

# 嵌入式流媒体服务器的设计和实现

曾 金,毛燕琴,沈苏彬

(南京邮电大学 计算机学院,江苏 南京 210003)

**摘 要:**为了解决嵌入式环境下的视频传输,提出了一个嵌入式流媒体服务器的应用设计方案。采用了流媒体传送技术、控制协议技术、进程间通信技术,使用开放源码流媒体服务器 live555 项目设计和实现了一个嵌入式流媒体服务器,解决了四个技术问题:管道读写视频操作问题,服务器传送视频控制字问题,服务器与客户端视频播放控制协议问题,流媒体服务器异常退出后恢复问题。该流媒体服务器能够正确传输视频数据,客户端能够播放实时视频,播放画面图像质量高、时延小、稳定可靠。

**关键词:**嵌入式系统;流媒体服务器;视频传输

**中图分类号:**TP37

**文献标识码:**A

**文章编号:**1673-629X(2011)07-0081-04

## Design and Implementation of Embedded Streaming Media Server

ZENG Jin, MAO Yan-qin, SHEN Su-bin

(School of Computer Science and Technology, Nanjing University of Posts and  
Telecommunications, Nanjing 210003, China)

**Abstract:** Present an embedded streaming media server application design for solving the problem of embedded video streaming. The embedded streaming media server is designed and implemented by open-source multimedia server live555 project using streaming media technology, control protocol technology and interprocess communication technology. Four problems are solved, that is the operation of reading and writing video in pipe, the sending video control words by server, the control protocols of playing video between server and client, the restoration after streaming media server quit unexpectedly. The server can stream video exactly and client can play the real-time video streaming with good image and small delay. The system is stable and reliable.

**Key words:** embedded system; streaming media server; video transmission

### 1 概 述

随着信息技术的快速发展,流媒体技术在视频监控<sup>[1]</sup>、视频点播和视频会议等方面的应用空前广泛,方便了人们全球范围内的信息交流。当今在全球市场范围内主要的流媒体产品有三个:Windows Media、Real System、QuickTime。这些流媒体产品针对的用户对象是具有雄厚资金和技术实力的大型企事业单位,一般具有很高的性能,能服务于大量的用户,同时其价格也相对昂贵,对硬件的要求较高,需要专业人士进行维护。中小企业或个人用户要求移动性更高的流媒体产品,这类用户的流媒体服务器可能经常需要变更位置,例如在不同位置进行视频监控。一个小型企业或个人

用户一般仅需要为数量较少的内部用户提供流媒体服务,在这种情况下高成本、高性能的流媒体产品显得过于浪费。嵌入式多媒体技术<sup>[2]</sup>是小型流媒体服务器的理想解决方案之一,它能充分满足众多数字视频产品对实时视频处理的需求,流媒体经过几年的发展,未来将向小型化、便携式、无缝交互等方向发展,嵌入式多媒体技术与流媒体技术的结合具有广阔的发展空间、良好的市场前景,具有重要的研究意义和应用价值。

文中基于开放源码流媒体服务器 live555,设计和实现一个嵌入式流媒体服务器,其中涉及到流媒体传送和控制协议,以及嵌入式系统进程间通信技术。

#### 1.1 流媒体传送和控制协议

流媒体传送和控制协议主要包括 RTP、RTCP 以及 RTSP 协议。实时传送协议(RTP)<sup>[3]</sup>是传输视频、音频、仿真数据等实时数据的运输层传输协议,和传统的注重高可靠性的数据传输层协议相比,RTP 更加注重于数据传输的实时性。RTP 提供在多播或单播的网络服务上的端到端网络传输功能,适合应用程序传输实时数据。RTP 协议<sup>[4]</sup>使用 UDP 协议来完成传输协议

收稿日期:2010-12-03;修回日期:2011-03-21

基金项目:国家高技术(863)计划项目(2006AA01Z208);江苏省科技支撑计划项目(BE2009157)

作者简介:曾 金(1985-),女,重庆人,硕士研究生,研究方向为计算机网络、多媒体网络传输;沈苏彬,研究员,博导,研究方向为计算机网络、下一代电信网、网络安全、嵌入式软件、网络计算。

功能,它只负责实时数据的传送,RTP 本身并不能提供可靠的按顺序传输的传送机制,也不提供流量控制和 QoS 服务质量等,这些服务需要 TCP 和 RTCP 协议来配合使用。

