MPlayer播放器在嵌入式平台上的应用

李　巍[1]，贾克斌[2]
（北京工业大学 电子信息与控制工程学院，北京　100022）

摘要：通过对MPlayer播放器进行分析和研究，设计出一种适用于嵌入式系统的便携式媒体播放器，并成功应用到三星ARM9嵌入式平台上。针对播放器整体结构设计、数据处理流程以及分流器，音、视频解码，音、视频同步等各个关键功能模块都作了详细的描述，并给出了详细的移植过程。

关键词：MPlayer；媒体播放器；嵌入式系统
中图分类号：TP36　　**文献标识码**：A　　**文章编号**：1000-8829(2007)S0-0286-04

Application of MPlayer in Embedded Platform

LI Wei[1], JIA Ke-bin[2]
(College of Electronic Information and Control Engineering, Beijing University of Technology, Beijing 100022, China)

Abstract: Through an analysis of MPlayer, a portable media player based on embedded system is designed, and successfully applied to the Samsung ARM9 platform. The structure design of media player and the data processing procedures are also mentioned in detail, as well as every key module such as demuxer, audio-video decoding, audio-video synchronization. The transplant process is also mentions.

Key words: MPlayer; media player; embedded system

随着多种技术的不断融合，移动娱乐市场的市场日益扩大，消费者日益要求在一个移动平台上欣赏照片、电影、电视节目等数字内容，伴随着这股热潮,媒体播放器领域也从红极一时的移动数字音乐播放器（MP3）发展到如今应运而生的便携式媒体播放器（PMP）。然而便携式媒体播放器的核心技术掌握在少数几家方案供应商手中，国内厂商很少具有自主知识产权，在各个方面不得不受制于人，极其被动，这对于国内PMP市场的发展是极为不利的。

本文通过对MPlayer媒体播放器系统构架的分析和研究，将其功能进行必要的修改，并成功应用到嵌入式系统中，设计出一套完整的、通用的PMP软硬件解决方案。该方案不仅可以满足了PMP产品的基本性能要求，更可以借助平台的开放性与兼容性，不断融入更多其他方案不能实现或难以实现的功能，使PMP由单一功能的影片播放向多元化方向发展。

收稿日期：2007-06-18
基金项目：国家自然科学基金项目资助(60672050)；北京市自然科学基金项目资助（4062005）
作者简介：李巍（1983—），男，河北文安人，硕士研究生，主要研究方向为数字多媒体信息处理；贾克斌（1962—），男，河南安阳人，教授，博士生导师，主要研究方向为图像/视频内容检索、编码和处理技术、基于Internet网的多媒体信息处理技术。

1　播放器硬件系统的架构

本系统所采用的嵌入式微处理器是三星ARM9系列的S3C2440芯片，其主频高达400 MHz[1]。嵌入式开发平台上还配有1 MB的Nor Flash，64 MB的Nand Flash和64 MB的SDRAM。Nor Flash可用于存放启动代码，Nand Flash用来放置操作系统和应用程序，这样嵌入式平台可以作为独立的设备而运行。除此之外，平台还提供了LCD控制器、以太网接口、USB接口、硬盘接口、SD/MMC接口、AC97控制器等，具有丰富的扩展功能。

考虑到便携式多媒体播放器的实际需求，媒体文件被存放在SD卡中。系统运行时，读取SD卡中要播放的媒体文件，通过SDRAM缓冲，分解为视频流和音频流。视频流解码后写入FrameBuffer，通过LCD屏幕进行播放；音频流通过AC97控制器解码,送到Speaker输出。硬件系统的结构如图1所示。

2　播放器软件系统的实现

播放器软件的主要功能是：①文件格式解析，即

解析 AVI、MP4、MOV、ASF 等多媒体文件；②将分解出视频和音频数据放到不同缓存区中，分别进行解码；③对音视频数据进行同步播放控制并传送到相应的输出设备上。

