# CPCI6310 视频采集板

# 使
# 用
# 手
# 册

北京九航星达科技有限公司
Beijing Jiuhang Xingda Technology Co.,Ltd.

版本：1.1

# 目　录

# 第一章

## 1 简介

感谢您购买键石 CPCI6310 型视频采集板，该板卡是一款高品质 CPCI 视频板。键石 CPCI6310 具有高品质的视频采集性能，采用 CPCI 总线，兼容即插即用（PNP），支持一机多卡。

我们提供功能全面的二次开发包（以下简称 SDK）。您可以选择 VisualBasic，Visual C++，Delphi 等多种编程语言通过 SDK 进行开发，SDK 中包含 DLL 动态库（VC 使用），OCX 控件（VB，Delphi 使用）及其详细说明。您可以通过 SDK 控制图像的输入端口，图像亮度，对比度，色度，灰度等输入信号，动态截取图像，以 AVI 格式进行录像侦测图像是否有移动目标等等。

在安装和使用键石 CPCI6310 型视频采集板前，建议您先阅读本手册，以便了解如何安装和使用该产品。

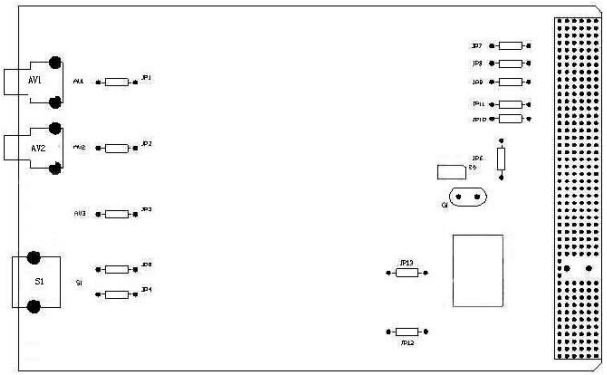
## 2 功能

### 2.1 功能与技术指标

- 3U CPCI 总线，兼容 Windows 即插即用（PNP），安装简易
- 显示画面流畅不间断，每秒可达 30 帧
- 显示分辨率可达 720*576(PAL)/720*480(NTSC)、24 位真彩
- 影像窗口大小随意调整，可全屏显示
- 动态采集影像以静态图像方式存盘，提供 BMP、JPG、TIF、TGA 等多种存盘格式
- 提供动态 AVI 影像捕获，捕获存盘影像大小随意选择
- 可连续动态捕获影像以静态图像方式自动存盘,提供照片浏览和电子相册等功能

- 符合 ITU H.324 视讯会议标准，完全兼容流行的视讯会议软件（如：NetMeeting，VDOPhone，CUSeeme 等）、画质清晰，性能稳定
- 兼容 5V/3.3V CPCI 总线规范
- 支持前/后出线选择
- **不支持热插拔**。

## 2.2 采集板示意图




AV1：第一路复合视频输入接口

AV2：第二路复合视频输入接口；

S1：S-Video 输入接口；

## 2.3 接口信号

J2 信号定义

| J2 针号 | 名称信号 | 含义 | 备注 |
|---|---|---|---|
| E9 | Y | Y 信号 | |
| E10 | C | C 信号 | |
| E11 | GND | 视频地 | |

| | | | |
|---|---|---|---|
| E16 | AV2 | 视频信号 | |
| E17 | GND | 视频地 | |
| E19 | AV1 | 视频信号 | |
| E20 | GND | 视频地 | |

## 2.4 硬件安装

该产品与天敏 2000 完全兼容，安装光盘附有天敏 2000 驱动程序，本板不再提供其它驱动程序。

第一步：将键石 CPCI6310 板装入 CPCI 机箱。

1. 关闭系统和所有外部设备，并断开所有电源。
   使用静电环或用手触摸一下计算机上裸露的金属板以便释放静电，卸掉 CPCI 机箱上的空挡板，露出插槽位置和 CPCI 导轨。
2. 找到一个空置的 CPCI 功能插槽，将键石 CPCI6310 板沿着导轨插入 CPCI 插槽中，并将 CPCI 挡板上的把手和螺钉固定好。
3. 将信号线接好，接通电源，打开计算机。