实时传送控制协议(RTCP)是进行流量控制和拥塞控制的实时传输控制协议,它与 RTP 协议一起使用。在一次 RTP 会话中,客户端周期性地传送 RTCP 包,该 RTCP 包中包含已发送的数据包的数量、丢失的数据包的数量等信息,服务器收到这些 RTCP 包后,利用这些信息调整传输速率。RTP 与 RTCP 一起使用,能以最小的开销和有效的反馈使传输效率最佳化,所以特别适合传送因特网上的实时数据。

实时传送流协议(RTSP)<sup>[5]</sup>是一种客户端和服务端之间多媒体传输描述协议,该协议定义了一对多的应用程序如何通过 IP 网络传送多媒体数据,它还可以控制流媒体数据在 IP 网络上的实时传送,并且提供用于音频和视频等多媒体的“VCR 模式”远程控制功能。RTSP 协议是一个应用层协议,位于 RTP 协议、RTCP 协议之上,它和 HTTP 协议类似,每个流媒体文件发布的形式都为 RTSP URL。RTSP 与 RTP、RTCP 一起完成流媒体的传输和控制,许多操作系统都支持这些协议。

文中介绍在 RTSP 协议交互过程中,需要使用的 5 种报文:

(1) Option 报文:客户端首先发送 Option 报文,服务器端回应客户端支持的交互方法集合;

(2) Describe 报文:客户端然后发送 Describe 报文,服务器端响应关于请求的媒体 SDP 信息;

(3) Setup 报文:客户端接着发送 Setup 报文,服务器端根据客户端请求,确定传输模式,建立会话,然后返回会话的相关信息;

(4) Play 报文:客户端继续发送 Play 报文,请求服务器端传送多媒体流,服务器收到该请求后,给客户端发送多媒体流;

(5) Teardown 报文:客户端最后发送 Teardown 报文,请求关闭流媒体传送,服务器端响应请求,终止指定 URL 的流媒体传送。

## 1.2 管道

管道<sup>[6]</sup>是 UNIX 操作系统最强大和最具特色的性能之一,管道是进程间通信的一种工具,为了减少系统存储资源的占用开销,一般情况下,系统将管道的最大长度限制在 4096 字节。当向管道中写入的消息超过 4096 字节时,就会产生读进程和写进程之间的同步问题。首先,写进程查找管道文件中当前指针的偏移量 F-OFFSET,然后从该位置开始尽可能写入消息,当管道中数据长度达到 4096 字节时,系统便使写进程进入

睡眠状态,等待读进程把全部消息取走,取走后,文件长度指针又设置为 0,此时,写进程才被唤醒继续向管道中写入数据。为防止多个进程同时读写同一个管道而产生数据读写混乱,可以设置管道文件的 ILOCK 标志项,使用软件锁方式实现多进程对同一管道文件的互斥使用。

管道<sup>[7]</sup>有两种:无名管道和有名管道。无名管道是只能在程序中由系统调用 PIPE 生成,它有两个缺点:第一、无名管道只能在有共同祖先的进程中传递数据;第二、无名管道是依附祖先进程临时存在的。有名管道弥补了无名管道的不足。有名管道是一种永久性的文件,它拥有普通 UNIX 系统的文件名。在 UNIX 系统下可利用 MKNOD 命令建立有名管道,除非刻意删除有名管道,否则它将一直保存在系统中。正因为有名管道用“文件名”来标识,所以只要事先规定某一文件名称,那样所有知道该规定的进程,不论它们之间是否具有继承关系,都可以使用该管道进行通信。

## 1.3 开源流媒体传送技术

live555 是一个为流媒体提供解决方案的跨平台的 C++ 开源项目,它实现了对标准流媒体传送协议如 RTP、RTCP、RTSP、SIP 等的支持。live555 实现了对多种音视频编码格式的音视频数据的流化、接收和处理等支持,包括 MPEG、H. 263+、DV、JPEG 视频和多种音频编码。同时由于良好的设计,live555 非常容易扩展对其他格式的支持。目前,live555 已经被用于多款播放器的流媒体播放功能的实现,如 VLC、MPlayer。live555 项目编译的平台有 unix(包括 Linux 和 Mac OS X), Windows and QNX(和其他 POSIX-compliant systems),适合运行在嵌入式或者低功耗的平台。