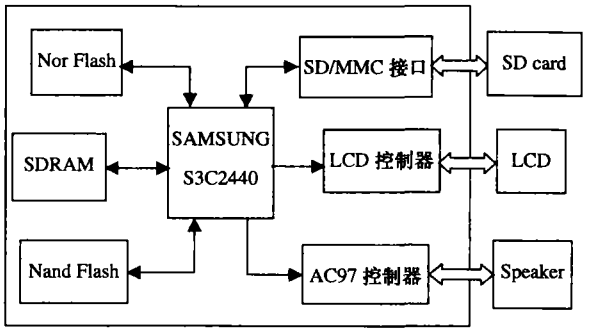

图 1 硬件平台结构图

由于播放器的结构复杂，自行开发需花费大量的时间，因此可以通过研究已有的播放器并对其进行必要的修改，将其移植到嵌入式系统中，构建嵌入式媒体播放器。笔者最终选择了 Linux 上的一款著名媒体播放器——MPlayer。

MPlayer 的功能十分强大，能够播放众多格式的媒体文件，在 X86 PC 机上运行很稳定，也可以把它移植到非 X86 CPU 上的嵌入式系统中来。MPlayer 能使用众多本地的 Xanim、RealPlayer 和 Win32 DLL 编解码器，播放大多数 MPEG、VOB、AVI、OGG、VIVO、ASF / WMV、QT / MOV、FLI、RM、NuppelVideo、yuv4mpeg、FILM、RoQ 文件[2]。除了支持的文件格式多，运行速度快等特点外。MPlayer 作为 Linux 下最优秀的媒体播放器之一，其最具吸引力的地方在于它符合 GNU 协议，是真正意义上的开源软件。

2.1 播放器的逻辑结构

图 2 展示的是 MPlayer 媒体播放器的逻辑结构。该结构主要分为 4 个功能层，分别是：

输入层（input layer）：包含读取媒体文件模块，主要负责将文件中的媒体数据按照流的方式读入进来，并将数据存放到一块缓冲区中，解析文件头从而能够判断出该流属于何种音、视频格式的文件。

分流层（demuxer layer）：即图中分流部分，其主要功能模块为分流器，它的功能是依靠数据头来判断音、视频在这段文件数据中各自的位置，继而对音、视频进行分离。分离后的音、视频数据将分别存入各自缓冲区。分流器同时将提取时间戳(PTS, Presentation time stamp)，随音、视频数据一同传送。通过 PTS，我们可以有效的控制音频和视频的同步输出。

解码层（decoder layer）：该层不但包含音、视频的解码模块，也包含了音、视频解码器的选择模块。在解码层中，由音、视频解码模块根据分流器分离出来的音、视频数据的压缩格式，初始化对应格式的音、视频解码器以便将音、视频数据分别进行解码，将解码出来的信息输出，传递给下一级的输出层。

输出层（output layer）：主要包含音、视频同步，音频输出和视频输出模块。这一层由输出模块选择最适合的输出设备驱动，根据 PTS 确定的同步机制（如丢帧或修正 PTS 跳变等）进行音、视频的播放以达到同步的目的。如果要达到一些特定的输出效果，可以先对即将输出的数据进行一些处理，然后再进行输出。

2.2 播放器的目录文件组织结构

为了实现 MPlayer 媒体播放器的强大功能，其源代码的文件非常庞杂，其中一部分功能是嵌入式媒体播放器所不需要的，因此我们要对其进行必要的删改。考虑到嵌入式媒体播放器的需求，我们去掉了其对 TV、VCD、DVD 支持的功能，主要目录及文件功能如表 1 所示。

