大恒图像系列板卡

图像采集卡软件开发说明书

2009 年 3 月版

本手册中所提及的其它软硬件产品的商标与名称，都属于相应公司所有。

网站：http://www.daheng-image.com

营销中心：sales@daheng-image.com

010-82828878 转 8028

支持信箱：support@daheng-image.com

010-82828878 转 8006

前　言

首先感谢您选用大恒图像产品，图像采集卡应用接口库是我公司提供的应用程序接口函数。它基本上概括了图像卡应用程序编程中涉及的各个方面，通过应用接口库，用户可以很好地了解图像卡的运行机制，并在此基础上进行应用程序的开发。

图像采集卡应用接口库按照模块分为应用功能模块和扩充功能模块。其中应用功能模块包括图像卡的控制、采集图像到屏幕、采集图像到内存、错误处理等功能，扩充功能模块包括采集图像到屏幕控制、采集图像到内存控制、数据传递等功能。

本手册按功能对接口库中的函数进行了划分，对于每一个功能部分的函数都具体给出了它们的函数原型、参数定义、说明、示例等信息。

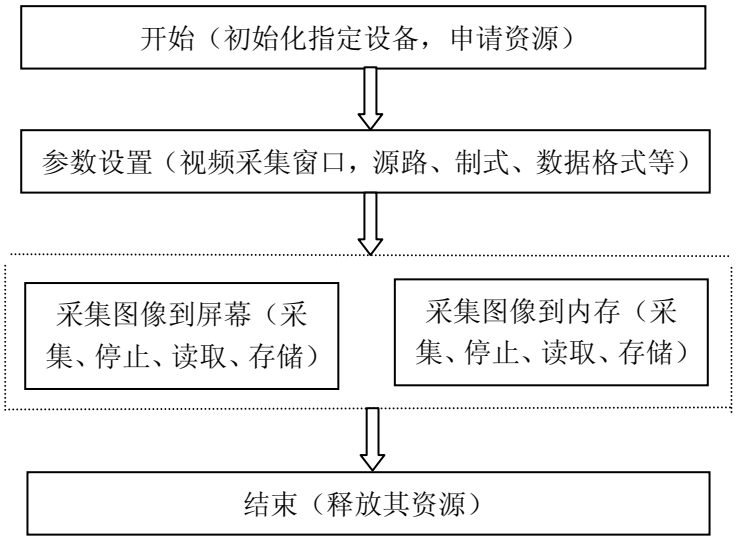
本手册为用户进行图像卡的二次开发提供了详细的说明，如果用户在使用过程中发现错误和纰漏，请与我公司的销售或技术服务部门联系，以便我们及时改进。

目　录

1 概述

在 Microsoft 的 32 位 Windows 操作系统中，图像采集卡应用接口库直接操作图像卡并提供给用户一个简单明确的应用接口。用户在编制自己的应用程序时，可以直接调用这些库函数来实现指定的功能。

图像采集卡工作流程：



一般情况下，图像卡的开始操作和初始化参数的设置，最好在用户应用程序的初始化中完成，图像卡的结束操作应在应用程序退出前执行。图像卡采集图像数据，不占用计算机 CPU 的时间，支持图像的实时处理。采集图像到屏幕和采集图像到内存的操作，不能同时进行，即同一时刻图像卡采集图像到屏幕，或者到内存。

1.1 功能

应用接口库划分为应用功能模块和扩充功能模块。

1.2 文件

1.2.1 应用功能模块

包含文件 CGVideo.h

动态链接库 CGVideo.dll

静态链接库 CGVideo.lib

1.2.2 扩充功能模块

包含文件 CGVidEx.h

动态链接库 CGVidEx.dll

静态链接库 CGVidEx.lib

1.3 开发工具

应用接口库支持 32 位编程开发工具 Microsoft Visual C/C++、Microsoft Visual Basic、Borland C/C++、Delphi、C++ Builder、Power Builder 等。

使用 C/C++编程工具，用户应在程序中调用相关的包含文件（.h），并将静态链接库（.lib）文件加入到工程文件中，供编译程序在链接（Link）时使用。需要说明的是，MS-Visual C/C++和 C++ Builder 定义了各自的静态链接库的文件格式，它们之间是不通用的。

使用 MS-Visual Basic、Delphi、Power Builder 等编程工具调用应用接口库时，应按照调用动态链接库的方法，在程序中重新声明函数原型，这时要注意正确定义参数的数据类型。

1.4 应用程序的发布

发布图像采集卡的应用程序，首先，安装图像卡的设备驱动程序（安装步骤可参见“安装图像卡设备驱动程序”说明），并确认安装成功；然后，将要发布的应用程序安装到相应的工作目录。这样，应用程序就可以运行了。

如果需要将驱动程序与二次开发的应用程序安装盘一并发布，请将随卡的驱动程序光盘或软盘中，相应操作系统目录下的驱动程序拷贝到准备发布的安装盘上即可。

2 说明

2.1 应用功能

应用功能模块包括图像采集卡的控制、采集图像到屏幕、采集图像到内存、错误处理、其它功能等功能。函数的原型声明在包含文件 CGVideo.h，动态链接库 CGVideo.dll，静态链接库 CGVideo.lib。

2.1.1 定义

模块中相关常量，数据结构，宏的说明。

2.1.1.1 常量

1. typedef HANDLE HCG;

HCG 标识图像卡设备句柄。

图像卡的控制，参数设置，图像采集等，都需要使用图像卡设备句柄。

2. typedef enum tagCGTYPE {

```
CG300TYPE = 0,
CG400TYPE = 1,
CG410TYPE = 2,
VT121TYPE = 3
```

} CGTYPE;

CGTYPE 标识当前图像卡的类型。

不同类型的图像卡有一些不同的硬件特性或参数，用户需要根据

图像卡类型，完成相关的操作。

3. typedef enum tagVIDEO_SOURCE_TYPE {

```
    COMPOSITE_VIDEO  = 0,
    S_VIDEO          = 1,
    COMPONENT_VIDEO  = 2
} VIDEO_SOURCE_TYPE;
```

VIDEO_SOURCE_TYPE 标识视频信号源路的类型。

COMPOSITE_VIDEO

复合视频信号；

S_VIDEO

S端子视频信号；

COMPONENT_VIDEO

YPbPr分量视频信号。

4. typedef enum tagVIDEO_ADJUST_PARAM {

```
    BRIGHTNESS  = 1,
    CONTRAST    = 2,
    HUE         = 3,
    SATURATION  = 4
} VIDEO_ADJUST_PARAM;
```

VIDEO_ADJUST_PARAM 标识视频信号的显示调节参数。

BRIGHTNESS

亮度；

CONTRAST

对比度；

HUE

色调；

SATURATION

色饱和度。

5. typedef enum tagVIDEO_FORMAT {

```
YUV422       = 0,
RGB888       = 1,
RGB565       = 2,
RGB555       = 3,
RGB8888      = 4,
ALL8BIT      = 5,
LIMITED8BIT  = 6
```

} VIDEO_FORMAT;

VIDEO_FORMAT 标识图像数据格式。

YUV422

YUV422 方式，16 位/像素；

RGB565

RGB565 方式，16 位/像素；

RGB555

RGB555 方式，15 位/像素；

RGB888

RGB888 方式，24 位/像素；

RGB8888

RGB8888 方式，32 位/像素；

ALL8BIT

黑白方式，8 位/像素，取值：CG300、CG400 为 0-255，CG410 为 1-254；

LIMITED8BIT

黑白方式，8位/像素，取值：CG300为16-253，CG400、CG410为16-235。

6. typedef enum tagVIDEO_STANDARD {

PAL = 0,

NTSC = 1

} VIDEO_STANDARD;

VIDEO_STANDARD 标识视频信号制式。

PAL

PAL制；

NTSC

NTSC制。

7. typedef enum tagVIDEO_SCAN {

FRAME = 0,

FIELD = 1,

FIELD1 = 2

} VIDEO_SCAN;

VIDEO_SCAN 标识视频信号扫描方式。

FRAME

帧方式，隔行扫描，一帧图像的全部数据；

FIELD

场方式，逐行扫描，一帧图像的一场数据。

FIELD1

场方式1，逐行扫描，一帧图像的另一场数据。

8. typedef enum tagLUMA_PARAM {

LUMA_CHROMA_TRAP = 0

} LUMA_PARAM;

