第 34 卷　第 12 期　　计　算　机　工　程　　2008 年 6 月
Vol.34　No.12　　Computer Engineering　　June 2008
·多媒体技术及应用·　文章编号：1000—3428(2008)12—0200—03　文献标识码：A　中图分类号：TP393

JPEG2000 高清采集系统软件设计与实现

李　强[1,2]，王兴东[1,2]
(1. 上海交通大学电子工程系图像通信与信息处理研究所，上海 200240；2. 上海交通大学数字媒体处理与传输重点实验室，上海 200240)

摘　要：介绍一套 Windows 平台下基于 JPEG2000 采集卡的高清采集系统，设计并实现了采集卡的驱动程序与应用程序。驱动程序、应用程序以及驱动程序与应用程序的同步分别采用 DMA 传输机制、多线程和 Event 方式。实验表明该系统，能实现高清数据的采集、实时编码与存储，对高清节目制作具有重要意义。

关键词：高清；JPEG2000 采集卡；应用程序

Design and Implementation of HD Capturing System Based on JPEG2000

LI Qiang[1,2], WANG Xing-dong[1,2]
(1. Institute of Image Communication and Information Processing, Department of Electronic Engineering, Shanghai Jiaotong University, Shanghai 200240; 2. Key Laboratory of Digital Media Processing and Transmissions, Shanghai Jiaotong University, Shanghai 200240)

【Abstract】This paper introduces a HD capturing system based on JPEG2000 capturing card in Windows, and the cards driver and application programme are designed and implemented. The Driver is designed in DMA mode, the application programme is designed in Multi-thread, and they are synchronized with event. Experiments show that the system runs well with designed software and the system can capture, code and store HD data at the real time. It is very important for compiling HD programmes.

【Key words】HD; JPEG2000 capturing card; application programme

目前高清节目制作主要采用非线性编辑手段，一般的工作流程是录像机输出无压缩高清数据，经专用采集卡采集处理后存储到专门的存储设备中，然后将采集到的数据导入专业的非线性编辑软件进行编辑。本文介绍了一种非线性编辑系统中利用 JPEG2000 采集卡对无压缩高清数据进行采集编码存储的方法，在 Windows 平台下设计并实现了采集卡的驱动程序及应用程序。

1　系统整体框架与驱动开发

1.1　系统整体框架

采集系统整体框架如图 1 所示，高清原始信号经录像机输出到采集卡，采集卡对接收到的数据进行采集与实时编码，通过计算机的 PCI 接口由驱动程序将编码后的码流上载到计算机的内存，再由应用程序将码流按 J2M 格式(自定义文件格式)打包保存到存储系统中。本系统之所以采用 JPEG2000 作为基本压缩手段，是因为影视制作一般不采用帧间压缩作为压缩方式，而 JPEG2000 帧内压缩效率极高，同时支持无损和有损两种模式，而且做到了一次采集、分级使用(具有多种分辨率)。

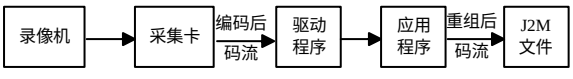

图 1　系统框架

采集卡总体框架如图 2 所示，采集卡内部起编码作用的是 JPEG2000 编码模块，起数据上传作用的是 PLX9656 芯片。

JPEG2000 编码模块采用基于小波变换的JPEG2000 压缩标准，利用该模块进行编码时可以设定多种参数，例如：信号源格式，无损压缩或有损压缩，压缩后码率，压缩后Y/C分量比等，这些参数需要软件层配载编码模块对应寄存器得以设定[1]。

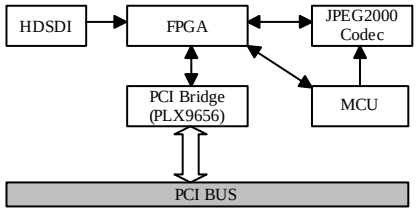

图 2　采集卡框架

由于采集对象是高清数据，数据量大，因此采集卡采用 PCI 接口(33 MHz,133 MB/s)-DMA(直接存储器存取)方式与计算机进行数据传输。本系统采用 PLX9656 作为 PCI 桥芯片。DMA 是由软件层配载 PLX9656 相关寄存器实现的。