文中在分析开源流媒体服务器 live555 基础上,设计和实现了一个嵌入式流媒体服务器,实现 MPEG-4 格式视频流的实时传送。

## 2 设计嵌入式流媒体服务器的问题

文中设计的流媒体服务器框架如图 1 所示,主要包括以下处理过程:采集视频<sup>[8]</sup>;采用编码芯片编码成 MPEG-4 格式;使用开源 live555 项目的流媒体服务器,从管道中读取视频,然后打包成 RTP 数据包,以多播<sup>[9]</sup>方式传输到客户端;客户端播放收到的视频流。

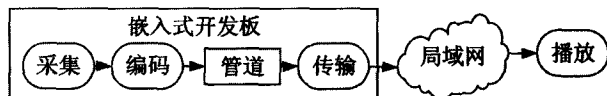


图 1 嵌入式流媒体服务器应用框架

未经改进的开源 live555 项目能够实现 MPEG-4 视频流传输功能,但是 MPEG-4 视频必须保存在本地, live555 项目是读取本地视频文件,然后打包传输。

由于嵌入式系统条件有限,视频不能大容量地保存在本地,需要通过网络传输到客户端播放、保存。因此,文中在嵌入式系统环境下利用开源 live555,设计和实现了实时传输 MPEG-4 视频的解决方案,其中主要解决以下四个技术问题:

#### 1) live555 项目不能传输管道内的视频数据。

管道文件和普通文件的区别:读取普通文件时,打开文件便可以读取文件中的数据;但是读取管道文件必须同时有读/写两个进程端,即读取管道文件时,如果没有对该管道文件的写进程,那么读进程将阻塞;同理如果只打开管道文件的写进程,没有对该管道文件的读进程,那么写进程也会阻塞。

为了传输实时视频,视频编码和 live555 项目使用一个事先约定好的有名管道传输视频数据,例如,该管道文件命名为 fifo. m4e,后缀. m4e 表示视频为 MPEG-4 格式。live555 项目在处理 RTSP<sup>[5]</sup> Describe 报文时,检查客户端请求的文件是否存在,需要对指定文件以读方式打开。如果客户端请求的文件是普通文件, live555 项目可以正常运行;但当客户端请求的文件是管道文件时,如果此时没有对该管道文件的写进程,那么流媒体服务器将阻塞,便无法实现实时视频流传输。

#### 2) live555 项目无法传输嵌入式系统的视频。

live555 项目传输 MPEG-4 视频流时,需要 VOSS、VOS 头部,即 Visual Object Sequence Start (VOSS) and the Visual Object Start (VOS) 头部:

VOSS = 000001B0 00000000;

VOS = 000001B5 00000005;

嵌入式系统的视频并没有产生 VOSS、VOS 的头部。其视频数据如图 2 所示,在视频流的最开始部分是 Video Object 和 Video Object Layer 头部,接下来是一个 I 帧,然后接下来就是 P 帧。

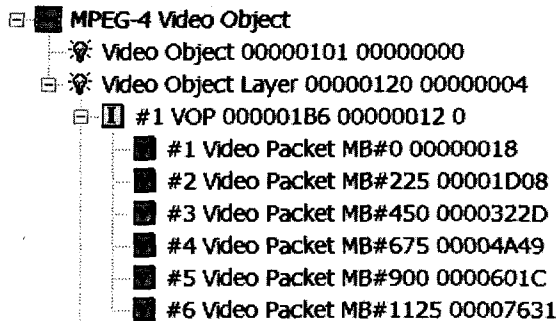


图 2 DM355 编码视频数据

为了使用 live555 传输嵌入式系统的视频,必须在视频中加入 VOSS 和 VOS 头部,格式如下:VOSS+VOS+EncodedStream。

#### 3) 客户端播放视频的问题。

##### (1) 客户端无法播放视频。

在嵌入式系统内编码后的视频,使用 live555 项目

传输到客户端,客户端无法播放视频画面。除非客户端播放器请求的时间早于视频编码程序启动的时间。原因在于 MPEG-4 标准<sup>[9]</sup>中定义了 VO、VOL,即 Video Object 和 Video Object Layer,如图 2 所示。客户端播放器读取到 VO 和 VOL 这两个字段才开始播放视频。然而在一次连续的视频中,VO、VOL 只出现在视频数据的最前面,仅出现一次。因此,当 live555 项目已经将 VO、VOL 传输出去以后,之后请求视频的客户端,将收不到 VO、VOL,所以客户端播放器将无法播放视频画面。

