理在控制信道上接收到的 PDU

· virtual void SendCapabilitySet(BOOL empty) ： 发送新的能力集

·virtual H323Channel * CreateLogicalChannel (

const H245_OpenLogicalChannel & open, BOOLstartingFast, unsigned & errorCode)：创建一个新的逻辑信道实体

·virtual BOOL OpenAudioChannel(BOOL isEncoding, unsigned bufferSize, H323AudioCodec & codec) ： 打开音频通道

·virtual BOOL OpenVideoChannel (BOOL isEncoding, H323VideoCodec & codec) ： 打开视频通道

·virtual void CloseLogicalChannel(unsigned number, BOOL fromRemote)：关闭

逻辑信道

上面给出了 OPENH323 类库中两个主要类的主要成员函数，而且这些函数在模拟呼叫器程序实现时用到。

5.3.3 新增类介绍

为了实现视频信号的存储和读取功能，新设计了 CH323File 和 CFileSink 两个类。CH323File 类完成文件的基本读写操作，CFileSink 类处理要写和读取的数据，同时解决视频发送速度和原媒体流的播发速度同步。

首先定义文件头的结构体：占 10 字节

```
typedef   struct{
                BYTE     video_type;        //视频类型
                BYTE     audio_type;        //音频类型
                WORD     video_width;       //视频宽度
                WORD     video_height;      //视频高度
                BYTE     fps;               //每秒的帧数
                BYTE     ver;               //版本号，初始化为 1
                BYTE     reserve[2];        //预留
                }H323_FILE_HEADER;
```

其中，视频类型中 0x80 表示 H.261CIF；0x81 表示 H.261QCIF；0x90 表示 H.263。音频类型中 0x80 表示 G.711A；0x81 表示 G.711U；0x90 表示 G.723。

文件包的结构体：

```
typedef   struct{
                BYTE     media_type;        //媒体类型
                DWORD    packet_size;       //包的大小
                DWORD    delay;             //包与包之间的延迟
                BYTE     reserved;          //预留
                BYTE     *data;             //媒体数据
                }H323_FILE_PACKET;
```

其中，媒体类型中假定 0x80 表示视频数据，0x81 表示音频数据。

5.3.3.1 类 H323File 的成员变量及函数：

主要成员变量：

PString　m_strFileName：文件名，PString 为 PWLIB 类库中的字符串类

BOOL　m_bCanRead：读文件标志

H323_FILE_HEADER　m_Header：文件头

FILE　*pfile：　文件

PINDEX　read_count：　所读包数

主要成员函数：

·void InitPacket（H323_FILE_PACKET &packet)：初始化文件数据包的五个成员变量，0x00，0，0，0x00，NULL

·void SetFileHeader(const LPH323FILEHERDER header)：设置文件头

·void GetFileHeader(H323_FILE_HEADER &header)：获取文件头

·void ReadFileHeader( )： 读文件头

·BOOL OpenFile(BOOL bread)：打开文件，bread=true 时打开文件的模式为“rb”，否则为“wb”

·void CloseFile( )： 关闭文件

·void WriteHeader( )： 写文件头

·void WritePacket(const H323_FILE_PACKET &packet)

void Write(BOOL &bvideo,const int &len,const int &delay,const void *data)： 写数据包到文件。文件打开成功后，如果文件为空则先写文件头，再写数据包头和数据；如不为空，则直接写包头和数据。

·BOOL ReadPacket(H323_FILE_PACKET & packet)