第二步：按照向导安装视频采集板驱动程序。

打开电源，启动计算机后，系统将检测到新硬件，紧接着出现找到新硬件向导，选择“从列表或指定位置安装(高级)”，点击“下一步”。

在对话框中，选中“在搜索中包括这个位置”，手工输入驱动程序所在目录或者是通过“浏览”按钮在安装光盘上定位驱动程序所在目录（安装光盘的 F:\Win9X2KDrv\Win2000_98\单卡或 F:\Win9X2KDrv\Win95\单卡 目录），然后单击“下一步”。

出现“完成找到新硬件向导”窗口，表示驱动程序安装完成，点击“完成”按钮，WDM Video Capture 安装结束。按此方法继续安装 WDM Crossbar 和 WDM Video Capture。重新启动计算机后就可以正常使用了。

# 第二章 软件开发说明 （for VC++ 6.0）

## 1 编译环境：

1、将SDK安装目录下的Inc 和 Lib 目录路径分别加入VC++工程设置的“C/C++ - Preprocressor - Additional include directories”和“Link - Input - Additional library path”中。

2、 将 Lib\DSStream.lib 文件加入工程设置的“Link - General - Object/library module”中。

3、 将 Inc\DSStream.h 文件加入工程，并 #include “DSStream.h”。

4、 将 Exe\DSStream.dll 文件拷入系统目录或其他系统能找到的目录中。

5、 开始编写代码。

## 2 函数说明：

### 2.1、初始化和反初始化函数：

（1）、HRESULT DSStream_Initialize()

初始化COM接口及一些参数。这是开始使用动态库的第一个函数，未初始化之前，任何其他函数的调用均无效。

（2）、void DSStream_Uninitialize()

释放COM接口，并断开已连接的所用设备。这是使用动态库的最后一个函数。这个函数应该在应用程序退出时、所有应用程序消息都处理完毕后调用，一般放在CWinApp::ExitInstance()或CWnd::OnNcDestroy()中。

（3）、HRESULT DSStream_ConnectDevice(int iCardID, BOOL bOverlay, HWND

hParentWnd = NULL)

连接视频采集板。只有连接后，才能对卡进行其他操作。

**iCardID：**

想要连接的卡号，以 0 为基数。计算机上的视频采集板的数量可由 DSStream_GetCardNumber 得到。卡号对应的设备名字可利用 DSStream_EnumVideoCaptureDev 取得。

**bOverlay：**

是否使用 Overlay 模式。TRUE-使用，FALSE-不使用。使用 Overlay 时，显示速度快、占用 CPU 资源极少，但是不能快照、录像、显示 Logo 等。另外，当有一块卡使用了 Overlay 或者系统中有其他的 Overlay 程序存在时，其他卡将不能使用这种模式。

**hParentWnd:**

指定视频图像的父窗口，图像将显示在 hParentWnd 窗口中。也可用 DSStream_SetOwnerWnd 来指定父窗口。

（4）、HRESULT DSStream_DisconnectDevice(int iCardID)

断开与视频采集板的连接。视频采集板的所有属性将被还原。

**iCardID：**

将要断开连接的卡号，以 0 为基数。

（5）、HRESULT DSStream_GetCardNumber(int * pCardNum)

得到计算机上的视频采集板的数量。

**pCardNum：**

指向一个 int 型变量。由它返回采集板的数目。

（6）、HRESULT DSStream_IsConnected(int iCardID, BOOL * bConnected)

查看某一片卡是否已经被连接。

**bConnected:**

指向一个 BOOL 型变量。由它返回是否已经被连接。

（7）、void DSStream_ConnectPin(int iCardID, int pin)

对显示脚或录像脚进行渲染。除非显示脚或录像脚已经被 DSStream_DisonnectPin 断开，否则不应该调用此函数。正在录像时，此函数无效。

（8）、void DSStream_DisonnectPin(int iCardID, int pin)

断开对显示脚或录像脚的渲染。

（9）、HRESULT DSStream_EnumVideoCaptureDev(char szDevName[MAX_DEVICE_NUM][MAX_DEVICE_NAME_LEN], int *pDevNum);

枚举计算机上的视频采集板，得到它们的名字和数量。

**szDevName:**

返回各个视频采集板的名字。

**pDevNum:**

返回视频采集板的数量。

## 2.2、设置视频窗口的属性：

（1）、HRESULT DSStream_SetOwnerWnd(int iCardID, HWND hParentWnd)