LUMA_PARAM 标识视频信号的亮度通道的控制。

LUMA_CHROMA_TRAP

亮度通道的色度陷波。

9. typedef enum tagGAMMA_CORR {

NO_GAMMACORR = 0,

Y_GAMMACORR = 1,

UV_GAMMACORR = 2,

YUV_GAMMACORR = 3

} GAMMA_CORR;

GAMMA_CORR 标识视频信号Gammma校正的控制。

NO_GAMMACORR

无 Gamma 校正;

Y_GAMMACORR

对亮度（Y）数据进行 Gamma 校正;

UV_GAMMACORR

对色度（U、V）数据进行 Gamma 校正;

YUV_GAMMACORR

对亮度和色度（Y、U、V）数据进行 Gamma 校正。

10. typedef enum tagMIRROR_DIRECTION {

HOR_DIR = 0,

VERT_DIR = 1

} MIRROR_DIRECTION;

MIRROR_DIRECTION 标识图像采集显示的镜像功能。

HOR_DIR

水平方向；

VERT_DIR

垂直方向。

11. typedef enum tagEXT_VIDEO_OUTPUT {

```
    EXT_VOUT_MODE           = 0,
    EXT_VOUT_SOURCE         = 1,
    EXT_VOUT_CYCLE_LENGTH   = 2,
    EXT_VOUT_CYCLE_SOURCE   = 3
} EXT_VIDEO_OUTPUT;
```

EXT_VIDEO_OUTPUT 标识视频信号的外接输出功能。

EXT_VOUT_MODE

视频输出方式；

EXT_VOUT_SOURCE

视频输出源路；

EXT_VOUT_CYCLE_LENGTH

视频输出循环显示间隔；

EXT_VOUT_CYCLE_SOURCE

视频输出循环显示源路。

12. typedef enum tagCRY_OSC {

```
    CRY_OSC_35M = 0,
```

CRY_OSC_28M = 1

} CRY_OSC;

CRY_OSC 标识图像卡配置的晶振类型。

CRY_OSC_35M

35M晶振;

CRY_OSC_28M

28M晶振。

13. typedef enum tagBOARD_INFO {

BOARD_TYPE = 0,

BOARD_SERIALNUM = 1,

BOARD_SUB_DEVICE = 3

} BOARD_INFO;

BOARD_INFO 标识图像卡的标志信息。

BOARD_TYPE

板型;

BOARD_SERIALNUM

序列号;

BOARD_SUB_DEVICE

设备子号

2.1.1.2 结构

1. typedef struct tagVIDEO_SOURCE {

    VIDEO_SOURCE_TYPE type;

    int nIndex;

} VIDEO_SOURCE;

VIDEO_SOURCE 结构包含视频信号的源路参数。

成员 type

说明视频信号源路的类型，包括复合视频、S端子、分量视频。

nIndex

说明视频信号源路的序号，从0开始。

例如，CG300图像卡有四个复合视频信号输入，则类型type为COMPOSITE_VIDEO，序号nIndex分别为0、1、2、3。

2. typedef struct tagCHECK_PARAM {

    BYTE byParam1;

    BYTE byParam2;

    BYTE byParam3;

    BYTE byParam4;

} CHECK_PARAM;

CHECK_PARAM 结构包含给定的图像卡查验参数。

成员 byParam1

参数 1。

byParam2

参数 2。

byParam3

参数 3。

byParam4

参数 4。

2.1.1.3 宏

1. CG_SUCCESS

BOOL CG_SUCCESS（status）

CG_SUCCESS 宏检查函数执行的返回状态。

参　数：CGSTATUS status;

函数执行的返回状态。

返回值：标识函数执行结果，成功为 TRUE；失败为 FALSE。

注　释：CG_SUCCESS 在 CGDef.h 中的定义如下，

#define CG_SUCCESS(status)（（status）== CG_OK ）

2. CG_VERIFY

CG_VERIFY（status）

CG_VERIFY 宏校验函数执行是否成功。CG_VERIFY 宏只在调试（DEBUG）版的执行程序中运行，函数执行成功，

不显示信息；函数执行不成功，则弹出对话框，显示错误信息。在发布（RELEASE）版的执行程序中不运行，也不产生任何代码。

参　数：CGSTATUS status;

函数执行的返回状态。

注　释：CG_VERIFY 在 CGDef.h 中的定义如下，

```
#ifdef _DEBUG
   #define CG_VERIFY（status） \
      if（（status） != CG_OK）{ \
         ::MessageBox（NULL， \
         CGGetErrorString（status）， \
         "Error"，MB_ICONWARNING | MB_OK）； \
      }
#else
   #define CG_VERIFY（status） （status）
#endif
```

2.1.2 函数说明

2.1.2.1 图像卡的控制

图像卡的开始（初始化）、结束，设置视频采集窗口、制式、源路、数据格式等。

1. 函　数：BeginCGCard

原　型：CGSTATUS __stdcall BeginCGCard（int nDevice，HCG *pHandle）

参　数：int nDevice；

图像卡序号，由 1 开始。

HCG *pHandle；

指向图像卡设备句柄。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：开始指定图像卡操作，初始化图像卡，获得其设备句柄，分配相应的资源。

输入参数 nDevice 是图像卡的逻辑序号，按照大恒图像系列板卡序列，如 CG300、CG400…CGxxx 等，由低到高排列确定。例如:用户系统中有 3 块 CG300 和 2 块 CG400，则 CG300 卡的序号分别为 1、2、3，而 CG400 卡的序号为 4、5。如果 nDevice 的值超过当前系统实际的图像卡数,则返回 CG_NO_CARD_FOUND。调用函数 CGGetCardType 可以确定当前图像卡类型。

图像卡使用完毕，调用函数 EndCGCard 结束。

范　例：如果系统中安装了两块或两块以上的图像卡，则可以按下面调用得到每块图像卡的设备句柄。

```
BeginCGCard（1，&hCard1）；  //打开图像卡 1
BeginCGCard（2，&hCard2）；  //打开图像卡 2
……        ……      //类推…
```

参　看：EndCGCard，CGGetCardType。

2．函　数：EndCGCard

原　型：CGSTATUS __stdcall EndCGCard（HCG hcg）

参　数：HCG hcg；

图像卡句柄。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：结束指定图像卡操作，释放其设备句柄和资源。

调用函数前，必须停止图像卡工作，如果图像卡正在采集图像到屏幕或内存时调用函数，则返回 CG_IN_WORK。

范　例：

```
EndCGCard（hCard1）；  //关闭图像卡 1
EndCGCard（hCard2）；  //关闭图像卡 2
```

参　看：BeginCGCard。

3．函　数：CGSetInputWindow

原　型：CGSTATUS __stdcall CGSetInputWindow（HCG hcg，int nStartX，int nStartY，int nWidth，int nHeight）

参　数：HCG hcg；

图像卡句柄。

int nStartX;

输入窗口左上角 x 坐标，数值应为 4 的倍数。

int nStartY;

输入窗口左上角 y 坐标，数值应为 2 的倍数。

int nWidth;

输入窗口宽度，数值应为 4 的倍数。

int nHeight;

输入窗口高度，数值应为 2 的倍数。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：设置视频采集输入（Input）窗口，输入窗口是指视频数据输入图像卡时的起始位置和大小。

PAL 制信号，视频输入窗口最大是 768×576，NTSC 制信号，视频输入窗口最大是 640×480。

与输入窗口相关的是图像显示输出窗口，两个窗口决定了图像的显示方式，参见函数 CGSetOutputWindow。

在采集图像到内存的过程中，不能设置输入窗口，否则返回 CG_IN_WORK。

范　例：CGSetInputWindow（hCard1，0，0，512，512）;

参　看：CGSetOutputWindow。

4. 函　数：CGSetOutputWindow

原　型：CGSTATUS __stdcall CGSetOutputWindow(HCG hcg, int nStartX, int nStartY, int nWidth, int nHeight)

参　数：HCG hcg;

图像卡句柄。

int nStartX;

输出窗口左上角 x 坐标，数值应为 4 的倍数。

int nStartY;

输出窗口左上角 y 坐标，数值应为 2 的倍数。

int nWidth;

输出窗口宽度，数值应为 4 的倍数。

int nHeight;

