◎研发、设计、测试◎

构建基于SIP与嵌入式多核架构的流媒体服务器

常 青,裴 科,张 刚
CHANG Qing,PEI Ke,ZHANG Gang

太原理工大学 信息工程学院,太原 030024
College of Information Engineering,Taiyuan University of Technology,Taiyuan 030024,China
E-mail:imaupk@163.com

CHANG Qing,PEI Ke,ZHANG Gang.Embedded stream media servers building based on SIP and multi-core architecture.Computer Engineering and Applications,2010,46(24):53-56.

Abstract: Based on the most popular and widely acknowledged framework for embedded digital video system—Davinci framework,this paper adopts Davinci technology typical processor—TMS320DM6446 as hardware platform,constructs a streaming media server.The streaming media server system can implement real-time coding,video storage,real-time supervision and remote control,which makes use of H.264,MPEG4 and AVS as video compression standard,the G.711 and G.728 as audio compression standard,the SIP as access control protocol,the RTP/RTCP/RTSP and UDP as multimedia control and transport protocol.The system has high performance and coding efficiency,low cost and delay time and hard real-time advantages,and it can affluently run in wide area network.

Key words: embedded multi-core system;Session Initiation Protocol(SIP);streaming media server;Davinci technology

摘 要:结合当前最流行且得到业界广泛认可的嵌入式数字视频系统框架——达芬奇技术框架,基于达芬奇技术典型处理器TMS20DM6446平台构建流媒体服务器,用AVS、H.264和MPEG4做视频压缩,用G.711和G.728做音频压缩算法,用SIP做访问控制协议,用RTP/RTCP和UDP等做媒体控制和传输协议;完成具有完整实时编码、录像存储、实时监控、远程控制等功能的网络音视频监控系统。该系统具有高性能、高编码效率、低成本、低延迟、强实时性等特点,能够在广域网上流畅运行。

关键词:嵌入式多核系统;会话初始协议;流媒体服务器;达芬奇技术
DOI:10.3778/j.issn.1002-8331.2010.24.017 **文章编号:**1002-8331(2010)24-0053-04 **文献标识码:A** **中图分类号:**TP368.2

1 引言

网络通信技术、多媒体技术和多核SoC微电子技术的发展,促进了音视频产品向着高清晰、网络化、数字化、智能化和便携化的方向发展,而网络化和数字化是核心。这使得在单一网络平台上实现(Video)、影像(Image)、语音(Speech)及音频(Audio)等多种业务成为可能。而一个视频监控系统网络内部一般还部署其他种类的多媒体系统,随着网络化的进一步发展,各类多媒体系统的融合逐渐成为一种趋势,需要一种灵活的会话控制机制来促进多媒体系统的可扩展性和兼容性。会话初始协议SIP(Session Initiation Protocol)正是一种能够满足需求的信令控制协议。SIP是由Internet工程任务组(IETF)提出的应用层控制(信令)协议,主要用于建立、修改与终止多媒体会话,它也是Internet多媒体通信和控制协议体系的一部分,但其操作和相应的功能独立于其他协议。SIP协议具有的众多优良特性,将使其成为下一代网络融合通信的标准。基于SIP协议所设计的嵌入式流媒体服务器系统可替代传统的视频监控系统,具有高度的兼容性和可扩展性,不仅可以支持多种流媒体文件,方便地与其他多媒体平台进行集成。还可以有效降低网络负载,结合嵌入式平台所建立的流媒体服务器将具有更高性能和更低的功耗。

TI(Texas Instruments)公司达芬奇技术的典型代表TMS320DM6446(简称DM6446)处理器是专为数字视频应用设计,具有专用集成芯片专用性的一面;此外ARM+DSP双核架构的嵌入式处理器兼具有用户可编程和FPGA等可编程器件的灵活性的特点。本文在DM6446处理器强大的硬件平台和软件框架[1]基础上构建了基于嵌入式多核架构的流媒体服务器。

基金项目:国家自然科学基金(the National Natural Science Foundation of China under Grant No.6772101);山西省青年科技基金项目(No.2010021019)。

作者简介:常青(1975-),男,博士生,讲师,主要研究领域为嵌入式个人/家庭瘦服务器,嵌入式系统等;裴科(1980-),通讯作者,男,博士生,主要研究领域为嵌入式高性能计算中间件,片上网络体系结构,嵌入式系统及其应用等;张刚(1953-),男,博士,教授,博士生导师,主要研究领域为语音编码,数字信号处理,嵌入式系统等。

