第 35 卷 第 4 期 计 算 机 工 程 2009 年 2 月
Vol.35 No.4 Computer Engineering February 2009
·多媒体技术及应用· 文章编号：1000—3428(2009)04—0232—03 文献标识码：A 中图分类号：TP391

基于 ARM 的便携式视频解码终端设计与实现

张云川，王正勇，卿粼波，汪华章
(四川大学电子信息学院，成都 610064)

摘 要：设计便携的嵌入式视频解码终端，从而在 ARM9 芯片和嵌入式 Linux 操作系统上实现 H.264 压缩标准的实时视频接收、解码和播放。介绍系统的研制过程，给出系统构架和实现的关键技术。该系统将先进的视频压缩技术与嵌入式系统相结合，建立了高效的便携式通信平台，有着较大的工程意义和市场价值。

关键词：Linux 嵌入式操作系统；解码终端；视频传输

Design and Implementation of Portable Video Decoding Terminal Based on ARM

ZHANG Yun-chuan, WANG Zheng-yong, QING Lin-bo, WANG Hua-zhang
(Institute of Electronics and Information, Sichuan University, Chengdu 610064)

【Abstract】 This paper presents a portable embedded video decoder terminals. Based on the ARM9 processor of Samsung and the embedded Linux operating system, receiving, decoding and playback of video in real time are achieved, and the video compression standard is H.264. The research background and research process of this system is interpreted. The framework of the system and the pivotal technologies to achieve are offered. The system integrates the advanced video compression technology with embedded system. Highly efficient communication platform is achieved, which has significance for the project and market value.

【Key words】 Linux embedded operating system; decoding terminal; video transmission

1 概述

随着 Internet 技术的发展，网络视频信息传输技术逐渐广泛应用于视频会议、消费电子、工业控制、军事等领域，极大地改变了人们的工作和生活方式。然而，随着社会的发展和应用环境要求的提高，人们更希望通过常备的便携式设备来满足视频信息的获取。因此，利用嵌入式系统实现视频通信功能的方式成了学术界以及工业界的热点 领域。

本文所论述的 H.264 以其优异的压缩性能在视频实时通信、网络视频流媒体传输等各个方面发挥重要作用。H.264 不仅比 H.263 和 MPEG-4 节约了 50%的码率，同时具有良好的网络亲和性，适用于各种传输网络，因此，研究并实现基于 H.264 的便携式视频终端是十分必要的[1]。

本文根据各种电子及视频应用领域对视频解码终端开发的需求，主要研究了新一代视频编码技术 H.264、ARM 嵌入式系统的实现技术，实现了由 ARM9 芯片构成的基于嵌入式 Linux 的便携式视频解码终端。

2 便携式视频解码终端的硬件设计及系统定制

嵌入式解码终端的设计实现首先是搭建硬件平台，然后将嵌入式 Linux 移植到硬件平台，由此建立开发环境。

2.1 硬件设计

S3C2410 芯片是三星公司推出的 32 位 ARM9 处理器，是一款为手持设备等相关应用设计的低功耗、高集成度的微处理器。它包含一个 ARM920T 内核，采用五级流水线和哈佛结构，提供 1.1 MIPS/MHz 的性能，是高性能和低功耗的硬宏单元。这些特征使这款芯片适合嵌入式视频解码终端的开发。

本系统涉及以下 2 个主要的通信过程：

(1)构建 Linux 系统时 PC 主机与目标板之间的通信。PC 主机通过串口用超级终端中的 Xmoden 协议将 Bootloader 下载到 Dataflash 中，复位启动后用 Bootloader 中的 load flash kernel 命令将内核镜像及其驱动程序下载到目标板上。通过以太网将文件系统及所需应用程序下载到 Dataflash 中。

(2)作为便携式视频解码终端时与服务器之间的通信。S3C2410 通过以太网口接收服务器端经过 H.264 编码的源文件，再调用 Dataflash 中的解码程序和播放程序对源文件进行解码处理，同时接收键盘中断的控制。最后将所得的视频文件在 LCD 上显示。

