

IKapBoard 用户开发手册

(V1.6.4 2023.05.16)



合肥埃科光电科技股份有限公司

<http://www.i-tek.cn/>

历史版本

版本	日期	更新内容描述
1.4.1	2016/1/26	<ul style="list-style-type: none"> ● 修改常量说明中的错误 ● 修改第 6 节库的使用方法中的配图 ● 修改函数说明的表述方法
1.4.2	2016/1/28	<ul style="list-style-type: none"> ● 修改文档格式
1.4.3	2016/8/10	<ul style="list-style-type: none"> ● 修改部分函数定义和表述方式 ● 修改文档格式
1.4.4	2016/8/21	<ul style="list-style-type: none"> ● 修改文档格式
1.4.5	2016/8/23	<ul style="list-style-type: none"> ● 修改产品名称 ● 修改常量表示方法 ● 增加对于采集超时和缓冲区溢出事件的检测
1.4.6	2016/8/26	<ul style="list-style-type: none"> ● 修改示例代码中的错误说明
1.4.7	2016/9/24	<ul style="list-style-type: none"> ● 修改函数定义 ● 增加读取和设置采集卡配置参数的功能
1.4.8	2016/10/17	<ul style="list-style-type: none"> ● 增加 C# API 使用方法 ● 增加配置参数 IKP_IMAGE_TYPE
1.4.9	2016/12/27	<ul style="list-style-type: none"> ● IKapGrabOnce() 支持同步和异步调用两种方式 ● 删除 IKapGetCameraStatus() 的第二个传入参数 <i>bModeChange</i> ● 删除 IKapGetAvaliableFrameCount() ● 添加事件类型： <ul style="list-style-type: none"> ➢ IKEvent_GrabLine + n ➢ IKEvent_GrabLineEnd + n
1.5.0	2017/01/04	<ul style="list-style-type: none"> ● 增加内存管理 ● 增加软件开发环境
1.5.3	2017/01/15	<ul style="list-style-type: none"> ● 更新 IKP_IMAGE_HEIGHT, IKP_IMAGE_WIDTH, IKP_BOARD_BIT 参数说明 ● 更新内存管理
1.5.4	2017/04/15	<ul style="list-style-type: none"> ● 增加时间统计功能 ● 修改彩色图像像素排列模式为 BGR
1.5.5	2017/09/21	<ul style="list-style-type: none"> ● 增加读取/配置的采集卡参数
1.5.6	2018/02/08	<ul style="list-style-type: none"> ● 增加读取/配置的采集卡参数 ● 增加等待采集操作结束的 API
1.5.7	2018/05/22	<ul style="list-style-type: none"> ● 增加配置采集卡参数的示例代码 ● 增加 RGBC 数据采集类型
1.5.8	2018/06/14	<ul style="list-style-type: none"> ● 增加采集卡软件触发模式
1.5.9	2018/09/03	<ul style="list-style-type: none"> ● 增加编码器倍频系数选项 ● 增加 3Tap-RGB-Arrangement1 模式 ● 增加 4Tap-Mono-Arrangement7 模式
1.5.10	2018/11/21	<ul style="list-style-type: none"> ● 修改关于 IKP_SHAFT_ENCODER1_PULSE_DROP 的描

		述
1.5.11	2019/06/20	<ul style="list-style-type: none"> ● 增加关于 IKP_IMAGE_ROI_X 的描述，并修改关于 IKP_IMAGE_OFFSET_X 的描述 ● 增加外部输入信号触发回调函数的功能
1.5.12	2019/10/30	<ul style="list-style-type: none"> ● 增加帧开始和帧完成回调函数功能
1.5.13	2020/01/19	<ul style="list-style-type: none"> ● 增加编码器输入信号滤波功能：IKP_SHAFT_ENCODER1_MIN_WIDTH ● 增加编码器输入信号有效方向控制功能：IKP_SHAFT_ENCODER1_VALID_DIRECTION ● 增加编码器输入信号反向补偿功能：IKP_SHAFT_ENCODER1_REVERSE_COMPENSATION
1.5.14	2020/09/24	<ul style="list-style-type: none"> ● 去除文档中 C# 部分
1.5.15	2020/11/26	<ul style="list-style-type: none"> ● 增加采集卡外部触发帧数的设置功能：IKP_BOARD_TRIGGER_OUTTER_MODE_FRAME_COUNT
1.5.16	2021/07/21	<ul style="list-style-type: none"> ● 更新函数说明
1.6.0	2022/04/12	<ul style="list-style-type: none"> ● 修改文档格式
1.6.1	2022/04/26	<ul style="list-style-type: none"> ● 更新公司 logo
1.6.2	2022/08/10	<ul style="list-style-type: none"> ● 添加 K6 采集卡远距离传输功能：IKP_CL_LONG_DISTANCE_TRANSMISSION ● 更新配置参数和回调事件常量列表
1.6.3	2023/02/11	<ul style="list-style-type: none"> ● 更新公司地址
1.6.4	2023/05/16	<ul style="list-style-type: none"> ● 添加 Vulcan-SFP-2 相关描述 ● 添加函数接口：IKapOpenGVB()

联系方式

警告

版权所有 © 2023 合肥埃科光电科技股份有限公司

本用户开发手册由合肥埃科光电科技股份有限公司编印，版权所有，翻版必究。

声明

本用户开发手册适用于合肥埃科光电科技股份有限公司 Vulcan-CL、Vulcan-CXP 和 Vulcan-SFP-2 PCIe 系列采集卡。在使用 Vulcan-CL、Vulcan-CXP 或 Vulcan-SFP-2 PCIe 采集卡前，请仔细阅读本手册，并妥善保管，以便备用。合肥埃科光电科技股份有限公司保留对本手册中的打印错误、与最新资料不一致、软件升级及产品改进等解释权及随时进行改动的权利。这些更改恕不另行通知，将直接编入新版手册中。

合肥埃科光电科技股份有限公司

电话: +86-551-65318597

传真: +86-551-65318597

网址: www.i-tek.cn

地址: 安徽省合肥市高新区望江西路中安创谷科技园二期 J2 栋 3F

邮编: 230088

目 录

历史版本	2
联系方式	4
目 录	5
1 概述	7
1.1 关于 IKapBoard	7
1.2 关于 IKapExpert	7
2 开始使用 IKapBoard	8
2.1 主要组成	8
2.2 产品版本	8
2.3 适用对象	8
2.4 一般约定	8
2.5 开发环境	9
3 创建 IKapBoard 应用	10
3.1 头文件和库文件	10
3.2 创建 PCIe 采集卡的 C 应用程序	10
4 IKapBoard 使用流程	11
4.1 初始化	11
4.2 设置参数	11
4.3 设置图像缓冲区	12
4.4 注册回调函数	13
4.5 采集图像	14
4.6 处理图像	14
4.7 结束采集	15
5 API 函数说明	16
5.1 函数概览	16
5.2 函数详细说明	17
6 图像缓冲区内存管理	42
6.1 图像缓冲区状态	42
6.2 图像缓冲区传输模式	43
7 IKapBoard 使用范例	44
7.1 枚举设备	44
7.2 配置采集卡	44
7.3 单帧采集	51
7.4 连续采集	53
7.5 多帧采集	58
7.6 处理图像数据	62
8 配置参数	67
8.1 配置参数概览	67
8.2 配置参数详细说明	73
9 附录	139
9.1 设备类型常量列表	139
9.2 回调事件常量列表	139

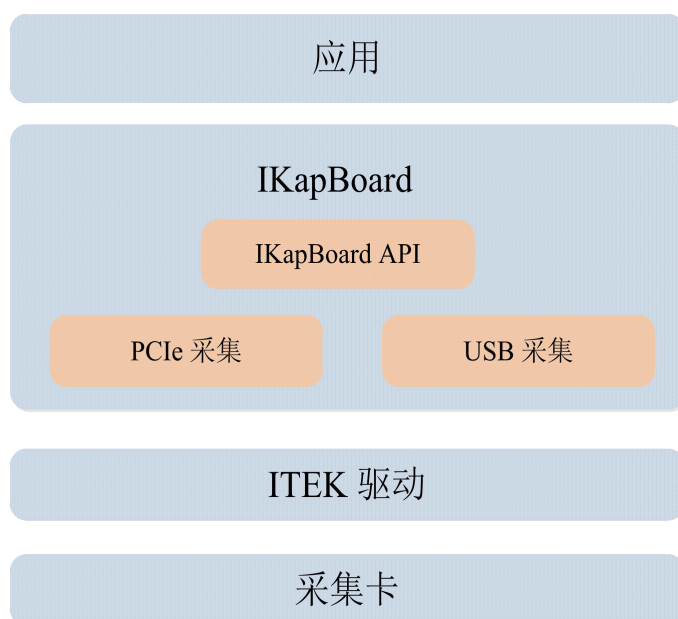
9.3 错误码类型常量列表	140
---------------------	-----

1 概述

1.1 关于 IKapBoard

IKapBoard 是合肥埃科光电科技股份有限公司针对 Vulcan-CL、Vulcan-CXP 和 Vulcan-SFP-2 系列采集卡开发的应用程序库。通过使用 IKapBoard，用户可以方便地管理和控制采集卡硬件设备，实现实时图像的采集和获取，从而简洁快速地进行高性能机器视觉应用的设计、开发和部署。IKapBoard 支持合肥埃科光电科技股份有限公司生产的所有的 CameraLink®、CoaXPress® 和 SFP® 接口采集卡。

IKapBoard 目前支持三种采集卡和计算机之间的数据传输方式：CL PCIe、CXP PCIe 和 SFP PCIe，用户可以依据实际应用需求选择合适的设备。无论通过 CL PCIe、CXP PCIe 还是 SFP PCIe 采集数据，IKapBoard 都提供了统一的编程接口。用户仅需要在开始时选择合适的设备型号进行初始化，无需关注设备的硬件差异，大大提高了应用开发的灵活性。



1.2 关于 IKapExpert

IKapExpert 是合肥埃科光电科技股份有限公司开发的一款基于 IKapBoard 库的高性能应用程序，具有图像采集、控制、处理和显示等功能。结合 Vulcan 系列采集卡，用户可以配置工业面扫描、线扫描相机。IKapExpert 支持对采集到的图像进行处理，包括：图像的放大和缩小、Bayer 图像转换、图像像素信息统计、直方图统计等。IKapExpert 同时还提供了相机状态和帧频等采集过程中的信息统计，便于用户进行机器视觉测试及部署。

IKapExpert 同时提供了 GUI 界面，用于配置相机的时序和控制参数。用户可以选择保存配置参数到指定文件中（*.vlcf），以便日后使用。

2 开始使用 IKapBoard

2.1 主要组成

- 如何使用 IKapBoard
- IKapBoard API 函数说明
- IKapBoard 使用范例

2.2 产品版本

与本文档对应的 Vulcan-CL 系列采集卡如下：

产品型号	总线	Camera Link 接口类型
Vulcan-CL PE1 Base	PCIe2.0/1.1 × 1	Base (MDR)
Vulcan-sCL PE4 DBase	PCIe2.0 × 4	Dual-Base (SDR)
Vulcan-sCL PE4 DBase Plus	PCIe2.0 × 4	Dual-Base (SDR)
Vulcan-CL PE4 Full	PCIe2.0 × 4	Full (MDR)
Vulcan-CL PE4 Full Pro	PCIe2.0 × 4	Full (MDR)
Vulcan-sCL PE4 Full	PCIe2.0 × 4	Full (SDR)
Vulcan-sCL PE4 Full Pro	PCIe2.0 × 4	Full (SDR)
Vulcan-sCL PE4 Full Plus	PCIe2.0 × 4	Full (SDR)

与本文档对应的 Vulcan-CXP 系列采集卡如下：

产品型号	总线	CoaXPress 接口类型
Vulcan-CXP6-2	PCIe Gen3 x 4	CXP6
Vulcan-CXP6	PCIe Gen3 x 4	CXP6
Vulcan-CXP6-8	PCIe Gen3 x 8	CXP6
Vulcan-CXP12-1	PCIe Gen3 x 4	CXP12
Vulcan-CXP12-2	PCIe Gen3 x 4	CXP12
Vulcan-CXP12	PCIe Gen3 x 8	CXP12

与本文档对应的 Vulcan-SFP-2 系列采集卡如下：

产品型号	总线	接口类型
Vulcan-SFP-2	PCIe Gen3 x 4	SFP

2.3 适用对象

本文档适用于以下工程师：

- 技术支持工程师
- 软件开发工程师

2.4 一般约定

文件名、目录以及网站以粗体文字表示（例如：**IKapExpert.exe**, **<http://www.i-tek.cn>**）。

函数参数以斜体文字表示（例如：*nHeight*）。

源码，示例代码以及文件列表以字符边框包含（例如：时间戳）。

2.5 开发环境

对于 32 位开发环境，支持下列编译平台：

- Microsoft Visual C++ 2008 (with Service Pack 1)
- Microsoft Visual C++ 2010 (with Service Pack 1)
- Microsoft Visual C++ 2012
- Microsoft Visual C++ 2013
- Microsoft Visual C++ 2015
- Microsoft Visual C++ 2017

对于 64 位开发环境，支持下列编译平台：

- Microsoft Visual C++ 2008 (with Service Pack 1)
- Microsoft Visual C++ 2010 (with Service Pack 1)
- Microsoft Visual C++ 2012
- Microsoft Visual C++ 2013
- Microsoft Visual C++ 2015
- Microsoft Visual C++ 2017

3 创建 IKapBoard 应用

3.1 头文件和库文件

创建 IKapBoard 应用所需的头文件和库文件在安装 IKapExpert 软件时会被自动安装在目标计算机上。用户可以在 IKapExpert 软件的安装位置找到相应的文件，具体如下表所示：

文件名	描述	位置
IKapBoard.h	Basic class header file	\ I-TEK OptoElectronics \ IKapLibrary \ Include
IKapBoard.lib	x64 basic library for Visual C++ version	\ I-TEK OptoElectronics \ IKapLibrary \ Lib \ x64
IKapBoard.lib	x86 basic library for Visual C++ version	\ I-TEK OptoElectronics \ IKapLibrary \ Lib \ x86
IKapBoard.dll	32bit system basic DLL for Visual C++ 2008 and up	<windir> \ System32
IKapBoard.dll	64bit system basic DLL for Visual C++ 2008 and up	<windir> \ SysWOW64

3.2 创建 PCIe 采集卡的 C 应用程序

下列步骤描述了如何在 VS2010 中创建一个 PCIe 采集卡的 C 应用程序：

- 在程序源代码中包含 **IKapBoard.h**（它会包含所有需要的头文件）
- 添加 **\$(ITEK_ROOT) \ Include** 或 **\$(IKAPC_ROOT) \ Include** 到 项目 - 属性 - 配置属性 - C/C++ - 常规 - 附加包含目录
- 如果是 x64 应用程序，添加 **\$(ITEK_ROOT) \ Lib \ x64 \ IKapBoard.lib** 或 **\$(IKAPC_ROOT) \ Lib \ x64 \ IKapBoard.lib** 到 项目 - 添加现有项
- 如果是 x86 应用程序，添加 **\$(ITEK_ROOT) \ Lib \ x86 \ IKapBoard.lib** 或 **\$(IKAPC_ROOT) \ Lib \ x86 \ IKapBoard.lib** 到 项目 - 添加现有项
- 项目 - 属性 - 配置属性 - C/C++ - 代码生成 - 运行库 选择 **多线程 DLL(/MD)** 或者 **多线程调试 DLL(/MDd)**

4 IKapBoard 使用流程

用户可以方便地使用 IKapBoard 库进行图像采集，其应用流程一般包括七个步骤：



4.1 初始化

在初始化的过程中，用户首先枚举连接到计算机上的采集卡设备，然后创建采集卡设备，打开相应的采集卡。

- 枚举计算机上的采集卡设备
 - **IKapGetBoardCount():** 此函数获取连接到计算机上的采集卡数量，可以通过设置参数分别枚举 PCIe 设备和 USB3.0 设备。
 - **IKapGetBoardName():** 此函数获取采集卡名称。
- 打开采集卡设备
 - **IKapOpen():** 此函数打开采集卡设备。

4.2 设置参数

采集卡需要的所有参数信息都在配置文件 (VLCF, *.vlcf) 中记录，用户可以打开 IKapExpert 应用软件设置相应的参数，然后保存到相应的配置文件中；也可以通过任意文本编辑软件打开配置文件，根据配置文件说明手动修改参数。当采集卡导入配置文件后，在首次采集时该配置信息才会生效。

- 导入采集卡配置文件
 - **IKapLoadConfigurationFromFile():** 将配置信息从配置文件中导入内存空间，配置文件是以 vlcf 为后缀的文件。
- 获取/设置采集卡参数
 - **IKapGetInfo():** 此函数获取采集卡参数信息。
 - **IKapSetInfo():** 此函数设置采集卡参数信息。

注意:

- (1) **IKapLoadConfigurationFromFile()** / **IKapSetInfo()** 设置的参数信息不会立即生效，这些配置信息将会在下次采集开始时生效。
- (2) 对于面扫描相机，单帧图像的尺寸不超过 128MB。
- (3) 对于线扫描相机，不限制扫描行数的数量。

4.3 设置图像缓冲区

设置相机传入图像数据的缓冲区，IKapBoard 以“帧”为单位管理图像缓冲区域。

帧的大小由相机输出决定，其值是图像宽度、图像高度和每个像素占据字节数目的乘积。每个像素占据的字节数由图像类型、数据格式和像素深度共同决定，详细信息如下表所示。

注意由于 IKapBoard 以字节为单位存储像素数据，当图像的数据格式超过 8bit 但是小于 16bit 时，IKapBoard 会以 16bit 存储数据。

相机			IKapBoard	
图像类型	数据格式	像素深度	采集卡数据位宽	每个像素占据的字节
0	8	8	8	1
0	10	10	16	2
0	12	12	16	2
0	14	14	16	2
0	16	16	16	2
1	8	24	24	3
1	10	30	48	6
1	12	36	48	6
1	16	48	48	6
2	8	32	32	4
2	10	40	64	8
2	12	48	64	8
2	14	56	64	8
2	16	64	64	8
3	8	24	24	3
3	10	30	48	6
3	12	36	48	6
3	14	42	48	6
3	16	48	48	6
4	8	32	32	4
4	10	40	64	8
4	12	48	64	8
4	14	56	64	8
4	16	64	64	8

表格说明:

- (1) Image Type 0 是灰度或 Bayer 图像，每个像素包含一个有效数据。
- (2) Image Type 1 是彩色 RGB 图像（真彩 RGB 相机），每个像素包含三个有效数据并按照

R=>G=>B 的顺序在内存排列，像素深度是数据格式的三倍。

- (3) Image Type 2 是彩色 RGBC 图像（真彩 RGBC 相机且包含图像灰度信息，其中 C 指示图像灰度信息），每个像素包含四个有效数据并按照 R=>G=>B=>C 的顺序在内存排列，像素深度是数据格式的四倍。
- (4) Image Type 3 是彩色 BGR 图像（真彩 BGR 相机），每个像素包含三个有效数据并按照 B=>G=>R 的顺序在内存排列，像素深度是数据格式的三倍。
- (5) Image Type 4 是彩色 BGRC 图像（真彩 BGRC 相机且包含图像灰度信息，其中 C 指示图像灰度信息），每个像素包含四个有效数据并按照 B=>G=>R=>C 的顺序在内存排列，像素深度是数据格式的四倍。

IKapBoard 默认维护一帧图像缓冲区，用户可以通过 **IKP_FRAME_COUNT** 参数设置多帧图像缓冲区。对于多帧图像缓冲区，IKapBoard 将会按照环形缓冲区的方式管理。关于图像缓冲区的详细描述，参见“[6 图像缓冲区的内存管理](#)”。