收稿日期:2010-04-12 **修回日期:**2010-06-25

2 系统方案

嵌入式流媒体系统包括服务器与客户机两部分。本文重点讨论流媒体服务器的设计。在系统设计中,采用SIP协议作为TCP通道中的双向控制协议[2],使用实时传输协议(RTP)进行UDP流的封装。因此对服务器端,流媒体系统的互操作部分主要采用SIP协议,并使用SIP中定义的用户代理来实现;RTP采用开源的ORTP库,服务器发送经DSP编码压缩后的码流时需要利用RTP的发送功能。对于客户机端,交互部分同样采用SIP,只需用RTP来实现接收功能,并将其接收到的数据发送到DSP进行解码和显示。

2.1 SIP系统结构

SIP会话使用4个主要组件:SIP用户代理(UA,User Agent)、SIP注册服务器、SIP代理服务器(AS,Agent Server)和SIP重定向服务器。系统通过传输包含有SDP(Standard Description Protocol)协议(用作定义消息的特点与内容)的消息来完成SIP会话。

UA也称SIP终端,它是SIP系统中的最终用户,在IETF文档RFC3261中将它定义成一个应用[3]。按照它们在会话当中扮演的角色的不同,又可分为用户代理客户端(UAC,UA Client)和用户代理服务器(UAS)。其中,由UAC发起呼叫请求,由UAS响应呼叫请求。

UAS的登录要通过SIP注册服务器来完成,为了使UAC可以通过注册服务器找到UAS,所有处于SIP系统网络中的UAS都要通过某个注册服务器登录。在SIP通信时,这些注册服务器会检索接入者IP及相关信息,并将其发送给AS。

AS不仅具有解析名字的能力,还能够作为其前面用户的代理向下一跳服务器发出呼叫请求,服务器再决定下一跳的地址信息,因此AS兼具客户机和服务器的功能。

SIP的呼叫路径通过SIP重定向服务器来规划。SIP重定向服务器得到下一跳地址信息后,立刻通知它前面的用户直接向下一跳的地址发出请求,而其自身则退出对该呼叫的控制。SIP代理服务器、SIP注册服务器及SIP重定向服务器可以在同一个硬件平台上[4]。下面给出一个SIP的呼叫建立过程,如图1所示。

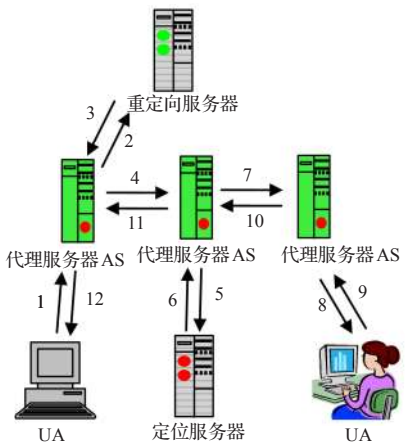

图1 SIP呼叫建立过程图

图1中,从1到12为呼叫建立的过程:(1)UA向本地AS发送Invite(呼叫建立请求);(2)AS向SIP重定向服务器发送Invite;(3)重定向服务器返回重定向的消息;(4)AS向重定向服务器指定AS发送呼叫建立请求;(5)被请求的AS使用域名查询或LDAP等协议(非SIP协议)到定位服务器查询被叫位置(Called-AS);(6)定位服务器返回Called-AS;(7)被请求的AS向Called-AS发送呼叫建立请求;(8)Called-AS向Called-UA(被叫)发呼叫建立请求(被叫振铃或显示);(9)Called-UA向Called-AS发同意或拒绝;(10)Called-AS向主叫(Calling)AS所请求的AS发同意或者拒绝;(11)Calling-AS所请求的AS向Calling-AS发同意或者拒绝;(12)Calling-AS向Calling-UA反馈被叫对呼叫请求的同意或拒绝信息。如果呼叫建立成功,双方会按照他们协商好的媒体形式及其压缩算法等信息进行通讯。

2.2 实时传输协议

实时传输协议(RTP)提供的服务主要包括传输控制、数据序列、时间载量标识和时间戳等,RTP和RTCP(RTP控制协议)共同获得一些数据传输的相关控制信息。UDP和RTP协议一起实现传输层功能。UDP仅传输数据包,不关心数据包传输的时间顺序。UDP分组在承载RTP数据包时,时间标签不是必须有的,因为有时一帧数据要被分割成若干个具有相同时间标签的包。UDP的多路复用技术能够使RTP利用支持显式的多点投递来满足多媒体会话的需求。一般来讲,RTP提供服务时要依据一个具体应用,RTP仅提供了一个协议框架,开发者可根据具体的应用需求对协议进行相应的扩展。从这个角度讲,RTP也属于应用层协议。现在对RTP的研究和设计主要是满足多对多的多用户情况下的网络多媒体会议系统的需要。此外,它也被广泛应用于存储连续数据、交互式分布仿真、IP电话和一些测量及控制等场合中。