·BOOL Read(BOOL &bvideo,int &len,int &delay,void *data)；

读数据包：第一次读时首先读文件头，再读数据包头和数据；其他直接读数据包头和数据。

· LPCSTR GetFileName( )： 获取文件名

·void SetFileName(LPCSTR filename)： 设置文件名

· BYTE GetVideoType( )： 获取视频类型

·void SetVideoType(BYTE videotype)： 设置文件类型

·BYTE GetAudioType( )： 获取音频类型

·void SetAudioType(BYTE audiotype)： 设置音频类型

·int GetVideoWidth( )： 获取视频宽度

·int GetVideoHeight( )： 获取视频高度

·void SetVideoWidth(WORD width)： 设置视频宽度

·void SetVideoHeight(WORD height)： 设置视频高度

5.3.3.2 类 CFileSink 的成员变量及函数：

主要成员变量：

CH323File m_ReadFile： 所读文件

CH323File m_WriteFile： 所写文件

PMutex m_mutex： 线程互斥对象

PSemaphore m_hReadData： 读数据信号量

PSemaphore m_hReadAudio： 读音频信号量

PSemaphore m_hReadVideo: 读视频信号量

主要成员函数:

·void SetVideoFormat(int width, int height, unsigned fps, LPCSTR fmt):

设置视频格式，调用 CH323File 类的 SetVideoWidth( )、SetVideoHeight ( )、SetFps ( )、SetVideoType ( ) 函数。

·void SetAudioFormat(LPCSTR fmt): 设置音频格式，调用 CH323File 类的 SetAudioType ( ) 函数。

·void WriteVideoData(const BYTE * buffer,, unsigned int & len，unsigned char sbit，unsigned char ebit，const RTP_DataFrame & frame)：写视频数据到缓冲区 *videoData* 中。

·void WriteData(const void *buffer, unsigned int len, BOOL isvideo)：先计算与前一包的延迟 delaytime，写缓冲区里的数据到文件，最后写入文件。

·BOOL ReadVideo( ): 读视频

·void ReadAudio(void *buffer, PINDEX &amount)：读音频

·BOOL ReadData( ): 读数据

5.3.4 存储和读写文件的调用链

从第四章提出的测试方案知道，在测试之前要为模拟呼叫器预存特定格式的音视频文件。H.323 终端对模拟呼叫器发起呼叫，应答并建立连接以后，终端以普通形式向模拟呼叫器发送视频数据，模拟呼叫器接收并按包顺序存放到文件。存储过程所要用到的函数之间的调用链如下，其中，MS300Connection、MS300MeetRoom、CFileSink 和 CH323File 是涉及到的类。

MS300Connection::OnIncomingVideo(const BYTE * buffer, unsigned int & len,
unsigned char sbit, unsigned char ebit,
const RTP_DataFrame & frame)

判断是否有视频数据进来，有则写入缓冲区 buffer 中

—> CFileSink::WriteVideoData(const BYTE * buffer, unsigned int & len,
*unsigned char sbit, unsigned char /*ebit*/,*
const RTP_DataFrame & frame)

把缓冲区 buffer 中的数据写入 CFileSink 类为数据设计的缓冲区

—> CFileSink::WriteData(const void *buffer, unsigned int len, BOOL isvideo)

计算出与前一视频包的时间间隔 delaytime

—> CH323File::Write(BOOL &bvideo,const int &len,const int &delay,const void *data)

把媒体类型、包的大小、延迟及数据写入文件

写的过程中，不断有视频数据进来，不断循环调用这些函数，就完成了数据

的写操作。

视频文件建立后，就可以开始测试了。第一个连接建立后，启动读文件线程，虚终端 T0 开始从视频文件中读取视频信号，并把视频信号作 N 份复制放到其余虚终端的发送缓冲区中。

读视频文件过程所要用到的函数之间的调用链如下：

首先在 MS300MeetRoom 类中定义一从文件读数据到缓冲区的线程 ReadFileThread（），在连接建立后，启动该线程。

MS300MeetRoom::OnConnectionEstablished(H323Connection& connection,
const PString & /*token*/)

判断是否有连接建立

—> MS300MeetRoom::ReadFileThread( ) 有则启动读文件线程

—> 循环调用 CFileSink:: ReadVideo（ ） 读取的数据写入缓冲区中

—> MS300MeetRoom::WriteFileVideo(const BYTE * buffer, unsigned int & len,unsigned char sbit, BOOL marker,DWORD timestamp)

—> MS300Connection::WriteBufferVideoFromBuffer(const BYTE * buffer, unsigned int & len, unsigned char sbit, BOOL marker,DWORD timestamp) 把 buffer 中的数据写到每个连接的发送缓冲区