设置视频显示窗口的父窗口，视频图像将在这个窗口中显示。

**hParentWnd:**

父窗口的句柄。

（2）、HRESULT DSStream_GetOwnerWnd(int iCardID, HWND* phParentWnd)

得到当前视频显示窗口的父窗口的句柄。

**phPraentWnd:**

返回父窗口的句柄。

（3）、HRESULT DSStream_SetWindowPos(int iCardID, RECT rc)

设置视频图像在父窗口中的位置。如果所设位置的宽、高与视频图像实际的宽、高不等,视频图像将被缩放。视频图像的实际宽、高可通过调用 DSStream_GetVideoInfo 得到。

**rc：**

视频图像在父窗口中的位置。

（4）、HRESULT DSStream_GetWindowPos(int iCardID, RECT* prc)

得到视频窗口在父窗口中的位置。

**prc：**

返回视频窗口在父窗口中的位置。

（5）、HRESULT DSStream_SetMessageDrain(int iCardID, HWND hWnd)

设置后，视频显示窗口的键盘和鼠标消息将发往窗口 hWnd。通常，与父窗口为同一个窗口。

**hWnd：**

承受键盘、鼠标消息的窗口的句柄。

（6）、HRESULT DSStream_SetNotifyWindow(int iCardID, HWND hWnd, long lMsg)

设置后，每当有视频错误产生时，窗口 hWnd 都将收到消息 lMsg。

**hWnd：**

接受 lMsg 消息的窗口。

**lMsg:**

自定义的消息。消息的 LPARAM 和 WPARAM 参数的意义可参见枚举量 - 错误代号。

（7）、void DSStream_ManageNotifyMessage(int iCardID)

此函数要等设置 DSStream_SetNotifyWindow 后才有效。它分析当前产生的事件，负责发送消息 lMsg 到窗口 hWnd。每当有错误 Error_FirstNotifyMsg 产生时，必须调用此函数。

## 2.3、设置视频图像的各种属性：

（1）、HRESULT DSStream_DisplayPropertyDialogs(int iCardID, PropertyDialog id, HWND hParentWnd, LPCTSTR szCaption=NULL)

调用系统对话框，对视频的各种属性进行设置。

**id：**

要调用的对话框的代号

**hParentWnd：**

对话框的父窗口。

**szCaption：**

对话框的标题。如果设为 NULL，使用默认标题。

（2）、HRESULT DSStream_IsPropertyExist(int iCardID, PropertyDialog id, BOOL* bReturn)

查看某一个属性对话框是否被系统支持。

**id：**

属性对话框的代号。

**bReturn：**

返回属性是否被支持。

（3）、HRESULT DSStream_GetVideoInfo(int iCardID,VIDEOSTREAMINFO * pVSI,

int pin)

得到显示脚或录像脚的视频属性，包括视频子类型、帧率、图像的大小、颜色位率等。

**pVSI：**

返回视频属性。

（4）、HRESULT DSStream_SetVideoInfo(int iCardID, VIDEOSTREAMINFO vsi, int pin)

设置显示脚或录像脚的视频属性，包括视频子类型、帧率、图像的大小、颜色位率等。

**vsi：**

VIDEOSTREAMINFO 结构，包括各种属性。

（5）、HRESULT DSStream_GetVideoPropertyValue(int iCardID, VideoProperty id, VIDEOPROPERTYRANGE* pVPR)

得到视频图像的各个属性的信息，包括亮度、对比度、饱和度等。

**id：**

图像属性的代号。

**pVPR：**

返回图像属性的信息。

（6）、HRESULT DSStream_SetVideoPropertyValue(int iCardID, VideoProperty id, long value)

设置视频图像的各个属性的值，包括亮度、对比度、饱和度等。

**id：**

图像属性的代号。

**value：**

属性的值。它的取值范围和默认值可由 DSStream_GetVideoPropertyValue 得到。

（7）、HRESULT DSStream_WhatInPinRouteToOutPin(int iCardID, int idOutPin, long* pInPin)

得到当前的视频源。

**idOutPin：**

必须为 0。

**pInPin：**

返回视频源。对于 SDK-2000 板，范围是 0-2；对于四路板，范围是 0-4。

（8）、HRESULT DSStream_RouteInPinToOutPin(int iCardID, int idInPin, int idOutPin)

