第 34 卷 第 1 期 计 算 机 工 程 2008 年 1 月
Vol.34 No.1 Computer Engineering January 2008
·开发研究与设计技术· 文章编号：1000—3428(2008)01—0267—03 文献标识码：A 中图分类号：TP391

基于 DSP 的 SIP 协议栈的设计与实现

叶晓舟[1,2]，王劲林[2]，张建东[1,2]，邓 峰[2]
（1. 中国科学院研究生院，北京 100080；2. 中国科学院声学研究所，北京 100083）

摘 要：设计了应用于单 DSP 芯片 VoIP 模拟电话适配器中的 SIP 协议栈。采用分层结构和会话状态机设计，完成基本呼叫功能和多种扩展的呼叫功能，实现 SIP 消息的 16 位构建、存储和解析，基于 μCosII 嵌入式实时操作系统创建 SIP 任务。该协议栈具有小巧、灵活的特点，经实践检验具有良好的运行稳定性和兼容性。

关键词：SIP 协议栈；DSP 技术；VoIP 适配器

Design and Implementation of DSP-based SIP Stack

YE Xiao-zhou[1,2], WANG Jin-lin[2], ZHANG Jian-dong[1,2], DENG Feng[2]
（1. Graduate School, Chinese Academy of Sciences, Beijing 100080; 2. Institute of Acoustic, Chinese Academy of Sciences, Beijing 100083）

【Abstract】This paper describes a SIP stack design and implementation for single DSP VoIP ATA. The SIP stack adopts layered structure and session state machine, which accomplishes both basic and extended call functions, and implements 16 bit construction, storage and parse for SIP messages. SIP task is created based on μCosII embedded operating system, it shows good stability and compatibility in practice.

【Key words】SIP stack; DSP technology; VoIP ATA

会话初始协议(SIP)[1]是由互联网工程工作组(IETF)提出并规范化的应用层控制协议，用于创建、修改和终结一个或多个参与者参加的多媒体会话。SIP协议独立于下层传输协议和应用层业务；基于文本方式，容易理解和调试；采用模块化设计，灵活简单；支持移动操作；类似Web的可扩展开放通信使SIP与因特网的其他协议很容易集成。鉴于SIP协议所具有的优点，下一代网络(NGN)将其作为多媒体软交换系统的重要协议，而 3GPP R5 提出的第三代移动通信全IP网络核心控制层——IP多媒体子系统(IMS)实际采用的就是SIP协议。SIP协议本身不提供媒体信息，必须与会话描述协议(SDP)[2]相配合才能提供完整的通信控制功能。SDP对应于SIP消息中的消息体，用于呼叫双方协商媒体特性(如要求带宽、媒体类型、编解码方式等)并提供媒体通道。SIP协议栈包括SIP协议和SDP协议。随着SIP协议的成熟完善和VoIP等SIP应用的快速发展，出现了越来越多的VoIP模拟电话适配器(ATA)、IP电话等嵌入式SIP协议栈终端设备。目前，大多数VoIP ATA的实现采用CPU/MCU＋DSP的处理器架构。SIP协议栈运行于CPU/MCU之上，而DSP负责媒体信号的实时处理，如语音编解码等。这种系统设计复杂、成本较高、功耗较大。嵌入式SIP终端设备的发展方向是使用单DSP处理器架构，基于嵌入式操作系统，由DSP实现从上层应用到底层传输的各种协议。这种方案能够简化系统设计，增强应用灵活性，降低成本，降低系统功耗。基于LSI Logic公司的低成本16 位定点DSP芯片LSI403LP，实现了单芯片VoIP ATA，本文研究了基于LSI403LP的SIP协议栈设计和实现。