系统基本框图如图 1 所示。

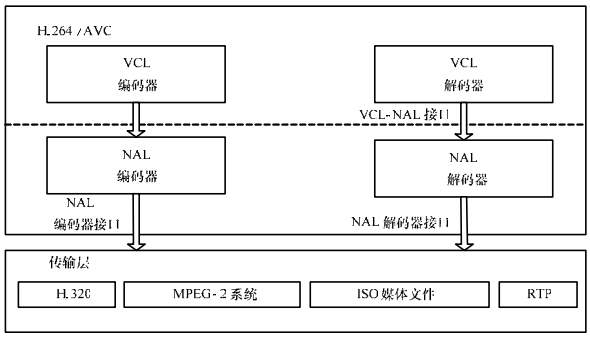

图 1 系统基本框图

作者简介：张云川(1983－)，男，硕士研究生，主研方向：嵌入式系统应用；王正勇，副教授；卿粼波、汪华章，博士研究生

收稿日期：2008-08-12 **E-mail：**cyz331@163.com

2.2 系统定制

嵌入式系统是以应用为中心、软硬件可裁减的、适用于对功能、可靠性、成本、体积、功耗等综合性能严格要求的专用计算机系统[2]。为此，本设计中采用开放源代码的嵌入式 Linux 作为操作系统。首先对 Linux 内核进行剪裁，保留本系统所需的 NAND Device Support、MTD 分区、UDP 协议以及套接字、NFS 文件系统、framebuffer 等驱动；裁减包括并口支持、WLAN 协议支持等驱动模块。将剪裁过后的内核通过交叉编译器编译成 ARM 处理器能够运行的镜像文件。在得到镜像文件后由引导加载程序 Bootloader 将镜像文件下载到目标板的 Dataflash 中。然后通过 NFS 将系统文件下载到目标板的 Dataflash 中。最后将视频解码程序和播放器通过 USB 控制器拷贝到 Dataflash 中。

3 网络传输与 H.264 解码器的实现

本设计中视频来源于网络服务器，经 H.264 解码得到播放的视频文件。

3.1 网络传输

本设计要实现 H.264 码流的网络传输功能，在实现方法上面采用 C/S 模式,所以存在两方面的设计：server 端和 client 端的程序设计。server 端作为服务器需要连接摄像头采集视频。在服务器端通过 USB 摄像头采集图像，然后编码将码流通过网络传送。在 client 端，ARM9 上采用嵌入式 Linux 操作系统，由 socket 编程实现。

现行网络通信协议主要有 TCP/IP, UDP 等协议[3]。考虑到视频解码终端对传输速度及实时性的要求，本设计中选择 UDP 协议作为 C/S 模型的网络通行协议。在 Linux 中，网络编程通过 socket 接口进行，调用包括 sendto()和 recvfrom()等函数完成客户端的监听和接收数据的功能[4]。编程过程中所用到主要 socket 函数如下：

(1)Int bind(int sockfd, struct sockaddr *my_addr, int addrlen)

sockfd 是一个 socket 描述符，my_addr 是一个指向包含本机 IP 地址及端口号等信息的 sockaddr 类型指针，addrlen 常设置为 sizeof(struct sockaddr)。

(2)int send(int sockfd, const void*msg, int len, int flags)

sockfd 是用于传输数据的 socket 的描述符，msg 是一个指向要发送的数据指针，len 是以字节为单位的数据长度，flags 一般情况下设置为 0。

(3)int recv(int sockfd, void *buf, int len, unsigned int flags)

sockfd 是接收函数的 socket 描述符，buf 是存放接收数据的缓冲区，len 是缓冲的长度，flags 也被置为 0。

(4)int sendto(int sockfd,const void*msg, int len, unsigned int flags,const struct sockaddr*to, int tolen)

sockfd 是用于传输数据的 socket 的描述符，msg 是一个指向要发送的数据指针，len 是以字节为单位的数据长度，flags 一般情况下设置为 0，to 表示目的机的 IP 地址和端口号信息，而 tolen 被赋值为 sizeof(struct sockaddr)。

3.2 H.264 解码器的实现

H.264/AVC 作为新一代视频压缩算法，吸收了以往编解码方案的优点，在视频压缩性能上得到了很大的提高。H.264 的算法在概念上可以分为 2 层：视频编码层(Video Coding Layer, VCL)，网络提取层(Network Abstraction Layer, NAL)。为了更好地适应网络传输，H.264 在系统层上提出了一个新的概念，即在视频编码层和网络适配层之间进行概念性分割。采用这样的分层结构有利于压缩编码与网络传输之间的分离，能够更好地适应网络数据信息的封装和对信息进行更好的视频流优先级分类控制。其分层结构如图 2 所示。