输出窗口高度，数值应为 2 的倍数。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：设置图像显示输出（Output）窗口，输出窗口是指图像数据从图像卡输出时的起始位置和大小。

输出窗口大小只能等于或小于输入窗口的大小，两个窗口的大小决定了图像显示方式。当两个窗口大小相同时，是裁剪方式，这时输出窗口中的图像为全部输入图像的一个局部或全部；当输出窗口小于输入窗口时，是按比例压缩方式，这时输出窗口中的图像为一个按比例缩小的局部或全部输入图像。

在采集图像到屏幕时，输入参数 nStartX，nStartY 是输出窗口在屏幕中的起始坐标（以屏幕左上角为原点）；采集图像到内存时，参数忽略，设置为 0，0 即可。

在采集图像到内存的过程中，不能设置输出窗口，否则返回 CG_IN_WORK。

范　例：CGSetOutputWindow（hCard1，0，0，512，512）；

参　看：CGSetInputWindow。

5．函　数：CGSetVideoSource

原　型：CGSTATUS __stdcall CGSetVideoSource（HCG hcg，VIDEO_SOURCE source）

参　数：HCG hcg；

图像卡句柄。

VIDEO_SOURCE source；

视频信号的源路参数，参见 VIDEO_SOURCE 结构说明。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：设置图像卡的视频信号源路。图像卡 CG300 视频源路有四路复合视频（0-3）、一路 S 端子（0），CG400 有六路复合视频（0-5）、三路 S 端子（0-2）、一路分量视频（0），CG410 有六路复合视频（0-5）、二路 S 端子（0-1）、二路 YPbPr 分量视频（0-1）。当使用 QP300 卡时，QP300 的每个 CG300 单元模块的源路只有复合视频 1 和 2 可以

选择，如果采用 J1-J4 进行输入，则每个模块的源路设定为复合视频输入 2；如果采用 JP2-JP5 输入，则每个模块的源路设定为复合视频输入 1。VT121 每个单元模块的视频源路有两路复合视频（0-1）、一路 S 端子（0）。如果参数 source 为图像卡不支持的视频源路，则返回 CG_PARAMETER_INVALID。

范　例：//设置视频源路为复合视频 3

```
source.type = COMPOSITE_VIDEO;
source.nIndex = 2;
CGSetSource (hCard1, source);
```

6. 函　数：CGAdjustVideo

原　型：CGSTATUS __stdcall CGAdjustVideo (HCG hcg,
VIDEO_ADJUST_PARAM param, BYTE byVal)

参　数：HCG hcg;

图像卡句柄。

AD_PARAMTER param;

调节参数的类型。

BYTE byVal;

参数值，取值范围为 0-255。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：设置图像卡的图像显示调节参数，包括亮度、对比度、

色调、色饱和度。参数的初始值都为：128。

范　例：CGAdjustVideo（hCard1，BRIGHTNESS，120）；

7．函　数：CGSetVideoFormat

原　型：CGSTATUS __stdcall CGSetVideoFormat（HCG hcg，VIDEO_FORMAT format）

参　数：HCG hcg；

图像卡句柄。

VIDEO_FORMAT format；

图像数据格式。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：设置图像卡输出的图像数据格式，数据格式包括 YUV422、彩色（15 位以上）、黑白（8 位）方式。

采集图像到计算机的显示卡（屏幕）时，显示卡的显示“颜色”数应与图像数据格式一致，256 色（8 位）对应 Limited8Bit 或 All8Bit 格式；32768 色（15 位）对应 RGB555 格式；65536 色（16 位）对应 RGB565 格式；16mil 色（24 位）对应 RGB888 格式；真彩色（32 位）对应 RGB8888 格式。

在采集图像到屏幕或内存的过程中，不能设置图像数据格式，否则返回 CG_IN_WORK。

范　例：CGSetVideoFormat（hCard1，RGB888）；

8. 函　数：CGSetVideoStandard

原　型：CGSTATUS __stdcall CGSetVideoStandard（HCG hcg，VIDEO_STANDARD mode）

参　数：HCG hcg；

图像卡句柄。

VIDEO_STANDARD mode；

视频信号制式。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：设置图像卡视频信号制式，包括 PAL 制和 NTSC 制。

在设置制式时，图像卡视频输入、输出窗口的大小和视频信号的行正程起始点、场正程起始行，也要做相应的调整。图像卡初始为 PAL 制。

CG300，VT121 图像卡在采集图像到屏幕或内存的过程中，不能设置视频制式，否则返回 CG_IN_WORK。

范　例：CGSetVideoStandard（hCard1，PAL）；

参　看：CGSetInputWindow，CGSetOutputWindow，CGSetDelay。

9. 函　数：CGSetDelay

原　型：CGSTATUS __stdcall CGSetDelay(HCG hcg,int nXDelay，int nYDelay）

参　数：HCG hcg；

图像卡句柄。

int nXDelay；

延迟的像素数。

int nYDelay；

延迟的行数，数值应为 2 的倍数。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：设置视频信号的行正程起始点、场正程起始行。行正程起始点是以像素为单位，从行同步有效时开始计数；场正程起始行是以行为单位，从场同步有效时开始计数。图像卡的初始值是 0，0。

范　例：CGSetDelay（hCard1，0，10）；

10．函　数：CGSetPLLFrequency

原　型：CGSTATUS __stdcall CGSetPLLFrequency（HCG hcg，BYTE byValue）

参　数：HCG hcg；

图像卡句柄。

BYTE byValue；

图像扭曲的调整参数，取值：0-255。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：当视频信号是 NTSC 制式时，对图像的扭曲进行微调。如果图像卡工作在 NTSC 制式，采集的图像扭曲，可用本函数调整。缺省设置是 245。

卡　型：CG300，VT121。

11．函　数：CGSetScanMode

原　型：CGSTATUS __stdcall CGSetScanMode（HCG hcg，
VIDEO_SCAN mode）

参　数：HCG hcg；
图像卡句柄。
VIDEO_SCAN mode；
视频信号扫描方式。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：视频信号的一帧图像由两场数据组成，通过设置视频信号的扫描方式，确定图像的帧方式、场方式或场方式 1。扫描方式为帧方式时，一帧图像的高度等于视频输入/输出窗口的高度；扫描方式为场方式或场方式 1 时，一场图像的高度为视频输入/输出窗口高度的一半。

范　例：CGSetScanMode（hCard1，FRAME）；

参　看：CGSetInputWindow，CGSetOutputWindow。

12．函　数：CGEnableMask

原　型：CGSTATUS __stdcall CGEnableMask（HCG hcg，BOOL bEnable）

参　数：HCG hcg；

图像卡句柄。

BOOL bEnable；

TRUE 允许，FALSE 禁止。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：图像卡的屏蔽（数据迭加）功能控制。屏蔽功能是指图像输出窗口内的任一坐标点的象素数据是否允许被图像数据刷新。函数控制屏蔽功能允许，屏蔽掩码的设置参见函数 CGSetPixelMask。

范　例：CGEnableMask（hCard1，TRUE)；

参　看：CGSetPixelMask。

13. 函　数：CGSetPixelMask

原　型：CGSTATUS __stdcall CGSetPixelMask（HCG hcg，int x，int y，BOOL bEnable）

参　数：HCG hcg；

图像卡句柄。

int x；

象素点的 x 坐标。

int y；

象素点的 y 坐标。

BOOL bEnable；

TRUE 屏蔽，FALSE 显示。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：设置指定象素点的屏蔽。对图像输出窗口内任一象素点(输出窗口左上角为原点)进行屏蔽，只有屏蔽功能允许时，函数才有效。

范　例：CGSetPixelMask (hCard1, 10, 10, TRUE);

参　看：CGEnableMask。

14. 函　数：CGLumaControl

原　型：CGSTATUS __stdcall CGLumaControl（HCG hcg，LUMA_PARAM param，BYTE byValue）

参　数：HCG hcg;

图像卡句柄。

LUMA_PARAM param;

亮度通道的控制参数。

BYTE byValue;

TRUE 允许，FALSE 禁止。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：设置视频信号的亮度通道控制参数，亮度通道的色度陷波允许。当视频源为彩色复合视频信号时，例如：采集彩色图像，允许色度陷波；当视频源为 S-Video 或者分量视频信号时，例如：采集黑白图像，禁止色