1 基于 DSP 的工作方式和设计要求

LSI403LP 为 16 位定点 DSP 处理器，采用哈佛结构，通过 4 路超标量流水结构，在 150 MHz 时钟频率下最高处理速度可达 600MIPS。LSI403LP 具有 48 K 字内存，本文为指令和数据分别分配 32 K 字和 16 K 字；片外指令和数据的可存、取空间分别为 256 K 字，采用了 FLASH 和 SRAM 存储器；实际运行时可执行的指令和数据分别为 64 K 字。LSI403LP 的处理速度完全可以满足 VoIP ATA 的运算要求，非实时性的 SIP 协议栈指令可以在片外 FLASH 上运行，数据在片外 SRAM 上运行，其瓶颈在于可执行的指令和数据空间受限。

由于 LSI403LP 采用 16 位字长，因此不能对数据进行单独的 8 位寻址。基于文本的 SIP 协议栈运行在 16 位 DSP 上会产生浪费 50%数据带宽、运行效率低等问题，这与单芯片 VoIP 应用的设计初衷相背。设计要求实现一种小巧灵活的 16 位嵌入式 SIP 协议栈。

2 SIP 协议栈的设计

2.1 功能设计

各功能应用的请求方法及其规范如表 1 所示。

表 1 SIP 请求方法及其 RFC 规范对照

请求方法	RFC	SIP 协议栈功能
Register	RFC3261 (2543)	注册
Invite ACK Bye Cancel	RFC3261 (2543)	基本呼叫功能
Option	RFC3261 (2543)	能力问讯
Refer	RFC3515	有、无人职守转移
Notify	RFC3265	无人职守转移
Info	RFC2976	DTMF 传送
PRACK	RFC3262	可靠临时应答

SIP协议栈设计需要完成基本的呼叫功能（如呼叫请求、应答和挂断等），还要实现扩展的呼叫功能（如可靠的临时应答、呼叫保持、呼叫等待、呼叫转移以及DTMF传送等）。考虑到现有SIP服务器支持的SIP协议版本不同，除了RFC3261

基金项目：国家发改委下一代互联网示范工程 2005 基金资助项目“视频多媒体点播系统”(CNGI-04-15-2A)
作者简介：叶晓舟(1980－)，男，博士研究生，主研方向：信号与信息处理；王劲林，研究员；张建东，博士研究生；邓 峰，副研究员
收稿日期：2007-01-30 **E-mail：**yexz@dsp.ac.cn

之外，还要兼容老版本的RFC2543[3]。

2.2 分层设计

SIP 协议栈采用分层设计，与 RFC3261 中定义的分层结构略有不同，按照 VoIP ATA 中 SIP 消息的实际处理过程将层次分为：传输适配层，编、解码层，事务层，事务用户层和应用层 API。其中，应用层 API、事务用户层、事务层、传输适配层在垂直方向上有明显的上下层关系，实现自上而下的调用，而编解码层与传输适配层之上的各层交叉，分层结构如图 1 所示。

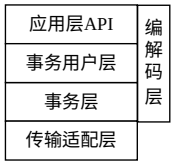

图 1 SIP 协议栈分层结构

传输适配层调用 UDP 套接字发送 SIP 数据，或者从 UDP 层接收 SIP 数据。SIP 协议端口默认值一般为 5060 位。由于大多数 UDP/IP 协议栈采用 8 位数据形式，而本文设计的 SIP 协议栈采用 16 位，因此传输适配层需要执行 8 位/16 位数据存储转换。

编、解码层用于实现 SIP 消息的构建和解析。SIP 消息的构建是指按照需要生成起始行、头域和消息体等有用信息，并将这些信息组合成完整的 SIP 请求或者响应。SIP 消息的解析是指从收到的 SIP 请求或者响应中获取所需信息，提供给 SIP 事务层和事务用户层。

事务层负责维护 SIP 事务状态机，调用传输适配层 API。事务分为客户端邀请（Invite）、非邀请（non-Invite）和服务器邀请、非邀请事务。非邀请事务比较简单，任何确认响应(2XX~6XX)均能够结束一个非邀请事务。邀请事务由 Invite 方法创建、需要复杂的状态机处理。