- 获取/设置图像帧大小

IKapGetInfo() / **IKapSetInfo()**用于获取和设置图像尺寸相关参数。图像尺寸和下列参数有关：

- **IKP_IMAGE_WIDTH**
- **IKP_IMAGE_HEIGHT**
- **IKP_IMAGE_TYPE**
- **IKP_DATA_FORMAT**
- **IKP_BOARD_BITS**

- 设置图像缓存

图像缓存的帧数由参数 **IKP_FRAME_COUNT** 指定，用户可以通过 **IKapGetInfo()** / **IKapSetInfo()**获取和设置。多帧图像缓存的传输模式参见“[6 图像缓冲区的内存管理](#)”。

注意：

- (1) 用户应该在设置 **IKP_FRAME_COUNT** 参数之前设置和图像尺寸有关的参数。
- (2) 用户应该校验设置 **IKP_FRAME_COUNT** 返回值。当系统没有足够资源申请内存时，IKapBoard 将可能无法正常工作。
- (3) 图像缓存区域由 IKapBoard 负责管理，用户不用释放申请的内存区域。

4.4 注册回调函数

在开始采集图像前，用户可以选择注册回调函数来对采集图像的过程进行控制，所有支持的事件都可以在“[9.2 回调事件常量列表](#)”中找到。

- 设置回调函数

IKapRegisterCallback()设置回调函数及对应的事件。

注意：

- (1) **IKapRegisterCallback()**函数应该在采集开始前调用。
- (2) **IKEvent_FrameReady** 事件对于线扫描相机和面扫描相机有不同的含义。对于面扫描相机，该事件在一帧图像采集完成后触发；对于线扫描相机，该事件在由 **IKP_IMAGE_HEIGHT** 指定行数采集完毕后触发。

- (3) 对于线扫描相机，用户可以通过 **IKapGetBufferStatus()**实时的查询当前有效行数来进行图像处理，不用等待 **IKEvent_FrameReady** 事件触发后再进行图像数据处理。
- (4) 相机断电、采集卡状态异常、停止采集都会导致图像停止采集，可以在停止采集图像阶段根据 **IKapGetLastError()**判断停止采集的原因。
- (5) 不要在回调函数进行太过费时的操作，这有可能导致相机最新采集的图像数据丢失。当相机采集的图像数据丢失时，**IKEvent_FrameLost** 事件会被触发。

4.5 采集图像

IKapBoard 可以选择连续采集图像或者采集有限帧的图像序列。当选择采集图像序列时，用户可以选择阻塞调用或者非阻塞调用；对于连续采集图像，始终以非阻塞方式进行工作。

- 开始采集
 - **IKapStartGrab()**函数启动连续采集图像或者采集有限帧的图像序列。
 - 第二个参数 *nFrameCount* 表示希望 IKapBoard 采集的帧数。
 - 如果 *nFrameCount* = 1, IKapBoard 会从相机中采集一帧图像；
 - 如果 *nFrameCount* = N 且 N>1, 则 IKapBoard 从相机中采集连续的 N 帧图像；
 - 如果 *nFrameCount* = 0, 则 IKapBoard 开始连续采集图像。
 - 每次调用 **IKapGrabStart()**都会从设置的缓冲区首帧开始顺序的写入数据。
- 图像序列的采集模式
 - **IKapSetInfo(HANDLE, IKP_GRAB_MODE, xxx)**可以用来设置采集图像序列的调用模式，用户可以选择阻塞调用或者非阻塞调用。
 - 在阻塞调用中，**IKapStartGrab()**将会一直等待直到由 *nFrameCount* 指定的所有图像采集完成或者采集超时。
 - 在非阻塞调用中，**IKapStartGrab()**将会立即返回而不等待所有图像数据传输完成。
- 停止采集
 - **IKapStopGrab()**停止采集数据，无论当前是否有正在进行的数据传输。

注意：

- (1) 每次调用 **IKapStartGrab()**都会清空当前缓冲区。
- (2) 对于尺寸较大的图像可能需要较多时间完成图像采集，可以通过设置 **IKP_TIME_OUT** 保证图像完整传输。
- (3) 不要在采集图像的过程中设置和图像尺寸有关的参数，这可能导致软件运行错误。
 - **IKP_IMAGE_WIDTH**
 - **IKP_IMAGE_HEIGHT**
 - **IKP_IMAGE_TYPE**
 - **IKP_DATA_FORMAT**
 - **IKP_FRAME_COUNT**

4.6 处理图像

图像采集开始后，图像数据将会存入预先设置的图像缓冲区内，用户可以通过获取该图像数据地址和状态进而进行图像处理工作。图像缓冲区工作原理，参见“[6 图像缓冲区的内存管理](#)”。

- 获取图像缓冲区域首地址
 - **IKapGetBufferAddress()**：此函数用来获取图像缓冲区的内存地址。

- 获取图像缓冲区域状态
 - **IKapGetBufferStatus()**: 此函数用来获取图像缓冲区的状态，只有当图像缓冲区为满时，该缓冲区的数据才为有效。关于图像缓冲区的状态切换，参见“[6 图像缓冲区的内存管理](#)”。

注意:

- (1) 单帧图像的大小可以通过参数 **IKP_FRAME_SIZE** 进行访问，用户对于图像缓冲区的操作不能超过由 **IKP_FRAME_SIZE** 指定的范围。
- (2) 图像缓冲区状态可以在空/满/传输之间切换，关于缓冲区的状态切换，参见“[6 图像缓冲区的内存管理](#)”。
- (3) 关于处理灰度图像、彩色 RGB 图像和彩色 RGBC 图像的具体方法，参见“[7 IKapBoard 使用范例](#)”。

4.7 结束采集

当采集图像完毕后，应该结束采集过程、关闭采集卡设备、销毁申请的系统资源。请按照下列步骤顺序执行来结束采集过程并释放采集卡资源。

- 结束采集过程
 - **IKapStopGrab()**: 此函数将停止正在进行的采集过程。
- 清除回调函数
 - **IKapUnRegisterCallback()**: 此函数用于清除注册的回调函数。
- 关闭采集卡设备
 - **IKapClose()**: 此函数关闭已经打开的采集卡设备。

5 API 函数说明

5.1 函数概览

函数名	概要
<u>IKapGetBoardCount</u>	获取连接到计算机上的采集卡数量
<u>IKapGetBoardName</u>	获取采集卡名称
<u>IKapOpen</u>	打开采集卡
<u>IKapOpenCXP</u>	打开 CXP 采集卡
<u>IKapOpenGVB</u>	打开 GVB 采集卡
<u>IKapClose</u>	关闭采集卡
<u>IKapLoadConfigurationFromFile</u>	导入配置文件
<u>IKapSaveConfigurationToFile</u>	保存配置文件
<u>IKapGetInfo</u>	获取配置参数
<u>IKapSetInfo</u>	设置配置参数
<u>IKapRegisterCallback</u>	注册回调函数
<u>IKapUnRegisterCallback</u>	清空回调函数
<u>IKapStartGrab</u>	开始采集图像
<u>IKapStopGrab</u>	停止采集图像
<u>IKapWaitGrab</u>	等待采集结束
<u>IKapClearGrab</u>	清空当前采集状态
<u>IKapGetBufferAddress</u>	获取缓冲区地址
<u>IKapSetBufferAddress</u>	设置缓冲区地址
<u>IKapGetBufferBusAddress</u>	获取图像缓冲区物理总线映射地址
<u>IKapSetBufferBusAddress</u>	设置图像缓冲区物理总线映射地址
<u>IKapGetBufferStatus</u>	获取缓冲区状态
<u>IKapGetBufferStatusEx</u>	获取更多图像缓冲区状态
<u>IKapReleaseBuffer</u>	释放缓冲区
<u>IKapGetCameraStatus</u>	获取相机状态
<u>IKapGetLastError</u>	获取错误码
<u>IKapGetSerialPort</u>	获取采集卡设备绑定的串口号
<u>IKapGetFrameRate</u>	获取图像传输的帧率
<u>IKapReadCXPUart</u>	读取 CXP 串口数据
<u>IKapWriteCXPUart</u>	写入 CXP 串口数据
<u>IKapWaitCXPUart</u>	等待 CXP 有效数据
<u>IKapSaveBuffer</u>	保存缓冲区到指定位置
<u>IKapLoadBuffer</u>	从指定位置加载缓冲区
<u>IKapWriteRegister</u>	采集卡寄存器直接写入
<u>IKapReadRegister</u>	采集卡寄存器直接读取

5.2 函数详细说明

IKapGetBoardCount 获取连接到计算机上的采集卡数量

C++/C	int IKapGetBoardCount(unsigned int resourceType, unsigned int* resourceCount);	
参数说明	resourceType	查询的资源类型 0: 连接到计算机上的所有类型采集卡 1: USB3.0 采集卡 2: PCIe 采集卡
	resourceCount	查询采集卡设备的数量
返回值	0: 调用失败 1: 调用成功	
说明	查询当前连接到主机上的某一类型采集卡的数量。 对于同一类型的采集卡，其序号从 0 开始，并且依次递增。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_BoardNotFound	设备未发现
	IKStatus_AllocMemoryFail	申请内存失败
	IKStatus_InvalidParameter	无效的参数
注意事项	如果没有查询类型的采集卡连接到计算机上，则返回 0。	

示例代码

```
// 获取当前连接到计算机上的所有类型的采集卡数量
IKapGetBoardCount( IKBoardALL, &nResourceCount );
printf("Enum Device Number : %u\n", nResourceCount);

// 获取当前连接到计算机上的 PCIe 类型的采集卡数量
IKapGetBoardCount( IKBoardPCIe, &nPCIEResourceCount );
printf("Enum PCIE Device Number: %u\n\n", nPCIEResourceCount);

// 获取当前连接到计算机上的 USB3.0 类型的采集卡数量
IKapGetBoardCount( IKBoardUSB30, &nUSBResourceCount);
printf("Enum USB Device Number: %u\n\n", nUSBResourceCount);
```

IKapGetBoardName 获取采集卡名称

C++/C	int IKapGetBoardName(unsigned int resourceType, unsigned int resourceIndex, char* resourceName, unsigned int* resourceNameSize);	
参数说明	resourceType	查询的资源类型 1: USB3.0 采集卡

		2: PCIe 采集卡
	resourceIndex	查询的资源序号，有效范围从 0 到(n-1)。其中 n 是通过 IKapGetBoardCount() 获得的 <i>resourceCount</i> 值
	resourceName	查询的资源名称，以\0 结尾
	resourceNameSize	作为输入时，指明的是输入缓冲区 <i>resourceName</i> 的最大长度；作为输出时，指明的是实际获取的名字的有效长度。当输入缓冲区过小时，函数调用将会失败，此时会返回 IKStatus_BufferTooSmall 错误代码并在 <i>resourceNameSize</i> 中存放采集卡名称的缓冲区大小
返回值	0: 调用失败 1: 调用成功	
说明	当 <i>resourceName</i> 为空时，函数调用会失败，并且返回 IKStatus_InvalidParameter 。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_BoardNotFound	设备未发现
	IKStatus_BufferTooSmall	输入参数缓存太小
	IKStatus_InvalidParameter	无效的参数
注意事项	采集卡序号 <i>resourceIndex</i> 从 0 开始递增，最大值为 n-1。其中 n 是通过 IKapGetBoardCount() 获得的 <i>nResourceCount</i> 。	
	<i>resourceNameSize</i> 作为输入参数指明 <i>resourceName</i> 的输入大小，作为输出参数指明 <i>resourceName</i> 的输出有效缓存大小	
相关函数	IKapGetBoardCount()	

示例代码

```
char* resourceName = NULL;
resourceNameSize = 0;
IKapGetBoardName( IBoardPCIE, i, resourceName, &resourceNameSize );
IKapGetLastError( &pIKErrInfo, true );
if ( pIKErrInfo.uErrorCode == IKStatus_BufferTooSmall )
{
    resourceName = new char[resourceNameSize];
    IKapGetBoardName( IBoardPCIE, i, resourceName, &resourceNameSize );
}
IKapGetLastError( &pIKErrInfo, true );
if ( pIKErrInfo.uErrorCode != IKStatus_Success )
    printf("Last Error: %08X\n", pIKErrInfo.uErrorCode );
else
    printf("PCIe Device-%u Name: %s\nDevice\n\n",i, resourceName );
```

```
if (resourceName != NULL)
delete[] resourceName;
```

IKapOpen 打开采集卡

C++/C	HANDLE IKapOpen(unsigned int resourceType, unsigned int resourceIndex);	
参数说明	resourceType	打开的采集卡类型 1: USB3.0 采集卡 2: PCIe 采集卡
	resourceIndex	从 0 开始的采集卡序号。对于 n 个连接到系统上相同类型的采集卡，resourceIndex 有效范围是 0 ~ (n-1)
返回值	INVALID_HANDLE_VALUE (-1): 调用失败 设备句柄: 调用成功	
说明	本函数获取指定类型和序号的采集卡的设备句柄，同时初始化采集卡设备。 如果在一个进程中多次打开相同类型和序号的采集卡，只有本函数的第一次调用会被执行；后续的调用会返回打开失败错误。同样的，如果在多个进程中打开相同类型和序号的采集卡，也只有第一个调用本函数的进程会被执行。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_InvalidParameter	无效的参数
	IKStatus_AllocMemoryFail	申请内存失败
	IKStatus_BoardNotFound	设备不存在
	IKStatus_Mutex_Locked	由于设备被占用导致无法打开
	IKStatus_Invalid_Mutex	无效的设备句柄
	IKStatus_WinError	Windows 系统错误，通过 Windows 系统函数 GetLastError() 获取系统错误信息
注意事项	本函数执行成功返回有效的设备句柄，执行失败返回-1。	
相关函数	IKapClose()	

示例代码

```
HANDLE hDev = IKapOpen( IKBoardPCIE, 0 );
```

IKapOpenCXP 打开 CXP 采集卡

C++/C	HANDLE IKapOpenCXP(unsigned int resourceType , unsigned int resourceIndex, IKAP_CXP_BOARD_INFO info);	
参数说明	resourceType	打开的采集卡类型 1: USB3.0 采集卡 2: PCIe 采集卡

	resourceIndex	从 0 开始的采集卡序号。对于 n 个连接到系统上相同类型的采集卡， <i>resourceIndex</i> 有效范围是 0 ~ (n-1)
	info	CXP 采集卡设备信息
返 回 值	INVALID_HANDLE_VALUE (-1): 调用失败 设备句柄: 调用成功	
说 明	本函数获取指定类型和序号的采集卡的设备句柄，同时初始化采集卡设备。 如果在一个进程中多次打开相同类型和序号的采集卡，只有本函数的第一次调用会被执行；后续的调用会返回打开失败错误。同样的，如果在多个进程中打开相同类型和序号的采集卡，也只有第一个调用本函数的进程会被执行。	
关联文件	IKapBoard.h	
关 联 库	IKapBoard.lib	
错 误 码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_InvalidParameter	无效的参数
	IKStatus_AllocMemoryFail	申请内存失败
	IKStatus_BoardNotFound	设备不存在
	IKStatus_Mutex_Locked	由于设备被占用导致无法打开
	IKStatus_Invalid_Mutex	无效的设备句柄
	IKStatus_WinError	Windows 系统错误，通过 Windows 系统函数 GetLastError() 获取系统错误信息
注意事项	本函数执行成功返回有效的设备句柄，执行失败返回-1。	
相关函数	IKapClose()	

示例代码

无

IKapOpenGVB 打开 GVB 采集卡

C++/C	HANDLE IKapOpenGVB(unsigned resourceType, unsigned resourceIndex, IKAP_GVB_BOARD_INFO info);	
参数说明	resourceType	打开的采集卡类型 1: USB3.0 采集卡 2: PCIe 采集卡
	resourceIndex	从 0 开始的采集卡序号。对于 n 个连接到系统上相同类型的采集卡， <i>resourceIndex</i> 有效范围是 0 ~ (n-1)
	info	GVB 采集卡设备信息
返 回 值	INVALID_HANDLE_VALUE (-1): 调用失败 设备句柄: 调用成功	
说 明	本函数获取指定类型和序号的采集卡的设备句柄，同时初始化采集卡设备。 如果在一个进程中多次打开相同类型和序号的采集卡，只有本函数的第一次调用会被执行；后续的调用会返回打开失败错误。同样的，如果在多个进程中打开相	

	同类型和序号的采集卡，也只有第一个调用本函数的进程会被执行。	
关联文件	IKapBoard.h	
关 联 库	IKapBoard.lib	
错 误 码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_InvalidParameter	无效的参数
	IKStatus_AllocMemoryFail	申请内存失败
	IKStatus_BoardNotFound	设备不存在
	IKStatus_Mutex_Locked	由于设备被占用导致无法打开
	IKStatus_Invalid_Mutex	无效的设备句柄
	IKStatus_WinError	Windows 系统错误，通过 Windows 系统函数 GetLastError() 获取系统错误信息
注意事项	本函数执行成功返回有效的设备句柄，执行失败返回-1。	
相关函数	IKapClose()	

示例代码

无

IKapClose 关闭采集卡

C++/C	int IKapClose(HANDLE hDev);	
参数说明	hDev	采集卡设备句柄
返 回 值	0: 调用失败 1: 调用成功	
说 明	关闭已经打开的采集卡设备。如果采集卡设备已经关闭或者没有打开，该函数不会重复关闭设备。	
关联文件	IKapBoard.h	
关 联 库	IKapBoard.lib	
错 误 码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_BoardNotFound	设备不存在
	IKStatus_WinError	Windows 系统错误，通过 Windows 系统函数 GetLastError() 获取系统错误信息
	IKStatus_Invalid_Handle	无效的设备句柄
相关函数	IKapOpen()	

示例代码

```
HANDLE hDev = IKapOpen( IKBoardPCIE, 0 );
IKapClose( hDev);
```

IKapLoadConfigurationFromFile 导入配置文件

C++/C	int IKapLoadConfigurationFromFile(HANDLE hDev, char *lpFileName);
-------	--

参数说明	hDev	设备句柄
	lpFileName	指明采集卡配置路径，注意所有采集卡配置文件都以.vlcf 为后缀
返回值	0: 调用失败 1: 调用成功	
说明	读取由 <i>lpFileName</i> 指定路径的文件, 该文件是一个以.vlcf 为后缀的采集卡配置文件. 采集卡配置文件可以通过 IKapExpert 软件中菜单的保存功能生成, 也可以由用户通过任意文档编辑工具手动修改生成。 本函数必须要指明配置文件的全部路径。 本函数同时会校验导入配置文件的参数, 如果设置参数错误则函数会返回 0。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败, 可调用 IKapGetLastError() 返回错误码。	
	IKStatus_ConfigFilePathInvalid	采集卡配置文件路径无效
	IKStatus_Invalid_Handle	无效的设备句柄
	IKStatus_ConfigParameterInvalid	无效的配置参数
注意事项	注意不要在采集图像的过程中调用该函数。	
相关函数	IKapSaveConfigurationToFile()	

示例代码

```
HANDLE hDev = IKapOpen(IKBoardPCIE, 0);
IKapLoadConfigurationFromFile( hDev, "Default.vlcf" );
```

IKapSaveConfigurationToFile 保存配置文件

C++/C	int IKapSaveConfigurationToFile(HANDLE hDev, char* lpFileName);	
参数说明	hDev	设备句柄
	lpFileName	指明采集卡配置路径，注意所有采集卡配置文件都以.vlcf 为后缀
返回值	0: 调用失败 1: 调用成功	
说明	保存当前采集卡的配置参数到指定文件中。 必须要指明配置文件的全部路径。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败, 可调用 IKapGetLastError() 返回错误码。	
	IKStatus_ConfigFilePathInvalid	采集卡配置文件路径无效
	IKStatus_Invalid_Handle	无效的设备句柄
注意事项	无	
相关函数	IKapLoadConfigurationFromFile()	