以上给出了模拟呼叫器实现时主要用到的类和类中的成员函数以及函数之间的调用关系。

5.4 模拟呼叫器的改进

模拟呼叫器开发完成后，通过对西安邮电学院通信技术研究所研制开发的 MCU 软件（MediaSwitch300）进行测试，验证其符合事先的设计要求，实现了预先设计的各项功能。针对第四章提出的方案中的不足，为了更好地更广泛地应用于不同厂商的 MCU 设备的测试，需要对设计的测试设备即模拟呼叫器加以改进。

就功能看，模拟呼叫器虽然能接收各接收媒体通道上的音视频信号，但却无法实时显示该信号。解决方案一，设计使模拟呼叫器具有像终端一样的解码能力，并把接收到的信号解码之后存储成文件，用媒体播放器播放；方案二，设计使模拟呼叫器与一个实终端相连，使得点击其上的某一在线用户，立即将该用户接收到的数据传送至该终端，从而显示出来。这是个比较复杂的问题，有待于进一步的研究和探讨。

在软件方面看，如何更有效的利用系统内存资源始终是一个问题。我们能改进的主要是软件结构和算法，使编译器能够更好的自动优化程序，提高系统效率。在关键的运算密集的地方优化程序可以大大提高运行速度。这些都是在软件开发

中应用的方法，也是今后改进的方向。

5.5 MCU 性能测试方案的应用

5.5.1 MCU 性能测试方案的应用

模拟呼叫器开发完成后，对西安邮电学院通信技术研究所研制开发的多点控制单元（MCU）进行了测试，测试结果表明，模拟呼叫器软件功能完善，操作简单方便，且能有效测试 MCU 的性能，符合事先的设计期望。

整个测试系统由西安邮电学院通信技术研究所提供的 10Mb/s 局域网环境、一台 BS3000 终端，一台网闸，模拟呼叫器和被测 MCU（MediaSwitch300）组成。测试的内容为 MCU 处理不同运动程度图像的能力。

整个系统的工作过程如下：

1．当模拟呼叫器和被测 MCU 连入网络时，就利用人工搜索的方式发出寻找网闸的请求，网闸接收到此请求之后发出确认消息。此后，模拟呼叫器和被测 MCU 就可以向网闸发出登记请求，如果网闸接受请求，它们就可以处于待机状态，可以发动呼叫和接纳呼叫。在此过程中，模拟呼叫器就在网闸上登记注册了多个别名 TestMCU-Alias0、TestMCU-Alias1、TestMCU-Alias2……，相当于多个虚终端；被测 MCU 同样也在网闸上登记注册了多个别名 MS300-Alias0、MS300-Alias1、MS300-Alias2……。（RAS 协议）

2．多个虚终端对被测 MCU 发起呼叫，首先将被测 MCU 的别名以及要求的带宽发送给网闸，如果被测 MCU 在网闸上已经注册且处于空闲状态，网闸就会同意接纳此呼叫，则将翻译后的被测 MCU 传输层地址送回。（H.225.0 协议）

3．此时虚终端可以根据被测 MCU 的传输层地址，向对方发出连接请求，对方接收这个请求后，双方就可以进行多媒体通信了。（H.225.0、H.245 协议）

4．整个通信过程中，虚终端将从文件读取来的视频信号经网络发出，同时将网络接收来的视频信号通过被测 MCU 的选路功能经由实终端显示。（H.26X、G.7XX 协议）

5．结束通信过程，需要先向网闸提出申请，网闸同意后，发送确认消息，带宽释放，整个通信过程结束。（RAS 协议）

整个测试过程严格地按照第四章提出的测试方案进行。

5.5.2 测试结果

在测试 MediaSwitch300 处理运动程度不同的图像的性能时，通信带宽和接入终端数量作为变量条件不断改变，如果两者同时改变，测试显得很混乱，而且结果没有规律可循，因此可以使带宽一定，接入终端数量改变，或者接入终端数量一定，带宽改变。最后分析出 MCU 的处理能力与通信带宽和接入终端数量的关系。