播放器开始运行后，主程序文件 mplayer.c 负责分流、解码、输出 3 个功能模块的文件调度，如图 3 所示。

A.分流。对输入的媒体数据进行分流所要调用的文件都存放在媒体数据分析分流库<libmpdemux>当中。

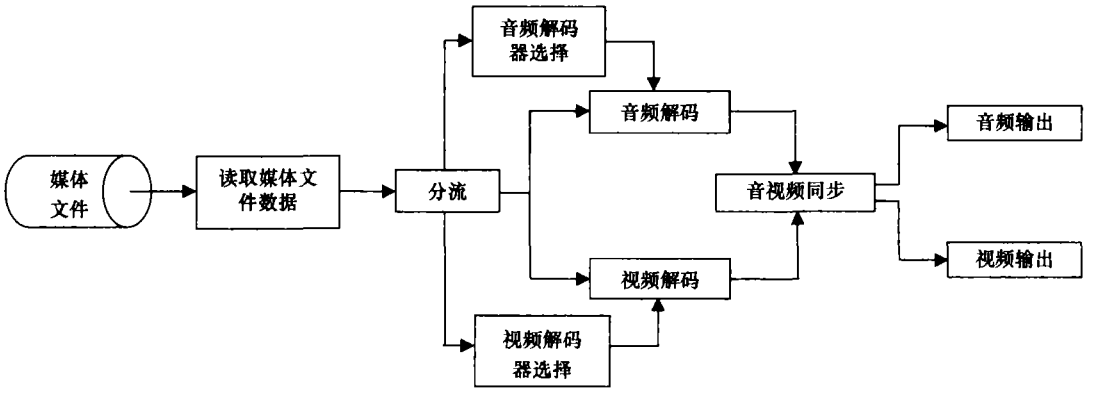

图 2 播放器的逻辑结构

表 1　播放器主要目录及文件功能

目录名	目录功能
<libmpdemux>	媒体数据分析分流库
<libmpcodecs>	媒体编解码库
<libao2>	音频输出库
<libvo>	视频输出库
<input>	外设输入
<osdep>	与 os 相关的文件
<libaf>	音频过滤器库
文件名	文件功能
mp_msg.c	播放器出错信息处理功能实现
playtree.c	播放列表功能实现
cputable.h	CPU 类型定义
cpudetect.c	检测 CPU 类型
mplayer.c	主程序

首先调用 steam.c 文件，判断数据流类型，播放器一共定义了 6 种文件流类型，因而存在 6 种文件，分别是 stream_ file.c， stream_ftp.c， stream_netstream.c，stream_null.c，stream_vcd.c，stream_vstream.c。从中挑选出正确的文件进行数据流的处理。下一步调用 demuxer.c 文件，判断媒体类型，继而调用对应的分流器文件对媒体数据进行音、视频的分流。播放器包含的分流器类型很多，常用到的文件有 demux_asf.c，demux_avi.c，demux_ mpg.c，demux_mov.c 等。

B.解码。解码分为视频解码和音频解码。以视频解码为例，媒体编解码库 <libmpcodecs> 中的 dec_video.c 文件读取视频数据的相关编码信息，判断视频格式。下一步调用视频编解码库<libavcodec>中的 vd.c 文件，从中选择对应的解码器。最后由 vd_ffmpeg.c（以 ffmpeg 解码器为例）对视频数据进行解码。若要对视频进行控制，如亮度设置、对比度设置、图像翻转、放大缩小等，还要从 vf.c 文件中选择相应滤波器对视频进行滤波处理。

C.输出。输出也分为视频输出和音频输出。以视频输出为例，视频输出库<libvo>中的文件 video_out.c 首先判断输出设备类型（如 framebuffer），继而调用相应的设备驱动。通过 vo_fbdev.c 将解码后的视频按照帧的方式从 framebuffer 设备上播放。