度陷波。参数的初始值为允许色度陷波。

范　例：CGLumaControl（hCard1，LUMA_CHROMA_TRAP，TRUE）；

卡　型：CG400。

15. 函　数：CGGammaCorrControl

原　型：CGSTATUS __stdcall CGGammaCorrControl（HCG hcg，GAMMA_CORR param）；

参　数：HCG hcg；

图像卡句柄。

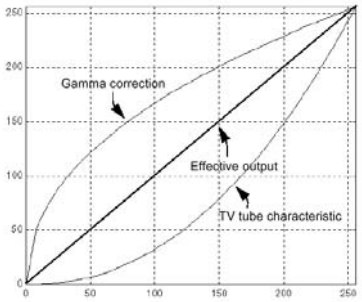
GAMMA_CORR param；

Gammma 校正的控制。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：设置图像卡视频信号 Gamma 校正的控制方式。

CG400 图像卡具有一个可编程的 Gamma 校正表，允许用户对视频的亮度信号和色度信号做多种形式的校正。当应用 Gamma 校正时，可产生一个下图中上方的曲线，它可以补偿图像卡 A/D 采样或者显示系统所产生的非线性失真，从而使图像输出总的传输特性是线性的。



初始值为无 Gamma 校正。

范　例：CGGammaCorrControl（hCard1，Y_GAMMACORR）；

参　看：CGSetGammaCorrCoef。

卡　型：CG400。

16. 函　数：CGSetGammaCorrCoef

原　型：CGSTATUS __stdcall CGSetGammaCorrCoef（HCG hcg，BYTE *pBuffer）；

参　数：HCG hcg；

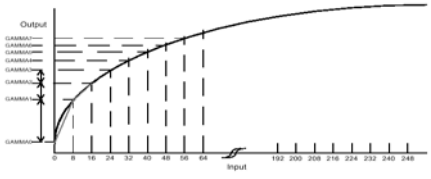
图像卡句柄。

BYTE *pBuffer；

指向存放 32 个校正段的段起始 Gamma 值的数据区。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：设置图像卡 Gamma 校正表的参数，这些参数是输入的原始数据，经过 Gamma 校正后得到的结果值。如图：



水平坐标为输入图像的 256 个亮度值或色度值，垂直坐标为经过 Gamma 校正后输出图像的亮度值或色度值

（Gamma 值），将输入值平均分成 32 段，则每段的起始值（$VALUE_n$= 0，8，16，24 等）对应一个输出值（Gamma 值）的段起始值（$GAMMA_n$）。

设在第 N 段中有一输入值 Q，则校正后的输出值 Q_{gamma} = INT（$GAMMA_n$ + M × K），其中 M = Q - $VALUE_n$（0≤M≤7），K =（$GAMMA_{n+1}$ - $GAMMA_n$）/8。

当 K ＞ 1 时，对该段进行线性插值；

当 K = 1 时，该段的线性关系不变；

当 K ＜ 1 时，对该段进行线性抽值。

根据校正结果，设置 32 段 Gamma 值的段起始值 GAMMA0、GAMMA1、GAMMA2、…、GAMMA31，并且 $GAMMA_{n+1}$ - $GAMMA_n$ ≤63；GAMMA31≤255。例如：

YUV 数据 Gamma 校正值

偏移量	Gamma	偏移量	Gamma
0	0	16	187
1	53	17	192
2	73	18	197
3	87	19	202
4	99	20	207
5	110	21	211
6	120	22	216
7	128	23	220
8	136	24	225
9	144	25	229

10	151	26	233
11	158	27	237
12	164	28	241
13	170	29	245
14	176	30	249
15	181	31	252

参　看：CGGammaCorrControl。

卡　型：CG400

17. 函　数：CGEnableVideoMirror

原　型：CGSTATUS __stdcall CGEnableVideoMirror（HCG hcg，MIRROR_DIRECTION dir，BOOL bEnable）

参　数：HCG hcg；

图像卡句柄。

MIRROR_DIRECTION dir；

图像采集的镜像方向。

BOOL bEnable；

TRUE 允许，FALSE 禁止。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：设置图像卡图像采集的沿水平、垂直方向镜像的功能。

卡　型：CG400，CG410。

18. 函　数：CGSetVideoExtOutput

原　型：CGSTATUS __stdcall CGSetVideoExtOutput（HCG hcg，EXT_VIDEO_OUTPUT mode，int nValue）

参　数：HCG hcg；

图像卡句柄。

EXT_VIDEO_OUTPUT mode；

视频信号的外接输出功能。

int nValue；

参数值，EXT_VOUT_MODE 功能：

0，输出显示的源路与图像卡的视频源路相同；

1，独立设置输出显示的源路；

2，循环显示各源路。

EXT_VOUT_SOURCE 功能（输出方式为独立设置输出显示的源路时有效）：

0，对应复合视频输入 1；

1，对应复合视频输入 2；

2，对应复合视频输入 3；

3，无效；

4，对应复合视频输入 4；

5，对应复合视频输入 5；

6，对应复合视频输入 6；

7，无效。

EXT_VOUT_CYCLE_LENGTH 功能，当输出方式为循环显示时，设置显示间隔时间，取值范围：0-65535，单位：29 毫秒。注意：板号为 AY0110702001 到 AY0110702052 的板卡，实际切换间隔为（EXT_VOUT_CYCLE_LENGTH & 0xff）×（EXT_VOUT_CYCLE_LENGTH & 0xff00）/256。

EXT_VOUT_CYCLE_SOURCE 功能，当输出方式为循环显示时，设置参加显示的源路，参数值的低 7 位有效，对应值：

第 0 位，对应复合输入 1；

第 1 位，对应复合输入 2；

第 2 位，对应复合输入 3；

第 3 位，无效；

第 4 位，对应复合输入 4；

第 5 位，对应复合输入 5；

第 6 位，对应复合输入 6；

第 7 位，无效。

以上各位 0，禁止显示；1，允许显示。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：控制图像卡的视频外接输出。包括输出的方式、循环显示等。

参　看：CGSetVideoSource。

卡　型：CG400，CG410。

19. 函　数：CGSelectCryOSC

原　型：CGSTATUS __stdcall CGSelectCryOSC（HCG hcg，CRY_OSC mode）

参　数：HCG hcg；

图像卡句柄。

CRY_OSC mode；

晶振类型。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：选择图像卡配置的晶振类型。参数 mode 的取值是由图像卡的硬件配置决定的，目前，CG300、CG320 图像卡采用 35M 晶振，CGMPEG 和 QP300 采用了 28M 晶振，在使用这些图像卡时要根据型号设定晶振类型。

卡　型：CG300，QP300。

20. 函　数：CGVideoPresent

原　型：CGSTATUS __stdcall CGVideoPresent（HCG hcg，BOOL *pStatus）

参　数：HCG hcg；

图像卡句柄。

BOOL *pStatus；

指向视频信号源的连接状态，TRUE 检测到源路的视频信号，FALSE 未检测到源路的视频信号。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：检测图像卡当前的源路是否已经连接了视频信号。

卡　型：CG300，QP300，VT121。

21. 函　数：CGWaitOddVSync

原　型：CGSTATUS __stdcall CGWaitOddVSync（HCG hcg）

参　数：HCG hcg；

图像卡句柄。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：等待奇场同步，奇场同步的到来表示奇场的开始，函数返回时正是奇场的开始，在需要控制图像的奇场数时，可以通过调用该函数实现。

范　例：CGWaitOddVSync（hCard1）；

参　看：CGWaitEvenVSync。

22. 函　数：CGWaitEvenVSync

原　型：CGSTATUS __stdcall CGWaitEvenVSync（HCG hcg）

参　数：HCG hcg；

图像卡句柄。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：等待偶场同步，偶场同步的到来表示偶场的开始，函数返回时正是偶场的开始，在需要控制图像的偶场数时，可以通过调用该函数实现。

范　例：CGWaitEvenVSync（hCard1）；

参　看：CGWaitOddVSync。

23. 函　数：CGWaitVSync

原　型：CGSTATUS __stdcall CGWaitVSync（HCG hcg）

参　数：HCG hcg；

图像卡句柄。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：等待场同步，场同步的到来表示视频信号一场的开始，函数返回时正是场的开始，在需要控制图像的场数时，可以通过调用该函数实现。