示例代码

```
HANDLE hDev = IKapOpen(IKBoardPCIE, 0);
IKapSaveConfigurationToFile( hDev, "Default.vlcf");
```

IKapGetInfo 获取配置参数

C++/C	int IKapGetInfo(HANDLE hDev, unsigned int uType, int* npValue);	
参数说明	hDev	设备句柄
	uType	获取值的 ID
	npValue	<i>uType</i> 指定 ID 的获取值
返回值	0: 调用失败 1: 调用成功	
说明	本函数获取由 <i>uType</i> 指定参数 ID 的值。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_Invalid_Handle	无效的设备句柄
	IKStatus_InvalidParameter	无效的参数
注意事项	无	
相关函数	IKapSetInfo()	

示例代码

```
HANDLE hDev = IKapOpen(IKBoardPCIE, 0);
// 获取采集超时时间的设定值，单位 ms
int nTimeout;
IKapGetInfo(hDev, IKP_TIME_OUT, &nTimeout);
```

IKapSetInfo 设置配置参数

C++/C	int IKapSetInfo(HANDLE hDev, unsigned int uType, int nValue);	
参数说明	hDev	设备句柄
	uType	设定值的 ID
	nValue	设定值
返回值	0: 调用失败 1: 调用成功	
说明	本函数设置由 <i>uType</i> 指定 ID 的参数值。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_BoardNotOpen	采集卡设备未打开

	IKStatus_Invalid_Handle	无效的设备句柄
	IKStatus_Insufficient_Resource	资源不足
注意事项	注意不要在图像传输的过程中调用该函数，可能导致系统错误。	
相关函数	IKapGetInfo()	

示例代码

```
HANDLE hDev = IKapOpen(IKBoardPCIE, 0);
// 设置采集时间为 5000ms
IKapSetInfo(hDev, IKP_TIME_OUT, 5000);
```

IKapRegisterCallback 注册回调函数

C++/C	int IKapRegisterCallback(HANDLE hDev, unsigned int uEventType, HookFnPtr fEventFunc, void* pParam);	
参数说明	hDev	设备句柄
	uEventType	<p>0: 当开始连续采集图像时触发该回调函数</p> <p>1: 在连续采集图像的过程中，图像缓冲区域有一帧或者超过一帧完整图像被采集完毕时触发该回调函数</p> <p>2: 当连续采集图像停止时触发该回调函数</p> <p>3: 当图像缓冲区溢出导致图像丢失时触发该回调函数</p> <p>4: 当图像采集超时触发该回调函数</p> <p>5: 当相机时钟信号从无效变为有效时触发该回调函数</p> <p>6: 当相机时钟信号从有效变为无效时触发该回调函数</p> <p>7: 当采集卡工作在外触发模式，外部触发信号被采集卡丢失时触发该回调函数。该回调函数可能在外触发频率快于采集帧率的情况下发生</p> <p>8: 当 CXP 采集卡发生过流保护时调用该函数</p> <p>9: 当 CXP 采集卡发生 CRC 错误时调用该函数</p> <p>10: 当 CXP 采集卡发生传输错误时调用该函数</p> <p>0x00100000+n: 当第 n 行图像从相机采集到采集卡时触发该回调函数，注意此时图像数据可能尚未传入到用户指定内存中，只适用于线扫描相机，最多指定 128 个行进行标记</p> <p>0x00200000+n: 当第 n 行图像从采集卡采集到电脑内存空间时触发该回调函数，只适用于线扫描相机，最多指定 128 个行进行标记</p> <p>0x00400000+n: 当检测到输入信号 n 的下降沿触发回调函数</p> <p>0x00800000+n: 当检测到输入信号 n 的上升沿触发回调函数</p> <p>0x01000000: 当采集卡开始一帧图像数据时触发该回调函数</p> <p>0x02000000: 当采集卡结束一帧图像数据时触发该回调函数</p> <p>0x04000000+n: 当第 n 各个缓冲区数据传输完成时触发该回调函数，缓冲区索引从 0 开始计数</p>
	fEventFunc	注册的回调函数指针

	pParam	通用参数
返回值	0: 调用失败 1: 调用成功	
说明	<p>在开始和结束采集一帧图像时，都会调用相应的回调函数。用户可以自己定义在回调函数中的功能，比如保存图像数据，但是请不要在回调函数中进行过于费时的操作，这有可能导致缓冲区中的图像数据被新采集的数据覆盖。</p> <p>如果多次设置回调函数，仅保证最新设置的回调函数会被调用。</p> <p>回调函数定义如下：</p> <p>void CALLBACK SampleFunc(void* pParam); <i>pParam</i>: IKapRegisterCallback()注册回调函数时的第四个参数。</p> <p>注意：由于 <i>pParam</i> 是用户申请内存地址的指针，因此如果在已经释放该内存的情况下，在回调函数中继续使用该函数可能会导致内存越界错误。</p> <p>支持边沿检测的输入信号：</p> <p>0x01: 内部触发信号 0x02: 通用输入信号 1 0x04: 通用输入信号 2 0x08: 编码器输入信号 A 相 0x10: 编码器输入信号 B 相 0x20: 板间同步输入信号 0x40: 积分信号 1 0x80: 积分信号 2</p>	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_Invalid_Handle	无效的句柄
	IKStatus_BoardNotOpen	采集卡设备未打开
注意事项	IKStatus_TimeOut	操作超时
	不要在回调函数中进行过于费时的操作，有可能导致缓冲区中的图像数据被新采集的数据覆盖或者图像数据丢失。	
	如果多次设置回调函数，仅保证最新设置的回调函数会被调用。	
相关函数	回调函数的形参必须和定义保持一致。	
	采集图像结束后请调用 IKapUnRegisterCallback() 清除回调函数。	
相关函数	IKapUnRegisterCallback()	

示例代码

```
// 回调函数，在连续采集开始时调用
void CALLBACK OnGrabStart(void *Context)
{
    printf("Start grabbing image(continuous)\n");
}
HANDLE hDev = IKapOpen( IKBoardPCIE, 0);

// 注册回调函数
```

```
if ( IKapRegisterCallback( hDev, IKeEvent_GrabStart, OnGrabStart, hDev ) == false )
{
    ErrorMessage("Regist Callback", "IKeEvent_GrabStart");
    goto FreeHandle;
}
```

IKapUnRegisterCallback 清除回调函数

C++/C	int IKapUnRegisterCallback (HANDLE hDev, unsigned int uEventType);	
参数说明	hDev	设备句柄
	uEventType	参见 IKapRegisterCallback()
返回值	0: 调用失败 1: 调用成功	
说明	当停止采集后，需要调用该函数来清除回调函数。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError()返回错误码。	
	IKStatus_Invalid_Handle	无效的句柄
	IKStatus_BoardNotOpen	采集卡设备未打开
注意事项	IKStatus_TimeOut	操作超时
	如果多次设置回调函数，仅保证最新设置的回调函数会被调用。	
相关函数	请在停止连续采集图像后删除所有设置的回调函数。	
	IKapRegisterCallback()	

示例代码

```
// 回调函数，在连续采集开始时调用
void CALLBACK OnGrabStart(void *Context)
{
    printf("Start grabbing image(continuous)\n");
}
HANDLE hDev = IKapOpen( IKBoardPCIE, 0 );

// 注册回调函数
if ( IKapRegisterCallback( hDev, IKeEvent_GrabStart, OnGrabStart, hDev ) == false )
{
    ErrorMessage("Regist Callback", "IKeEvent_GrabStart");
    goto FreeHandle;
}

// 清除回调函数
IKapUnRegisterCallback ( hDev, IKeEvent_GrabStart );
```

IKapStartGrab 开始采集图像

C++/C	int IKapStartGrab (HANDLE hDev, int nFrameCount);	
参数说明	hDev	设备句柄
	nFrameCount	0: 连续采集图像 1: 单次采集 2 – 2147483647: 采集图像序列
返回值	0: 调用失败 1: 调用成功	
说明	调用该函数，采集卡将根据当前导入的配置文件信息采集图像序列或者连续采集图像。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_Invalid_Handle	无效的设备句柄
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_TimeOut	操作超时
	IKStatus_GrabPending	当前有未完成的单次采集或者连续采集过程
注意事项	请确保在采集图像前，采集卡设备已经打开，配置参数已经导入。	
	对于较大尺寸的图像，可能需要较长的时间完成一次图像采集。可以通过增长采集时间保证一帧完整图像的采集。	
	配置参数不合理，相机状态异常都会导致采集失败。	
	每次调用该函数都会清空原有的图像缓冲数据。	
	如果单次采集图像（ <i>nFrameCount</i> =1），本函数仅会采集一帧图像，如果 IKapStopGrab() 在尚未采集完成前调用，则会终止当前的采集；	
	如果是采集图像序列（ <i>nFrameCount</i> >=2），本函数将会采集由 <i>nFrameCount</i> 指定的图像序列，如果 IKapStopGrab() 在尚未采集完成前调用，则会终止当前的采集；	
	如果是连续采集（ <i>nFrameCount</i> =0），本函数将会连续采集图像直到用户调用 IKapStopGrab() 。	
	当调用该函数时，IKapBoard 始终从缓冲区的第一帧开始写入区域并清空以前写入的数据。	
	设置缓冲区有效帧数为 4（ IKP_FRAME_COUNT ），采集到的图像将按照下面顺序排列：	
	单次采集时(<i>nFrameCount</i> =1): 「1→1→1→1→...」 序列采集时(<i>nFrameCount</i> =2): 「(1→2)→(1→2)→...」 序列采集时(<i>nFrameCount</i> =6): 「(1→2→3→4→1→2)→(1→2→3→4→1→2)→...」 连续采集时(<i>nFrameCount</i> =0): 「(1→2→3→4)→(1→2→3→4)→...」	
相关函数	IKapOpen() / IKapStopGrab()	

示例代码

参见“[7 IKapBoard 使用范例](#)”

IKapStopGrab 停止采集图像

C++/C	int IKapStopGrab (HANDLE hDev);	
参数说明	hDev	设备句柄
返回值	0: 调用失败 1: 调用成功	
说明	停止正在进行的连续采集或者单次采集。如果当前并没有正在进行的采样，则该函数不会产生作用。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	IKStatus_Invalid_Handle	无效的设备句柄
	IKStatus_BoardNotOpen	采集卡设备未打开
注意事项	请确保采集前，采集卡设备已经打开，配置参数已经导入。	
	请在停止图像数据采集时调用该函数用来停止采集。	
	停止采集图像后可以调用 IKapUnRegisterCallback()清除设置的回调函数。	
相关函数	IKapStartGrab()	

示例代码

参见“[7 IKapBoard 使用范例](#)”

IKapWaitGrab 等待采集结束

C++/C	int IKapWaitGrab (HANDLE hDev);	
参数说明	hDev	设备句柄
返回值	0: 调用失败 1: 调用成功	
说明	用户调用 IKapStartGrab()以非阻塞模式采集图像序列时($nFrameCount > 0$)，调用该函数将会等待异步采集操作结束或者采集超时。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	IKStatus_Invalid_Handle	无效的设备句柄
	IKStatus_BoardNotOpen	采集卡设备未打开
注意事项	请确保采集前，采集卡设备已经打开，配置参数已经导入。	
	如果采集超时，该函数会返回 0 且设置错误码为 IKStatus_Timeout。	
相关函数	IKapStartGrab()	

示例代码

无

IKapClearGrab 清空当前采集状态

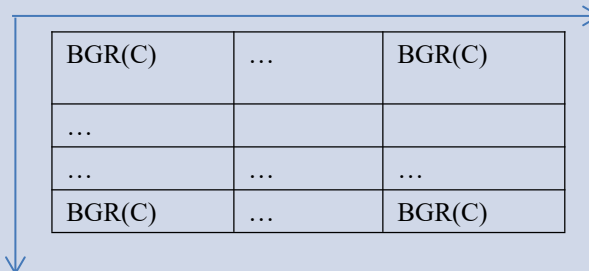
C++/C	int IKapClearGrab(HANDLE hDev);	
参数说明	hDev	设备句柄
返回值	0: 调用失败 1: 调用成功	
说明	当采集卡处于采集图像的过程时，调用该函数将会清除所有已经采集到的缓冲区图像数据并复位采集状态。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_Invalid_Handle	无效的设备句柄
	IKStatus_BoardNotOpen	采集卡设备未打开
注意事项	该函数会清除 PC 缓冲区和 FPGA 中的图像缓存数据，保证在确实需要清空当前缓冲区的情况下执行该 API。	
相关函数	IKapStartGrab()	

示例代码

无

IKapGetBufferAddress 获取缓冲区地址

C++/C	int IKapGetBufferAddress(HANDLE hDev, int nFrameNum, void** pAddress);	
参数说明	hDev	设备句柄
	nFrameNum	图像缓冲区索引，有效范围是 [0 ~ IKP_FRAME_COUNT - 1]
	pAddress	图像缓冲区首地址
返回值	0: 调用失败 1: 调用成功	
说明	当相机采集图像数据到图像缓冲区后，如果需要访问采集的内容，则应该调用该函数获取图像缓冲区首地址。注意访问地址的界限不能超过 (<i>pAddress</i> + IKP_FRAME_SIZE)。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_Invalid_Handle	无效的句柄
注意事项	像素排列方式：从左上角到右下角依次递增。	
	对于数据格式是 8bit 的灰度或者 Bayer 图像，每个像素的每个通道仅需要一个字节的缓存区域保存。	
	对于数据格式大于 8bit 的灰度或者 Bayer 图像，每个像素的每个通道需要两个字	

	节的缓存区区域保存。
	对于彩色图像，各通道数值按照如下形式存储：
	
	如果图像像素深度为 8bit，则图像缓冲区域中的每个字节对应一个像素点；如果图像像素深度超过 8bit，则相邻的两个字节拼凑成一个像素的像素值，像素的有效数据采用 little-endian 方式排列。
相关函数	IKapGetInfo()

示例代码
无

IKapSetBufferAddress 设置缓冲区地址

C++/C	int IKapSetBufferAddress(HANDLE hDev, int nFrameNum, void* pAddress);	
参数说明	hDev	设备句柄
	nFrameNum	图像缓冲区索引，有效范围是 [0~IKP_FRAME_COUNT - 1]
	pAddress	图像缓冲区首地址
返回值	0：调用失败 1：调用成功	
说明	设置图像数据写入的首地址，当采集卡采集到图像数据后将会直接写入该用户缓冲区内。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_Invalid_Handle	无效的句柄
	IKStatus_InvalidParameter	<i>nFrameNum</i> 索引越界
注意事项	像素排列方式：从左上角到右下角依次递增。	
	对于数据格式是 8bit 的灰度或者 Bayer 图像，每个像素的每个通道仅需要一个字节的缓存区域保存。	
	对于数据格式大于 8bit 的灰度或者 Bayer 图像，每个像素的每个通道需要两个字节的缓存区区域保存。	
	如果图像像素深度为 8bit，则图像缓冲区域中的每个字节对应一个像素点；如果图像像素深度超过 8bit，则相邻的两个字节拼凑成一个像素的像素值，像素的有效数	

	据采用 little-endian 方式排列。
相关函数	IKapGetInfo()

示例代码
无

IKapGetBufferBusAddress 获取图像缓冲区物理总线映射地址

C++/C	int IKapGetBufferBusAddress(HANDLE hDev, int nFrameNum, PIKAP_BUS_MEMORY pBusAddress);	
参数说明	hDev	设备句柄
	nFrameNum	图 像 缓 冲 区 索 引 ， 有 效 范 围 是 [0~IKP_FRAME_COUNT-1]
	pBusAddress	物理总线地址
返 回 值	0: 调用失败 1: 调用成功	
说 明	获取图像缓冲区物理总线映射地址，仅针对 CXP 采集卡。	
关联文件	IKapBoard.h	
关 联 库	IKapBoard.lib	
错 误 码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_Invalid_Handle	无效的句柄
注意事项	无	
相关函数	IKapSetBufferBusAddress()	

示例代码
无

IKapSetBufferBusAddress 设置图像缓冲区物理总线映射地址

C++/C	int IKapSetBufferBusAddress(HANDLE hDev, int nFrameNum, IKAP_BUS_MEMORY BusAddress);	
参数说明	hDev	设备句柄
	nFrameNum	图 像 缓 冲 区 索 引 ， 有 效 范 围 是 [0 ~ IKP_FRAME_COUNT - 1]
	BusAddress	物理总线地址
返 回 值	0: 调用失败 1: 调用成功	
说 明	设置图像缓冲区物理总线映射地址，仅针对 CXP 采集卡。	
关联文件	IKapBoard.h	

关 联 库	IKapBoard.lib	
错 误 码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_Invalid_Handle	无效的句柄
	IKStatus_InvalidParameter	<i>nFrameNum</i> 索引越界
注意事项	无	
相关函数	IKapGetBufferBusAddress()	

示例代码

无

IKapGetBufferStatus 获取缓冲区状态

C++/C	int IKapGetBufferStatus(HANDLE hDev, int mFrameNum, PIKAPBUFFERSTATUS pIKapBufferStatus);	
参数说明	hDev	设备句柄
	nFrameNum	图 像 缓 冲 区 索 引，有效范围是 [0 ~ IKP_FRAME_COUNT - 1]
	pIKapBufferStatus	图像缓冲区状态
返 回 值	0: 调用失败 1: 调用成功	
说 明	<p>当单次采集或者连续采集的过程中，可以调用该函数获取图像缓冲区的状态。 图像缓冲区状态结构体定义如下：</p> <pre>typedef struct { unsigned uFull; unsigned uEmpty; unsigned uTransfer; unsigned uOverflow; unsigned uLineNum; } IKAPBUFFERSTATUS, *PIKAPBUFFERSTATUS;</pre> <p>参数说明： <i>uFull</i>: 指明缓冲区是否为满 <i>uEmpty</i>: 指明缓冲区是否为空 <i>uTransfer</i>: 指明缓冲区是否正在传输 <i>uOverflow</i>: 指明缓冲区是否越界 <i>uLineNum</i>: 对于线扫描相机，该参数指明当前传输的图像有效行数；对于面扫描相机，该参数始终为 0</p>	
关联文件	IKapBoard.h	
关 联 库	IKapBoard.lib	
错 误 码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_Invalid_Handle	无效的句柄

注意事项	无
相关函数	IKapOpen(); IKapLoadConfigurationFromFile(); IKapSetInfo()

示例代码
无

IKapGetBufferStatusEx 获取更多图像缓冲区状态

C++/C	int IKapGetBufferStatusEx(HANDLE hDev, int nFrameNum, PIKAPBUFFERSTATUSEX pIKapBufferStatusEx);	
参数说明	hDev	设备句柄
	nFrameNum	图像缓冲区索引，有效范围是 [0 ~ IKP_FRAME_COUNT - 1]
	pIKapBufferStatusEx	图像缓冲区状态
返回值	0: 调用失败 1: 调用成功	
说明	<p>当单次采集或者连续采集的过程中，可以调用该函数获取图像缓冲区的状态。图像缓冲区状态结构体定义如下：</p> <pre>typedef struct { unsigned uFull; unsigned uEmpty; unsigned uTransfer; unsigned uOverflow; unsigned uLineNum; unsigned long long uCompressSize; unsigned long long uTimestamp; unsigned Reserved[62]; } IKAPBUFFERSTATUSEX, *PIKAPBUFFERSTATUSEX;</pre> <p>参数说明：</p> <p><i>uFull</i>: 指明缓冲区是否为满</p> <p><i>uEmpty</i>: 指明缓冲区是否为空</p> <p><i>uTransfer</i>: 指明缓冲区是否正在传输</p> <p><i>uOverflow</i>: 指明缓冲区是否越界</p> <p><i>uLineNum</i>: 对于线扫描相机，该参数指明当前传输的图像有效行数；对于面扫描相机，该参数始终为 0</p> <p><i>uCompressSize</i>: 对于压缩图像，存储当前缓冲区中有效长度</p> <p><i>uTimestamp</i>: 缓冲区时间戳</p> <p><i>Reserved[62]</i>: 预留字段</p>	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_BoardNotOpen	采集卡设备未打开