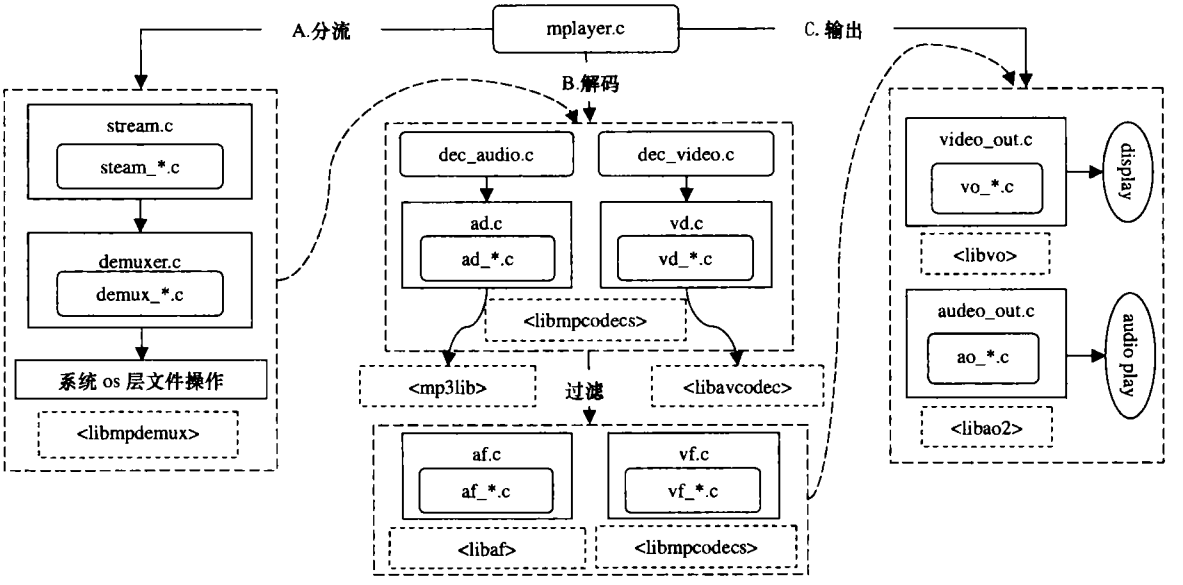

图 3　播放器的目录文件组织结构

3　MPlayer 在嵌入式平台上的移植

要使 MPlayer 在嵌入式平台上顺利的运行，还需要有引导加载程序（bootloader）、操作系统、文件系统的支持。笔者所采用的交叉编译器是 arm-linux-gcc 3.2。该版本运行稳定，并提供了所需的 Linux 库文件。

Bootloader 是系统加电后运行的第一段代码[3]。尽管它在系统启动期间执行的时间非常短，但对于嵌入式系统来说，这是一个非常重要的系统组成部分。其任务是将内核映像读到 SDRAM 中，然后跳转到内核的入口点去运行，即开始启动操作系统。笔者使用的 Bootloader 为 U-Boot 1.1.2，编译后生成 u-boot.bin 文件，通过 JTAG 烧入 Nor Flash 中。

MPlayer 是在 Linux 环境下运行的，因此需要将 Linux 操作系统移植到嵌入式平台上，笔者采用的版本为 2.4.20，已加入对 S3C2440 处理器的支持，但需要用户自己添加 SD 卡的驱动。

根文件系统是 Linux 不可或缺的组件，Linux 内核在系统启动期间的最后操作之一就是安装根文件系统[4]。用户可将编译好的 MPlayer 添加到根文件系

统当中，使其成为系统的一部分，这样每次上电都无须再安装 MPlayer 应用程序了。

需要注意的是，在编译 MPlayer 时，由于 MPlayer 播放器使用嵌入式平台上的 framebuffer 作为视频输出设备，因此要修改 libvo 文件夹下 vo_fbdev.c 文件中所指定的设备路径，使其与根文件系统中 framebuffe 设备的路径相一致。除了 MPlayer 自带的音视频编解码库，用户还可根据自身需要对其进行扩充，从而增加 MPlayer 对各种音视频文件的支持，笔者对其增加了 libpng、libmad、libfaad 这 3 个编解码库，只需在 config 的时候添加“--with-extralibdir=编解码库路径”选项即可。

4 结束语

将编译好的文件依次烧写到嵌入式平台的 flash 中，上电后进入到 Linux 文件系统，通过 mount 指令将 SD 卡挂载到系统中，以便 MPlayer 能够播放其中的音视频文件。