设运动程度缓慢的图像为 A，运动程度一般的图像为 B，运动程度剧烈的图像为 C。为便于比较 MediaSwitch300 处理运动程度不同的图像的性能对视频会议系统的各项性能指标的不同影响，下面将主要给出 MediaSwitch300 接入 16、24 和 32 个用户时传输 A、B、C 三种类型图像的测试结果。

图 5.4、5.5 分别示出了接入用户数为 16 和 24 个时对 MediaSwitch300 进行测试的结果。限于篇幅，只给出了其中比较典型的 6 次测试结果。从两个图可以看出，在会议能正常进行的前提下，在通信带宽允许范围之内，MediaSwitch300 在接入 16 和 24 个用户时，点对点图像时延满足视频会议系统的要求。

图 5.6 示出了欲接入更多用户时的测试结果。只给出了其中比较典型的八次测试结果。从图中可看出，在通信带宽允许范围之内，欲测试 MediaSwitch300 接入更多用户时的结果时，当接入数大于 26 且不论用户传送哪种类型的视频信号，点对点图像时延都比较长，不满足视频会议系统的要求；当接入数为 25 时，传送剧烈运动图像时时延较长，传送缓慢和一般运动图像时人眼感觉不到时延，但是已经超过了视频会议规定的要求（150ms）。

测试参数 / 测试序号	图像类型	接入终端数	通信带宽	图像时延	视频质量
第一次	A	16	768K	121ms	画面连续清晰，人眼感觉不到时延
第二次	A	16	1120K	115ms	画面连续清晰，人眼感觉不到时延
第三次	B	16	768K	127ms	画面连续清晰，人眼感觉不到时延
第四次	B	16	1500K	123ms	画面连续清晰，人眼感觉不到时延
第五次	C	16	768K	142ms	画面连续清晰，人眼感觉不到时延
第六次	C	16	1120K	138ms	画面连续清晰，人眼感觉不到时延

图 5.4 接入 16 个用户的测试结果

测试参数 / 测试序号	图像类型	接入终端数	通信带宽	图像时延	视频质量
第一次	A	24	768K	142ms	画面连续清晰，人眼感觉不到时延
第二次	A	24	1120K	126ms	画面连续清晰，人眼感觉不到时延
第三次	B	24	1120K	145ms	画面连续清晰，人眼感觉不到时延
第四次	B	24	1500K	138ms	画面连续清晰，人眼感觉不到时延
第五次	C	24	1120K	153ms	画面连续清晰，人眼感觉不到时延
第六次	C	24	1920K	140ms	画面连续清晰，人眼感觉不到时延

图 5.5　接入 24 个用户时的测试结果

测试参数 / 测试序号	图像类型	接入终端数	通信带宽	图像时延	视频质量
第一次	C	32	1120K	时延明显	画面连续清晰，人眼明显能感到时延
第二次	A	32	1920K	时延明显	画面连续清晰，人眼明显能感到时延
第三次	B	28	1920K	时延明显	画面连续清晰，人眼明显能感到时延
第四次	A	28	1920K	明显时延	画面连续清晰，人眼明显能感到时延
第五次	C	26	1920K	明显时延	画面连续清晰，人眼明显能感到时延
第六次	A	26	1500K	明显时延	画面连续清晰，人眼明显能感到时延
第七次	B	25	1500K	179ms	画面连续清晰，人眼能感决到时延
第八次	A	25	1120K	167ms	画面连续清晰，人眼感觉不到时延

图 5.6　接入更多用户的测试结果

比较上述测试结果，可以得出以下结论：

对于 MediaSwitch300，当接入用户数目小于等于 24 时，在带宽允许范围内，不论用户传送的是哪种类型的视频信号（缓慢、一般、剧烈），在对视频信号进行处理转发之后，能够提供连续清晰的画面，且无马赛克现象。当接入用户数大于 24 时，点对点图像时延相对较大，特别是接入数超过 26 时时延很明显。 因此，不论传送哪种类型的视频信号，测得 MCU 的最大接入终端数都为 24。而且图像运动程度越剧烈，传输所占用的带宽越多。