	IKStatus_Invalid_Handle	无效的句柄
注意事项	无	
相关函数	IKapOpen(); IKapLoadConfigurationFromFile(); IKapSetInfo()	

示例代码

无

IKapReleaseBuffer 释放缓冲区

C++/C	int IKapReleaseBuffer(HANDLE hDev, int nFrameNum);	
参数说明	hDev	设备句柄
	nFrameNum	图像缓冲区索引，有效范围是 [0 ~ IKP_FRAME_COUNT - 1]
返回值	0: 调用失败 1: 调用成功	
说明	调用该函数来情况状态为满的传输缓冲区，然后该缓冲区可以用来写入新到来的图像数据。 本函数在连续采集 (<i>nFrameCount</i> = 0) 且关闭自动清理机制 (<i>IKP_FRAME_AUTO_CLEAR</i> = 0) 时候生效。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_BoardNotOpen	采集卡未打开
	IKStatus_Invalid_Handle	无效的句柄
注意事项	IKapBoard 默认自动管理传输缓冲区的状态，因此用户很少会直接调用该函数。 如果你希望手动管理缓冲区的状态，必须先调用 IKapSetInfo (HANDLE, IKP_FRAME_AUTO_CLEAR, 0) 来关闭 IKapBoard 的缓存自动清空机制。	
相关函数	IKapGetBufferStatus()	

示例代码

无

IKapGetCameraStatus 获取相机状态

C++/C	int IKapGetCameraStatus(HANDLE hDev, int* npValue);	
参数说明	hDev	设备句柄
	npValue	相机当前状态
返回值	0: 调用失败 1: 调用成功	
说明	用户可以在任意时刻调用该函数查询当前相机状态判断采集发生异常的原因。 如果返回 0 或相机状态 <i>npValue</i> 为 0xFFFFFFFF，则采集卡获取相机信号失败，用	

	<p>户可以在间隔一段时间后（如 100ms），重新调用该函数获取相机状态。</p> <p>相机状态 <i>npValue</i> 描述如下：</p> <p>Camera Link 相机：</p> <p>[7:3]: 预留</p> <p>[2]: 行有效状态位，1 表示检测到行信号，0 表示未检测到行信号</p> <p>[1]: 帧有效状态位，1 表示检测到帧信号，0 表示未检测到帧信号</p> <p>[0]: 时钟有效状态位，1 表示检测到时钟，0 表示未检测到时钟</p> <p>CoaXPress 相机：</p> <p>[4]: 相机连接状态位，1 表示连接到相机，0 表示未连接到相机</p> <p>[3:0]: 链路速度状态位，0 表示 1.25G，1 表示 2.5G，2 表示 3.125G，3 表示 5G，4 表示 6.25G</p>	
关联文件	IKapBoard.h	
关 联 库	IKapBoard.lib	
错 误 码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_TimeOut	操作超时
	IKStatus_Invalid_Handle	无效的句柄
注意事项	无	
相关函数	IKapOpen(); IKapLoadConfigurationFromFile(); IKapSetInfo()	

示例代码

```

HANDLE hDev = IKapOpen(IKBoardPCIE, 0);
// 获取相机状态
int nCameraStatus;
IKapGetCameraStatus(hDev, &nCameraStatus);

```

IKapGetLastError 获取错误码

C++/C	void IKapGetLastError(PIKAPERRORINFO pIKapErrorInfo, bool bErrorReset);	
参数说明	pIKapErrorInfo	指向错误信息结构体的指针
	bErrorReset	重置错误信息结构体中的值 1: 重置错误信息结构体中的值 0: 不重置错误信息结构体中的值
返 回 值	无	
说 明	<p>当函数执行过程中出现错误时，会产生相应的错误信息码，调用该函数可以获取相应详细信息。注意如果产生多个错误，则始终保留最近一次产生的错误信息。</p> <p>如果 <i>bErrorReset</i> 设置为 1，则会重置错误信息到初始状态。</p> <p>错误信息结构体定义如下：</p> <pre> typedef struct { unsigned int uBoardType; unsigned int uBoardIndex; </pre>	

	<pre>unsigned int uErrorCode; } IKAPERRORINFO, *PIKAPERRORINFO;</pre> <p>参数说明:</p> <p><i>uBoardType</i>: 发生函数调用错误的设备类型</p> <p><i>uBoardIndex</i>: 发生函数调用错误的设备编号</p> <p><i>uErrorCode</i>: 错误码, 查阅“9.3 错误码类型常量列表”获取详细信息</p>
关联文件	IKapBoard.h
关联库	IKapBoard.lib
注意事项	如果产生多个错误, 则始终保留最近一次产生的错误信息。

示例代码

```
IKAPERRORINFO pIKErrInfo;
memset( &pIKErrInfo, 0, sizeof(IKAPERRORINFO) );
IKapGetLastError( &pIKErrInfo, true );
```

IKapGetSerialPort 获取采集卡设备绑定的串口号

C++/C	int IKapGetSerialPort(HANDLE hDev, int* npPortNumber);	
参数说明	hDev	设备句柄
	npPortNumber	采集卡设备绑定的串口号
返回值	0: 调用成功 1: 调用失败	
说明	根据采集卡类型和采集卡编号查询绑定的串口号, 如果采集卡不存在或者采集卡并未绑定到串口则调用成功, 否则调用失败。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败, 可调用 IKapGetLastError() 返回错误码。	
	IKStatus_Invalid_Handle	无效的句柄
	IKStatus_BoardNotFound	设备未发现
	IKStatus_BoardNotBindingCOM	设备尚未绑定串口
注意事项	如果采集卡设备尚未绑定串口, 用户可以使用 IKComManager.exe 工具为采集卡配置串口, IKComManager.exe 工具可以在 I-TEK OptoElectronics \ IKapLibrary \ Bin 中找到。	

示例代码

```
HANDLE hDev;
int nPortNumber = 0;
hDev = IKapOpen(IKBoardPCIE, 0);
IKapGetSerialPort(hDev, &nPortNumber);
```

IKapGetFrameRate 获取图像传输的帧率

C++/C	int IKapGetFrameRate(HANDLE hDev, double* dpFrameRate);	
参数说明	hDev	设备句柄
	dpFrameRate	连续采集图像时相机输出图像的帧率
返回值	0: 调用失败 1: 调用成功	
说明	获取连续采集图像时相机输出的帧率。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_Invalid_Handle	无效的句柄
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_TimeOut	操作超时
注意事项	单次采集图像时无法获得相机输出帧率。 连续采集过程中，在采集初始阶段(0~10 帧内)可能会出现帧率为 0 的情况。	

示例代码

```
HANDLE hDev;
double dFrameRate;
hDev = IKapOpen(IKBoardPCIE, 0);
IKapGetFrameRate(hDev, &dFrameRate);
```

IKapReadCXPUart 读取 CXP 串口数据

C++/C	int IKapReadCXPUart (HANDLE hDev, char* buffer, unsigned length);	
参数说明	hDev	设备句柄
	buffer	缓冲区数据
	length	作为输入，输入缓冲区长度；作为输出，读取缓冲区实际长度
返回值	0: 调用失败 1: 调用成功	
说明	读取 CXP 串口数据。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_Invalid_Handle	无效的句柄
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_TimeOut	操作超时
注意事项	无	
相关函数	IKapWriteCXPUart()	

示例代码

无

IKapWriteCXPUart 写入 CXP 串口数据

C++/C	int IKapWriteCXPUart (HANDLE hDev, char* buffer, unsigned length);	
参数说明	hDev	设备句柄
	buffer	缓冲区数据
	length	作为输入，输入缓冲区长度
返回值	0: 调用失败 1: 调用成功	
说明	写入 CXP 串口数据。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_Invalid_Handle	无效的句柄
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_TimeOut	操作超时
注意事项	无	
相关函数	IKapReadCXPUart()	

示例代码

无

IKapWaitCXPUart 等待 CXP 有效数据

C++/C	int IKapWaitCXPUart (HANDLE hDev,int timeout);	
参数说明	hDev	设备句柄
	timeout	等待超时时间
返回值	0: 调用失败 1: 调用成功	
说明	等待 CXP 有效数据。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_Invalid_Handle	无效的句柄
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_TimeOut	操作超时
注意事项	无	
相关函数	IKapReadCXPUart(); IKapWriteCXPUart()	

示例代码

无

IKapSaveBuffer 保存缓冲区到指定位置

C++/C	int IKapSaveBuffer (HANDLE hDev, int nFrameNum, const char* fileName, int nFlag = IKP_DEFAULT_COMPRESSION);	
参数说明	hDev	设备句柄
	nFrameNum	图像缓冲区索引，有效范围是 [0 ~ IKP_FRAME_COUNT -1]
	fileName	保存 buffer 名称，目前支持 bmp、jpeg、png、tif 和 raw 类型
	nFlag	图像压缩标志 对 jpeg 图像可设置值为： IKP_JPEG_QUALITYSUPERB IKP_JPEG_QUALITYGOOD IKP_JPEG_QUALITYNORMAL IKP_JPEG_QUALITYAVERAGE IKP_JPEG_QUALITYBAD 对 tiff 图像可设置值为： IKP_TIFF_NONE IKP_TIFF_LZW 对 png 图像可设置值为： IKP_PNG_Z_NO_COMPRESSION IKP_PNG_Z_BEST_SPEED IKP_PNG_Z_DEFAULT_COMPRESSION IKP_PNG_Z_BEST_COMPRESSION
返回值	0：调用失败 1：调用成功	
说明	保存缓冲区到指定位置。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_Invalid_Handle	无效的句柄
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_TimeOut	操作超时
注意事项	目前支持 bmp、jpeg、png、tif 和 raw 类型。	
相关函数	IKapLoadBuffer()	

示例代码

无

IKapLoadBuffer 从指定位置加载缓冲区

C++/C	int IKapLoadBuffer (HANDLE hDev, int nFrameNum, const char* fileName);	
参数说明	hDev	设备句柄
	nFrameNum	图像缓冲区索引，有效范围是 [0 ~ IKP_FRAME_COUNT - 1]
	fileName	加载 buffer 名称，目前支持 bmp、jpeg、png、tif 和 raw 类型
返回值	0: 调用失败 1: 调用成功	
说明	从指定位置加载缓冲区。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_Invalid_Handle	无效的句柄
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_TimeOut	操作超时
注意事项	目前支持 bmp、jpeg、png、tif 和 raw 类型。	
相关函数	IKapSaveBuffer()	

示例代码

无

IKapWriteRegister 采集卡寄存器直接写入

C++/C	int IKapWriteRegister(HANDLE hDev, unsigned addr , unsigned data);	
参数说明	hDev	设备句柄
	Address	寄存器地址
	data	寄存器值
返回值	0: 调用失败 1: 调用成功	
说明	采集卡寄存器直接写入。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_Invalid_Handle	无效的句柄
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_TimeOut	操作超时
注意事项	无	
相关函数	IKapReadRegister()	

示例代码

无

IKapReadRegister 采集卡寄存器直接读取

C++/C	int IKapReadRegister(HANDLE hDev, unsigned addr , unsigned* data);	
参数说明	hDev	设备句柄
	Address	寄存器地址
	data	寄存器值
返回值	0: 调用失败 1: 调用成功	
说明	采集卡寄存器直接读取。	
关联文件	IKapBoard.h	
关联库	IKapBoard.lib	
错误码	函数调用失败，可调用 IKapGetLastError() 返回错误码。	
	IKStatus_Invalid_Handle	无效的句柄
	IKStatus_BoardNotOpen	采集卡设备未打开
	IKStatus_TimeOut	操作超时
注意事项	无	
相关函数	IKapWriteRegister()	

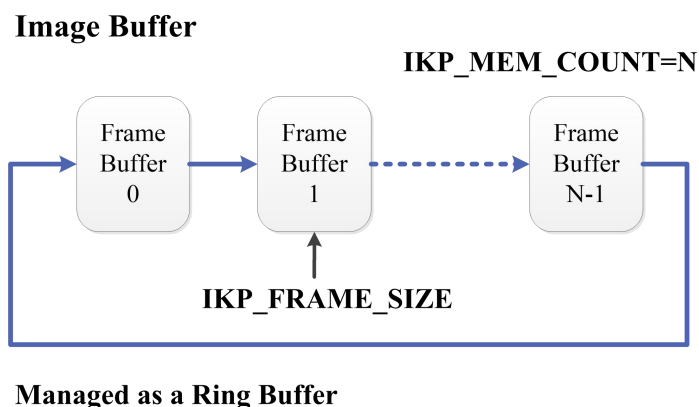
示例代码

无

6 图像缓冲区内内存管理

相机采集到的图像数据将会存储到 IKapBoard 的图像缓存中。IKapBoard 提供了不同的方法来控制图像的传输流程。通过设置合理的图像传输机制，可以更好的保证图像数据的安全，提高应用程序性能。

如 4.3 小节中所述，IKapBoard 以“帧”为单位管理图像缓存区域，而且每帧只能保存一张完整图像。每帧图像的大小可以通过 **IKP_FRAME_SIZE** 参数获取。IKapBoard 默认维护一帧图像缓存区域存储相机采集到的数据，用户可以通过 **IKP_FRAME_COUNT** 设置多帧缓存区域。当缓冲区中有多帧缓存时，IKapBoard 将会按照环形缓冲区的方式来存储相机采集到的数据，如下图所示。



6.1 图像缓冲区状态

- 图像缓冲区的三种状态：
 - **Empty**: 空状态，意味着当前缓冲区没有相机采集到的图像数据。
 - **Full**: 满状态，意味着当前缓冲区已经被相机采集到的图像数据填满。
 - **Transfer**: 传输状态，意味着相机正在向该缓冲区内传输数据。
 用户可以通过 **IKapGetBufferStatus()** 来获取指定传输缓冲区的状态。
- 图像缓冲区的状态切换

图像缓冲区的状态会在如下时刻切换：

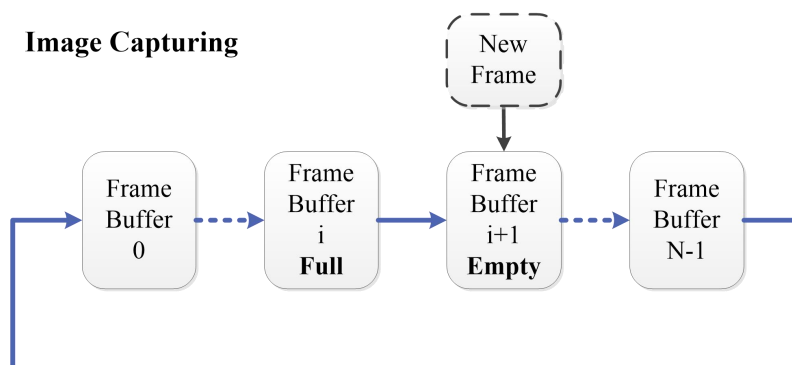
 - 图像采集启动时，所有图像缓冲区被设置为 **Empty**。
 - 当一帧完整的数据被传输到该图像缓冲区后，设置该图像缓冲区状态为 **Full**。
 - 当图像数据正在向该图像缓冲区传输时，设置该图像缓冲区状态为 **Transfer**，对于线扫描相机，可以在 **Transfer** 状态获取图像缓冲区已经传输的有效行数（仅适用于线扫描相机）。
 - 当 **IKEvent_FrameReady** 的回调函数执行完成后（无论是否设置），都会自动设置图像缓冲区状态为 **Empty**。
- 图像缓冲区的自动清空机制
 - 当一帧完整的图像传输到图像缓冲区后，会触发 **IKEvent_FrameReady** 事件并且调用相应的回调函数。当回调函数完成后（无论是否设置），会将当前帧的图像缓冲区设置为 **Empty**。
 - 用户需要在回调函数中读取当前帧的缓冲区，否则新的图像数据有可能会覆盖旧数据。
 - 如果用户需要手动控制缓存区域的状态，可以调用 **IKapSetInfo(HANDLE, IKP_FRAME_AUTO_CLEAR, 0)** 来关闭 IKapBoard 的自动清空机制。注意该函数应该在

采集过程开始前调用。

- 当用户关闭自动清空机制后，需要手动清空状态为 **Full** 的图像数据。通过调用 **IKapReleaseBuffer()** 将会设置传输缓存区域状态为 **Empty**，新的图像数据将会再次写入状态为 **Empty** 的缓存中。

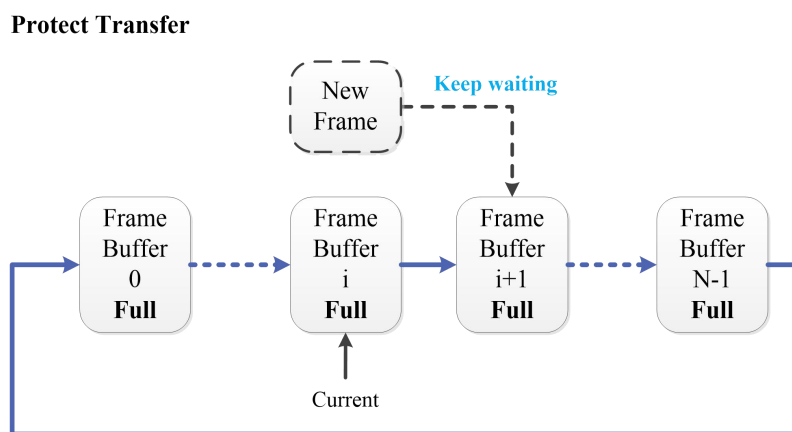
6.2 图像缓冲区传输模式

IKapBoard 支持缓冲区的同步保护模式。



● 同步保护模式

在这种模式下，IKapBoard 不会写入图像数据直到下一帧缓存状态为 **Empty**。注意在这种模式下，原有图像数据的安全性可以得到保证，但是可能导致新到来图像数据的丢失。



注意：

- (1) 图像缓存区域的数量由参数 **IKP_FRAME_COUNT** 设定。增加缓冲区数量可以避免图像丢失或者覆盖。
- (2) 当用户禁用 IKapBoard 自动清空机制后，必须手动调用 **IKapReleaseBuffer()** 来释放已经处理完毕的缓存区域。
- (3) 图像数据将会从第一帧开始次序写入。为了避免采集图像乱序，用户应该保存访问的图像编号。

7 IKapBoard 使用范例

7.1 枚举设备

本例展示了如何枚举所有连接到计算机上的采集卡设备，并且输出采集卡的名字信息。具体工程参见 \ I-TEK OptoElectronics \ IKapLibrary \ Examples \ C。