2.3 流媒体播放方式

流媒体播放方式主要分为单播、组播和点播广播三种方式。单播是将数据包复制多个备份,并以点对点的传输方式分别将其发送到需要它的用户。广播式将数据包单独的一个拷贝发送给网络上所有用户。这两种传输方式会非常浪费网络带宽。组播方式兼顾了两种传输方式的优点与不足,将数据包一个独立的拷贝发送给需要它的客户。本系统支持单播、组播和点播广播方式。

3 系统组件

基于SIP和多核架构的视频监控系统包括流媒体视频服务器、视频监控客户端以及SIP服务器三部分。由SIP进行信令控制,并配合会话描述协议(SDP)、RTP构成了一个完整的通信系统,并使系统能够穿越局域网在广域网范围内运行[5]。在本系统中,流媒体服务器基于TMS320DM6446处理器平台建立,完成音视频的采集、压缩、传输和系统管理等功能;视频监控客户端在OMAP3530开发平台实现,实现SIP注册、与视频服务器建立连接、音视频数据流控制、播放和存储等功能。SIP网络服务器软件采用SIP Server 2008,它是一个SIP代理服务器(SIP Proxy)、注册服务器(Registrar)和计费服务器。基于SIP和多核架构的嵌入式视频监控系统整体框架如图2所示。

3.1 系统硬件平台

本系统采用TI公司的TMS320DM6446双核处理器[6],ARM采用ARM926EJ-S内核,主频为297 MHz,CPU内置16 KB SRAM和8 KB ROM,16 KB的数据Cache和8 KB指令Cache,外部存储器使用了256 MB的DDR2型SDRAM,64 MB的NAND Flash和16 MB的NOR Flash,外设包括DDR2控制

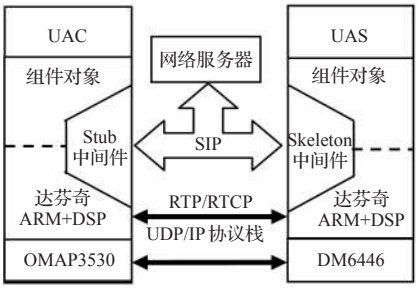

图2　基于SIP的视频监控系统整体框架图

器、ATA硬盘控制器、RS-232标准接口、10/100 Mb/s自适应以太网接口、USB2.0高速接口、CCD和CMOS图像接口、LCD接口、VLYNQ™接口及IEEE 1149.1 JTAG标准扫描接口等[7]；DSP采用C64x+™内核，主频达594 MHz，指令数更是高达4 752 MIPS，CPU具有32 KB L1程序Cache/RAM、80 KB L1数据Cache/RAM和64 KB L2 Cache/RAM，并包含丰富的外设接口[8]。视频处理子系统VPSS包括一个视频前端VPFE和一个视频后端VPBE，包括一个视频图像协处理器（VICP）支持CFA插值、自动聚焦/曝光/白平衡、图像缩放、画中画、隔行逐行转换、OSD数据混合等一系列图像处理功能[9]。其他性能指标本文不再讨论，感兴趣的读者可参考相关文档了解详细的硬件信息。

客户机采用的也是基于达芬奇技术的OMAP3530处理器平台，该处理器集成了600 MHz的ARM Cortex™-A8内核及430 MHz的具有高级数字信号处理算法的C64x+™DSP核，并包含丰富的外设接口[10]。选用的交叉编译工具链为ARM GNU/Linux EABI 2008q1，在客户机端通过该工具链交叉编译以下库文件后即可搭建SIP呼叫界面进而与基于SIP的流媒体服务器进行交互。SIP中主要包含exosip2.h、osip2.h与osipparser2.h等头文件，包含的库有exosip2.lib、osip2.lib及osipparser2.lib。客户机系统设计部分本文不作详细讨论。

3.2　系统软件设计

在宿主机上建立一套针对UAS硬件平台上ARM端的交叉编译环境以及DSP端的DSP/BIOS Link编译环境。移植osip、eXosip和oRTP协议栈使得硬件平台兼具SIP服务器/客户机双重角色，进而灵活地利用SIP和RTP协议实现各种压缩码流的网络传输。这样整个软件开发平台搭建完成。