经过测试，该系统可以支持 DivX 3.11，DivX 4.X，DivX 5.X，XviD，WMV，MPEG-1 等视频格式及 MP3、aac 等音频格式，播放分辨率为 320×240 的多媒体影片，其帧率可以达到 30 fps，音视频时间差不超过 0.1 s。同时播放器可以完成包括快进快退、图像旋转、屏幕捕获、字幕显示等功能。另外通过添加解码库，可以支持更多的音、视频格式。由此可见，该播放器完全满足便携式多媒体播放器的需求。

本文通过分析 MPlayer 媒体播放器，提出了一套便携式多媒体播放器的解决方案。该方案能够大大缩短开发周期，降低成本，且系统的扩展能力强，便于维护和升级，具有良好的市场前景。

参考文献：

[1] SMDK2440 board application note[R]. Samsung Electronics, 2003.

[2] Mplayer—movie player[EB/OL]. http://www.mplayerhq.hu/DOCS/HTML/en/intro.html, 2005.

[3] 孙天泽，袁文菊，张海峰. 嵌入式设计及 Linux 驱动开发指南——基于 ARM9 处理器[M]. 北京：电子工业出版社，2005.

[4] Karim Yagbmour. 构建嵌入式 LINUX 系统[M]. O' Reilly Taiwan 公司，译. 北京：中国电力出版社，2005.

□

（上接第 285 页）

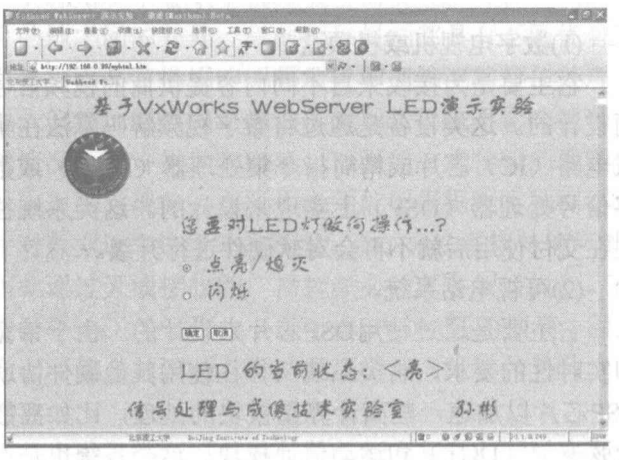

图 2 Web Server LED 远程监控演示图

以上功能可以通过 SSI 技术实现。SSI（Server Side Includes），是 Web Server 提供的一套命令，直接由 Web Server 解释执行，对客户端没有限制。这些命令直接嵌入到 HTML 文档的注释内容，当服务器发现客户端请求的网页中由 SSI 命令标记时，嵌入式 WebServer 解释标记语言，并把相应监视函数的结果作为 HTML 的内容替换标记语言。例如网页内容中包含 HTML 代码“ LED 的当前状态：<!--#echo var="stat"--> ”。其中，“#echo”命令的作用是显示变量 stat 的数值，WebServer 解析命令后，先通过监视函数读取 GPIO 的当前电平为 1（高），然后用“亮”代替原网页代码中的 SSI 命令，将网页内容发送到客户端，我们就会在客户端的浏览器上看到“LED 的当前状态：亮”。由于监视函数是实时读取的，所以网页内容也是动态生成的，从而达到实时监视的目的。

4 结束语

经过实际测试，移植到 VxWorks 系统下的 GoAhead Web Server 能够方便地实现远程监控功能，工作高效、稳定。基于嵌入式 Web Server 的控制器是中小型机电设备远程监控较好的选择方案，在智能家居和楼宇自动化方面也将会得到广泛应用。本文介绍的 GoAhead Web Server 移植及应用具有一般性，对其他平台下的 Web Server 构建也有参考价值。

参考文献：

[1] 彭绍林. 基于 S3C2440A 嵌入式平台的 VxWorks 移植[D]. 北京理工大学, 2006.

[2] 彭绍林，周治国，刘志文. VxWorks 实时操作系统下网络芯片 DM9000 驱动的实现[J]. 测控技术，2006, 25(S0).

[3] Tornado user's guide, 2.2[z]. Wind River Systems, Inc.,2002.

□