事务用户层创建和取消事务，处理各种 SIP 请求和响应。事务层和事务用户层共同维护会话状态机。

应用层 API 向 SIP 任务提供各种会话的控制 API，用于完成呼叫、应答、挂断、保持等动作引发的 SIP 处理。

2.3 会话状态机设计

一个正常的会话由 Invite 事务发起，通过一系列响应和 ACK 建立，并通过 Bye 终结。主叫与被叫分别维护各自的会话状态机。主叫端的会话状态包括：空闲状态，邀请状态，连接进行状态和连通状态。被叫的会话状态包括：空闲状态，连接进行状态和连通状态。

主叫的状态转移图如图 2 所示。在空闲状态下，主叫发起 Invite 呼叫进入邀请状态；在邀请状态下，如果收到 1XX 临时响应则转到连接进行状态，如果收到 4XX~6XX 响应或者本地挂机，则返回空闲状态，如果收到 3XX 响应则返回空闲状态并重新发送 Invite；在连接进行状态下，如果收到 200 OK 响应则进入连通状态并发送 ACK，如果收到 4XX~6XX 响应则返回空闲状态，如果收到 3XX 响应则返回空闲状态并重新发送 Invite，如果本地挂机则发送 Cancel 请求并返回空闲状态；在连通状态下，如果本地挂机则发送 Bye 请求并返回空闲状态，如果对方挂机则接收 Bye 请求、发送 200 OK 响应并返回空闲状态。

被叫的状态转移图如图 3 所示。在空闲状态下，如果接收 Invite 请求则发送 1XX 响应并进入连接进行状态；在连接进行状态下，如果本地发送 200 OK 响应并收到 ACK 请求则进入连通状态，如果收到 Cancel 请求则返回空闲状态，如果本地发送 3XX~6XX 响应则返回空闲状态；在连通状态下，如果收到 Bye 请求则发送 200 OK 响应并返回空闲状态，如果本地挂机则发送 Bye 请求并返回空闲状态。

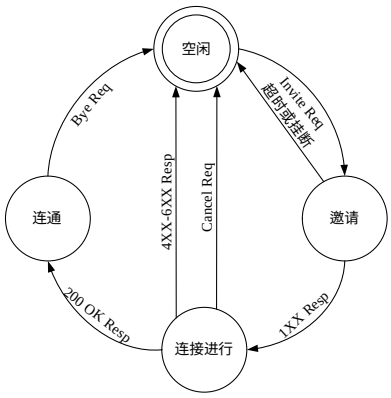

图 2 主叫状态转移图

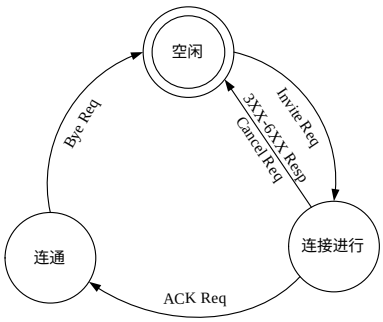

图 3 被叫状态转移图

3 SIP 协议栈的实现

3.1 SIP 消息的 16 位实现

在 16 位 DSP 中，文本数据存储占用低 8 位，高 8 位补 0，造成 50%数据带宽的浪费。而采用 16 位存储方式，即每 2 个文本数据合并成一个 16 位数据，奇数字节放在高 8 位，偶数字节放在低 8 位，数据带宽利用率接近 100%。

SIP 消息由很多信息段组合而成，这些信息段有些按照文本方式存储，有些按照 16 位字段方式存储，SIP 消息的构建需要调用 8 位/16 位转换函数，最终生成的 SIP 消息是 16 位存储的。

SIP 消息的解析采用“懒汉”算法，即将收到的 SIP 消息按照起始行、头域和消息体等内容拆开并分别存储，在解析时对每部分内容分别解析，获取其中有用的信息。为了节约数据空间，收到 SIP 消息的各部分采用 16 位存储方式。直接在 16 位存储的字段上解析文本内容需要大量的移位和文本操作，因此，本文开辟了一段缓冲区，用于将 16 位存储内容转换成文本方式，然后再进行解析。这样，既可以减少 SIP 消息解析所需的数据空间，同时又不增加代码量。