第六章 总 结

视频会议是网络技术发展的产物。它为不同领域的人们提供了类似于面对面的交流方式，成为远程医疗、远程教育等应用的技术基础。视频、音频和数据相结合的多媒体通信技术已经发展成为信息时代的一种全新信息交递和交流方式，并保持着强劲的发展势头。视频会议技术是融计算机技术、通信网络技术和微电子技术等于一体的产物，它要求将各种媒体信息数字化，利用各种网络进行实时传输并能与用户进行友好的交流。

视频会议系统主要是基于 IP 网络的，IP 网络的特点是传送速率高，但仍然存在一些瓶颈问题。而多点控制单元（MCU）作为视频会议系统中实现多点控制功能的重要控制部件，系统终端通过网络与 MCU 相连，MCU 主要完成会议主席终端和其他终端之间信令以及多媒体数据的传输，它还负责切换、混合来自各终端的多媒体数据，因此系统的瓶颈问题主要发生在 MCU 上，它的性能好坏直接影响了视频会议系统的质量。但是传统的测试方法占用资源多不易实现，而且业界还没有公开的方法，本文就是针对这个问题提出了一套测试 MCU 性能的方案，并实现了一个测试设备模拟呼叫器，而且应用该方案完成了 MCU 处理不同运动程度的图像的性能测试。该方案占用资源少，耗费成本低，简单易行，具有很强的实用性。

本文工作的意义在于：根据测试方案，应用模拟呼叫器能简单方便且有效地测试 MCU 的所有性能，如 MCU 最大接入终端数、MCU 处理图像时延、H.323 协议内容以及多画面性能等等。从而，可以根据 MCU 的实际情况灵活地改变带宽或终端接入数以达到最佳的音视频效果，同时保证视频会议的正常进行。甚至可以根据网络的情况灵活的改变编解码方式，来自适应的调整图像的清晰度，最大可能的利用带宽。另外，对 MCU 软件不够理想的模块可以进一步完善。

此外，由于标准理解的程度不同，使得很多厂家的视频会议系统兼容性不太好，以至于测试方案和测试设备不能广泛地应用。为了更好的和其他产品实现兼容，可以扩充模拟呼叫器的功能，以满足更多的需求。

致 谢

感谢尊敬的导师裴昌幸教授，感谢裴老师两年多的关心、帮助和指导。裴老师渊博的知识，严谨的治学态度，平易近人的作风对我有深厚的影响，在我的学习、工作和生活中给我以勇气和信心。

感谢尊敬的导师朱志祥教授，感谢朱老师为我的设计工作提供了良好的环境，并在我的工作中提供了许多有益的指导和帮助。朱老师丰富的实践经验，渊博的专业知识， 一丝不苟、精益求精的科研作风给我留下了深刻的印象，将使我终生受益。同时感谢西安邮电学院通信技术研究所的所有老师给予本论文工作的关心和帮助。

感谢西安电子科技大学的 104 实验室所有的成员，彼此的友谊是我奋进的动力，他们在工作上和生活上对我的关心和帮助使我可以愉快，顺利地完成硕士阶段地学习生活。

特别感谢辛勤养育我多年的父母和我的姐姐，正是他们在我漫漫的求学生涯中给了我最大的关心、理解、支持和鼓励，并且始终竭尽全力地给予我最好的学习和生活条件。

最后，感谢所有参加论文评审和对本文提出宝贵意见的各位专家、教授以及老师们。

谨借此机会向所有给予我关心、支持和帮助的人士表示由衷的感谢！

参考文献

[1] ITU-T Recommendation H.323v4(2000)，Packet-based multimedia communications systems.

[2] ITU-T Recommendation H.225.0(1998)，Call signalling protocols and media stream packetization for packet-based multimedia communication systems.

[3] ITU-T Recommendation H.245(1998)，Control protocol for multimedia communication.

[4] ITU-T Recommendation Q.931(1993)，ISDN user-network interface layer 3 specification for basic call control.

