黑龙江科技信息

DM6446 的视频编解码及播放实现

王 磊 白 桦

（武警工程学院 研究生管理大队，陕西 西安 710086）

摘 要：DM6446 开发平台将控制全局的 ARM 内核和用于数据及图像视频处理的 DSP 子系统集成在一起，更加适合处理数据复杂的视频信号。主要介绍了视频处理应用程序的构成，着重讲述了视频编解码的过程与播放显示。

关键词：达芬奇技术；DM6446；视频编解码

1 达芬奇技术综述

达芬奇(DaVinci)技术是一种专门针对数字视频应用、基于信号处理的解决方案，能为视频设备制造商提供集成处理器、软件、工具和支持，以简化设计进程，加速产品创新。DM6446 是针对编解码双向的系统，该平台配备一个视频输入，在编码通道方面是可以实现的，另外又增加了一个视频协处理器，协助内部处理单元更快速实现视频编解码，进行视频播放。

Codec Engine 是一个 Codec 执行框架，自动地请求和实现符合 eXpressDSP 的 Codec 和算法，在视频开发过程中，Codec Engine 为算法的执行提供一个标准的软件结构和界面，使得视频编解码过程更具有效率，简化了调试进程。

2 视频编解码实现

DM6446 的视频编码/解码 Engine 模块为：

描 述	模块缩写	包 名	头文件
视频编码器接口	VIDENC_	ti.sdo.ce.video	videnc.h
视频解码器接口	VIDDEC_	ti.sdo.ce.video	viddnc.h

包名对应于路径，应用程序必须使用该路径，引用其包含的头文件，以便使用视频编解码模块，其 #include 说明为：

描 述	#include 说明
视频编码	#include <ti/sdo/ce/video/videnc.h>
视频解码	#include <ti/sdo/ce/video/viddnc.h>

视频编码模块接收来自捕获线程的帧缓冲，调用接口函数并使用视频编码算法对其进行编码，解码模块收到了编码数据后对其先进行解码，而后再把缓存数据送到显示线程中显示出来。

2.1 视频编码流程

视频程序收到采集进来的视频数据后，送到编码缓存中对其进行编码，再送到传输程序将编码后的数据发送出去，其处理流程如下：

(1)视频程序首先通过 FifoUtil_open()函数打开与采集程序之间的通信缓冲，调用 Fifoutil_get()函数和 FifoUtil_put()函数作为视频线程与采集线程之间的数据交流通道；

(2)使用 CE 的 Engine_open()来创建视频编码算法引擎——Engine；返回一个句柄，所有使用相同 Engine 的线程都需要单独的句柄，用来确定线程的安全；

(3)使用 videoEncodeAlgCreate()创建编码算法，这包括：

a.使用 VIDENC_create()里的静态参数来创建 Codec；

b. 使用 VIDENC_control () 和 XDM_SETFARAMS 来设置动态的视频编码参数，查询编码缓冲区大小；

(4)使用 Memory_contigAlloc()函数来申请一段连续的视频数据缓冲并用 XDM_GETBUFINFO 来设置缓冲区的大小；

(5)2 个采集缓存也使用 Memory_contigAlloc()函数来申请连续的内存空间，这 2 个缓存轮流从采集程序读数据，以下为 2 个采集缓存申请连续内存代码示意：

```
#define CAP_BUFFERS 2;
for (bufIdx=0; bufIdx < CAP_BUFFERS; bufIdx++) {
captureBuffers[bufIdx] =
        Memory_contigAlloc (imageSize,Memory_DEFAULTALIGNMENT);}
```

(6)使用 VIDENC_process()调用 H.264 算法对数据进行编码，而后送到传输程序，编码结束。

2.2 视频解码流程

视频程序收到传输程序传送来的编码数据后，先将其进行解码，最后送到显示线程显示出来，其处理流程如下：

(1)视频线程首先通过 FifoUtil_open()函数打开与显示线程之间的通信缓冲，调用 Fifoutil_get()函数和 FifoUtil_put()函数作为视频线程和显示线程之间的数据交流通道；