##### (2) 客户端绿屏、花屏现象。

客户端播放时最开始有绿屏、花屏现象。原因在于客户端播放器读取的第一帧不是 I 帧,I 帧是关键帧,之后的帧需要通过 I 帧恢复画面。MPEG-4 标准<sup>[9]</sup>中定义了 VOP\_START\_CODE, 字段值为 000001B6,如图 2 所示。查看 VOP\_START\_CODE 之后的最高两位,00,表示 I 帧;01,表示 P 帧;10,表示 B 帧;11,表示 sprite 帧。

##### 4) live555 项目遇到异常退出,不能自动重启。

嵌入式设备有无人看管的特性,所以在嵌入式设备上运行的应用程序必须具有自动恢复以及记录错误信息日志的功能。live555 项目不具备这样的功能。

### 3 嵌入式流媒体服务器的实现

根据图 1 的嵌入式流媒体服务器框架,下面介绍实时视频传输过程的具体实现以及上面讨论的 4 个问题的解决方法。流媒体服务器和视频编码通过管道传送数据。根据实际情况,流媒体服务器的传输模式为多播<sup>[10]</sup>,如图 3 所示。

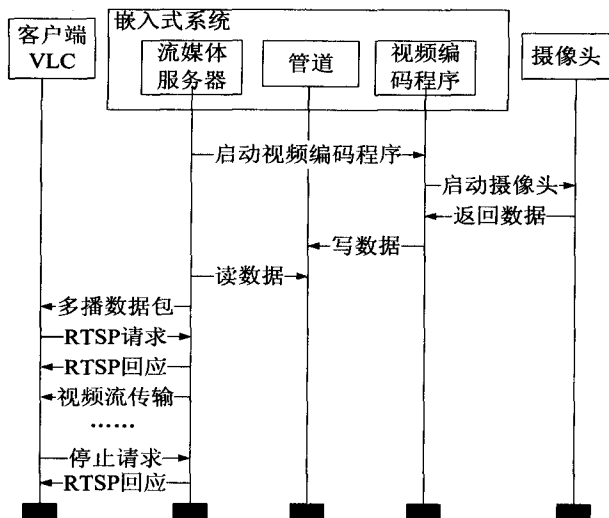


图 3 基于 RTSP 协议的嵌入式实时视频应用系统处理流程

(1) 启动流媒体服务器,初始化流媒体服务器的基本环境。

(2) 启动视频编码程序, 首先对摄像头进行初始化, 设置视频采集相关参数。

(3) 摄像头启动后, 将采集的视频数据返回给视频编码程序。

(4) 视频采集程序接收到摄像头的原始视频数据后, 采用 MPEG-4 视频压缩算法对视频进行编码, 然后将编码后的视频数据发送到管道中。

(5) 流媒体服务器读取管道中的数据, 按照 RTP MPEG-4 规则<sup>[11]</sup> 打包, 然后以多播方式, 向局域网发送视频数据包。

(6) 客户端播放器向服务器请求视频时, 发送 RTSP 请求建立连接, 即 4 个交互过程: Option、Describe、Setup、Play。

(7) 流媒体服务器收到 RTSP 请求后, 回应 RTSP 请求以及视频的相关信息, 然后发送视频数据包。

(8) 客户端停止视频播放时, 发送 RTSP 停止请求报文 Teardown 给流媒体服务器。

(9) 流媒体服务收到 RTSP 停止请求后, 回应客户端, 停止发送数据包。

### 3.1 管道实现

为了避免流媒体服务器运行阻塞, 文中提出两种解决办法:

1) 在编码程序没有运行的情况下, 流媒体服务器在以读方式打开 fifo. m4e 之前, 创建一个以写方式打开 fifo. m4e 的进程, 不写入数据。这种方法的优点是 live555 项目源码改动较少, 缺点是创建额外进程, 带来更多系统开销。

2) 设置 O\_NONBLOCK 标志来打开管道。open(filename, O\_RDONLY) 是以只读方式打开管道, open 调用将阻塞, 直到有一个进程以写方式打开同一个管道为止。open(filename, O\_RDONLY | O\_NONBLOCK) 是以只读非阻塞状态方式打开管道, open 调用立刻返回, 当没有进程以写方式打开同一个管道时, 读进程将返回 0, 不会阻塞。这种方法的优点是不会带来额外开销, 缺点是 live555 项目源码改动较多。文中采用这种方法, 系统开销小。