示例代码

```
unsigned int nResourceCount = 0, nPCIEResourceCount = 0, nUSBResourceCount = 0;
char* resourceName = NULL;
unsigned int resourceNameSize = 0;
IKAPERRORINFO pIKErrInfo;

// 获取当前计算机上连接的所有类型的采集卡数量
IKapGetBoardCount( IKBoardALL, &nResourceCount );
printf("Enum Device Number : %u\n", nResourceCount);

// 获取当前计算机上连接的PCIe类型的采集卡数量
IKapGetBoardCount( IKBoardPCIE, &nPCIEResourceCount );
printf("Enum PCIe Device Number: %u\n\n", nPCIEResourceCount);

// 获取当前计算机上连接的USB3.0类型的采集卡数量
IKapGetBoardCount( IKBoardUSB30, &nUSBResourceCount);
printf("Enum USB Device Number: %u\n\n", nUSBResourceCount);

// 打印所有类型采集卡的名称信息
for (unsigned i=0; i < nPCIEResourceCount; i++)
{
    resourceNameSize = 0;
    IKapGetBoardName( IKBoardPCIE, i, resourceName, &resourceNameSize );
    IKapGetLastError( &pIKErrInfo, true );
    if ( pIKErrInfo.uErrorCode == IKStatus_BufferTooSmall )
    {
        resourceName = new char[resourceNameSize];
        IKapGetBoardName( IKBoardPCIE, i, resourceName, &resourceNameSize );
    }
    IKapGetLastError( &pIKErrInfo, true );
    if ( pIKErrInfo.uErrorCode != IKStatus_Success )
        printf("Last Error: %08X\n", pIKErrInfo.uErrorCode );
    else
        printf("PCIe Device-%u Name: %s\nDevice\n\n", i, resourceName );
    delete[] resourceName;
}
```

7.2 配置采集卡

本例展示了如何配置采集卡并采集图像。具体工程参见 \ I-TEK OptoElectronics \

IKapLibrary \ Examples \ C。

示例代码

```
int g_nImageIndex = 0;    // 采集到的图像索引
int g_nGrabImageTotalCount = 10;    // 需要采集的图像总数

/* 返回 IKapBoard 错误信息 */
void printErrorMessage(char* msg1, char* msg2);

/* 释放采集卡句柄 */
void freeFrameGrabberHandle(HANDLE hVul);

/* 一帧图像采集完成时触发该回调函数 */
void CALLBACK OnFrameReady(void* Context);

/* 配置采集卡参数 */
void configureBoard(HANDLE hVul);

/* 设置传输模式为同步模式 */
bool demonstrateGrabMultiSync(HANDLE hVul);

int main(int argc, char *argv[])
{
    HANDLE hVul = INVALID_HANDLE_VALUE;
    unsigned int PCIeDevCount = 0;
    int res = 0;

    /* 枚举所有 Vulcan 采集卡 */
    res = IKapGetBoardCount(IKBoardPCIE, &PCIeDevCount);
    if (PCIeDevCount == 0)
    {
        fprintf(stderr, "Not enough frame grabber devices found.\n");
        exit(EXIT_FAILURE);
    }

    /* 打开采集卡 */
    hVul = IKapOpen(IKBoardPCIE, 0);
    if(hVul == INVALID_HANDLE_VALUE)
    {
        fprintf(stderr, "Fail to open frame grabber device.\n");
        exit(EXIT_FAILURE);
    }

    /* 配置采集卡参数 */
    configureBoard(hVul);
}
```

```
/* 开始图像采集 */
if (!demonstrateGrabMultiSync(hVul))
{
    fprintf(stderr, "Fail to grab image from camera.\n");
    freeFrameGrabberHandle(hVul);
    break;
}

/* 关闭采集卡 */
IKapClose(hVul);

return EXIT_SUCCESS;
}

/* 返回 IKapBoard 错误信息 */
void printErrorMessage(char* msg1, char* msg2)
{
    IKAPERRORINFO pIKErrInfo;

    memset(&pIKErrInfo, 0, sizeof(IKAPERRORINFO));
    IKapGetLastError(&pIKErrInfo, true);
    printf("%s(%s)\nType= %d\nIndex= %08x\nError Code= %08x\n", msg1, msg2,
        pIKErrInfo.uBoardType, pIKErrInfo.uBoardIndex, pIKErrInfo.uErrorCode);
}

/* 释放采集卡句柄 */
void freeFrameGrabberHandle(HANDLE hVul)
{
    if (hVul != INVALID_HANDLE_VALUE)
    {
        IKapClose(hVul);
        hVul = INVALID_HANDLE_VALUE;
    }
}

/* 一帧图像采集完成时触发该回调函数 */
void CALLBACK OnFrameReady(void* Context)
{
    HANDLE hVul = (HANDLE)Context;
    int frameSize;
    char* pBuffer;
    IKAPBUFFERSTATUS status;
```

```

/* 获取缓冲区大小 */
if(IKapGetInfo(hVul, IKP_FRAME_SIZE, &frameSize) == false)
{
    printErrorMessage("IKapGetInfo", "IKP_FRAME_SIZE");
    freeFrameGrabberHandle(hVul);
    return;
}

for (int i = 0; i < g_nGrabImageTotalCount; i++)
{
    /* 获取缓冲区状态 */
    if(IKapGetBufferStatus( hVul, g_nImageIndex, &status ) == false)
    {
        printErrorMessage("IKapGetBufferStatus", "status");
        freeFrameGrabberHandle(hVul);
        return;
    }

    /* 处理被图像填充的缓冲区 */
    if (status.uFull == 1)
    {
        /* 获取缓冲区地址 */
        if(IKapGetBufferAddress(hVul, g_nImageIndex, (void**)&pBuffer) == false)
        {
            printErrorMessage("IKapGetBufferAddress", "pBuffer");
            freeFrameGrabberHandle(hVul);
            return;
        }
        /* 此处处理图像数据 */
        g_nImageIndex = (g_nImageIndex + 1) % g_nGrabImageTotalCount;
    }
    else
        break;
}
}

/* 配置采集卡参数 */
void configureBoard(HANDLE hVul)
{
    int TimeOut = INFINITE;    // 超时时间
    int ImageType = IKP_IMAGE_TYPE_VAL_MONOCHROME;    // 图像类型
    int dataFormat = IKP_DATA_FORMAT_VAL_8BIT;    // 数据格式
    int tapNum = 8;    // Tap 数

```

```

/* 相机 CC1 触发源 */
int CC1Source = IKP_CC_SOURCE_VAL_INTEGRATION_SIGNAL1;
/* 采集卡触发源 */
int integrationSource = IKP_INTEGRATION_TRIGGER_SOURCE_VAL_GENERAL_INPUT1;
/* 采集卡触发方法 */
int integrationMethod = IKP_INTEGRATION_METHOD_VAL_4;
int ImageWidth = 2048;    // 图像宽度
int ImageHeight = 2048;   // 图像高度

/* 设置相机扫描类型 */
if (IKapSetInfo(hVul, IKP_SCAN_TYPE, IKP_SCAN_TYPE_VAL_AREA) == false)
{
    printErrorMessage("IKapSetInfo", "IKP_SCAN_TYPE");
    freeFrameGrabberHandle(hVul);
    return;
}

/* 设置超时时间 */
if (IKapSetInfo(hVul, IKP_TIME_OUT, TimeOut) == false)
{
    printErrorMessage("IKapSetInfo", "IKP_TIME_OUT");
    freeFrameGrabberHandle(hVul);
    return;
}

/* 设置图像类型 */
if (IKapSetInfo(hVul, IKP_IMAGE_TYPE, ImageType) == false)
{
    printErrorMessage("IKapSetInfo", "IKP_IMAGE_TYPE");
    freeFrameGrabberHandle(hVul);
    return;
}

/* 设置数据格式 */
if (IKapSetInfo(hVul, IKP_DATA_FORMAT, dataFormat) == false)
{
    printErrorMessage("IKapSetInfo", "IKP_DATA_FORMAT");
    freeFrameGrabberHandle(hVul);
    return;
}

/* 设置图像宽度 */
if (IKapSetInfo(hVul, IKP_IMAGE_WIDTH, (int)ImageWidth) == false)
{

```



```

        printErrorMessage("IKapSetInfo", "IKP_IMAGE_WIDTH");
        freeFrameGrabberHandle(hVul);
        return;
    }

    /* 设置图像高度 */
    if(IKapSetInfo(hVul, IKP_IMAGE_HEIGHT, (int)ImageHeight) == false)
    {
        printErrorMessage("IKapSetInfo", "IKP_IMAGE_HEIGHT");
        freeFrameGrabberHandle(hVul);
        return;
    }

    /* 设置 Tap 数 */
    if(IKapSetInfo(hVul, IKP_TAP_NUMBER, tapNum) == false)
    {
        printErrorMessage("IKapSetInfo", "IKP_TAP_NUMBER");
        freeFrameGrabberHandle(hVul);
        return;
    }

    /* 设置相机 CC1 触发源 */
    if(IKapSetInfo(hVul, IKP_CC1_SOURCE, CC1Source) == false)
    {
        printErrorMessage("IKapSetInfo", "IKP_CC1_SOURCE");
        freeFrameGrabberHandle(hVul);
        return;
    }

    /* 设置采集卡触发源 */
    if(IKapSetInfo(hVul, IKP_INTEGRATION_TRIGGER_SOURCE, integrationSource) == false)
    {
        printErrorMessage("IKapSetInfo", "IKP_INTEGRATION_TRIGGER_SOURCE");
        freeFrameGrabberHandle(hVul);
        return;
    }

    /* 设置采集卡触发方法 */
    if(IKapSetInfo(hVul, IKP_INTEGRATION_METHOD, integrationMethod) == false)
    {
        printErrorMessage("IKapSetInfo", "IKP_INTEGRATION_TRIGGER_SOURCE");
        freeFrameGrabberHandle(hVul);
        return;
    }
}

```

```

/* 一帧图像采集完成时触发该回调函数 */
if(IKapRegisterCallback(hVul, IKEvent_FrameReady, OnFrameReady, hVul) == false)
{
    printErrorMessage("IKapRegisterCallback", "IKEvent_FrameReady");
    freeFrameGrabberHandle(hVul);
    return;
}
}

/* 设置传输模式为同步模式 */
bool demonstrateGrabMultiSync(HANDLE hVul)
{
    /* 设置缓冲区个数 */
    if(IKapSetInfo(hVul, IKP_FRAME_COUNT, g_nGrabImageTotalCount) == false)
    {
        printErrorMessage("IKapSetInfo", "IKP_FRAME_COUNT");
        return false;
    }

    /* 设置图像传输方式 */
    if(IKapSetInfo(hVul, IKP_FRAME_TRANSFER_MODE,
        IKP_FRAME_TRANSFER_SYNCHRONOUS) == false)
    {
        printErrorMessage("IKapSetInfo", "IKP_FRAME_TRANSFER_MODE");
        return false;
    }

    /* 设置图像采集方式 */
    if(IKapSetInfo(hVul, IKP_GRAB_MODE, IKP_GRAB_NON_BLOCK) == false)
    {
        printErrorMessage("IKapSetInfo", "IKP_GRAB_MODE");
        return false;
    }

    /* 清除缓冲区 */
    IKapClearGrab(hVul);

    /* 开始采集图像 */
    if(IKapStartGrab(hVul, g_nGrabImageTotalCount) == false)
    {
        printErrorMessage("IKapStartGrab", NULL);
        return false;
    }
}

```

```

/* 等待图像采集完成 */
if(IKapWaitGrab(hVul)==false)
{
    printErrorMessage("IKapWaitGrab", NULL);
    return false;
}
return true;
}

```

7.3 单帧采集

本例展示了如何单次采样图片、导入采集卡配置参数并输出图像的信息。具体工程参见 \ **I-TEK OptoElectronics \ IKapLibrary \ Examples \ C**。

示例代码

```

HANDLE hDev = INVALID_HANDLE_VALUE;
unsigned int nPCIEDevCount = 0;
char* configFilename=new char[MAX_PATH];
int nFrameSize = 0;
int nImageWidth = 0;
int nImageHeight = 0;
int nDataFormat = 0;
int nBoardBit = 0;
unsigned char* pUserBuffer;

if ( GetOption( configFilename) == false )
{
    MessageBox( GetFocus(), _T("Read Vulcan configuration file failed."), NULL, MB_OK );
    return 0;
}

// 获取 PCIE 采集卡数量
IKapGetBoardCount( IKBoardPCIE, &nPCIEDevCount );
if ( nPCIEDevCount == 0 )
{
    ErrorMessage( "IKapGetBoardCount", "0" );
    goto FreeHandle;
}

// 打开采集卡
hDev = IKapOpen( IKBoardPCIE, 0 );
if ( hDev == INVALID_HANDLE_VALUE )
{
    ErrorMessage("IKapOpen", NULL);
    goto FreeHandle;
}

```

```

}

// 导入采集卡配置文件
if ( IKapLoadConfigurationFromFile( hDev, configFilename ) == false )
{
    ErrorMessage("IKapLoadConfigurationFromFile", configFilename );
    goto FreeHandle;
}

// 采集一帧图像（阻塞模式）
if ( IKapStartGrab ( hDev, 1 ) == false )
{
    ErrorMessage("IKapStartGrab (block mode)", NULL );
    goto FreeHandle;
}

// 获取图像大小
if ( IKapGetInfo( hDev, IKP_FRAME_SIZE, &nFrameSize) == false )
{
    ErrorMessage( "IKapGetInfo", "IKP_FRAME_SIZE" );
    goto FreeHandle;
}

// 获取图像宽度
if ( IKapGetInfo( hDev, IKP_IMAGE_WIDTH, &nImageWidth ) == false )
{
    ErrorMessage("IKapGetInfo", "IKP_IMAGE_WIDTH");
    goto FreeHandle;
}

// 获取图像高度
if ( IKapGetInfo( hDev, IKP_IMAGE_HEIGHT, &nImageHeight ) == false )
{
    ErrorMessage("IKapGetInfo", "IKP_IMAGE_WIDTH" );
    goto FreeHandle;
}

// 获取数据格式
if ( IKapGetInfo( hDev, IKP_DATA_FORMAT, &nDataFormat ) == false )
{
    ErrorMessage("IKapGetInfo", "IKP_DATA_FORMAT" );
    goto FreeHandle;
}

```

```
// 获取采集卡 bit 大小
if ( IKapGetInfo( hDev, IKP_BOARD_BIT, &nBoardBit ) == false )
{
    ErrorMessage("IKapGetInfo", "IKP_BOARD_BIT");
    goto FreeHandle;
}
printf("Frame Size:%d\nImage Width:%d\nImage Height:%d\nPixel Format:%d\nBoard Bit:%d\n",
        nFrameSize, nImageWidth, nImageHeight, nDataFormat, nBoardBit);

// 获取缓冲区地址
if ( IKapGetBufferAddress( hDev, (void**)&pUserBuffer ) == false )
{
    ErrorMessage("IKapGetBufferAddress", NULL );
    goto FreeHandle;
}
else
{
    /**
     * 此处处理图像数据
     * e.g. ProcessImage(unsigned char* pUserBuffer, int nFrameSize )
     */
}

FreeHandle:
if ( hDev != INVALID_HANDLE_VALUE )
{
    IKapClose( hDev );
    hDev = INVALID_HANDLE_VALUE;
}
delete[] configFilename;
```

7.4 连续采集

本例展示了如何连续采集图像并通过回调函数在一帧图像采集完后获取图像信息。具体工程参见 \I-TEK OptoElectronics \ IKapLibrary \ Examples \ C。

示例代码

```
// 连续采集开始时触发该回调函数
void CALLBACK OnGrabStart(void *Context)
{
    printf("Start grabbing image(continuously)\n");
}

// 采集丢帧时触发该回调函数
void CALLBACK OnFrameLost(void *Context)
{
}
```

```

    printf("Frame lost\n");
}

// 采集超时时触发该回调函数
void CALLBACK OnTimeout(void *Context)
{
    printf("Grab image timeout\n");
}

// 一帧图像采集完成时触发该回调函数
void CALLBACK OnFrameReady(void *Context)
{
    HANDLE hDev = (HANDLE)Context;
    double dFrameRate = 0.0;
    int nFrameSize = 0;
    unsigned char* pBuffer = NULL;

    IKapGetFrameRate( hDev, &dFrameRate );
    IKapGetInfo( hDev, IKP_FRAME_SIZE, &nFrameSize );
    IKapGetBufferAddress( hDev, 0, (void**)&pBuffer );
    printf("End grab one frame FrameRate:%.2lffps\n", dFrameRate );

    /*****
    /* 此处处理图像数据
    /* e.g. ProcessImage(unsigned char* imageBuffer, int imageSize)
    /*
    *****/

}

// 连续采集结束时触发该回调函数
void CALLBACK OnGrabStop(void *Context)
{
    printf("Stop grabbing image(continuously)\n");

    HANDLE hDev = (HANDLE)Context;
    IKAPERRORINFO pIKErrInfo;

    memset( &pIKErrInfo, 0, sizeof(IKAPERRORINFO) );
    IKapGetLastError( &pIKErrInfo, true );
    if ( pIKErrInfo.uErrorCode != IKStatus_Success )
    {
        /*****
        /* 采集中断处理操作
        /* e.g. DoErrorHandle( uErrorCode )*/

```

```

    /*******
    }
}

int main(int argc, char* argv[])
{
    HANDLE hDev = INVALID_HANDLE_VALUE;
    unsigned int nPCIEDevCount = 0;
    char* configFilename = new char[MAX_PATH];
    int nTimeout = 2000;

    if ( GetOption( configFilename) == false )
    {
        MessageBox( GetFocus(), _T("Read vulcan configuration file failed."), NULL, MB_OK );
        return 0;
    }

    // 获取 PCIe 采集卡数量
    IKapGetBoardCount( IKBoardPCIE, &nPCIEDevCount );
    if ( nPCIEDevCount == 0 )
    {
        ErrorMessage("IKapGetBoardCount", "0");
        goto FreeHandle;
    }

    // 打开采集卡
    hDev = IKapOpen( IKBoardPCIE, 0 );
    if ( hDev == INVALID_HANDLE_VALUE )
    {
        ErrorMessage("IKapOpen", NULL );
        goto FreeHandle;
    }

    // 导入采集卡配置文件
    if ( IKapLoadConfigurationFromFile( hDev, configFilename ) == false )
    {
        ErrorMessage("IKapLoadConfigurationFromFile", configFilename );
        goto FreeHandle;
    }

    // 注册回调函数
    if ( IKapRegisterCallback( hDev, IKEvent_GrabStart, OnGrabStart, hDev ) == false )
    {
        ErrorMessage("Register Callback", "IKEvent_GrabStart");
    }
}
```

```

        goto FreeHandle;
    }

    // 注册回调函数
    if ( IKapRegisterCallback( hDev, IKEvent_FrameReady, OnFrameReady, hDev ) == false )
    {
        ErrorMessage(“Register Callback”, “IKEvent_FrameReady” );
        goto FreeHandle;
    }

    // 注册回调函数
    if ( IKapRegisterCallback( hDev, IKEvent_GrabStop, OnGrabStop, hDev ) == false )
    {
        ErrorMessage( “Register Callback”, “IKEvent_GrabStop”);
        goto FreeHandle;
    }

    // 注册回调函数
    if ( IKapRegisterCallback( hDev, IKEvent_TimeOut, OnTimeout, hDev ) == false )
    {
        ErrorMessage(“Register Callback”, “IKEvent_TimeOut” );
        goto FreeHandle;
    }

    // 注册回调函数
    if ( IKapRegisterCallback( hDev, IKEvent_FrameLost, OnFrameLost, hDev ) == false )
    {
        ErrorMessage( “Register Callback”, “IKEvent_FrameLost” );
        goto FreeHandle;
    }

    // 开始连续采集
    if ( IKapStartGrab( hDev, 0 ) == false )
    {
        ErrorMessage(“Start grabbing image(continuously)”, NULL );
        goto FreeHandle;
    }

    Sleep(5000);

    // 结束连续采集
    if ( IKapStopGrab ( hDev ) == false )
    {
        ErrorMessage(“Stop grabbing image(continuous)”, NULL );
    }