范　例：CGWaitVSync（hCard1）；

参　看：CGWaitOddVSync，CGWaitEvenVSync。

卡　型：CG400，CG410。

2.1.2.2 采集图像到屏幕

图像卡将采集的图像数据传输到计算机的显示卡中，直接显示在屏幕上。该过程不需要主机 CPU 的干预，图像实时显示。

1. 函　数：CGCapture

原　型：CGSTATUS __stdcall CGCapture（HCG hcg，BOOL bEnable）

参　数：HCG hcg；

图像卡句柄。

BOOL bEnable；

TRUE 采集，FALSE 冻结。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：控制图像卡向计算机的显示卡(屏幕)采集图像。输入参数 bEnable 为 TRUE，开始采集图像；为 FALSE，关闭图像采集。

在采集图像到内存的过程中，不能向屏幕采集图像，否则返回 CG_IN_WORK。

范　例：CGCapture（hCard1，TRUE）；　　//采集图像

CGCapture（hCard1，FALSE）；　//冻结图像

参　看：CGCaptureShot。

2. 函　数：CGCaptureShot

原　型：CGSTATUS __stdcall CGCaptureShot（HCG hcg）

参　数：HCG hcg；

图像卡句柄。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：抓取一帧/场图像到计算机的显示卡(屏幕)，图像卡打开图像采集，待采完一帧/场完整的图像后立即关闭。函数完成后图像卡处于冻结状态。

当图像卡正在执行采集图像到屏幕或内存的操作时，应先结束操作，然后再调用向屏幕抓取图像的功能，否则返回 CG_IN_WORK。

范　例：CGCaptureShot (hCard1);

参　看：CGCapture。

2.1.2.3 采集图像到内存

图像卡将采集的图像数据传输到计算机的内存中，用户可以对内存中的数据进行处理，也可以利用位图函数将图像显示在计算机的屏幕上。采集图像到内存时，不占用主机 CPU 的时间。

·静态内存

图像卡使用计算机内存采集图像数据，这段内存应该是物理连续的，在计算机系统启动时分配、预留出来，供图像卡使用，用户访问，并且内存在系统运行过程中一直存在，直到系统关闭时才被释放。我们定义这部分内存为静态内存。

·静态内存的分配

有两种方式可以分配静态内存。

1. 安装设备驱动程序时分配

使用随图像卡提供的驱动程序，安装设备驱动程序时，分配静

态内存 1000 页，即 4000KB。如果用户需要改变分配静态内存的大小，可通过修改 INF 文件或配置文件中的参数实现，下面以 CG300 图像卡驱动程序为例，进行说明。

Win9x 系统中，修改文件 CG300vi.inf 中的节[DriverAddReg]，其中的“HKLM，%STMEMKEY%，BlockSize，0x00010001，1000”项中的 1000 值，这个值的单位为页，即分配 1000 页静态内存。修改这个值来设置所需的内存大小。

类似的，在 Win2000/XP 系统中，修改文件 CG300vi.inf 中的节[DriverAddReg.NTx86]，其中的“HKLM，%DVCSERVPARAM%，RequiredMemorySize，0x00010001，1000”项中的 1000 值，设定所需静态内存的大小。

在 WinNT 系统中，通过修改配置文件 CG300NT.ini 中的节[Parameters]，其中的“RequiredMemorySize”项的赋值，设定所需静态内存的大小。

这种分配方式，适合于发布系统时使用，用户将图像卡设备驱动程序和自己的应用系统发布时，可定制分配静态内存的大小。

2. 在应用程序中调用函数 CGSetStaticMem 指定静态内存大小，然后重新启动计算机，分配静态内存。

重新设置要分配的静态内存大小后，每次启动计算机，系统都会按最新指定的大小分配完成，而不必再重新申请分配。不申请静态内存时，设为 0 页。

申请静态内存的大小与计算机的内存配置以及所使用的操作系统有关，例如：1GB 内存配置在 WinXP 下分配了大约 800MB。

·静态内存的使用

在使用静态内存之前，应该先获取当前静态内存状态。静态内存状态主要包括分配状态和锁定状态。分配状态是指用户指定大小的内存是否分配（预留）成功；锁定状态是指静态内存能否在程序中被锁定，静态内存分配成功后，用户不能直接使用，必须在自己的程序中将内存锁定后才能使用。

首先通过函数 CGGetStaticMem 获取当前静态内存的分配状态，分配成功的内存才能使用，如果要分配的静态内存过大，系统分配不成功。用户只能减少分配的数量，重新分配。重新分配静态内存方法参见“静态内存的分配”说明。

然后使用函数 CGStaticMemLock 对指定位置和大小的静态内存进行锁定，一般情况下，分配的内存都可以锁定成功。如果分配的内存较大，不能一次全部锁定成功，那么可以减少锁定内存的大小，对内存进行分段锁定。对于锁定成功的静态内存，用户可以通过锁定后得到的指针进行访问，例如，用户可以将图像卡采集到静态内存的图像数据，拷贝到用户自己分配的内存缓冲区，这里用户分配的内存是指利用操作系统 API 提供的函数，如 GlobalAlloc、HeapAlloc、new 等分配的内存。

在接口库中，图像卡采集图像到静态内存的操作都使用偏移 Offset 和长度 Length 来定位内存位置。例如，启动连续采集到内

存函数 CGStartSnap（HCG hcg， DWORD dwMemOffset， BOOL bInterline，WORD wSum）等。用户访问静态内存要通过内存的指针，例如，图像数据传递函数 CGDataTransform。

静态内存访问结束后，使用函数CGStaticMemUnlock解锁内存，应用程序每次对某个内存对象调用CGStaticMemLock时，最后都必须对该对象调用CGStaticMemUnlock。

1. 函　数：CGSetStaticMem

原　型：CGSTATUS __stdcall CGSetStaticMem（DWORD dwPages）

参　数：DWORD dwPages；

申请静态内存的页数，每页为 4KB。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：申请静态内存，图像卡采集图像数据到内存时，需要通过此函数申请静态内存。调用函数改变申请静态内存大小，计算机重新启动后，申请才能有效。内存分配是否成功可调用函数 CGGetStaticMem 查看。同时，申请分配的静态内存，在每次启动计算机后，都会自动分配完成，而不必再重新申请。

参　看：CGGetStaticMem。

2. 函　数：CGGetStaticMem

原　型：CGSTATUS __stdcall CGGetStaticMem（DWORD *pPages）

参　数：DWORD *pPages；

指向所申请静态内存的页数，每页为 4KB。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：获取静态内存的分配状态和申请的大小。

参　看：CGSetStaticMem。

3. 函　数：CGStaticMemLock

原　型：CGSTATUS __stdcall CGStaticMemLock（DWORD dwOffset，DWORD dwLength，HANDLE *pHandle，PVOID *ppLineAddr）

参　数：DWORD dwOffset；

内存偏移量，以字节为单位，相对于所申请到的静态内存起始地址。

DWORD dwLength；

内存锁定的长度，以字节为单位。

HANDLE *pHandle；

指向被锁定内存的句柄。

PVOID *ppLineAddr；

指向被锁定内存的指针。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：锁定指定位置和大小的静态内存，锁定成功后，就可以使用指针 ppLineAddr 访问内存数据，例如，调用函数 CGDataTransform 传递图像数据。

使用时，可以在应用程序初始化时锁定静态内存，在应用程序退出时解锁，这样在程序运行过程中都可以通过指针直接访问静态内存。也可以在访问内存前锁定静态内存，访问完毕后，解除锁定。

参　看：CGStaticMemUnlock。

4. 函　数：CGStaticMemUnlock

原　型：CGSTATUS __stdcall CGStaticMemUnlock（HANDLE handle）

参　数：HANDLE handle;

被锁定内存的句柄

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：解锁函数 CGStaticMemLock 锁定的静态内存。解锁内存时，输入参数 handle 的值应与锁定内存后得到的值保持一致，否则返回错误 CG_PARAMETER_INVALID。

参　看：CGStaticMemLock。

5. 函　数：CGSnapShot

原　型：CGSTATUS __stdcall CGSnapShot（HCG hcg，DWORD dwMemOffset，WORD wIntervSyncs，BOOL bInterline，WORD wSum）