1.2　驱动程序开发

驱动程序在操作系统中的位置如图 3 所示[2]。驱动程序是一个包含了许多操作系统可调用例程的容器，这些例程可以使硬件进行相应的操作。任何驱动程序都有一个入口点函数DriverEntry()，它的内部指定系统可调用哪些例程。主要例程包括：

(1)DriverExtension->AddDevice：创建设备对象，初始化设备扩展，注册备接口，以便应用程序能知道设备的存在。

基金项目：上海市科委基金资助项目“网络化高清影视节目制作播出系统关键技术研究”(03DZ15022)
作者简介：李　强(1981－)，男，硕士研究生，主研方向：高清晰度数字电视技术；王兴东，副教授
收稿日期：2007-06-30　**E-mail：**liqiang821015@sjtu.edu.cn

(2)MajorFunction[IRP_MJ_PNP] :处理系统发送过来的各种 IRP(I/O 请求包)，包括启动设备、关闭设备等。

(3)MajorFunction[IRP_MJ_DEVICE_CONTROL] :内部有一个 switch(controlCode){case:…;case:…;…}函数响应用户层各种自定义操作，controlCode 由应用程序传递给驱动程序。

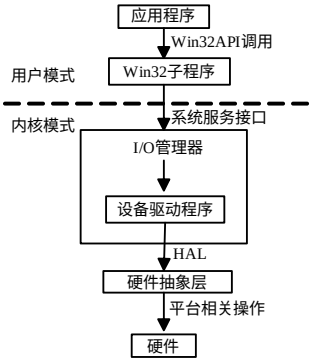

图 3　驱动程序在系统中的位置

驱动程序与应用程序的交互主要是通过定义一系列唯一的IOCTL码(即上文中的controlCode)完成[3]。驱动程序与硬件的交互主要是通过各种硬件抽象层宏实现，主要包括：

(1) READ_PORT_XXX 从硬件 I/O 端口读取一个单值。

(2)WRITE_PORT_XXX 将一个单值写入硬件的一个 I/O 端口。

2　软件设计与实现

2.1　JPEG2000 采集卡驱动程序

本系统采集卡驱动程序的主要任务是控制 JPEG2000 编码模块设定编码参数，利用 PLX9656 完成 DMA 传输，难点在于 DMA 传输及与应用程序的同步。

2.1.1　采集卡初始化

驱动程序需要为设备注册接口，以便应用程序获取设备句柄。采集卡注册接口的实现是：首先用 GUIDGEN 生成一个 GUID，本文驱动程序生成的是 DEF-INE_GUID (HD_GUID,0x6d0666d9, 0x2ba6, 0x4eed, 0x8e, 0xed, 0xa2, 0xb8, 0x1e, 0xa6, 0x66, 0xd2)，其中，HD_GUID 是自定义的变量名，而后在 AddDevice 例程中以 HD_GUID 为参数调用函数 IoRegisterDeviceInterface 完成注册。

不论读写采集卡寄存器还是 DMA 传输都需要知道采集卡的资源，如 I/O 基地址、中断向量号等，这一过程可以在 MajorFunction[IRP_MJ_PNP]处理 IRP_MN_START_ DEVICE (启动设备)时实现。系统描述设备资源的结构是 CM_ RESOURCE_LIST，具体实现代码如下:

```
PCM_RESOURCE_LIST pResourceList;
PCM_FULL_RESOURCE_DESCRIPTOR pFullDes-criptor;
PCM_PARTIAL_RESOURCE_LIST pPartialList;
PCM_PARTIAL_RESOURCE_DESCRIPTOR
pParti-alDescriptor;
NTSTATUS status = STATUS_SUCCESS ;
PHYSICAL_ADDRESS portbase ;
pResourceList=pIrpStack->Parameters.StartDevice.
AllocatedResourcesTranslated;// pIrpStack 可由系统传递的参数获取
pFullDescriptor = pResourceList->List;
pPartialList=&pFullDescriptor->PartialResourceLi-st;
for (i=0; i<(int)pPartialList->Count; i++)
{
pPartialDescriptor=&pPartialList->PartialDescriptors[i];
switch (pPartialDescriptor->Type)
{
case CmResourceTypeInterrupt:…//中断
case CmResourceTypeDma:…     //系统 DMA
case CmResourceTypePort:…    //I/O 资源
case CmResourceTypeMemory:…//Memory 资源
}}
```