程序主要包括一个主线程和两个子线程。首先执行RTP的初始化代码，接着打开一个获取实时码流的子线程（即RTP包的发送线程）并准备发送数据；然后主线程进行SIP初始化和注册，接着打开一个SIP监听子线程，主线程进入阻塞状态，该监听线程控制整个程序是否结束；若选择结束，则关闭RTP、SIP及编码器相关的资源。图3是流媒体服务器端的主程序流程图。

3.2.1　SIP的实现过程

3.2.1.1　建立及配置相关的库

实现SIP主要依赖于oSIP库，oSIP库的首个版本在2001年5月发布，目前已经发展到了3.3.0版本。oSIP库采用ANSI C语言编写，代码结构简单小巧，运行速度快。该库主要提供了解析SDP/SIP消息的API及一些事务处理的状态机，不提供高层的SIP会话控制的API。开发UA还需要用到的一个库是eXosip，eXosip库是一个Osip2的扩展协议集，它封装了Osip2

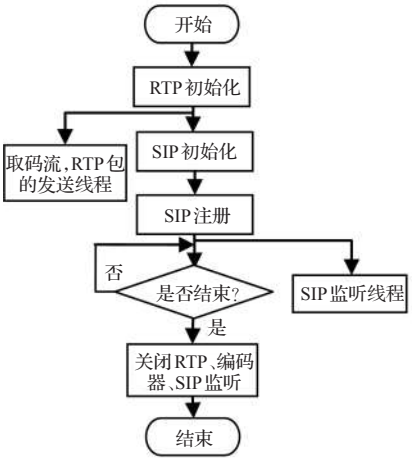

图3　流媒体服务器UAS端主程序流程图

的部分协议栈，使其更加易用。eXosip增加了注册（registration）、呼叫（call）和会话（dialog）等过程的解析，使其具有更强的实用性。可以使用ORTP库来实现语音和视频数据的实时传输。以上这些库在它们的官方网站或其开源社区上都可免费下载得到。

服务器端软件在MontaVista Pro v4.0版Linux系统下实现，编译过程如下。

```
user $ CC=arm_v5t_le-gcc./configure --prefix=$SIP_DIR --host=arm-elf --enable-shared --disable-static
user $ make clean
user $ make install
```

CC=arm_v5t_le-gcc表明采用的交叉编译工具，编译完成的库文件都将安装到SIP_DIR目录下，SIP_DIR是一个用户自定义的环境变量。分别在lib-osip库和lib-eXosip库所在的目录下使用这种方法编译即可。