3.2 事务层和事务用户层的实现

在单芯片 VoIP ATA 中，实现了呼叫保持、人工转移等多种扩展呼叫功能，需要同时保存两路 SIP 会话信息。因此，本文为客户端和服务器分别提供了 8 个事务信道，可以分别处理和维护 8 个事务状态。由于采用了 UDP 传输协议，因此事务层通过重传机制保证传输可靠性。每一个 SIP 消息都需要在事务层进行保存，在需要重传时通过传输适配层发送出去。对应事务层客户端和服务器的各 8 个事务信道，保存的重传数据也需要为为客户端和服务器各开辟 8 个存储空间。

由于在这里存储的是完整的 SIP 消息，因此重传 SIP 消息数据区是整个 SIP 协议栈中数据量最大的，约占 80%，必须采用 16 位数据存储。事务的数据结构如下：

```
{     int SIP_TRANS_STATE,
int SIP_TRANS_METHOD,
int SIP_TRANS_RX,
int SIP_TRANS_Timer,
UDPSoc SIP_TRANS_UDPSocket,
BOOLEAN SIP_TRANS_PRACK          }
```

其中，SIP_TRANS_STATE 表示事务当前状态；SIP_TRANS_METHOD 表示事务由何种请求方法创建；SIP_TRANS_RX 表示重传消息的存储空间地址；SIP_TRANS_Timer 用于重传计时以及超时处理；SIP_TRANS_UDPSocket 表示事务层调用的 UDP 头信息，用于发送重传消息；SIP_TRANS_PRACK 表示邀请事务是否需要可靠的临时应答。

事务用户在创建新的事务时，在 8 个事务信道中寻找 SIP_TRANS_STATE 为空闲状态的信道，并在发现的空闲信道上存储事务信息。事务用户层根据接收 SIP 消息中的 CallID 信息，判断该 SIP 消息是否属于已存在的会话。事务层根据请求方法和 branch 信息，判断该 SIP 消息属于哪一个事务。事务状态的变化需要改变事务相应的 SIP_TRANS_STATE 信息。在事务结束时，需要将 SIP_TRANS_STATE 置为空闲状态，同时将其他信息清空。

按照 RFC3261 实现了邀请(Invite)事务，但是在单芯片 VoIP ATA 的实际运行过程中，发现少数 SIP 服务器执行 RFC2543 的 SIP 标准，造成了 SIP 协议栈事务层信道耗尽。其原因是 RFC3261 在发送 Cancel 消息后，除了收到 Cancel 消息的 200 OK 响应外，还需要 487 响应及 Ack 等后续信令流程才能结束邀请事务。RFC2543 不需要后续信令，这就导致事务层一直等待而无法结束邀请事务。一种解决方法是在收到 Cancel 请求的响应 200 OK 后设置等待时间，超时则自动清空该事务。

3.3 SIP 任务

在 LSI403LP 上使用 μCosII 操作系统建立 SIP 任务。SIP 任务通过接收操作系统消息调用相应 SIP 协议栈应用层 API，包括：SIP_Register()发送注册消息；SIP_CallReq()发起呼叫请求；SIP_Hangup()本地挂机处理；SIP_LocAnsw()本地呼叫应答；SIP_Forward()呼叫转移；SIP_UnAttendTransfer()无人职守呼叫转移；SIP_AttendTransfer()有人职守呼叫转移；SIP_CallHold()呼叫保持；SIP_DTMFInfo()发送 DTMF 键值；SIP_UDPHandle()解析 UDP 上传的 SIP 消息；SIP_TimerHandle() SIP 负责协议栈计时处理。以上 SIP 协议栈应用层 API 由 SIP 任务消息触发，SIP 任务在完成某一函数调用后，通过 OSQPend()等待新的消息到来。