2.1.2　编码模块的控制

采集卡初始化后，可以获得它的 PortBase(I/O 端口基地址)，通过硬件抽象层宏写编码模块寄存器，设定编码参数。本文采用的是 WRITE_PORT_ULONG((ULONG_PTR) PortBase+(ULONG)(pShare[0])),(ULONG)(pShare[1]))，其中，pShare[0]是编码模块寄存器偏移地址；pShare[1]是要写入的参数值。但是用户只能在应用层选择编码参数，而不能直接通过驱动程序向寄存器写参数值，这就需要应用程序将要写的参数值传递给驱动程序，本文采用的方法是：应用程序调用 DeviceIoControl，在参数中指明存放寄存器偏移地址与编码参数值的内存地址；驱动程序对应例程函数响应 DeviceIoControl，通过 pShare=pIrp->AssociatedIrp. System Buffer 获取该地址，其中，pIrp 由系统提供。

2.1.3　DMA 设计与实现

DMA 设备使用 DMA Adapter(DMA 适配器)，用来在内存和外围设备间传输数据。申请 DMA 适配器是在 Major Function[IRP_MJ_PNP]处理 IRP_MN_STA-RT_DEVICE 时调用 IoGetDmaAdapter 完成的。本系统采用基于通用缓冲区的 DMA 传输。申请通用缓冲区可以紧跟在获得 DMA 适配器后，通过 AdapterObject(适配器对象)->DmaOperations-> Allocate CommonBuffer 获得通用缓冲区内核模式虚拟物理地址。

PLX9656 负责DMA具体实现，需要配载其内部 3 个参数寄存器：传输字节数寄存器，系统端传输起始物理基地址，设备端传输起始物理基地址[4]。第 1 个设定为 4 MB(经验值)，第 2 个是通用缓冲区物理地址。

由于传输的是高清数据，既有视频也有音频，因此设计了如下的 DMA 传输机制：开辟 2 个 DMA 通道(PLX9656 支持 4 个 DMA 通道)，一个用来传视频，另一个用来传音频；每个通道申请 2 个 4 MB 的通用缓冲区，采用乒乓操作。具体实现是：定义 2 个 IOCTL 码 IOCTL_HDCARD_ STARTDMA_UP0, IOCTL_HDCARD_STARTDMA_UP1，分别对应视频通道、音频通道；MajorFunction[IRP_MJ_ DEVICE_CONTROL]中针对 2 个IOCTL 码分别通过 PLX9656 实现 DMA，视频通道中 PLX9656 的第 3 个参数设定为 0，音频通道中 PLX9656 第 3 个参数设定为 0x400000，这样硬件就知道应该传视频还是音频。

由于每个 DMA 通道采用乒乓操作，因此每完成一次 DMA 传输后必须更改通用缓冲区地址,这个操作是在中断处理程序中完成的。注册中断处理程序是在 MajorFunction [IRP_MJ_PNP]处理 IRP_MN_START_DEVICE 时通过调用函数 IoConnectInterrupt 做的。每个 DMA 通道及通道的缓冲区都设定对应的 Flag，这样中断处理程序就能判断是哪个通道的哪个缓冲区完成了传输。

DMA传输完成后必须通知应用程序，应用程序才能及时处理通用缓冲区内的数据，本文是通过Event的方式通知应用程序的。首先在应用程序中用数组创建 4 个Event，对应 4 个通用缓冲区，通过DeviceIoControl将Event的地址传递给驱动程序；同时驱动程序定义 4 个HANDLE，为hEvent[4]，对

应接收到的 4 个应用层的Event，以hEvent[4]为参数调用ObReferenceObjectByHandle，这样驱动程序中的 4 个hEvent就与应用层的 4 个Event建立了关联，应用程序就可以通过应用层的Event及时获知驱动层DMA传输完成情况。应用程序还需要知道通用缓冲区在用户模式下的虚拟地址才能处理数据，这个问题的解决是依靠系统的MDL(内存描述符列表)。MDL可以跟踪一个与内核虚拟地址缓冲区相关的物理页，进而获得该物理页在用户模式下的虚拟地址。实现的主要代码如下：

```
mdl =IoAllocateMdl(通用缓冲区内核模式虚拟地址,4M,FALSE,FALSE,NULL);
MmBuildMdlForNonPagedPool(mdl);
Pointer=MmMapLockedPages(mdl, UserMode);
```