设置视频源。

**idInPin：**

视频源。对于 SDK-2000 板，范围是 0-2；对于四路板 ，范围是 0-4。

**idOutPin：**

必须为 0。

（9）、HRESULT DSStream_GetVideoStandard(int iCardID, VideoStandard* pVideoStandard, long* pAvailable)

得到当前的视频制式。

**pVideoStandard：**

返回当前使用的视频制式。

**pAvailable：**

返回采集板支持的所有视频制式。

（10）、HRESULT DSStream_SetVideoStandard(int iCardID, VideoStandard vs)

设置视频制式。

**vs：**

视频制式。

（11）、HRESULT DSStream_GetStreamStatus(int iCardID, int* pStatus)

得到当前视频显示的状态。

**pStatus：**

返回视频显示状态。

（12）、HRESULT DSStream_SetStreamStatus(int iCardID, int status)

设置视频显示的状态。

**status：**

显示状态。

（13）、HRESULT DSStream_IsVideoSignalLocked(int iCardID, BOOL * blsSignalLocked)

检测当前的视频输入端口上是否有信号输入

## 2.4、录像、快照、数据流函数：

（1）、void DSStream_SetCaptureFile(int iCardID, LPCTSTR szFileName)

设置录像时的输出文件。

**szFileName：**

.AVI 文件的名字。

（2）、void DSStream_StartCapture(int iCardID)

开始录像。

（3）、void DSStream_StopCapture(int iCardID)

停止录像。

（4）、void DSStream_IsCapturing(int iCardID, BOOL* blsCapturing)

查看某一片卡是否正在录像。

**bIsCapturing：**

返回结果。

（5）、void DSStream_IsCaptureAudio(int iCardID, BOOL* bCaptureAudio)

查看某一片卡是否正在使用声卡。

**bCaptureAudio：**

返回结果。

（6）、void DSStream_SetCaptureAudio(int iCardID, BOOL* bCaptureAudio)

将声卡分配给某一片卡使用，或取消分配。

**bCaptureAudio：**

指示是否分配声卡，并返回操作结果。

（7）、void DSStream_WhoIsCapturingAudio(int * pCardID)

查看是哪一片视频卡在使用声卡。

**pCardID：**

返回使用声卡的视频卡的卡号。若无人使用声卡，返回 AUDIO_DEVICE_NO_USED。若声卡不存在，返回 AUDIO_DEVICE_NO_EXIST。

（8）、void DSStream_SetFrameRateOnCapture(int iCardID, BOOL bUseFrameRate, double duFrameRate)

设置录像时是否使用自定义的帧率，以及帧率的大小（0-30）。

**bUseFrameRate：**

录像时是否使用自定义的帧率。

**duFrameRate：**

帧率的大小（0-30）。

（9）、void DSStream_IsUseFrameRate(int iCardID, BOOL* pYesOrNo)

查看某一片卡是否在录像时使用自定义帧率。

**pYseOrNo：**

返回结果。

（10）、HRESULT DSStream_SaveToJpgFile(int iCardID, LPCTSTR szFileName, int iQuality)

保存当前图像为 JPG 文件。视频流状态为 STOP 时，保存失败。

**szFileName：**

.JPG 文件的名字。

**iQuality：**

JPG 图片的压缩质量（0-100）。

（11）、HRESULT DSStream_SaveToBmpFile(int iCardID, LPCTSTR szFileName)

保存当前图像为 BMP 文件。视频流状态为 STOP 时，保存失败。

**szFileName：**

.BMP 文件的名字。

（12）、HRESULT DSStream_GetCurrentDib(int iCardID, BYTE* pBuffer, long* pSize)

将当前图像的 DIB 数据保存到内存中。视频流状态为 STOP 时，保存失败。

**pBuffer：**

指向预先分配的内存。可以为 NULL，此时 pSize 将得到保存图像需要的内存大小。

**pSize：**

若 pBuffer 不为 NULL，pSize 指示 pBuffer 的大小。为 pBuffer 为 NULL，pSize 得到保存图像需要的内存大小。

（13）、HRESULT DSStream_GetVideoStream(int iCardID, VideoStreamProc proc,

LPVOID pParam)