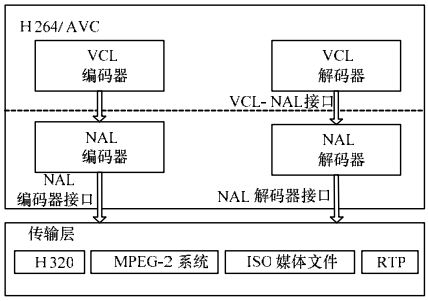

图 2 H.264 分层结构

VCL 包括核心压缩引擎和块、宏块及片的语法级别的定义，其设计目标是在尽可能独立于网络的情况下进行高效的编解码；而 NAL 负责将 VCL 产生的比特字符串适配到各种网络和多元环境中，它覆盖了所有片级别以上的语法级别，同时支持独立片解码、起始码唯一保证、流格式编码数据传送等功能。

NAL 解码器负责将符合 H.264 码流的压缩视频流解码，并进行图像重建。解码流程如下：解码器从 NAL 中接收压缩的比特流，经过对码流进行熵解码获得一系列量化系数 X；这些系数经过反量化和反变换得到残差数据 D；解码器使用从码流中得到的头信息创建一个预测块 PRED，PRED 与残差数据 D 求和得到图像块数据 uF；最后每个 uF 通过去方块滤波得到重建图像的解码块 F。解码器结构如图 3 所示。

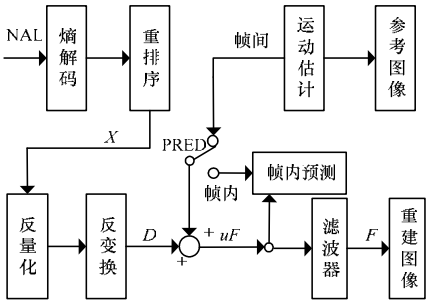

图 3 H.264 解码器结构[5]

3.3 解码器的主要函数

(1)int H264DECInitialize( )

此函数的作用是初始化解码器，如果初始化成功，函数返回 0。

(2)int H264Decode(unsigned char*_puc264Picture, unsigned long _ul264PictureSize)

此函数的作用是针对 1 帧 H.264 压缩数据的解码。如果解码成功，函数返回 0。

(3)int H264GetDecodedPicture (unsigned char*& _puc YUVPicture, unsigned long*& _ulYUVPictureSize, unsigned long*& _ulYUVPictureWidth, unsigned long*& _ulYUV PictureHeight)

此函数的作用是获取解码以后的 YUV 数据相关信息。函数调用后自动为这 4 个参数赋值，得到的_ulYUVPicture Width 与_ulYUVPictureHeight 再赋值给新的变量，得到图像尺寸信息。

(4)int H264DECExit( )

销毁解码器，释放资源。

4 视频终端播放器设计与实现

4.1 播放器逻辑结构分层

整个播放器的逻辑结构为分层次的系统结构。从下往上

依次为数据源访问层、解码器层和视频界面层。数据源访问层负责媒体数据的读取，包括本地数据和网络数据，解码器以动态库的形式加载。在视频界面层主要进行视频播放控制。播放器分层结构如图 4 所示。

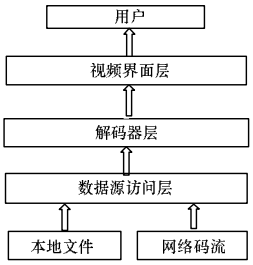

图 4　播放器分层结构

4.2　播放器实现

在本系统中，图像是通过 LCD 显示的。Linux 为显示设备提供了一个叫做帧缓冲(framebuffer)的设备接口，它把显存抽象为一种设备，允许上层应用程序在图形模式下直接对显示缓冲区进行读写操作。这种操作是抽象的、统一的。用户不必关心物理显存的位置、换页机制等具体细节，这些都是由 framebuffer 设备驱动完成的。帧缓冲设备对应的设备文件为/dev/fb*。首先通过 open()函数打开/dev/fb 设备文件，然后调用 ioctl()函数获得屏幕参数，最后将屏幕缓冲区映射到用户空间[6]。这样用户就可以直接读写屏幕缓冲区，进行绘图和图片显示。