### 3.2 视频传送实现

要实现 live555 传送 DM355<sup>[12]</sup> 编码视频, 需要加入 VOSS、VOS 头部, 有两种办法:

1) 视频编码程序将视频数据写入管道之前, 先写入 VOSS、VOS, 即 000001B0 00000000 000001B5 00000005;

2) 修改 live555 项目中对 MPEG-4 视频处理规则, 在文件 MPEG4VideoStreamFramer. cpp 中, 添加对 VOSS、VOS 的处理。文中采用这种方法, 处理灵活, 可按需多次传送 VOSS、VOS。

### 3.3 客户端播放视频的实现

#### 1) 客户端无法播放视频。

为了解决这个问题, 文中让流媒体服务器一开始读取视频时, 就保存 VO、VOL 字段, 当后来有客户请求视频时, 首先发送 VO、VOL, 然后再发送视频数据。这样就可以保证客户端播放器能够读取到 VO、VOL, 便可以播放视频画面。

#### 2) 客户端绿屏、花屏现象。

为了解决客户端播放中的绿屏、花屏现象, 文中的解决办法是: 当有客户请求视频时, 流媒体服务器在发送 I 帧之前发送 VO、VOL, 然后再发送其他的帧。这样就可以保证客户端播放器既能读取到 VO、VOL, 又能从 I 帧开始播放, 可以解决绿屏、花屏问题。

### 3.4 自启动实现

文中简单设计一个看门狗监控流媒体服务器, 同时记录流媒体服务器运行信息日志。流媒体服务器出现异常退出后, 能够在  $n$  秒内被检测, 并重新启动,  $n$  根据实际情况取值。使用 shell 脚本实现, 过程如下:

```
if(流媒体服务器正常运行)
sleep n;
else
重新启动流媒体服务器;
```

## 4 系统测试

### 4.1 测试环境

(1) 服务器配置: TMS320DM355, 操作系统 MontaVista, encode 采集编码程序, live555 项目。

(2) 客户端配置: 一般商用 PC, 操作系统 windows, VLC0.8.6 播放器。

(3) 视频文件: MPEG-4 格式的实时视频文件。

### 4.2 测试步骤

1) 开启流媒体服务器, 流媒体服务器会打开采集编码程序。

2) 在客户端打开 VLC 播放器, 点击菜单“文件”, “打开网络串流”, 选择 RTSP, 输入 RTSP://IP/ fifo. m4e, 其中 IP 为服务器端 IP 地址, fifo. m4e 为管道名称, 缓存值, 选择 300, 点击“确定”。

测试表明, 视频传送系统稳定可靠, 实时视频播放画面图像质量高, 达到实时监控的目的。

## 5 结束语

文中设计了一个嵌入式流媒体服务器传送视频方案, 利用嵌入式技术和开源 live555 项目实现了该方案, 讨论了实现过程中的具体问题, 并给出了解决办法, 最后测试了视频传送系统, 达到实时视频传送的效果, 稳定可靠。

(下转第 89 页)

是最优的发布时间,当用户要求的可靠度不那么高的时候, $C_1(T)$ 中的最优发布时间不是整个模型的最优发布时间,但是随着用户要求的可靠度不断提高的时候, $C_1(T)$ 中的最优发布时间比 $C_2(T)$ 中的最优发布时间更优,即使总的软件开发费用最小,成为整个模型的最优发布时间。

表1的数据来自文献[11,12],是取的实际开发过程当中的经验数据。从例一中可以看到, $T_0$ 为 $C_1(T)$ 中的最优发布时间,也是 $C_2(T)$ 中的最优发布时间,所以为此例中最优发布时间,这说明只要一达到用户的要求的可靠度即可马上释放;从例二和例四中能看到尽管 $C_1(T)$ 中的最优发布时间相对比较小,但此时发布软件的总费用不是最小,这说明尽管已经达到用户要求的可靠度,但是继续测试纠错提高软件的可靠度,这样后续维护阶段所减少的成本要大于加强测试期间的测试成本,从而降低总成本;从例三中可以看到当测试的成本很高的时候,尽管还没有达到用户要求的可靠度,但也应该提前释放,使软件总的成本最小;从例六中不难发现当软件的人力测试成本很高的时候,为了使开发总费用最小,应该开发出来之后直接释放。