启动或停止数据流回调。如果启动数据流回调，每当有一帧新的图像产生时，函数 proc 都将被调用。

**proc：**

数据流回调函数的函数指针。若 proc 不为 NULL，开始数据流回调；若为 NULL，停止数据流回调。回调函数的定义为 extern "c" void proc (const BYTE* pDIBHead, const BYTE* pDIBits, LPVOID pParam)。pDIBHead 指向 DIB 头（即 BITMAPINFO*），pDIBits 指向点阵数据，pParam 是用户在 DSStream_GetVideoStream 中自定义的参数。

**pParam：**

用户自定义的参数。每次回调函数 proc 被调用时，pParam 都被传回给用户。

## 2.5、视频压缩函数：

（1）HRESULTDSStream_EnumVideoCompressor (VideoCompressorInfo * pInfo, int * piVidCompNum)

得到系统中安装的视频压缩算法的信息。

**pInfo：**

指向一片预先分配的 VideoCompressorInfo 结构。如果为 NULL，piVidCompNum 将得到系统中安装的视频压缩算法的数目。

**piVidCompNum：**

如果 pInfo 不为 NULL，piVidCompNum 表示 pInfo 中包含多少个 VideoCompressorInfo 结构。如果 pInfo 等于 NULL，piVidCompNum 返回系统中安装的视频压缩算法的数目。

（2）、HRESULT DSStream_ChooseVideoCompressor(DWORD dwCompHandle)

选择录像时使用的压缩算法。

**dwCompHandle：**

压缩算法的句柄，即 VideoCompressorInfo 结构的 dwHandle 成员。

（3）、HRESULTDSStream_GetCurrentVideoCompressor

(DWORD * pdwCompHandle)

得到当前正在使用的视频压缩算法的句柄。

**pdwCompHandle：**

返回压缩算法的句柄。

（4）、HRESULT DSStream_GetVideoCompressorQuality(int * piQuality)

得到视频压缩时的质量。

**piQuality：**

返回视频压缩时的质量设置。

（5）、HRESULT DSStream_SetVideoCompressorQuality(int iQuality)

设置视频压缩时的质量。并非对所有压缩算法都有效。

**iQuality：**

设置视频压缩时的质量。

## 2.6、Logo、日期、时间的显示和设置：

（1）、HRESULT DSStream_SetLogoFile(int iCardID, LPCTSTR szFilename)

设置用作 Logo 的 BMP 文件名。此 BMP 文件必须为 24 位格式，最左下角的一点为透明色。

（2）、HRESULT DSStream_ShowLogo(int iCardID, BOOL bShow, int x, int y)

是否显示 Logo，及 Logo 的位置。

（3）、HRESULT DSStream_ShowTime(int iCardID, BOOL bShow, int x, int y)

是否显示日期，及日期的位置。

（4）、HRESULT DSStream_ShowDate(int iCardID, BOOL bShow, int x, int y)

是否显示时间，及时间的位置。

（5）、HRESULT DSStream_ShowLogoTimeOnUserStream(int iCardID, BOOL bShow)

是否在由 DSStream_GetVideoStream 得到的数据流中显示 Logo、日期、时间。

**2.7、四路卡专用的函数：**

（1）、HRESULT DSStream_GetVideoPropertyValue_Ex(int iCardID, long lWay, VideoProperty id, long * pValue)

四路同屏时，得到其中某一路图像的亮度、对比度等。参见 DSStream_GetVideoPropertyValue。

（2）、HRESULT DSStream_SetVideoPropertyValue_Ex(int iCardID, long lWay, VideoProperty id, long lValue)

四路同屏时，设置其中某一路图像的亮度、对比度等。参见 DSStream_SetVideoPropertyValue。

（3）、HRESULT DSStream_ChooseCamera_Ex(int iCardID, BYTE idWay, BYTE idCamera)

四路卡的矩阵调节。在 32 个摄像头中选择一个，连接到 4 路中的一路。

**idWay：**

连到 4 路中第几路，范围 0-3。

**idCamera：**

选择第几个摄像头，范围 0-31。大于或等于 32 时，第 idWay 路图像被取消，变为蓝屏。

## 3 使用的结构、枚举量和常量：

### 3.1、结构：

（1）、视频流的信息

```
typedef struct
{
VideoSubType subtype; //视频子类型
RECT rcSource;
RECT rcTarget;
DWORD dwBitRate;
DWORD dwBitErrorRate;
LONGLONG AvgTimePerFrame; //帧率 = 10000000. / AvgTimePerFrame
BITMAPINFOHEADER bmiHeader; //宽、高、颜色位率等
} VIDEOSTREAMINFO;
```