3.4 实现尺寸

本文实现了小巧灵活的 16 位 SIP 协议栈，在 LSI403LP 芯片上编译的数据和代码如表 2 所示。SIP 协议栈全部代码在片外 FLASH 中运行，所有数据在片外 SRAM 中运行，运行速度满足要求。

表 2　SIP 协议栈所需数据代码量

代码/字长	数据/字长
14 371	20 671

4　结束语

本文实现了应用于单 DSP 芯片 VoIP ATA 中的 SIP 协议栈。针对 LSI403LP 的特点，设计实现了一种小巧灵活的 16 位 SIP 协议栈，既能够满足基本呼叫功能，又能够实现多种扩展的呼叫功能，实现的单芯片 VoIP ATA 已经实际应用，实践表明本文设计实现 SIP 协议栈具有良好的运行稳定性和兼容性。随着 VoIP 嵌入式终端的广泛应用，实现的 SIP 协议栈将具有资源和成本优势。

参考文献

[1] Rosenberg J, Schulzrinne H, Camarillo G, et al. SIP Session Initiation Protocol[S]. RFC 3261, 2002-06.

[2] Handley M, Jacobson V. SDP Session Description Protocol[S]. RFC 2327, 1998-04.

[3] Handley M, Schulzrinne H, Schooler E, et al. SIP Session Initiation Protocol[S]. RFC 2543, 2002-06.

（上接第 266 页）

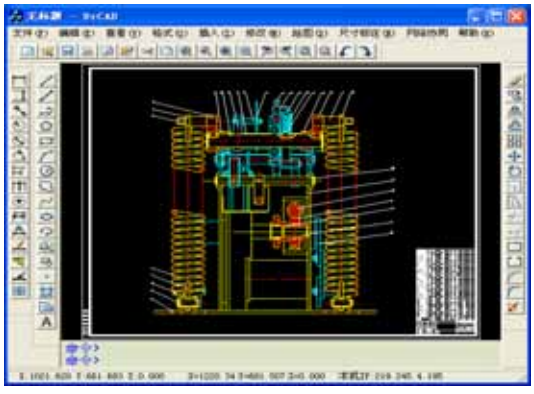

图 4　DrCAD 运行界面

本文的 DrCAD 系统是对原系统进行体系结构层面上彻底重构而成的。它具有更强的开放性与集成性，运行效率也大大提高。但在功能上，除了新增协同工作的功能，和原系统的功能没有太大的变化。将来的工作重点是对系统功能的扩充。例如：支持三维绘图的操作、可以读取和操作更多格式类型的 CAD 图形文件等。本文提出的总线模型及其一些原则也可以作为同类系统组件与体系结构设计的参考。

参考文献

[1] 张友生. 软件体系结构[M]. 北京: 清华大学出版社, 2004: 3-6.

[2] 周小健, 余冬梅, 张聚礼. 基于设计模式的软件体系结构研究[J]. 甘肃工业大学学报, 2003, 29(4): 99-102.

[3] 孙昌爱, 金茂忠, 刘　超. 软件体系结构研究综述[J]. 软件学报, 2002, 13(7): 1228-1237.

[4] Oki B, Pfluegl M, Siegel A, et al. The Information Bus——An Architecture for Extensible Distributed Systems[J]. Operating Systems Review, 1993, 27(5): 58-68.

[5] 王云鹏, 黄松波, 雷　毅. CAXA 电子图板的体系结构与核心技术[J]. 计算机辅助设计与图形学报, 2002, 14(3): 283-286.

[6] 陆　薇, 孙家广. CAD 支撑系统构件－软件总线模型[J]. 计算机辅助设计与图形学报, 2001, 13(1): 1-7.

[7] 张秋余, 袁占亭, 翟志万, 等. 分布式计算机软件总线体系结构研究与设计[J]. 计算机工程, 2004, 30(20): 109-110.