参　数：HCG hcg；

图像卡句柄。

DWORD dwMemOffset；

内存偏移量，以字节为单位，相对于所申请到的静态内存起始地址。

WORD wIntervSyncs；

间隔帧数。

BOOL bInterline；

图像数据的存放方式，在视频信号扫描方式为帧方式时有效，TRUE 隔行存放，FALSE 奇偶场逐行存放。

WORD wSum；

采集图像的数量。在视频信号扫描方式为帧方式时，以帧为单位，场方式或场方式 1 时，以场为单位。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：图像卡采集若干帧/场图像数据到内存中。内存的大小由采集图像的尺寸和数量决定，图像的宽度、高度等于当前图像输出窗口的宽度和高度，图像的大小（字节数）为：输出窗口宽 × 高 × 图像数据格式位数 / 8。当

视频信号按场方式或场方式 1 扫描时，图像的高度是输出窗口高度的一半，参见 CGSetOutputWindow，CGSetVideoFormat，CGSetScanMode 的函数说明。

函数可以在视频扫描为帧方式、场方式或场方式 1 下采集图像，特别的，在帧方式中，图像数据存放方式有：隔行存放，即奇、偶场的图像数据交叉存放，组成一帧完整图像；奇偶场逐行存放，即一幅图像的上半部分为偶场图像，下半部分为奇场图像，图像数据是连续的。

在采集过程中可通过设定间隔的帧数，控制图像采集频率。

在采集图像到屏幕过程中，不能采集图像到内存，否则返回 CG_IN_WORK。

范　例：CGSnapShot（hCard1，0，0，TRUE，1）；　//采集一帧图像到内存

参　看：CGStartSnap。

6. 函　数：CGStartSnap

原　型：CGSTATUS __stdcall CGStartSnap（HCG hcg，DWORD dwMemOffset，BOOL bInterline，WORD wSum）

参　数：HCG hcg；

图像卡句柄。

DWORD dwMemOffset；

内存偏移量，以字节为单位，相对于所申请到的静态内存起始地址。

BOOL bInterline;

图像数据的存放方式，在视频信号扫描方式为帧方式时有效，TRUE 隔行存放，FALSE 奇偶场逐行存放。

WORD wSum;

循环采集图像的数量。在视频信号扫描方式为帧方式时，以帧为单位，场方式或场方式 1 时，以场为单位，取值 ≥ 2。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：启动图像卡采集若干帧/场图像数据到内存功能。

与函数 CGSnapShot 不同，在采集图像数据的过程中，用户可以运行自己的程序，例如，图像处理程序，并且因为图像卡采集图像时不占用计算机 CPU 的时间，用户可以使用 CPU 的全部资源。

输入参数 wSum 设置内存中图像缓冲区的数量（以帧或场为单位），且 wSum ≥ 2，函数执行后，图像卡就在这个区域内自动循环、反复采集图像数据，用户通过函数 CGGetSnappingNumber 查看图像的采集状态（当前采集到了第几帧（场）图像），来决定是处理前面的图像数据；还是等待当前图像采集完毕，处理当前的图像数据。例如，设置两帧图像缓冲区采集图像，当一帧图像正在

采集时，可以处理另一帧图像数据，图像卡采集图像，第一帧采集完毕，开始向第二帧的内存采集图像时，用户就可以处理第一帧图像数据了；当第二帧图像采集完成，并且开始向第一帧的内存采集图像时，就可以处理第二帧图像数据了……如此反复，当数据处理时间 PAL 制图像小于 1/25 秒,NTSC 制图像小于 1/30 秒就可以实现图像的实时处理。

图像缓冲区（静态内存）大小的计算，帧方式下，图像数据的存放参见函数 CGSnapShot 的相关说明。

结束图像卡采集操作，调用函数 CGStopSnap。

参　看：CGSnapShot，CGGetSnappingNumber，CGStopSnap。

7. 函　数：CGGetSnappingNumber

原　型：CGSTATUS __stdcall CGGetSnappingNumber（HCG hcg，int *pNumber）

参　数：HCG hcg；

图像卡句柄。

int *pNumber；

指向当前图像的采集状态字。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：获取当前采集图像的状态，这个采集状态是指执行函数 CGStartSnap 后，图像卡当前正在向内存采集那一场图

像。无论视频信号扫描方式是帧方式、场方式或场方式1，状态字的值都以场为单位，从 0 开始。例如CGStartSnap 在帧方式下采集 10 帧图像，状态字的值为第 0 至 19 场；在场方式或场方式 1 下采集 10 场图像，状态字的值为第 0 至 9 场。

采集状态字的读取，必须在函数 CGStartSnap 执行后，否则返回 CG_NOT_START_SNAP。

参　看：CGStartSnap，CGStopSnap。

8. 函　数：CGStopSnap

原　型：CGSTATUS __stdcall CGStopSnap（HCG hcg）

参　数：HCG hcg；

图像卡句柄。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：停止图像卡采集图像数据到内存。

调用函数 CGStartSnap 开始采集图像，图像采集结束后，要执行 CGStopSnap 关闭采集。

参　看：CGStartSnap，CGGetSnappingNumber。

2.1.2.4 错误处理

对系统运行状态和函数调用返回值的处理。

1. 函　数：CGGetErrorString

原　型:char＊__stdcall CGGetErrorString(CGSTATUS status)

参　数：CGSTATUS status;

错误代码。

返回值：返回指向错误代码说明的字符串指针。

说　明：根据输入的错误代码，返回相应的说明。

2.1.2.5 其它功能

一些附加的软/硬件应用功能。

1. 函　数：CGGetCardType

原　型：CGSTATUS __stdcall CGGetCardType（HCG hcg，

CGTYPE *pType）

参　数：HCG hcg;

图像卡句柄。

CGTYPE *pType;

指向图像卡类型。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：获取指定图像卡的类型。

2. 函　数：CGGetCardTotal

原　型：CGSTATUS __stdcall CGGetCardTotal（int *pNumber）

参　数：int *pNumber;

指向图像卡的总数。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：获取当前系统中图像采集卡的总数。

3．函　数：CGGetBoardInfo

原　型：CGSTATUS __stdcall CGGetBoardInfo（HCG hcg，
BOARD_INFO mode，char *pInfo）

参　数：HCG hcg；

图像卡句柄。

BOARD_INFO mode；

图像卡的标志信息。

char *pInfo；

指向存放标志信息的缓冲区。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：读取图像卡的标志信息，包括板型、序列号和设备子号。板型由 2 个字节组成，DH-CG400 的板型为“CG”，DH-CG410 为“CH”，DH-VT121 板型为“VT”。序列号由 12 个字节组成，每个 CG400、CG410、VT121 有唯一的序列号。VT121 设备子号取值为 0 或 1。

卡　型：CG400，CG410，VT121。

4. 函　数：CGCheckBoard

原　型：CGSTATUS __stdcall CGCheckBoard（HCG hcg，
CHECK_PARAM param，BOOL *pStatus）

参　数：HCG hcg；
图像卡句柄。
CHECK_PARAM param；
查验参数，参数的定义参见 CHECK_PARAM 结构说明。
这些参数由图像卡制造商提供。
BOOL *pStatus；
指向查验状态字，TRUE 正确，FALSE 错误。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：使用给定的参数对图像卡进行查验。

卡　型：CG400，CG410，VT121。

5. 函　数：CGReadSCMParam

原　型：CGSTATUS __stdcall CGReadSCMParam（HCG hcg，BYTE
*pBuffer）

参　数：HCG hcg；
图像卡句柄。
BYTE *pBuffer；
指向数据缓冲区。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：对图像卡上的单片机进行读操作，读取单片机的数据到缓冲区，数据长度 2 个字节。数据的低 9 位有效，其中第 0 － 7 位为数据位（第 0 个字节）；第 8 位为状态位（第 1 个字节的第 0 位），表示单片机上的读缓冲区是否已被更新，1 未更新，0 已更新。用户通过接口使用单片机，实现自己的功能。

参　看：CGWriteSCMParam。

卡　型：CG410。

6. 函　数：CGWriteSCMParam

原　型：CGSTATUS __stdcall CGWriteSCMParam（HCG hcg，BYTE *pBuffer，BYTE byEntries）

参　数：HCG hcg；

图像卡句柄。

BYTE *pBuffer；

指向数据缓冲区。

BYTE byEntries；

数据的数目。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：对图像卡上的单片机进行写操作，将缓冲区的数据写入单片机。由于单片机采用串口模式 2 的通信方式，所以每个数据的长度是 9 位，这样我们规定第 0 个数据（第