```



```
        goto FreeHandle;
    }

    // 清除回调函数
    if ( IKapUnRegisterCallback ( hDev, IKeEvent_GrabStart) == false )
    {
        ErrorMessage(“UnRegister Callback”, “IKeEvent_GrabStart” );
        goto FreeHandle;
    }

    // 清除回调函数
    if ( IKapUnRegisterCallback ( hDev, IKeEvent_GrabStop) == false )
    {
        ErrorMessage( “UnRegister Callback”, “IKeEvent_GrabEnd” );
        goto FreeHandle;
    }

    // 清除回调函数
    if ( IKapUnRegisterCallback ( hDev, IKeEvent_FrameReady) == false )
    {
        ErrorMessage( “UnRegister Callback”, “IKeEvent_FrameReady” );
        goto FreeHandle;
    }

    // 清除回调函数
    if ( IKapUnRegisterCallback ( hDev, IKeEvent_FrameLost) == false )
    {
        ErrorMessage(“UnRegister Callback”, “IKeEvent_FrameLost” );
        goto FreeHandle;
    }

    // 清除回调函数
    if ( IKapUnRegisterCallback ( hDev, IKeEvent_TimeOut) == false )
    {
        ErrorMessage(“UnRegister Callback”, “IKeEvent_TimeOut” );
        goto FreeHandle;
    }

FreeHandle:
    if ( hDev != INVALID_HANDLE_VALUE )
    {
        IKapClose( hDev );
        hDev = INVALID_HANDLE_VALUE;
    }
```

```
delete[] configFilename;
```

7.5 多帧采集

本例展示了如何采集多帧图像并通过回调函数在一帧图像采集完毕后获取图像信息。具体工程参见 \I-TEK OptoElectronics \ IKapLibrary \ Examples \ C。

示例代码

```
// 等待图像采集
bool bWait;
// 采集到的图像索引
int nIndex;

// 当一帧图像采集完成时触发该回调函数
void CALLBACK OnFrameReady(void *Context)
{
    HANDLE hDev = (HANDLE)Context;
    unsigned char* pUserBuffer = NULL;
    int nFrameSize = 0;
    int nFrameCount = 0;
    IKAPBUFFERSTATUS status;
    IKapGetInfo( hDev, IKP_FRAME_COUNT, &nFrameCount );
    IKapGetBufferStatus( hDev, nIndex, &status );
    while ( status.uFull == 1 && nIndex < nFrameCount )
    {
        IKapGetInfo( hDev, IKP_FRAME_SIZE, &nFrameSize );
        IKapGetBufferAddress( hDev, nIndex, (void**)&pUserBuffer );
        /* *****
        /* 此处处理图像数据
        /* e.g. ProcessImage(unsigned char* pUserBuffer, int nFrameSize)
        /* *****
        IKapGetBufferStatus( hDev, ++nIndex, &status );
    }
}

// 采集超时时触发该回调函数
void CALLBACK OnTimeOut(void* pContext)
{
    printf("Timeout Event Happen!\n");
    bWait = false;
}

// 采集停止时触发该回调函数
void CALLBACK OnGrabStop(void *Context)
{
    printf("Stop grabbing image(continuous)\n");
```

```

HANDLE hDev = (HANDLE)Context;
IKAPERRORINFO pIKErrInfo;

memset( &pIKErrInfo, 0, sizeof(IKAPERRORINFO) );
IKapGetLastError( &pIKErrInfo, true );
if ( pIKErrInfo.uErrorCode != IKStatus_Success )
{
    /******
    /* 采集中断处理操作
    /* e.g. DoErrorHandle( uErrorCode )*/
    /******
}
bWati = false;
}

int main (int argc, char* argv[])
{
    HANDLE hDev = INVALID_HANDLE_VALUE;
    unsigned int nPCIEDevCount = 0;
    char configFilename[MAX_PATH] = {0};
    int nFrameCount = 0;

    if ( GetOption( configFilename) == false )
    {
        MessageBox( GetFocus(), _T("Read vulcan configuration file failed."), NULL, MB_OK );
        return 0;
    }

    printf("Please input image number: \n");
    scanf_s("%d", &nFrameCount);

    // 获取 PCIe 采集卡数量
    IKapGetBoardCount( IKBoardPCIE, &nPCIEDevCount );
    if ( nPCIEDevCount == 0 )
    {
        ErrorMessage("IKapGetBoardCount", "0");
        goto FreeHandle;
    }

    // 打开采集卡
    hDev = IKapOpen( IKBoardPCIE, 0 );
    if (hDev == INVALID_HANDLE_VALUE )
    {

```

```

        ErrorMessage( "IKapOpen", NULL );
        goto FreeHandle;
    }

    // 导入采集卡配置文件
    if( IKapLoadConfigurationFromFile( hDev, configFilename ) == false )
    {
        ErrorMessage( "IKapLoadConfigurationFromFile", configFilename );
        goto FreeHandle;
    }

    // 设置帧数
    if ( IKapSetInfo(hDev, IKP_FRAME_COUNT, nFrameCount) == false )
    {
        ErrorMessage( "IKapSetInfo", "IKP_FRAME_COUNT" );
        goto FreeHandle;
    }

    // 设置传输模式
    if ( IKapSetInfo(hDev, IKP_FRAME_TRANSFER_MODE,
        IKP_FRAME_TRANSFER_ASYNCHRONOUS) == false )
    {
        ErrorMessage("IKapSetInfo", "IKP_FRAME_TRANSFER_MODE" );
        goto FreeHandle;
    }

    // 设置采集模式
    if ( IKapSetInfo(hDev, IKP_GRAB_MODE, IKP_GRAB_NON_BLOCK) == false )
    {
        ErrorMessage( "IKapSetInfo", "IKP_GRAB_STRAT_MODE" );
        goto FreeHandle;
    }

    // 注册回调函数
    if ( IKapRegisterCallback( hDev, IKeyEvent_FrameReady, OnFrameReady, hDev ) == false )
    {
        ErrorMessage( "Register Callback", "IKeyEvent_FrameReady" );
        goto FreeHandle;
    }

    // 注册回调函数
    if ( IKapRegisterCallback( hDev, IKeyEvent_GrabStop, OnGrabStop, hDev ) == false )
    {
        ErrorMessage( "Register Callback", "IKeyEvent_GrabStop" );
    }

```

```
        goto FreeHandle;
    }

    // 注册回调函数
    if ( IKapRegisterCallback( hDev, IKeyEvent_TimeOut, OnTimeOut, hDev) == false )
    {
        ErrorMessage( "Register Callback", "IKeyEvent_TimeOut" );
        goto FreeHandle;
    }

    // 开始采集
    if ( IKapStartGrab( hDev, nFrameCount ) == false )
    {
        ErrorMessage( "Start grabbing image(sequence)", NULL );
        goto FreeHandle;
    }

    while ( bWait );

    // 停止采集
    if ( IKapStopGrab ( hDev ) == false )
    {
        ErrorMessage( "Stop grabbing image(sequence)", NULL );
        goto FreeHandle;
    }

    // 清除回调函数
    if ( IKapUnRegisterCallback ( hDev, IKeyEvent_GrabStop) == false )
    {
        ErrorMessage( "UnRegister Callback", "IKeyEvent_GrabEnd" );
        goto FreeHandle;
    }

    // 清除回调函数
    if ( IKapUnRegisterCallback ( hDev, IKeyEvent_FrameReady) == false )
    {
        ErrorMessage( "UnRegister Callback", "IKeyEvent_FrameReady" );
        goto FreeHandle;
    }

    // 清除回调函数
    if ( IKapUnRegisterCallback ( hDev, IKeyEvent_TimeOut) == false )
    {
        ErrorMessage( "UnRegister Callback", "IKeyEvent_TimeOut");
    }
```

```

        goto FreeHandle;
    }

FreeHandle:
    if ( hDev != INVALID_HANDLE_VALUE )
    {
        IKapClose( hDev );
        hDev = INVALID_HANDLE_VALUE;
    }
    delete[] configFilename;
    return 0;
}

```

7.6 处理图像数据

本例展示了如何处理采集卡获取的图像数据。IKapBoard 当前支持的图像类型有灰度图像、彩色 RGB 图像和彩色 RGBC 图像，对于不同类型图像数据的处理方式本例都有详细说明。具体工程参见 \I-TEK OptoElectronics\IKapLibrary\Examples\C。

示例代码

```

int main( int argc, char *argv[] )
{
    HANDLE hDev = INVALID_HANDLE_VALUE;
    unsigned int nPCIEDevCount = 0;
    char configFilename[MAX_PATH] = { 0 };

    // 获取 PCIe 采集卡数量
    IKapGetBoardCount( IKBoardPCIE, &nPCIEDevCount );
    if ( nPCIEDevCount == 0 )
    {
        ErrorMessage( "IKapGetBoardCount", "0" );
        goto FreeHandle;
    }

    // 打开采集卡
    hDev = IKapOpen( IKBoardPCIE, 0 );
    if ( hDev == INVALID_HANDLE_VALUE )
    {
        ErrorMessage( "IKapOpen", NULL );
        goto FreeHandle;
    }

    // 导入采集卡配置文件
    if ( IKapLoadConfigurationFromFile( hDev, configFilename ) == false )
    {
        ErrorMessage( "IKapLoadConfigurationFromFile", configFilename );
    }
}

```

```

        goto FreeHandle;
    }

    // 单帧采集（阻塞模式）
    if ( IKapStartGrab( hDev, 1 ) == false )
    {
        ErrorMessage( "IKapStartGrab(block)", NULL );
        goto FreeHandle;
    }
    /*****
    /* 此处处理图像数据
    *****/
    ProcessImage(hDev);

FreeHandle:
    if ( hDev != INVALID_HANDLE_VALUE )
    {
        IKapClose( hDev );
        hDev = INVALID_HANDLE_VALUE;
    }
    return 0;
}

void ErrorMessage( char* msg1, char* msg2 )
{
    IKAPERRORINFO pIKErrInfo;

    memset( &pIKErrInfo, 0, sizeof(IKAPERRORINFO) );
    IKapGetLastError( &pIKErrInfo, true );

    printf( "%s(%s)\nType= %d\nIndex= %08x\nError Code= %08x\n", msg1, msg2,
        pIKErrInfo.uBoardType, pIKErrInfo.uBoardIndex, pIKErrInfo.uErrorCode );
}

void ProcessImage( HANDLE hDev )
{
    int nImageWidth = 0;
    int nImageHeight = 0;
    int nDataFormat = 0;
    int nImageType = 0;
    void* pImageBuffer = NULL;

    // 获取图像宽度
    if ( IKapGetInfo( hDev, IKP_IMAGE_WIDTH, &nImageWidth ) == false )

```

```

{
    ErrorMessage( "IKapGetInfo", "IKP_IMAGE_WIDTH" );
}

// 获取图像高度
if ( IKapGetInfo( hDev, IKP_IMAGE_HEIGHT, &nImageHeight ) == false )
{
    ErrorMessage( "IKapGetInfo", "IKP_IMAGE_HEIGHT" );
}

// 获取数据格式
if ( IKapGetInfo( hDev, IKP_DATA_FORMAT, &nDataFormat ) == false )
{
    ErrorMessage( "IKapGetInfo", "IKP_DATA_FORMAT" );
}

// 获取图像类型
if ( IKapGetInfo( hDev, IKP_IMAGE_TYPE, &nImageType ) == false )
{
    ErrorMessage( "IKapGetInfo", "IKP_IMAGE_TYPE" );
}

// 获取缓冲区地址
if ( IKapGetBufferAddress( hDev, 0, &pImageBuffer ) == false )
{
    ErrorMessage( "IKapGetInfo", "Image_Buffer_Address" );
}

// 打印每个像素各个通道的值
for (int i = 0; i < nImageHeight; i++)
    for (int j = 0; j < nImageWidth; j++)
    {
        switch (nImageType)
        {
            // 灰度图像
            case IKP_IMAGE_TYPE_VAL_MONOCHROME:
                if (nDataFormat == IKP_DATA_FORMAT_VAL_8BIT)
                {
                    // 当数据格式是 8 bit 时候，每个像素的每个通道占据 1 个字节
                    char* pData = (char*)pImageBuffer + i * nImageWidth + j;
                    printf("Location(x:%d,y:%d), Gray: %x.\n", j, i, *pData);
                }
                else
                {

```



```

        // 当数据格式是 10/12/14/16 bit 时候,
        // 每个像素的每个通道占据 2 个字节
        short* pData = (short*)pImageBuffer + i * nImageWidth + j;
        printf("Location(x:%d,y:%d), Gray: %x.\n", j, i, *pData);
    }
    break;

// 对于真彩 RGB 相机或者 BGR 相机, 像素按照 B=>G=>R 的顺序在内存排列
case IKP_IMAGE_TYPE_VAL_RGB:
case IKP_IMAGE_TYPE_VAL_BGR:
    if (nDataFormat == IKP_DATA_FORMAT_VAL_8BIT)
    {
        // 当数据格式是 8 bit 时候, 每个像素的每个通道占据 1 个字节
        char* pData = (char*)pImageBuffer + i * nImageWidth * 3 + j * 3;
        printf("Location(x:%d,y:%d), Blue: %x, Green: %x, Red: %x.\n",
            j, i, *pData, *(pData+1), *(pData+2));
    }
    else
    {
        // 当数据格式是 10/12/14/16 bit 时候,
        // 每个像素的每个通道占据 2 个字节
        short* pData = (short*)pImageBuffer + i * nImageWidth * 3 + j * 3;
        printf("Location(x:%d,y:%d), Blue: %x, Green: %x, Red: %x.\n",
            j, i, *pData, *(pData+1), *(pData+2));
    }
    break;

// 对于真彩 RGBC 相机或者 BGRC 相机,
// 像素按照 B=>G=>R=>C 的顺序在内存排列
case IKP_IMAGE_TYPE_VAL_RGBC:
case IKP_IMAGE_TYPE_VAL_BGRC:
    if (nDataFormat == IKP_DATA_FORMAT_VAL_8BIT)
    {
        // 当数据格式是 8 bit 时候, 每个像素的每个通道占据 1 个字节
        char* pData = (char*)pImageBuffer + i * nImageWidth * 4 + j * 4;
        printf("Location(x:%d,y:%d), Blue: %x, Green: %x, Red: %x, Clear: %x.\n",
            j, i, *pData, *(pData+1), *(pData+2), *(pData+3));
    }
    else
    {
        // 当数据格式是 10/12/14/16 bit 时候,
        // 每个像素的每个通道占据 2 个字节
        short* pData = (short*)pImageBuffer + i * nImageWidth * 4 + j * 4;
        printf("Location(x:%d,y:%d), Blue: %x, Green: %x, Red: %x.\n", j, i,

```

```
        *pData, *(pData+1), *(pData+2), *(pData+3));  
    }  
    break;  
}  
}  
}
```

8 配置参数

8.1 配置参数概览

注意： Vulcan-CXP 采集卡与 Vulcan-SFP-2 采集卡的配置参数是共用的。

定义	值	代表概要
<u>IKP_IMAGE_WIDTH</u>	0x10000001	图像宽度
<u>IKP_IMAGE_HEIGHT</u>	0x10000002	图像高度
<u>IKP_DATA_FORMAT</u>	0x10000003	图像数据格式
<u>IKP_BOARD_BIT</u>	0x10000004	图像位宽
<u>IKP_TIME_OUT</u>	0x10000005	超时时间 (ms)
<u>IKP_SCAN_TYPE</u>	0x10000006	相机扫描类型
<u>IKP_FPGA_VERSION</u>	0x10000007	采集卡固件版本
<u>IKP_INTERNAL_BUFFER_SIZE</u>	0x10000008	系统预留缓冲区大小
<u>IKP_FRAME_SIZE</u>	0x10000009	每帧图像大小
<u>IKP_IMAGE_TYPE</u>	0x1000000a	图像类型
<u>IKP_FRAME_COUNT</u>	0x1000000b	图像缓冲区帧数
<u>IKP_FRAME_TRANSFER_MODE</u>	0x1000000c	图像缓冲区传输模式
<u>IKP_FRAME_AUTO_CLEAR</u>	0x1000000d	图像缓冲区自动清空机制
<u>IKP_GRAB_MODE</u>	0x1000000e	图像序列采集模式
<u>IKP_FRAME_TIME_STAMP_LOW</u>	0x1000000f	上帧图像采集完成时间低字节
<u>IKP_FRAME_TIME_STAMP_HIGH</u>	0x10000010	上帧图像采集完成时间高字节
<u>IKP_BLOCK_TIME_STAMP_LOW</u>	0x10000011	图像单元采集完成时间低字节
<u>IKP_BLOCK_TIME_STAMP_HIGH</u>	0x10000012	图像单元采集完成时间高字节
<u>IKP_TAP_NUMBER</u>	0x10000013	相机 Tap 个数
<u>IKP_TAP_ARRANGEMENT</u>	0x10000014	相机 Tap 排列方式
<u>IKP_BAYER_PATTERN</u>	0x10000015	Bayer 图像解析模式
<u>IKP_PIXEL_CLOCK</u>	0x10000016	CL 采集卡像素时钟【已弃用】
<u>IKP_DATA_VALID_ENABLE</u>	0x10000017	CL 采集卡数据使能【已弃用】
<u>IKP_CC1_SOURCE</u>	0x10000018	CL 采集卡 CC1 信号源
<u>IKP_CC2_SOURCE</u>	0x10000019	CL 采集卡 CC2 信号源
<u>IKP_CC3_SOURCE</u>	0x1000001a	CL 采集卡 CC3 信号源
<u>IKP_CC4_SOURCE</u>	0x1000001b	CL 采集卡 CC4 信号源
<u>IKP_BOARD_TRIGGER_MODE</u>	0x1000001c	采集卡触发模式
<u>IKP_BOARD_TRIGGER_SOURCE</u>	0x1000001d	采集卡触发信号源
<u>IKP_GENERAL_INPUT1_SAMPLE_MODE</u>	0x1000001e	CL 采集卡通用输入信号 1 采样模式【已弃用】
<u>IKP_GENERAL_INPUT1_PROTECT_MODE</u>	0x1000001f	CL 采集卡通用输入信号 1 脉冲保护模式【已弃用】
<u>IKP_GENERAL_INPUT1_MINIMUM_INTERVAL</u>	0x10000020	CL 采集卡通用输入信号 1 最小脉冲间隔【已弃用】