#### 4 结束语

软件开发出来成败主要还是取决于用户的反应,而且用户对软件的可靠度是具有一定要求的;所以本模型从用户角度考虑了软件的可靠度要求,而且从开发者角度出发考虑了软件发布后失效给开发商带来的影响,更加具有一般化,也更符合实际情况,因此具有一定的实际意义。

#### 参考文献:

- [1] Koch H S, Kubat P. Optimal release time of computer software [J]. IEEE Trans on Software Engineering, 1983, 9(3): 323-327.
- [2] Leung Y. Optimum software release time with a given cost budget [J]. J. System Software, 1992, 17(3): 233-242.
- [3] Gokhale S S. Optimal software release time incorporating fault correction [J]. IEEE Trans on Reliability, 2003, 6(3): 168-169.
- [4] 孙 勇. 软件可靠性模型应用研究 [D]. 南京: 东南大学, 2004.
- [5] 王 容, 唐应辉. 基于用户要求的软件最优时间发布问题 [C]//第四届中国不确定系统年会论文集. 桂林: 桂林电子科技大学, 2006: 193-201.
- [6] Xie M A. Software Reliability Modeling [M]. Singapore: World Scientific Publisher, 1991.
- [7] 胡华军, 杨 波. 软件最优发布时间决策研究 [D]. 成都: 电子科技大学, 2007.
- [8] Yamada S, Ichimori T, Nishiwaki M. Optimal allocation policies for testing-resource based on software reliability growth model [J]. Mathematical and Computer Modeling, 1995, 22(10-12): 295-301.
- [9] Pham H. Software reliability and cost models: Perspectives, comparison and practice [J]. European Journal of Operational Research, 2003, 149(3): 475-489.
- [10] 陈光宇, 黄锡滋. 软件可靠性学科发展现状及展望 [J]. 电子科技大学学报, 2002, 4(3): 99-102.
- [11] Pham H, Zhang X A. Software Cost Model with Warranty and Risk Costs [J]. IEEE Trans. on Computers, 1999, 48(1): 71-75.
- [12] 腾云龙, 唐应辉. 软件可靠性与费用模型的研究 [D]. 成都: 电子科技大学, 2008: 31-33.

(上接第84页)

#### 参考文献:

- [1] 焦 铭, 易小波, 李仁发. 基于嵌入式 Internet 的远程视频监控系统设计 [J]. 计算机技术与发展, 2009, 19(5): 176-179.
- [2] 沈连丰, 宋铁成, 叶芝慧, 等. 嵌入式系统及其开发应用 [M]. 北京: 电子工业出版社, 2005: 85-95.
- [3] Schulzrinne H, Casner S, Fredrick R, et al. RTP: A Transport Protocol for Real-Time Applications [S]. IETF RFC3550, 2003.
- [4] 操龙敏, 蒋建国, 齐美彬. RTP 协议在嵌入式网络摄像机中的设计及实现 [J]. 计算机技术与发展, 2008, 18(3): 214-216.
- [5] Schulzrinne H, Rao A, Lanphier R. Real Time Streaming Protocol (RTSP) [S]. IETF RFC 2326, 1998.
- [6] Matthew N, Stones R. Linux 程序设计 [M]. 第3版. 陈 健, 宋健健, 译. 北京: 人民邮电出版社, 2007: 433-476.
- [7] 李小群, 赵慧斌, 孙玉芳. 进程间通信机制的分析与比较 [J]. 计算机科学, 2002, 29(11): 16-21.
- [8] 陈 亮, 裴海龙, 伍 越. 基于 Video4Linux2 的图像采集程序设计 [J]. 微计算机信息, 2009, 25(7): 65-67.
- [9] Koenen R. Overview of the MPEG-4 Standard [S]. ISO/IEC JTC1/SC29/WG11 N4668, 2002.
- [10] 林予松, 王 荣, 王宗敏. 因特网组管理协议 ICMP 研究 [J]. 计算机应用研究, 2005(1): 200-203.
- [11] Kikuchi Y, Nomura T, Fukunaga S, et al. RTP payload format for MPEG-4 audio/visual streams [S]. IETF RFC 3016, 2000.
- [12] TI. 面向便携高清视频应用 DaVinci DM355 处理器 [J]. 世界电子元器件, 2008(4): 62-65.