（2）、图像属性的信息

```
typedef struct
{
long lValue; //当前值
long lMin; //最小值
long lMax; //最大值
long lStepDelta; //最小步进值
long lDefault; //默认值
long lCapsFlags; //风格
} VIDEOPROPERTYRANGE;
```

（3）、视频压缩算法的信息

```
typedef struct VIDEOCOMPRESSORINFO
{
char szName[256];
DWORD dwHandle;
} VideoCompressorInfo;
```

**3.2、枚举量：**

（1）、视频子类型

```
typedef enum
{
VideoSubType_None = -1,
VideoSubType_RGB555 = 0,
VideoSubType_RGB24,
VideoSubType_YUY2,
VideoSubType_YVU9,
VideoSubType_YV12,
} VideoSubType;
```

（2）、属性对话框的代号

```
typedef enum
{
PropertyDlg_VideoCaptureFilter = 0,
PropertyDlg_VideoCapturePin, // *
PropertyDlg_VideoPreviewPin, // *
PropertyDlg_VideoCrossbar,
PropertyDlg_AudioCaptureFilter,
PropertyDlg_AudioCapturePin,
```

```
PropertyDlg_AudioCrossbar,
PropertyDlg_TVAudioFilter,
PropertyDlg_TVTuner,
PropertyDlg_VfwCaptureFormat, // *
PropertyDlg_VfwCaptureSource,
PropertyDlg_VfwCaptureDisplay,
} PropertyDialog;
```

（3）、视频属性的代号

```
typedef enum
{
VideoProperty_Brightness = 0, //亮度
VideoProperty_Contrast, //对比度
VideoProperty_Hue, //色度
VideoProperty_Saturation, //饱和度
VideoProperty_Gamma, //Gamma 校验
VideoProperty_ColorEnable,
VideoProperty_WhiteBalance, //白平衡
VideoProperty_BacklightCompensation,
} VideoProperty;
```

（4）、视频制式

```
    typedef enum
{
VideoStandard_None = 0x00000000,
VideoStandard_NTSC_M = 0x00000001,
VideoStandard_NTSC_M_J = 0x00000002,
VideoStandard_NTSC_433 = 0x00000004,
```

```
VideoStandard_PAL_B = 0x00000010,
VideoStandard_PAL_D = 0x00000020,
VideoStandard_PAL_H = 0x00000080,
VideoStandard_PAL_I = 0x00000100,
VideoStandard_PAL_M = 0x00000200,
VideoStandard_PAL_N = 0x00000400,
VideoStandard_PAL_60 = 0x00000800,
VideoStandard_SECAM_B = 0x00001000,
VideoStandard_SECAM_D = 0x00002000,
VideoStandard_SECAM_G = 0x00004000,
VideoStandard_SECAM_H = 0x00008000,
VideoStandard_SECAM_K = 0x00010000,
VideoStandard_SECAM_K1 = 0x00020000,
VideoStandard_SECAM_L = 0x00040000,
VideoStandard_SECAM_L1 = 0x00080000,
} VideoStandard;
```

（5）、视频错误的代号

```
DSStreamError = (LPARAM & 0x0000FFFFL)，Card ID = (LPARAM>>16) & 0x0000FFFFL。
typedef enum
{
Error_FirstNotifyMsg = 0x1234, //收到这个错误，必须调用
DSStream_ManageNotifyMessage
Error_DiskFull, //磁盘已满：wparam=0
Error_VideoSizeChange, //视频流尺寸改变：height=HIWORD(wparam),
width=LOWORD(wparam)
Error_StreamStatusChange, //视频流状态改变：wparam=0
Error_CaptureError, //录像异常：wparam=HRESULT: error value
} DSStreamError;
```

### 3.3、常量：

（1）、最多支持 32 片卡，卡的名字最长为 80 个字符。

```
#define MAX_DEVICE_NUM 32
#define MAX_DEVICE_NAME_LEN 80
```

（2）、视频的显示状态

```
#define RUN 0
#define PAUSE 1
#define STOP 2
```

（3）、设备管脚的代号

```
 #define CAPTURE 1 //录像脚
#define PREVIEW 2 //显示脚
```