<u>IKP_GENERAL_INPUT2_SAMPLE_MODE</u>	0x10000021	CL 采集卡通用输入信号 2 采样模式【已弃用】
<u>IKP_GENERAL_INPUT2_PROTECT_MODE</u>	0x10000022	CL 采集卡通用输入信号 2 脉冲保护模式【已弃用】
<u>IKP_GENERAL_INPUT2_MINIMUM_INTERVAL</u>	0x10000023	CL 采集卡通用输入信号 2 最小脉冲间隔【已弃用】
<u>IKP_SHAFT_ENCODER1_PULSE_DROP</u>	0x10000024	编码器分频系数
<u>IKP_SHAFT_ENCODER_PROTECT_MODE</u>	0x10000025	CL 采集卡编码器脉冲保护模式【已弃用】
<u>IKP_SHAFT_ENCODER1_MINIMUM_INTERVAL</u>	0x10000026	CL 采集卡编码器最小脉冲间隔【已弃用】
<u>IKP_INTEGRATION_TRIGGER_SOURCE</u>	0x10000027	积分触发信号源
<u>IKP_INTEGRATION_TRIGGER_FREQUENCY</u>	0x10000028	积分触发信号频率
<u>IKP_STROBE_TRIGGER_SOURCE</u>	0x10000029	闪光触发信号源
<u>IKP_BOARD_SYNC_OUTPUT1_SOURCE</u>	0x1000002a	CL 采集卡多卡同步输出 1 信号源
<u>IKP_BOARD_SYNC_OUTPUT2_SOURCE</u>	0x1000002b	CL 采集卡多卡同步输出 2 信号源
<u>IKP_GENERAL_OUTPUT1_SOURCE</u>	0x1000002c	通用输出信号 1 信号源
<u>IKP_GENERAL_OUTPUT2_SOURCE</u>	0x1000002d	通用输出信号 2 信号源
<u>IKP_INTEGRATION_METHOD</u>	0x1000002e	积分控制方法索引
<u>IKP_INTEGRATION_PARAM1</u>	0x1000002f	积分控制方法参数 1
<u>IKP_INTEGRATION_PARAM2</u>	0x10000030	积分控制方法参数 2
<u>IKP_INTEGRATION_PARAM3</u>	0x10000031	积分控制方法参数 3
<u>IKP_INTEGRATION_PARAM4</u>	0x10000032	积分控制方法参数 4
<u>IKP_INTEGRATION_POLARITY1</u>	0x10000033	积分控制方法信号 1 极性
<u>IKP_INTEGRATION_POLARITY2</u>	0x10000034	积分控制方法信号 2 极性
<u>IKP_STROBE_METHOD</u>	0x10000035	闪光控制方法索引
<u>IKP_STROBE_PARAM1</u>	0x10000036	闪光控制方法参数 1
<u>IKP_STROBE_PARAM2</u>	0x10000037	闪光控制方法参数 2
<u>IKP_STROBE_PARAM3</u>	0x10000038	闪光控制方法参数 3
<u>IKP_STROBE_PARAM4</u>	0x10000039	闪光控制方法参数 4
<u>IKP_STROBE_POLARITY</u>	0x1000003a	闪光控制方法信号极性
<u>IKP_GENERAL_OUTPUT1_POLARITY</u>	0x1000003b	通用输出信号 1 极性
<u>IKP_GENERAL_OUTPUT1_DELAY</u>	0x1000003c	CL 采集卡通用输出信号 1 输出延时【已弃用】
<u>IKP_GENERAL_OUTPUT2_POLARITY</u>	0x1000003d	通用输出信号 2 极性
<u>IKP_GENERAL_OUTPUT2_DELAY</u>	0x1000003e	CL 采集卡通用输出信号 2 输出延时【已弃用】
<u>IKP_GENERAL_INPUT1_TRIGGER_MODE</u>	0x1000003f	通用输入信号 1 触发模式
<u>IKP_GENERAL_INPUT2_TRIGGER_MODE</u>	0x10000040	通用输入信号 2 触发模式
<u>IKP_BOARD_SYNC1_TRIGGER_MODE</u>	0x10000041	CL 采集卡多卡同步信号 1 触发模式
<u>IKP_BOARD_SYNC2_TRIGGER_MODE</u>	0x10000042	CL 采集卡多卡同步信号 2 触发模

		式
<u>IKP_SHAFT_ENCODER1_CHANNEL</u>	0x10000043	编码器 A/B 通道选择
<u>IKP_SHAFT_ENCODER1_MULTIPLY_FACTOR</u>	0x10000044	编码器倍频系数
<u>IKP_PCB_VERSION</u>	0x10000045	CL 采集卡 PCB 版本号
<u>IKP_LVAL_FILTER</u>	0x10000046	CL 采集卡 LVAL 滤波控制
<u>IKP_FRAME_TRANSFER_PERIOD</u>	0x10000047	CL 采集卡面阵图像 DMA 传输周期数
<u>IKP_LINE_TRANSFER_PERIOD</u>	0x10000048	CL 采集卡线阵图像 DMA 传输周期数
<u>IKP_FPGA_EXTERNAL_TRIGGER_TIMEOUT</u>	0x10000049	CL 采集卡外触发超时系数
<u>IKP_IMAGE_OFFSET_X</u>	0x10000050	相机 Tap 水平偏移
<u>IKP_GENERAL_INPUT1_POLARITY</u>	0x10000051	通用输入信号 1 极性
<u>IKP_GENERAL_INPUT1_MIN_WIDTH</u>	0x10000052	通用输入信号 1 最小脉冲宽度
<u>IKP_GENERAL_INPUT2_POLARITY</u>	0x10000053	通用输入信号 2 极性
<u>IKP_GENERAL_INPUT2_MIN_WIDTH</u>	0x10000054	通用输入信号 2 最小脉冲宽度
<u>IKP_CAMRAR_PIXEL_CLOCK</u>	0x10000055	CL 相机时钟频率
<u>IKP_PCIE_KERNEL_BLOCK_SIZE</u>	0x10000056	CL 采集卡 PCIe 内部缓冲区块大小
<u>IKP_SOFTWARE_TRIGGER_WIDTH</u>	0x10000057	软件触发脉冲宽度
<u>IKP_SOFTWARE_TRIGGER_PERIOD</u>	0x10000058	CL 采集卡软件触发脉冲周期
<u>IKP_SOFTWARE_TRIGGER_COUNT</u>	0x10000059	CL 采集卡软件触发次数
<u>IKP_SOFTWARE_TRIGGER_START</u>	0x10000060	CL 采集卡开始软件触发
<u>IKP_SOFTWARE_TRIGGER_DELAY</u>	0x10000061	软件触发初始延迟时间
<u>IKP_SOFTWARE_TRIGGER_POLARITY</u>	0x10000062	软件触发极性
<u>IKP_GRAB_STATUS</u>	0x10000063	采集卡采集状态
<u>IKP_CHECK_FRAME_VALID_SIGNAL</u>	0x10000064	CL 采集卡是否校验 FVAL 信号
<u>IKP_PIXEL_CLOCK_POLLING_TIME</u>	0x10000065	CL 相机像素时钟轮询时间
<u>IKP_SOFTWARE_TRIGGER_SYNC_MODE</u>	0x10000066	CL 采集卡软件触发同步模式
<u>IKP_HARDWARE_TRIGGER_GENERAL_INPUT1_DELAY</u>	0x10000067	外部触发通用输入信号 1 延迟
<u>IKP_HARDWARE_TRIGGER_GENERAL_INPUT2_DELAY</u>	0x10000068	外部触发通用输入信号 2 延迟
<u>IKP_IMAGE_ROI_X</u>	0x10000069	CL 采集卡图像感兴趣区域水平偏移
<u>IKP_SHAFT_ENCODER1_MIN_WIDTH</u>	0x10000070	CL 采集卡编码器输入信号最小脉冲宽度
<u>IKP_SHAFT_ENCODER1_VALID_DIRECTION</u>	0x10000071	CL 采集卡编码器输入信号有效方向控制
<u>IKP_SHAFT_ENCODER1_REVERSE_COMPENSATION</u>	0x10000072	CL 采集卡编码器输入信号反向输入补偿功能
<u>IKP_FRAME_SIZE_64_LOW</u>	0x10000073	每帧图像低 32 位大小
<u>IKP_FRAME_SIZE_64_HIGH</u>	0x10000074	每帧图像高 32 位大小
<u>IKP_BOARD_TRIGGER_OUTTER_MODE_FRAME_COUNT</u>	0x10000075	CL 采集卡外触发模式触发帧数

<u>IKP_SHAFT_ENCODER1_QUAD_FREQUENCY_SOURCE_TYPE</u>	0x10000076	CL 采集卡编码器 4 倍频以上信号来源
<u>IKP_CURRENT_BUFFER_INDEX</u>	0x10000077	当前缓冲区帧索引
<u>IKP_FPGA_SERIAL_NUMBER</u>	0x20000001	采集卡序列号
<u>IKP_PCIE_LINK_STATE</u>	0x20000002	采集卡 PCIe 链路状态
<u>IKP_PCIE_SPEED_MISS_REQUIREMENT</u>	0x20000003	CL 采集卡 PCIe 链路速度不足导致的丢帧次数
<u>IKP_PCI_CONFIGURATION</u>	0x20000004	采集卡 PCIe 配置空间
<u>IKP_HARDWARE_TRIGGER_GENERAL_INPUT_DELAY_MODE</u>	0x20000005	CL 采集卡通用输入信号延迟模式
<u>IKP_HARDWARE_TRIGGER_GENERAL_INPUT1_DELAY_IN_LINES</u>	0x20000006	CL 采集卡通用输入信号 1 行延迟
<u>IKP_HARDWARE_TRIGGER_GENERAL_INPUT2_DELAY_IN_LINES</u>	0x20000007	CL 采集卡通用输入信号 2 行延迟
<u>IKP_EVENT_INPUT_INTERNAL_COUNT</u>	0x20000008	采集卡内部信号计数器
<u>IKP_EVENT_INPUT_GENERAL_1_COUNT</u>	0x20000009	采集卡通用输入信号 1 计数器
<u>IKP_EVENT_INPUT_GENERAL_2_COUNT</u>	0x2000000a	采集卡通用输入信号 2 计数器
<u>IKP_EVENT_INPUT_SHAFT_ENCODER_A_COUNT</u>	0x2000000b	采集卡编码器 A 通道信号计数器
<u>IKP_EVENT_INPUT_SHAFT_ENCODER_B_COUNT</u>	0x2000000c	采集卡编码器 B 通道信号计数器
<u>IKP_EVENT_INPUT_BOARD_SYNC_IN_1_COUNT</u>	0x2000000d	采集卡板间同步信号 1 计数器
<u>IKP_EVENT_INPUT_INTEGRATION_SIG_1_COUNT</u>	0x2000000e	采集卡积分控制信号 1 计数器
<u>IKP_EVENT_INPUT_INTEGRATION_SIG_2_COUNT</u>	0x2000000f	采集卡积分控制信号 2 计数器
<u>IKP_RCV_MORE_DATA_IN_TRIGGER_MODE</u>	0x20000010	CL 采集卡行间超时情况下是否丢弃已采集的未完成帧的数据
<u>IKP_DISABLE_IO_EVENT</u>	0x20000011	禁用 IO 事件
<u>IKP_PUBLIC_VERSION_SUPPORT</u>	0x20000012	是否支持公版库
<u>IKP_SHAFT_ENCODER1_REVERSE_COMPENSATION_LIMIT</u>	0x20000013	CL 采集卡编码器反转补偿最大值
<u>IKP_SHAFT_ENCODER1_CLOCK_DUTY_COMPENSATION_TYPE</u>	0x20000014	CL 采集卡行信号占空比补偿功能类型
<u>IKP_SHAFT_ENCODER1_CLOCK_DUTY_COMPENSATION_WIDTH</u>	0x20000015	CL 采集卡行信号占空比不是 50% 时的补偿宽度
<u>IKP_CL_VALID_COLUMN</u>	0x20000016	CL 采集卡有效列
<u>IKP_CL_SIGNAL_ENHANCE_MODE</u>	0x20000017	CL 采集卡信号增强模式
<u>IKP_CL_LONG_DISTANCE_TRANSMISSION</u>	0x20000018	K6 采集卡远距离传输功能
<u>IKP_FPGA_SERIAL_NUMBER_HIGH</u>	0x20000019	采集卡序列号高 32 位
<u>IKP_CXP_TEST_IMAGE</u>	0x30000001	CXP 采集卡测试图像
<u>IKP_CXP_TRIGGER_OUTPUT_SELECTOR</u>	0x30000002	CXP 采集卡触发输出选择器
<u>IKP_LAST_FRAME_INDEX</u>	0x30000003	最新帧索引
<u>IKP_CXP_VOLTAGE_SUPPLY_STATUS</u>	0x30000004	CXP 采集卡电压源状态
<u>IKP_CXP_POWER_SWITCH</u>	0x30000005	CXP 采集卡 PoCXP 功能开关
<u>IKP_CXP_POWER_STATUS</u>	0x30000006	CXP 采集卡电源状态

<u>IKP_CXP_SENSE_CURRENT</u>	0x30000007	CXP 采集卡当前选中通道的供电电流
<u>IKP_CXP_BUS_VOLTAGE</u>	0x30000008	CXP 采集卡当前选中通道的供电电压
<u>IKP_CXP_RESET_OCP</u>	0x30000009	CXP 采集卡过流保护重置
<u>IKP_CXP_SENSE_VOLTAGE_HIGH</u>	0x3000000a	CXP 采集卡检测状态监测电压上限
<u>IKP_CXP_SENSE_VOLTAGE_LOW</u>	0x3000000b	CXP 采集卡检测状态监测电压下限
<u>IKP_CXP_SUPPLY_VOLTAGE_LOW</u>	0x3000000c	CXP 采集卡供电状态监测电压下限
<u>IKP_CXP_SUPPLY_CURRENT_LOW</u>	0x3000000d	CXP 采集卡供电状态监测电流下限
<u>IKP_CXP_FPGA_FRAME_TIMEOUT</u>	0x3000000e	采集卡帧触发超时时间【已弃用】
<u>IKP_CXP_FPGA_FRAME_TIMEOUT_MULTIPLE</u>	0x3000000f	采集卡帧触发超时时间【已弃用】
<u>IKP_CXP_CRC_ERROR_COUNT</u>	0x30000010	CXP 采集卡 CRC 错误个数
<u>IKP_CXP_PoCXP_CHANNEL</u>	0x30000011	CXP 采集卡 PoCXP 通道
<u>IKP_CXP_PoCXP_LOCKED_VOLTAGE</u>	0x30000012	CXP 采集卡切换至高压前检测到的 PoCXP 电压值
<u>IKP_CXP_SHAFT_ENCODER_DEBOUNCE</u>	0x30000013	CXP 采集卡编码器输入信号去抖窗口长度
<u>IKP_CXP_SHAFT_ENCODER_TICK_MODE</u>	0x30000014	CXP 采集卡编码器计数器模式
<u>IKP_CXP_SHAFT_ENCODER_TICK_MAX</u>	0x30000015	CXP 采集卡编码器的计数器最大值
<u>IKP_CXP_SHAFT_ENCODER_TICK_RESET</u>	0x30000016	CXP 采集卡编码器计数器重置
<u>IKP_CXP_SHAFT_ENCODER_TICK_COUNT</u>	0x30000017	CXP 采集卡编码器计数器当前值
<u>IKP_CXP_SHAFT_ENCODER_REVERSE_MODE</u>	0x30000018	CXP 采集卡编码器工作模式
<u>IKP_CXP_SHAFT_ENCODER_REVERSE_MAX</u>	0x30000019	CXP 采集卡编码器反向计数器最大值
<u>IKP_CXP_SHAFT_ENCODER_REVERSE_RESET</u>	0x30000020	CXP 采集卡编码器反向计数器重置
<u>IKP_CXP_SHAFT_ENCODER_REVERSE_COUNT</u>	0x30000021	CXP 采集卡编码器反向计数器当前值
<u>IKP_CXP_GENERAL_OUTPUT1_THRESHOLD</u>	0x30000022	CXP 采集卡通用输出信号 1 阈值
<u>IKP_CXP_GENERAL_OUTPUT2_THRESHOLD</u>	0x30000023	CXP 采集卡通用输出信号 2 阈值
<u>IKP_CXP_FIRMWARE_TYPE</u>	0x30000024	CXP 采集卡固件类型
<u>IKP_CXP_TEMPERATURE</u>	0x30000025	CXP 采集卡温度
<u>IKP_CXP_GENERAL_INPUT_THRESHOLD</u>	0x30000026	CXP 采集卡通用输入信号阈值
<u>IKP_CXP_GENERAL_OUTPUT1_SOURCE_CHANNEL</u>	0x30000027	CXP 采集卡通用输出信号 1 通道
<u>IKP_CXP_GENERAL_OUTPUT2_SOURCE_CHANNEL</u>	0x30000028	CXP 采集卡通用输出信号 2 通道
<u>IKP_CXP_GENERAL_INPUT1_TYPE</u>	0x30000029	CXP 采集卡通用输入信号 1 模式
<u>IKP_CXP_GENERAL_INPUT2_TYPE</u>	0x3000002a	CXP 采集卡通用输入信号 2 模式

<u>IKP_CXP_TRANSFER_CHANNEL_SELECTOR</u>	0x3000002b	CXP 采集卡传输通道选择器
<u>IKP_CXP_CHANNEL_LOST_COUNT</u>	0x3000002c	CXP 传输包丢失数量
<u>IKP_CXP_CHANNEL_ERROR_COUNT</u>	0x3000002d	CXP 传输包错误数量
<u>IKP_TRIGGER_FRAME_ACTIVE_MODE</u>	0x3000002e	帧触发不定长采集模式
<u>IKP_JPEG_COMPRESS_ENABLE</u>	0x3000002f	使能 JPEG 压缩模式
<u>IKP_JPEG_COMPRESS_QUALITY</u>	0x30000030	JPEG 压缩质量
<u>IKP_CXP_FRAME_BURST_COUNT</u>	0x30000031	CXP 采集卡接受一个触发信号后产生的给相机的连续信号的个数
<u>IKP_CXP_FRAME_BURST_PERIOD</u>	0x30000032	CXP 采集卡产生连续信号的周期
<u>IKP_CXP_CHANNEL_CRC_ERROR_COUNT</u>	0x30000033	CXP 采集卡通道 CRC 错误发生次数
<u>IKP_CXP_TRIG_EDGE_MODE</u>	0x30000034	CXP 采集卡边沿触发模式
<u>IKP_CXP_DATA_PACKED_TRANSFER</u>	0x30000035	CXP 采集卡压缩传输模式
<u>IKP_CXP_TRIG_LEVEL</u>	0x30000036	光口采集卡传输给相机的触发包电平极性

8.2 配置参数详细说明

IKP_IMAGE_WIDTH 图像宽度

属性	Get	Set	vlcf parameter
	支持	支持	IMAGE_WIDTH
SetInfo	hDev	设备句柄	
	uType	IKP_IMAGE_WIDTH (0x10000001)	
	nValue	设置采集图像的宽度（水平像素个数），有效值 1 ~ max_width 注意： （1）max_width 由相机型号决定，查阅使用相机手册获得相关信息； （2）用户设置图像宽度应遵循下列原则。当接口设置为 1Tap 时，nValue 可以在 1 到 max_width 之间；当接口设置为 2Taps 时，nValue 应该是 2 的整倍数，对应关系如下所示。	
		接口 Tap 数量	nValue
		1Tap	1 ~ max_width (1 的整倍数)
		2Tap	2 ~ max_width (2 的整倍数)
		3Tap	3 ~ max_width (3 的整数倍)
		4Tap	4 ~ max_width (4 的整倍数)
		8Tap	8 ~ max_width (8 的整倍数)
		10Tap	10 ~ max_width (10 的整倍数)
GetInfo	hDev	设备句柄	
	uType	IKP_IMAGE_WIDTH (0x10000001)	
	npValue	获取采集图像的宽度	
说明	本参数表示采集卡采集图像的宽度。 对于面扫描相机，一帧图像占据的内存大小不能超过 IKP_INTERNAL_BUFFER_SIZE 指定的大小。 如果采集卡设置的采集图像宽度大于相机输出图像的实际有效宽度，采集卡会采集失败。		

IKP_IMAGE_HEIGHT 图像高度

属性	Get	Set	vlcf parameter
	支持	支持	IMAGE_HEIGHT
SetInfo	hDev	设备句柄	
	uType	IKP_IMAGE_HEIGHT (0x10000002)	
	nValue	设置采集图像的高度（对于线扫描相机，该参数代表扫描行数），有效值 1~ max_height	
GetInfo	hDev	设备句柄	
	uType	IKP_IMAGE_HEIGHT (0x10000002)	
	npValue	获取采集图像的高度	
说明	本参数表示采集图像的高度，行数最大值 max_height 由相机决定，具体请查阅相		

	机手册获得相关信息。
	对于面扫描相机，本参数表示图像高度，其最大值由相机前端传感器决定，单帧图像尺寸 IKP_FRAME_SIZE 不能超过 IKP_INTERNAL_BUFFER_SIZE 指定的大小。如果设置参数大于相机输出图像的高度，则无法正常采集。
	对于线扫描相机，本参数表示采集卡每次采集的图像行数，当有 IKP_IMAGE_HEIGHT 的图像采集完成后则会触发一次 IKEvent_FrameReady 事件，参数上限为 $2^{31}-1$ 。