(2)使用 CE 的 Engine_open()来创建视频解码算法引擎，同样返回一个句柄，所有使用相同 engine 的线程都需要单独的句柄，来确保线程的安全；

(3)使用 videoDecodeAlgCreate()创建编码算法，这包括：

a.使用 VIDDEC_create()里的静态参数来创建 Codec；

b. 使用 VIDDEC_control () 和 XDM_SETFARAMS 来设置动态的视频解码参数，查询编解码缓冲区大小；

(4)使用 Memory_contigAlloc()函数来申请一段连续的视频数据的缓冲并用 XDM_GETBUFINFO 参数来设置缓冲区的大小；

(5)3 个显示缓存也使用 Memory_contigAlloc()函数来申请连续的内存空间，这 3 个缓存轮流给显示程序送数据，以下为 3 个显示缓存申请连续内存代码示意：

```
#define DISPLAY_BUFFERS 3;
for (bufIdx=0; bufIdx < DISPLAY_BUFFERS; bufIdx++) {
    bufferElements[bufIdx].virtBuf=(char *)
        Memory_contigAlloc (imageSize,Memory_DEFAULTALIGNMENT); }
```

(6)使用 VIDDEC_process()调用 H.264 算法对数据进行解码，而后送到显示程序，解码流程结束。

3 视频播放实现

DM6446 平台可以通过 Linux 的 FBDev (Frame Buffer Device)驱动来访问视频输入输出硬件。帧缓冲设备(Frame Buffer Device)是视频硬件设备的一个抽象表示，与视频硬件设备的帧缓冲相对应，允许应用程序通过定义好的接口来访问视频硬件设备，这样应用程序就不必了解任务低层次的接口。视频播放程序即显示线程，选择帧缓冲设备/dev/fb/3 进行视频的显示播放，步骤如下：

(1) 以可读可写方式打开帧缓冲设备/dev/fb/3，返回帧缓冲设备文件描述符；

(2) 用 mmap 函数将帧缓冲设备映射到用户虚拟空间，映射的空间大小为 3 帧图像的大小，即分配了 3 个图像缓冲区与帧缓冲设备对应；

(3)将帧缓冲映射的虚拟地址转换成物理地址；

(4)通过 ioctl 命令配置帧缓冲设备；

(5)解码程序通过物理地址循环向 3 个图像缓冲区写视频数据；

(6)显示线程根据 DSP 端发来的信息决定显示哪个图像缓冲区的数据，如果整个解码过程未结束则重复此步骤，否则执行步骤(7)；

(7)调用 ummap 函数释放帧缓冲映射的虚拟空间；

(8)关闭帧缓冲设备，实现视频播放。

小结

DM6446 平台应用达芬奇双核技术，能够有效的完成数据复杂且庞大的视频信息的编解码过程，实现高质量的视频播放，对数字视频产品的加速创新具有很强的实际应用意义，也为在该平台上进行其他多媒体应用提供了参考。

参考文献

[1]白桦.数字单兵头盔音视频处理与网络传输相关技术研究[D].西安：武警工程学院，2008.

[2]彭启琮.达芬奇技术——数字图像/视频信号处理新平台[M].北京：电子工业出版社，2008.

责任编辑：鲁艳

（上接 229 页）间的关系，中国建筑科学研究院对各类钢筋做过详尽的分析研究，并将试验结果绘制成粘结应力—滑移(T-s)曲线，进行比较。螺旋肋外表比较合理，一方面螺旋肋连续，基面积削弱较少，肋部面积仍可承受拉力；而带肋钢筋因横肋不能受力，有效面积要损失 6%~11%左右。另一方面螺旋肋钢筋具有较高的锚固强度和刚度，且大滑移时仍具有较大承载力(锚固延性好)，有利于抗震。螺旋肋钢筋表面凸起多条螺旋肋，混凝土咬合齿宽厚且连续不间断，不易破碎、剪断。挤压面积大，相对肋面积大，劈裂力均匀无方向性，因此早期滑移小，而后期大变形时又具有较高的锚固延性，一般带肋钢筋横咬合齿分散易挤碎切断，大滑移下即失去锚固力，锚固延性差。