Pointer 即是通用缓冲区用户模式下虚拟地址。应用程序通过 DeviceIoControl 即可获得通用缓冲区用户模式下的虚拟地址。主要代码如下:

```
PULONG
pShare=(PULONG)pIrp->AssociatedIrp.SystemBuffer;
*pShare = (ULONG)Pointer;
```

至此，驱动程序可以控制采集卡设定编码参数完成 DMA 传输，传输完成后通知应用程序，应用程序可以相应获取缓冲区地址进行存储工作，驱动程序任务完成。

2.2 应用程序

应用层的主要任务是查找到采集卡，控制采集卡采集编码，同时完成存储工作。查找到编码卡是通过 HD_GUID 做到的，后两个任务决定了程序架构采用多线程，一个线程负责视频通道采集，另一个线程负责音频通道采集，还有一个线程负责存储。应用程序一共定义 6 个 Event，4 个用来对应通用缓冲区，还有 2 个 Video_Event, Audio_Event 用来对应视频、音频数据 DMA 传输完成情况。前 2 个线程始终监听与通用缓冲区对应的 4 个 Event，当 Event 被置为信号态即表明一次 DMA 传输完成，立刻进行下一次 DMA 传输，并 SetEvent 对应的 Video_Event 或者 Audio_Event 通知存储线程 ;存储线程始终监听 Video_Event 与 AudioEvent，当两者中其中之一被置为信号态时，即调用对应的视频存储函数或音频存储函数。程序框架如下：

```
视(音)频采集线程
While(TRUE)
{ if(停止采集)
return;
else
{监听视(音)频缓冲区对应 Event;
{
开始下一次 DMA 传输；SetEvent(Video_Event/Audio_Event);
}}}
存储线程
While(TRUE)
{ if(停止采集)
        return;
  else
  {监听 Video_Event 和 Audio_Event;
{
调用相应存储函数;
}}}
```

应用程序、驱动程序、硬件之间的通信机制如图 4 所示。

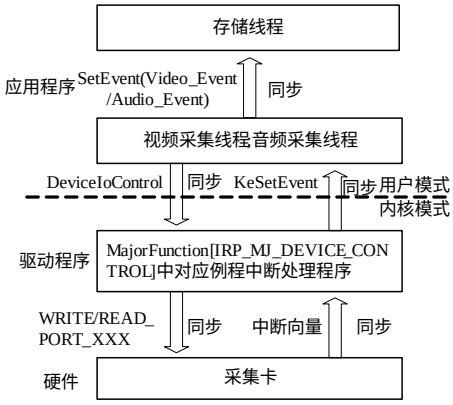

图 4　通信机制

3　实验结果

图 5 为应用程序界面。图 6 为采集到的 J2M 文件，播放器是安装了 J2M 播放插件的 QucikTime。

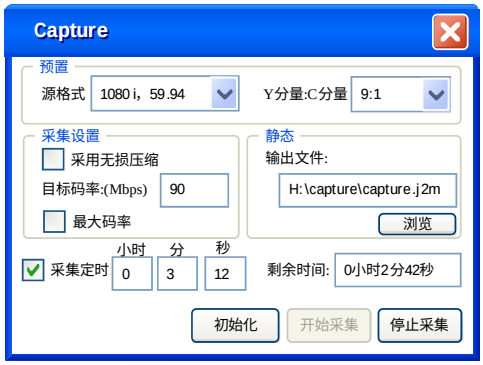

图 5　应用程序界面

图 6　采集到的 J2M 文件

4　结束语

JPEG2000 采集卡驱动程序采用 DMA 进行数据传输，应用程序采用多线程架构，均充分考虑了高清大数据量的特点，并配合硬件实现了高清数据的实时采集编码与存储，对于构建 Windows 平台下高清非线性编辑系统具有重要意义。

参考文献

[1] ADI 公司. ADV202_Users_Manual_rev_2_8.pdf[Z]. 2005.

[2] Oney W. Programming the Microsoft Windows Driver Model[M]. [出版地不祥]: 微软出版社, 1999.

[3] Baker A, Lozano J. Windows2000 设备驱动程序设计指南[M]. 北京: 机械工业出版社, 2001.

[4] PLX 公司. 9656BA_DataBook_v1.1.pdf[Z]. 2003.