4.3　播放控制

播放控制是主控制模块的核心部分。在播放器的工作过程中涉及 2 种状态转换：播放状态的转换和各模块之间的转换。转换过程如图 5 所示。

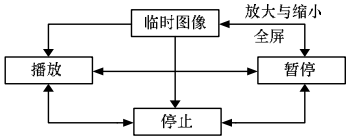

图 5　状态转换

4.4　播放器控制程序设计

播放器程序各状态之间切换部分的实现程序如下：

```
If( SDL_KEYDOWN)
{   // SDL_KEYDOWN 为键盘事件标志
switch(event.key.keysym.sym)
{ // event.key.keysym.sym 为键盘具体事件
case SDLK_SPACE:  {//当有空格键按下，调用解码器进行解码，
//播放解码所得文件；
//调用 recvform()函数接收网络 H.264 码流，并调用解码器解码，
//同时调用 displayframe()播放解码后的视频          }
case SDLK_w:          {//当有 w 键按下，图像放大；调用函数将
//图像放大并显示}
case SDLK_s:
{//当有 s 键按下，图像放大；调用函数将图像缩小并显示}
case SDLK_f:   {//当有 f 键按下，全屏显示；调用函数将图像设
//置为全屏并显示}
 case SDLK_q:
  {   //当有 q 键按下，退出播放程序；
   //关闭 recvform()函数，停止解码，退出 displayframe()函数}
(略)} }
```

5　系统测试

系统测试的工作平台如下：

软件平台：在 PC 上采用 redhat9 作为操作系统，内核版本为 2.4.20，编译器版本为 gcc3.2.2，程序调试工具为 gdb。在 ARM 平台上采用的嵌入式 Linux 内核版本为 2.4.18，交叉编译器版本为 arm-linux-gcc2.95.3。

硬件平台：采用前面介绍的硬件开发平台。其中，显示设备采用 SHARP 公司的 L80T64 型 TFT 数字 LCD，显示分辨率为 640×480，完全可以满足便携式设备的使用需求。

在系统搭建完毕后，为了检验解码器是否能流畅解码 H.264 文件，首先对本地的 H.264 视频文件进行解码测试，期间也对播放器各模块功能及各模块状态转换进行了测试。

视频图像采用 H.264 标准测试图像。测试效果见图 6。

(a)原始图像　　(b)放大一倍　　(c)全屏显示

图 6　测试结果

所测试的 H.264 视频格式为 QCIF(176×144)，在解码播放时，任意时刻可以在此 3 种模式下切换，不影响解码器的工作。

针对网络码流也进行了系统测试。首先，将服务端与客户端通过路由器相连接，保证网络有效。双方通过 IP 地址进行呼叫连接。服务端接收请求后，通过摄像头采集视频并编码打包发送数据，客户端打开相应的端口接收 UDP 数据包，实时解码显示。经测试在 QCIF 格式下，可达每秒 20 帧左右。能流畅地显示网络 H.264 码流。

6　结束语

本文实现了由 ARM9 芯片构成的基于嵌入式 Linux 的便携式视频解码终端。同时将 H.264 视频压缩标准的先进技术与嵌入式处理系统相结合，建立了高效的便携式通信平台。系统下一步将从 2 个方面改进：(1)现行的有线信道更改为无线信道；(2)网络传输采取更为严格的控制协议，使传输丢包率更小、速度更快。

参考文献

[1] 毕厚杰. 新一代视频压缩编码标准——H.264/AVC[M]. 北京: 人民邮电出版社, 2005.

[2] 罗　蕾. 嵌入式实时操作系统及应用开发[M]. 北京: 北京航空航天大学出版社, 2005.

[3] 谢希仁. 计算机网络[M]. 大连: 大连理工大学出版社, 2000.

[4] 孙　琼. 嵌入式 Linux 应用程序开发详解[M]. 北京: 人民邮电出版社, 2007.

[5] Joint Video Team of ISO/IEC MPEG and ITU-T VCEG. Joint Video Team of ITU-T and ISO/IEC JTC 1. ITU Recommendation and International Standard of Joint Video Specification[S]. 2003-03.

[6] 郭盛荣. 基于 Linux 的开放式媒体播放器研究[D]. 重庆: 重庆大学, 2006.