0 个字节）的最低位用于表示每个数据的第 9 位，即本次写操作的所有数据的第 9 位均相同，同时在设置参数 byEntries 确定要写入数据的数目时，在实际写入数据的数目上要加 1。用户通过接口使用单片机，实现自己的功能。

范　例：BYTE bywBuf0[5] = {0x00,0x11,0x12,0x13,0x14};
BYTE bywBuf1[5] = {0x01,0x11,0x12,0x13,0x14};
//要写入 4 个数据：0x11、0x12、0x13、0x14，每个数据的第九位均为 0
CGWriteSCMParam(hCard, bywBuf0, 5);
//要写入 4 个数据：0x11、0x12、0x13、0x14，每个数据的第九位均为 1
CGWriteSCMParam(hCard, bywBuf1, 5);

参　看：CGReadSCMParam。

卡　型：CG410。

7. 函　数：CGLoadCFGFile

原　型：CGSTATUS __stdcall CGLoadCFGFile（HCG hcg, char *filename）

参　数：HCG hcg;
图像卡句柄。
char *filename;

指向配置文件名。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：打开图像卡硬件参数配置文件，读取参数，根据参数设置图像卡。配置文件由相应的工具生成，如果配置文件格式不正确，则返回 CG_CFG_FILE_INVALID。

卡　型：CG400，CG410。

2.2 扩充功能

扩充功能模块包括采集图像到屏幕控制、采集图像到内存控制、数据传递等功能。函数的原型声明在包含文件 CGVidEx.h，动态链接库 CGVidEx.dll，静态链接库 CGVidEx.lib。

2.2.1 定义

模块中相关常量，数据结构，宏的说明。

2.2.1.1 常量

1. typedef enum tagINTERCHANGE_TYPE {

```
    HOR_ODD_COLUMN  = 0,
    HOR_EVEN_COLUMN = 1,
    VERT_ODD_ROW    = 2,
    VERT_EVEN_ROW   = 3,
    VERT_DOUBLE     = 4
} INTERCHANGE_TYPE;
```

INTERCHANGE_TYPE 标识相邻行、列图像数据的变换方式。

HOR_ODD_COLUMN

水平方向奇数列抽取；

HOR_EVEN_COLUMN

水平方向偶数列抽取；

VERT_ODD_ROW

垂直方向奇数行抽取；

VERT_EVEN_ROW

垂直方向偶数行抽取；

VERT_DOUBLE

垂直方向逐行复制。

2.2.1.2 结构

```
1. typedef struct tagSANP_INFO {
       HCG hcg;
       int nDevice;
       int nNumber;
       void *pParam;
   } SNAP_INFO;
```

SNAP_INFO 结构包含执行采集图像到内存的操作中，图像卡的工作状态。在连续循环采集图像到内存的过程中，标示当前那一图像卡设备，采集完成了第多少序号的图像数据和用户定义的参数。

成员 hcg

标示当前工作的图像卡的句柄。

nDevice

标示当前工作的图像卡的序号(从1开始)。

nNumber

标示当前采集完成的图像的序号(从0开始)。

pParam

指向用户定义的参数。

2. typedef int （CALLBACK *SNAPPROC）（SNAP_INFO *pInfo）；

SNAPPROC标示回调函数的声明。在采集图像到内存控制程序中，用户定义回调函数，在函数体中编写自己的程序，如，图像数据的处理等。使用函数CGOpenSnapEx注册回调函数，由采集控制程序调用。

2.2.2 函数说明

2.2.2.1 采集图像到屏幕控制

控制图像卡采集图像到屏幕的功能。

1. 函　数：CGCaptureEx

原　型：CGSTATUS __stdcall CGCaptureEx（HCG hcg，HWND hwnd，BOOL bEnable）

参　数：HCG hcg；

图像卡句柄。

HWND hWnd;

采集图像显示窗口句柄。

BOOL bEnable;

TRUE 采集，FALSE 冻结。

返回值：调用成功，返回CG_OK，否则返回错误代码。

说　明：控制图像卡向计算机的显示卡(屏幕)采集图像。功能与函数CGCapture相似，不同的是，在图像采集过程中，用户可以在图像显示区域上同屏迭加其它窗口，函数自动屏蔽相关区域，保证了采集图像和窗口内容的显示完整性。

输入参数hWnd为图像显示窗口的句柄。该窗口用来限定图像输出显示范围，即函数CGSetOutputWindow所确定的视频输出窗口应在hWnd所标识的窗口范围之内。

图像采集时，不能使用CGSetPixelMask设置图像象素点的屏蔽，在图像冻结后才能设置屏蔽。

函数CGCaputreEx和CGCapture不能同时使用，函数的图像采集、冻结操作也不能混合使用，例如，不能启动采集用CGCaptureEx，而停止采集用CGCapture。

参 看：CGCapture。

2.2.2.2 采集图像到内存控制

控制图像卡采集图像到内存的功能。

1. 函　数：CGOpenSnapEx

原　型：CGSTATUS __stdcall CGOpenSnapEx（HCG hcg，SNAPPROC pSnapFunc，void *pParam）

参　数：HCG hcg；

图像卡句柄。

SNAPPROC pSnapFunc；

指向回调函数。

void *pParam；

指向用户定义参数，参数对应于回调函数参数SNAP_INFO的字段pParam，它是传递给回调函数的。

返回值：调用成功，返回CG_OK，否则返回错误代码。

说　明：初始化图像卡采集图像到内存的控制，指定回调函数和用户定义的参数，分配资源。在使用采集图像到内存的控制功能前，必须进行初始化。

用户可以在回调函数中编写自己的程序，例如图像处理程序、控制程序等，如果图像处理时间，PAL制图像小于1/25秒，NTSC制图像小于1/30秒就可以实现图像的实时处理。

通过回调函数定义的参数SNAP_INFO，可以获得图像卡的句柄、序号、采集完成的图像的序号和用户定义的参数信息。

控制功能使用完毕，调用CGCloseSnapEx关闭。

参　见：CGCloseSnapEx。

2．函　数：CGCloseSnapEx

原　型：CGSTATUS __stdcall CGCloseSnapEx（HCG hcg）

参　数：HCG hcg；

图像卡句柄。

返回值：调用成功，返回CG_OK，否则返回错误代码。

说　明：关闭采集图像到内存的控制，释放申请的资源。

参　见：CGOpenSnapEx。

3．函　数：CGStartSnapEx

原　型：CGSTATUS __stdcall CGStartSnapEx（HCG hcg，DWORD dwMemOffset，BOOL bInterlace，WORD wSum）

参　数：HCG hcg；

图像卡句柄。

DWORD dwMemOffset；

内存偏移量，以字节为单位，相对于所申请到的静态内存起始地址。

BOOL bInterline；

图像数据的存放方式，在视频信号扫描方式为帧方式时有效，TRUE 隔行存放，FALSE 奇偶场逐行存放。

WORD wSum；

循环采集图像的数量。在视频信号扫描方式为帧方式时，以帧为单位，场方式或场方式 1 时，以场为单位，取值 ≥ 2。

返回值：调用成功，返回CG_OK，否则返回错误代码。

说　明：启动图像卡采集图像到内存。采集图像到内存控制的初始化成功后，启动采集功能。

与函数CGStartSnap不同的是，在图像采集过程中，用户可以通过回调函数运行自己的程序，执行图像数据访问，系统控制等操作。不需要通过函数CGGetSnappingNumber查看采集状态。回调函数的说明参见函数CGOpenSnapEx。

停止图像卡采集操作，调用函数CGStopSnapEx。

参　见：CGStopSnapEx。

4. 函　数：CGStopSnapEx

原　型：CGSTATUS __stdcall CGStopSnapEx (HCG hcg)

参　数：HCG hcg;

图像卡句柄。

返回值：调用成功，返回CG_OK，否则返回错误代码。

说　明：停止采集图像到内存。调用函数CGStartSnapEx可以再次启动采集。

参　见：CGStartSnapEx。

2.2.2.3 数据传递

在使用数据传递函数时，根据图像大小和传递方式，保证申请的源和目标数据缓冲区是足够大和有效的。

1. 函　数：CGDataTransform

原　型:CGSTATUS __stdcall CGDataTransform(BYTE *pDestBuf, BYTE *pSrcBuf, int nWidth, int nHeight, int nBitCount, BOOL bVerFlip)