[5] H.Schulzrinne，VJacobson， etc. RFC 1889， RTP: A Transport Protocol for Real-Time Applicationm，1996，01

[6] OpenH323 Project. http://www.openh323.org

[7] 糜正琨. IP 网络电话技术. 第一版. 北京：人民邮电出版社，2000

[8] 蒋林涛. 多媒体通信网. 第一版. 北京：人民邮电出版社, 1999

[9] Olivier Hersent,David Grule,Jean-Pierre Petit 著，邝坚，戴志涛 译. IP 电话—基于分组的多媒体通信系统. 第一版. 北京：人民邮电出版社，2000

[10] 张磊，王阿禅等. VoIP 语音技术及应用. 第一版. 北京：机械工业出版社，2000

[11] 张明德，王永东. 视频会议系统原理与应用. 第一版. 北京：北京希望电子出版社，1999

[12] 钟玉琢. 多媒体技术（高级）. 北京：清华大学出版社，1999

[13] 基于 IP 可视电话的 MCU 的设计与实现. 浙江大学硕士论文.

[14] 朱志祥，李燕，汪陈伍等. 基于 IP 网络的多点视频会议系统的实现. 通信世界. 2000 年 6 月. 10－11 页

[15] 黄韵，朱志祥，裴昌幸. IP 视频会议系统中 MCU 容量测试方案探讨. 电信科学. 2003 年 11 月，第 11 期. 42－45 页

[16] 张华忠，刘学. 基于 TCP/IP 的多媒体协议的研究. 计算机工程. 2001 年 6 月，第六期. 15－18 页

[17] H.Willebeek-Lemair， Zon-Yin. Videoconferencing over Packet-Based networks. IEEE Journal on Selected Areas in Communications. 1997，15，1101-1114

[18] 徐时新等. 一个基于 Internet 的实时多媒体通信实现模型. 计算机工程与应用. 1998 年 9 月，第 9 期. 18－20 页

[19] 胡绍海，赵帅峰等. 基于 H.323 系统的多点视听会议系统的实现. 北方交通大学学报. 2000，第 6 期

[20] 于波，陈平，张中兆. 视频会议系统中 MCU 的实现及其性能研究. 通信技术. 2001，第 7 期. 17-19 页

[21] 于波，徐松涛等. H.323：基于分组网的多媒体通信标准. 通信技术. 2000，第 2 期. 65-67 页

[22] 陈默，王晓东. 在 Windows 98/NT 下实现 H.323 终端. 电子技术. 2001，第 4 期. 51－53 页

[23] 罗强强，李燕，陆海虹. H.323 视频会议终端的设计与实现. 西安邮电学院学报. 1999，第 3 期

[24] Yu Hen Hu. The past，present，and future of multimedia signal processing. IEEE Signal Processing Magazine. 1997，7，28-51 页

[25] Yangzhen Zou，Changjia Chen. MCU System Software in Video Conference Network. IEEE Communication Transations. 1996，3，173－177 页

[26] Macedonia M R，Brutzman D P. Mbone provides audio and video across the internet. *IEEE* Computer Mag. 1994, 119-123

[27] Guy Vonderweit. A multipoint communication service for interactive applications. IEEE Trans on Communications. 1991, 1875-1885

[28] 张国峰. C++程序设计实用教程. 北京：清华大学出版社，1996

[29] Bruce Eckel 著，刘宗田，袁兆山，潘秋菱等译. C++ 编程思想（第 2 版）. 第一版. 北京：机械工业出版社，2002

[30] David J. Kruglinski 著，潘爱民，王国印 译. Visual C++技术内幕. 第四版. 北京：清华大学出版社，1999

研究生在读期间的研究成果

[1] 党薇，朱志祥，裴昌幸. 基于分组网的多媒体通信标准（H.323）. 电信交换. 2004 年 6 月，第 2 期

[2] 党薇，朱志祥，裴昌幸. 视频会议系统中 MCU 性能测试方法的研究. 西安电子科技大学 2004 年硕士研究生学术论坛