3.2.1.2　SIP的代码实现及流程

服务器端，SIP初始化完成后发送注册信息，当收到注册成功的信息时，打开一个监听线程；接着进行循环的监听，主线程进入等待状态，并由监听线程控制程序的结束。图4为

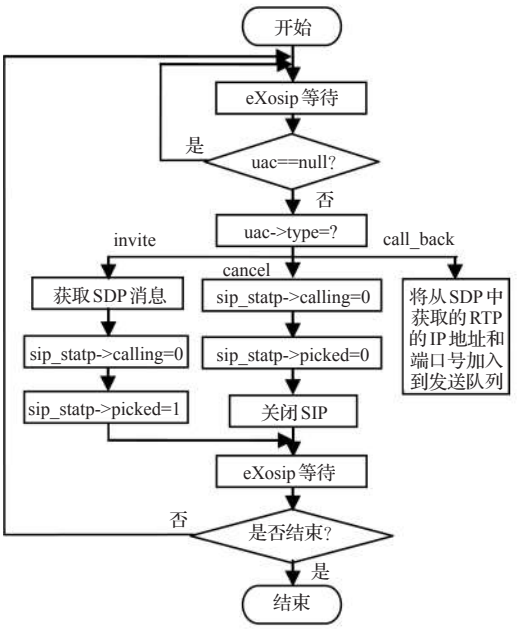

图4　流媒体服务器UAS端SIP流程图

UAS端SIP流程图。

SIP初始化部分的具体步骤如下。

(1)定义了一个数据结构,如下所示:

```
struct sip_stat{
  int codec_id;      //编码器类型ID;
  int call_id;      //呼叫ID条件判断变量;
  int dialog_id;      //会话ID条件判断变量;
  bool doing;      //事件监测变量用于SIP监听循环
  bool calling;      //判断SIP被叫的监听状态;
  bool picked;      //判断SIP接听是否已经连接成功
  bool dtmfing;      //DTMF条件判断变量;
  char *name;      //UAC呼叫用户名;
  char *pwd;      //UAC呼叫密码;
  char *hostip;      //本机的IP地址;
  int hostport;      //本机的端口号;
  char *hostportstr;      //本机的端口描述;
  int rtpport;      //RTP端口号;
  char *rtpportstr;      //RTP端口描述;
  char *server;      //服务器信息
  char *serverip;      //注册服务器的IP地址
  int  serverport;      //注册服务器的端口号
  char *serverportstr;      //注册服务器的端口号描述
  char *callee;      //客户端信息
  char *calleeip;      //客户机的IP地址
  int calleeport;      //客户机的端口号
  char *calleeportstr;      //客户机的端口描述
};
```

(2)变量赋值:主要对本地IP、端口和已知的一些服务器的IP地址及端口进行赋值,使用SIP_init()函数对osip库中的函数进行封装,以下给出了一些由它封装的主要函数及其功能。

eXosip_init(),eXosip的初始化,这个库函数没有输入和输出参数,开发者不必关心其实现细节;eXosip_set_UA(),设置本机(即UAS)的用户名,使得其他SIP用户可以连接,这里使用的用户名为DM6446;eXosip_listen_addr()函数用于设置监听方式,包括UDP/TCP及IPV4/IPV6的选择等。

(3)注册部分:首先要将本机用户名及密码传给eXosip中的变量,然后发送注册信息,并指定其等待回复的时间;如果在指定的时间内收到注册服务器的回复后继续执行下一步,否则返回。

3.2.2 RTP的实现过程

对于发送端,首先对RTP的相关信息进行初始化,然后判断是否可发送。如果可以发送,则把它获得的多媒体数据封装成RTP包发送给客户端。发送结束后释放RTP的信息。

在接收端,同样要对RTP的相关信息进行初始化,接着打开一个端口并开始接收RTP包。当其接收到包时,对RTP包头进行处理即可获得实际的多媒体数据。接收完成后释放掉RTP的信息。使用ORTP库后RTCP的信息接收与发送会自动完成。

发送端使用到的主要代码:使用ortp_init()函数初始化RTP的相关参数;rtp_session_new()函数是对RTP接收或发送的设置;rtp_session_remote_set_addr()函数用于设置客户端RTP的IP地址和端口号;rtp_session_payload_set_type()函数实现媒体属性的设置;rtp_session_send_with_ts()函数完成RTP包的发送;ortp_free()函数用来释放缓冲资源;ortp_exit()函数表示退出RTP主程序。接收端与发送端的实现过程相反,不再赘述。

4 结束语

基于达芬奇技术的双核架构平台上实现了一个高性能、低功耗和服务可靠的流媒体服务器。将流媒体播放技术,SIP、RTP/RTCP和UDP等网络媒体协议和G.711/728、H.264/AVS等音视频编解码技术很好地融合在一起,实现了从流媒体服务器硬件平台完成现场采集->音视频编码->网络传输->客户端软件实时解码显示的平滑结合。使系统具有优异的编解码效率和灵活的网络适应性,是网络流媒体监控系统的新模式。

参考文献:

[1] 张起贵,张胜,张刚,等.最新DSP技术:“达芬奇”系统、框架和组件[M].北京:国防工业出版社,2009.

[2] 刘勇,陈延雄.SIP协议的研究及呼叫控制实现[J].微处理器,2008(3):54-56.

[3] Rosenberg J,Schulzrinne H.RFC3261 IETF SIP:Session Initiation Protocol[S].2002.

[4] 赵哲峰,张刚,谢克明,等.基于SIP的视频监控服务器设计[J].太原理工大学学报,2009,40(4):337-340.

[5] 韩存环,陈利学,宋吉慧.几种SIP穿越NAT解决方案的比较[J].电信快报,2008(5):33-36.

[6] Texas Instruments Inc.TMS320DM6446 digital media system-on-chip reference guide[Z].SPRS283D,Texas,USA,2006.

[7] Texas Instruments Inc.TMS320DM644x DMSoC ARM subsystem reference guide[Z].SPRUE14,Texas,USA,2005.

[8] Texas Instruments Inc.TMS320DM644x DMSoC DSP subsystem reference guide[Z].SPRUE15,Texas,USA,2005.

[9] Texas Instruments Inc.TMS320DM644x DMSoC peripherals overview reference guide[Z].SPRUE19B,Texas,USA,2006.

[10] Texas Instruments Inc.OMAP3530/25 applications processor[Z].SPRS507F,Revised October 2009.