参　数：BYTE *pDestBuf;

指向目标缓冲区。

BYTE *pSrcBuf;

指向源缓冲区。

int nWidth;

图像宽度。

int nHeight;

图像高度。

int nBitCount;

图像数据格式（位数/像素）。

BOOL bVerFilp;

图像沿垂直方向翻转，TRUE 翻转，FALSE 正常。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：将源缓冲区的图像数据传递到目标缓冲区中，同时进行数据格式转换。

数据在缓冲区中按先行后列的顺序存放，目标缓冲区的数据格式为 8 位灰度图像，每个像素对应其灰度值；或 24 位彩色图像，每个像素有三个颜色值红、绿、蓝各 8 位。如果源缓冲区的数据格式与目标缓冲区不一致，则要进行转换。数据格式应是 8 位灰度图像或红绿蓝的彩色图像。

范　例：CGDataTransform（pDestBuffer，pSourceBuffer，768，576，24，TRUE）；

2．函　数：CGGetDisplayBits

原　型：CGSTATUS __stdcall CGGetDisplayBits(HDC hdc，int nLeft，int nTop，int nWidth，int nHeight，int *pBitCount，BYTE *pImageBuffer)

参　数：HDC hdc；

显示屏幕设备描述表句柄。

int nLeft；

窗口左上角 x 坐标。

int nTop；

窗口左上角 y 坐标。

int nWidth；

窗口宽度。

int nHeight；

窗口高度。

int *pBitCount;

指向图像数据格式（位数/像素）。

BYTE *pImageBuffer;

指向图像数据缓冲区。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：读取计算机显示屏幕中任意窗口内的图像数据，数据在缓冲区 pImageBuffer 中按先行后列的顺序存放，数据格式为 8 位灰度图像，每个像素对应其灰度值；或 24 位彩色图像，每个像素有三个颜色值红、绿、蓝各 8 位。

3．函　数：CGDataInterchange

原　型：CGSTATUS __stdcall CGDataInterchange(BYTE *pDestBuf, BYTE *pSrcBuf, int nWidth, int nHeight, int nBitCount, INTERCHANGE_TYPE type)

参　数：BYTE *pDestBuf;

指向目标缓冲区。

BYTE *pSrcBuf;

指向源缓冲区。

int nWidth;

图像宽度。

int nHeight;

图像高度。

int nBitCount;

图像数据格式（位数/像素）。

INTERCHANGE_TYPE type;

数据变换方式。

返回值：调用成功，返回 CG_OK，否则返回错误代码。

说　明：对相邻行、列的图像数据进行变换。

在视频信号场扫描方式下，采集的一场图像数据，垂直方向被压缩了一半，这时可以通过函数将数据沿垂直方向逐行复制，得到一幅放大的图像；也可以沿水平方向抽取奇数或偶数列，得到一幅缩小的图像。

一帧图像可以沿垂直方向抽取奇场、偶场数据。

3.附录

3.1 函数返回值代码表

CG_OK	0
成功	
CG_ALLOCATE_DEVICE_MEMORY_ERROR	-1
静态内存分配不成功	
CG_LOCK_DEVICE_MEMORY_ERROR	-2
静态内存锁定失败	
CG_ACCESS_DEVICE_MEMORY_DENIED	-3
静态内存访问越界	
CG_PARAMETER_INVALID	-4
参数无效	
CG_NOT_SUPPORT_INTERFACE	-5
不支持该接口	
CG_OPEN_DRIVER_FAILED	-6
装载驱动失败	
CG_CARD_HANDLE_INVALID	-7
句柄无效	
CG_NO_CARD_FOUND	-8
没有发现图像卡	
CG_HW_MAP_MEMORY_ERROR	-9
映射内存失败	

CG_NOT_ENOUGH_SYSTEM_MEMORY -10

没有足够系统内存

CG_HW_INIT_ERROR -11

硬件初始化错误

CG_PARAMETER_OUT_OF_BOUND -12

参数越界

CG_HW_INIT_I2C_ERROR -13

初始化I2C错误

CG_HW_INIT_AD_ERROR -14

初始化AD错误

CG_HW_BOARD_TYPE_ERROR -15

板型错误

CG_SNAP_SHOT_TIME_OUT -16

SnapShot超时

CG_INIT_DDRAW_ERROR -17

初始化DirectDraw失败

CG_IN_WORK -18

正在采集，不能执行该操作

CG_NOT_START_CAPTURE -19

没有启动采集到屏幕

CG_NOT_START_CAPTURE_EX -20

没有启动采集到屏幕控制

CG_NOT_START_SNAP -21

没有启动采集到内存

CG_NOT_START_SNAP_EX -22

没有启动采集到内存控制

CG_NOT_OPEN_SNAP_EX -23

没有初始化采集到内存控制

CG_VIDEO_FORMAT_INVALID -24

视频格式无效

CG_NOT_SUPPORT_CARD_TYPE -25

不支持该图像卡类型

CG_FILE_CREATE_ERROR -26

创建文件失败

CG_FILE_INVALID -27

文件格式无效

CG_VGA_BASE_INVALID -28

显示卡的基地址无效

CG_VIDEO_WINDOW_INVALID -29

视频（输入/输出）窗口无效

CG_INTERNAL_ERROR -30

内部错误

3.2 函数对照表

应用接口库是大恒系列图像采集卡应用的通用库，接口库中的函数和原来定义的图像卡接口函数的对应关系如下表。

1	BeginCGCard	BeginCG300 、BeginCG400
2	EndCGCard	EndCG300 、EndCG400
3	CGSetInputWindow	CG300SetInpVideoWindow CG400SetInpVideoWindow
4	CGSetOutputWindow	CG300SetDispWindow、 CG400SetDispWindow
5	CGSetVideoSource	CG300SetADParam、CG400SetADParam
6	CGAdjustVideo	CG300SetADParam、 CG400SetADParam
7	CGSetVideoFormat	CG300SetColorSpace、 CG400SetColorSpace
8	CGSetVideoStandard	CG300SetVideoStandard CG400SetVideoStandard
9	CGSetDelay	CG300SetDelay 、CG400SetDelay
10	CGSetPLLFrequency	CG300SetPLLFreqParam
11	CGSetScanMode	CG300SetDispMode、 CG400SetDispMode
12	CGEnableMask	CG300EnableMask、 CG400EnableMask
13	CGSetPixelMask	
14	CGLumaControl	CG400LumaControl
15	CGGammaCorrControl	CG400GammaCorrControl
16	CGSetGammaCorrCoef	CG400SetGammaCorrCoef
17	CGEnableVideoMirror	CG400EnableVideoMirror
18	CGSetVideoExtOutput	CG400SetVideoOutput

19	CGSelectCryOSC	CG300SelectCryOSC
20	CGVideoPresent	CG300VideoPresent
21	CGWaitOddVSync	CG300WaitOddVSync 、CG400WaitOddVSync
22	CGWaitEvenVSync	CG300WaitEvenVSync、 CG400WaitEvenVSync
23	CGWaitVSync	CG400WaitVSync
24	CGCapture	CG300Capture 、CG400Capture
25	CGCaptureShot	CG300Snap 、CG400Snap
26	CGSetStaticMem	SetStaticMemAlloc
27	CGGetStaticMem	StaticMemAlloc
28	CGStaticMemLock	
29	CGStaticMemUnlock	
30	CGSnapShot	CG300CaptureToMem、 CG400CaptureToMem
31	CGStartSnap	CG300SnapToMem、 CG400SnapToMem
32	CGGetSnappingNumber	CG300GetSnapToMemNumber CG400GetSnapToMemNumber
33	CGStopSnap	CG300Capture 、CG400Capture
34	CGGetErrorString	
35	CGGetCardType	
36	CGGetCardTotal	
37	CGGetBoardInfo	CG400GetBoardInfos
38	CGCheckBoard	CG400Check
39	CGLoadCFGFile	CG400LoadCFGFile
40	CGDataTransform	CG300ReadFromMem、 CG400ReadFromMem

4.修改历史

版本	修改内容	发布日期
2008 年 7 月	修改第 2 页的营销中心和技术支持的联系方式	2008-7-4
2009 年 3 月	增加了一种扫描方式：FIELD1， 修改：第 8 页的常量 VIDEO_SCAN， 第 24 页的函数 CGSetScanMode。	2009-3-17