IKP_DATA_FORMAT 图像数据格式

属性	Get	Set	vllcf parameter	
	支持	支持	PIXEL_DEPTH	
SetInfo	hDev	设备句柄		
	uType	IKP_DATA_FORMAT (0x10000003)		
	nValue	设置采集图像的数据格式		
		IKP_DATA_FORMAT_VAL_8Bit	8bit	
		IKP_DATA_FORMAT_VAL_10Bit	10bit	
		IKP_DATA_FORMAT_VAL_12Bit	12bit	
		IKP_DATA_FORMAT_VAL_14Bit	14bit	
		IKP_DATA_FORMAT_VAL_16Bit	16bit	
GetInfo	hDev	设备句柄		
	uType	IKP_DATA_FORMAT (0x10000003)		
	npValue	获取采集图像的数据格式		
说明	本参数表示采集图像的数据格式，即相机产生图像时的接口有效位宽。 如果 IKP_DATA_FORMAT 参数与相机输出图像的数据格式不一致，采集卡会采集图像失败。			

IKP_BOARD_BIT 图像位宽

属性	Get	Set	vllcf parameter	
	支持	不支持	无	
SetInfo	本参数不支持手动设定，由 IKP_DATA_FORMAT 和 IKP_IMAGE_TYPE 计算得到			
GetInfo	hDev	设备句柄		
	uType	IKP_BOARD_BIT (0x10000004)		
	npValue	获取图像在内存中的有效位宽，由 IKP_DATA_FORMAT 和 IKP_IMAGE_TYPE 参数计算得到，对应关系如下。		
		IKP_DATA_FORMAT	IKP_IMAGE_TYPE	IKP_BOARD_BIT
		8	0	8
		10/12/14/16	0	16
		8	1	24

		10/12/14/16	1	48
		8	2	32
		10/12/14/16	2	64
		8	3	24
		10/12/14/16	3	48
		8	4	32
		10/12/14/16	4	64
说明	本参数表示图像在内存中的有效位宽，即单个像素在 PC 内存中占据的位宽。具体信息参见“ 4.3 设置图像缓冲区 ”。			

IKP_TIME_OUT 超时时间（ms）

属性	Get	Set	vlcf parameter
	支持	支持	TIME_OUT
SetInfo	hDev	设备句柄	
	uType	IKP_TIME_OUT (0x10000005)	
	nValue	设置采集卡每次传输等待的超时时间，单位 ms	
		0	没有等待时间（立即返回）
		-1(0xFFFFFFFF)	无限等待
GetInfo	hDev	设备句柄	
	uType	IKP_TIME_OUT (0x10000005)	
	npValue	获取采集卡每次传输等待的超时时间，单位 ms	
说明	对单次图像采集，本参数表示单次采集等待的超时时间；对于连续采集，本参数表示两帧间隔的最大等待时间。对于较大尺寸的图像，可能需要较长的时间完成一次图像采集，可以通过增长采集时间保证一帧完整图像的采集。		

IKP_SCAN_TYPE 相机扫描类型

属性	Get	Set	vlcf parameter
	支持	支持	SCAN_TYPE
SetInfo	hDev	设备句柄	
	uType	IKP_SCAN_TYPE (0x10000006)	
	nValue	设置相机扫描类型	
		IKP_SCAN_TYPE_VAL_LINEAR	线扫描相机
		IKP_SCAN_TYPE_VAL_AREA	面扫描相机
GetInfo	hDev	设备句柄	
	uType	IKP_SCAN_TYPE (0x10000006)	
	npValue	获取相机扫描类型	
说明	本参数表示相机扫描类型。		

IKP_FPGA_VERSION 采集卡固件版本

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_FPGA_VERSION (0x10000007)	
	npValue	获取当前采集卡固件版本号	
说明	本参数表示采集卡固件版本号。		

IKP_INTERNAL_BUFFER_SIZE 系统预留缓冲区大小

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_INTERNAL_BUFFER_SIZE (0x10000008)	
	npValue	获取当前系统预留图像缓存区域大小，有效值 32/64/128 MB	
说明	本参数表示系统预留的图像缓存区域大小。 对于面扫描相机，单帧图像的大小(IKP_FRAME_SIZE)不能超过该参数，否则无法正常采集图像。		

IKP_FRAME_SIZE 每帧图像大小

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定，由 IKapBoard 自动计算得到		
GetInfo	hDev	设备句柄	
	uType	IKP_FRAME_SIZE (0x10000009)	
	npValue	返回每帧图像在缓存区中的尺寸，单位 byte，有效范围 1 ~ 2 ³¹ -1	
说明	本参数表示返回每帧图像在缓存区中的大小。		
	图像有效缓存区域大小计算方式： IKP_FRAME_SIZE= (IKP_IMAGE_WIDTH*IKP_IMAGE_HEIGHT*IKP_BOARD_BIT / 8) (bytes)		

IKP_IMAGE_TYPE 图像类型

属性	Get	Set	vlcf parameter
----	-----	-----	----------------

	支持	支持	IMAGE_TYPE
SetInfo	hDev	设备句柄	
	uType	IKP_IMAGE_TYPE (0x1000000a)	
	nValue	设置采集图像的图像类型	
		IKP_IMAGE_TYPE_VAL_MONOCHROME	灰度或者 Bayer 图像
		IKP_IMAGE_TYPE_VAL_RGB	彩色 RGB 图像
		IKP_IMAGE_TYPE_VAL_RGB_C	彩色 RGBC 图像，其中 C 是图像的灰度信息
		IKP_IMAGE_TYPE_VAL_BGR	彩色 BGR 图像
IKP_IMAGE_TYPE_VAL_BGR_C	彩色 BGRC 图像，其中 C 是图像的灰度信息		
GetInfo	hDev	设备句柄	
	uType	IKP_IMAGE_TYPE (0x1000000a)	
	npValue	获取采集图像的图像类型	
说明	本参数表示采集图像的图像类型，一帧图像占据的内存大小不能超过 IKP_INTERNAL_BUFFER_SIZE 指定的大小。如果 IKP_IMAGE_TYPE 参数与相机输出图像的图像类型不一致，采集卡则会采集图像失败。		

IKP_FRAME_COUNT 图像缓冲区帧数

属性	Get	Set	vlcf parameter
	支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_FRAME_COUNT (0x1000000b)	
	nValue	设置图像缓冲区的帧数，如果系统没有足够的内存，则无法进行图像采集	
GetInfo	hDev	设备句柄	
	uType	IKP_FRAME_COUNT (0x1000000b)	
	npValue	获取图像缓冲区的帧数	
说明	<p>本参数表示图像缓冲区的缓存帧数。</p> <p>当用户调用该函数时，IKapBoard 自动申请\管理图像缓冲区，用户无需释放该段内存缓冲区域。</p> <p>请在设置该参数前先设置 IKP_IMAGE_WIDHT，IKP_IMAGE_HEIGHT，IKP_DATA_FORMAT，否则可能会出现参数冲突导致采集失败。</p> <p>用户应该校验设置 IKP_FRAME_COUNT 的返回值，如果当前系统资源不足则无法申请到足够的内存数据则无法开始采集工作，用户应该减少 IKP_FRAME_COUNT 或者和图像尺寸的相关参数来保证内存申请的成功。</p> <p>请不要在连续采集的过程中设置该参数，这可能导致采集失败。</p>		

IKP_FRAME_TRANSFER_MODE 图像缓冲区传输模式

属性	Get	Set	vlcf parameter
	支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_FRAME_TRANSFER_MODE (0x1000000c)	
	nValue	设置图像缓冲区的循环模式	
		IKP_FRAME_TRANSFER_SYNCHRONOUS_NEXT_EMPTY_WITH_PROTECT	同步保护传输模式
GetInfo	hDev	设备句柄	
	uType	IKP_FRAME_TRANSFER_MODE (0x1000000c)	
	npValue	获取图像缓冲区的循环模式	
说明	本参数表示图像缓冲区的循环模式，具体描述参见章节 6。		

IKP_FRAME_AUTO_CLEAR 图像缓冲区自动清空机制

属性	Get	Set	vlcf parameter
	支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_FRAME_AUTO_CLEAR (0x1000000d)	
	nValue	设置自动清空缓存机制是否使能	
		IKP_FRAME_AUTO_CLEAR_VAL_DISABLE	禁止自动清空机制
		IKP_FRAME_AUTO_CLEAR_VAL_ENABLE	使能自动清空机制
GetInfo	hDev	设备句柄	
	uType	IKP_FRAME_AUTO_CLEAR (0x1000000d)	
	npValue	获取自动清空机制是否使能	
说明	<p>本参数表示 IKapBoard 的自动清空机制。</p> <p>IKapBoard 默认在 IKEvent_FrameReady 回调函数执行后自动清空缓冲区，当缓冲区为空时，新的图像数据可以写入该缓冲区内。因此当用户未及时取走缓冲区数据时，旧的图像数据将会被覆盖。</p> <p>为了确保图像的正确性，用户可以禁用自动清除机制并且选择 IKP_FRAME_TRANSFER_MODE 为同步保护传输模式，此时新的图像数据将不会覆盖原有数据直到缓冲区变为空。</p> <p>在禁用自动清除机制时，用户需要在使用完缓冲区后调用 IKapReleaseBuffer() 来释放缓冲区。</p> <p>缓存自动清空机制仅使用于连续图像采集过程。</p>		

IKP_GRAB_MODE 图像序列采集模式

属性	Get	Set	vlcf parameter
	支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_GRAB_MODE (0x1000000e)	
	nValue	设置图像序列采集模式，可以是以下值	
		IKP_GRAB_BLOCK	对于有限图像序列，IKapStartGrab()函数将会阻塞直到所有图像传输完毕
		IKP_GRAB_NON_BLOCK	IKapStartGrab()非阻塞式调用，不等待图像开始传输立即返回
GetInfo	hDev	设备句柄	
	uType	IKP_GRAB_MODE (0x1000000e)	
	npValue	获取图像序列采集模式	
说明	本参数表示图像序列采集模式，可以是阻塞或者非阻塞式调用。		

IKP_FRAME_TIME_STAMP_LOW 上帧图像采集完成时间低字节

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_FRAME_TIME_STAMP_LOW (0x100000f)	
	npValue	获取上帧图像采集完成时间戳的低字节	
说明	本参数表示上帧图像采集完成时间戳的低字节，时间戳信息占据 64bit。		

IKP_FRAME_TIME_STAMP_HIGH 上帧图像采集完成时间高字节

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_FRAME_TIME_STAMP_HIGH (0x1000010)	
	npValue	获取上帧图像采集完成时间戳的高字节	
说明	本参数表示上帧图像采集完成时间戳的高字节，时间戳信息占据 64bit。		

IKP_BLOCK_TIME_STAMP_LOW 图像单元采集完成时间低字节

属性	Get	Set	vlcf parameter
----	-----	-----	----------------

	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_BLOCK_TIME_STAMP_LOW(0x1000011)	
	npValue	获取单元图像采集完成时间戳的低字节	
说明	本参数表示图像块采集完成时间戳的低字节，时间戳信息占据 64bit。		

IKP_BLOCK_TIME_STAMP_HIGH 图像单元采集完成时间高字节

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_BLOCK_TIME_STAMP_HIGH(0x1000012)	
	npValue	获取单元图像采集完成时间戳的高字节	
说明	本参数表示图像块采集完成时间戳的高字节，时间戳信息占据 64bit。		

IKP_TAP_NUMBER 相机 Tap 个数

属性	Get	Set	vlcf parameter
	支持	支持	TAP_NUMBER
SetInfo	hDev	设备句柄	
	uType	IKP_TAP_NUMBER (0x10000013)	
	nValue	设置相机 Tap 个数	
GetInfo	hDev	设备句柄	
	uType	IKP_TAP_NUMBER (0x10000013)	
	npValue	获取相机 Tap 个数	
说明	本参数表示相机 Tap 个数，设置该参数时请参考具体相机文档。		

IKP_TAP_ARRANGEMENT 相机 Tap 排列方式

属性	Get	Set	vlcf parameter
	支持	支持	TAP_ARRANGEMENT
SetInfo	hDev	设备句柄	
	uType	IKP_TAP_ARRANGEMENT (0x10000014)	
	nValue	设置相机 Tap 排列方式	
GetInfo	hDev	设备句柄	
	uType	IKP_TAP_ARRANGEMENT (0x10000014)	
	npValue	获取相机 Tap 排列方式	
说明	本参数表示相机 Tap 排列方式，在不同的 IKP TAP NUMBER 取值下，该参数表		

示的图像排列方式会有所不同。

IKP_BAYER_PATTERN Bayer 图像解析模式

属性	Get	Set	vlcf parameter
	支持	支持	BAYER_PATTERN
SetInfo	hDev	设备句柄	
	uType	IKP_BAYER_PATTERN (0x10000015)	
	nValue	设置 Bayer 图像解析模式，可以是以下值	
		IKP_BAYER_PATTERN_VAL_NULL	非 Bayer 图像
		IKP_BAYER_PATTERN_VAL_BGGR	图像左上角四个像素按照 BGGR 颜色方式排布
		IKP_BAYER_PATTERN_VAL_RGGB	图像左上角四个像素按照 RGGB 颜色方式排布
		IKP_BAYER_PATTERN_VAL_GBRG	图像左上角四个像素按照 GBRG 颜色方式排布
		IKP_BAYER_PATTERN_VAL_GRBG	图像左上角四个像素按照 GRBG 颜色方式排布
GetInfo	hDev	设备句柄	
	uType	IKP_BAYER_PATTERN (0x10000015)	
	npValue	获取 Bayer 图像解析模式	
说明	本参数表示 Bayer 图像解析模式。		

IKP_PIXEL_CLOCK CL 采集卡像素时钟【已弃用】

属性	Get	Set	vlcf parameter
	支持	支持	PIXEL_CLOCK
SetInfo	hDev	设备句柄	
	uType	IKP_PIXEL_CLOCK (0x10000016)	
	nValue	设置像素时钟频率	
GetInfo	hDev	设备句柄	
	uType	IKP_PIXEL_CLOCK (0x10000016)	
	npValue	获取像素时钟频率	
说明	本参数表示 CL 采集卡像素时钟【已弃用】。		

IKP_DATA_VALID_ENABLE CL 采集卡数据使能【已弃用】

属性	Get	Set	vlcf parameter
	支持	支持	DATA_VALID_ENABLE
SetInfo	hDev	设备句柄	

	uType	IKP_DATA_VALID_ENABLE (0x10000017)	
	nValue	设置 CL 采集卡数据有效使能标志位	
		0	不使能
		1	使能
GetInfo	hDev	设备句柄	
	uType	IKP_DATA_VALID_ENABLE (0x10000017)	
	npValue	获取 CL 采集卡数据有效使能标志位	
说明	本参数表示 CL 采集卡数据有效使能标志位【已弃用】。		

IKP_CC1_SOURCE CL 采集卡 CC1 信号源

属性	Get	Set	vlcf parameter
	支持	支持	CC1_SOURCE
SetInfo	hDev	设备句柄	
	uType	IKP_CC1_SOURCE (0x10000018)	
	nValue	设置 CL 采集卡 CC1 信号源	
		IKP_CC_SOURCE_VAL_NOT_USE	不使用 CC1 信号
		IKP_CC_SOURCE_VAL_INTEGRATION_SIGNAL1	积分方法信号 1
		IKP_CC_SOURCE_VAL_INTEGRATION_SIGNAL2	积分方法信号 2
		IKP_CC_SOURCE_VAL_LOW	低电平信号
		IKP_CC_SOURCE_VAL_HIGH	高电平信号
		IKP_CC_SOURCE_VAL_SOFTWARE	软件触发源
		IKP_CC_SOURCE_VAL_GENERAL_INPUT1	通用输入信号 1
		IKP_CC_SOURCE_VAL_GENERAL_INPUT2	通用输入信号 2
		IKP_CC_SOURCE_VAL_BOARD_SYNC1	多卡同步信号 1
		IKP_CC_SOURCE_VAL_BOARD_SYNC2	多卡同步信号 2
		IKP_CC_SOURCE_VAL_STROBE	闪光信号
		GetInfo	hDev
uType	IKP_CC1_SOURCE (0x10000018)		
npValue	获取 CL 采集卡 CC1 信号源		
说明	本参数表示 CL 采集卡 CC1 信号源。		

IKP_CC2_SOURCE CL 采集卡 CC2 信号源

属性	Get	Set	vlcf parameter
----	-----	-----	----------------

	支持	支持	CC2_SOURCE
SetInfo	hDev	设备句柄	
	uType	IKP_CC2_SOURCE (0x10000019)	
	nValue	设置 CL 采集卡 CC2 信号源	
		IKP_CC_SOURCE_VAL_NOT_USE	不使用 CC2 信号
		IKP_CC_SOURCE_VAL_INTEGRATION_SIGNAL1	积分方法信号 1
		IKP_CC_SOURCE_VAL_INTEGRATION_SIGNAL2	积分方法信号 2
		IKP_CC_SOURCE_VAL_LOW	低电平信号
		IKP_CC_SOURCE_VAL_HIGH	高电平信号
		IKP_CC_SOURCE_VAL_SOFTWARE	软件触发源
		IKP_CC_SOURCE_VAL_GENERAL_INPUT1	通用输入信号 1
		IKP_CC_SOURCE_VAL_GENERAL_INPUT2	通用输入信号 2
		IKP_CC_SOURCE_VAL_BOARD_SYNC1	多卡同步信号 1
		IKP_CC_SOURCE_VAL_BOARD_SYNC2	多卡同步信号 2
		IKP_CC_SOURCE_VAL_STROBE	闪光信号
GetInfo		hDev	设备句柄
	uType	IKP_CC2_SOURCE (0x10000018)	
	npValue	获取 CL 采集卡 CC2 信号源	
说明	本参数表示 CL 采集卡 CC2 信号源。		

IKP_CC3_SOURCE CL 采集卡 CC3 信号源

属性	Get	Set	vlcf parameter
	支持	支持	CC3_SOURCE
SetInfo	hDev	设备句柄	
	uType	IKP_CC3_SOURCE (0x1000001a)	
	nValue	设置 CL 采集卡 CC3 信号源	
		IKP_CC_SOURCE_VAL_NOT_USE	不使用 CC3 信号
		IKP_CC_SOURCE_VAL_INTEGRATION_SIGNAL1	积分方法信号 1
		IKP_CC_SOURCE_VAL_INTEGRATION_SIGNAL2	积分方法信号 2
		IKP_CC_SOURCE_VAL_LOW	低电平信号
		IKP_CC_SOURCE_VAL_HIGH	高电平信号
		IKP_CC_SOURCE_VAL_SOFTWARE	软件触发源
		IKP_CC_SOURCE_VAL_GENERAL_IN	通用输入信号 1

		NPOT1	
		IKP_CC_SOURCE_VAL_GENERAL_I NPOT2	通用输入信号 2
		IKP_CC_SOURCE_VAL_BOARD_SY NC1	多卡同步信号 1
		IKP_CC_SOURCE_VAL_BOARD_SY NC2	多卡同步信号 2
		IKP_CC_SOURCE_VAL_STROBE	闪光信号
GetInfo	hDev	设备句柄	
	uType	IKP_CC3_SOURCE (0x10000018)	
	npValue	获取 CL 采集卡 CC3 信号源	
说明	本参数表示 CL 采集卡 CC3 信号源。		

IKP_CC4_SOURCE CL 采集卡 CC4 信号源

属性	Get	Set	vlcf parameter
	支持	支持	CC4_SOURCE
SetInfo	hDev	设备句柄	
	uType	IKP_CC4_SOURCE (0x1000001b)	
	nValue	设置 CL 采集卡 CC4 信号源	
		IKP_CC_SOURCE_VAL_NOT_USE	不使用 CC4 信号
		IKP_CC_SOURCE_VAL_INTEGRATI ON_SIGNAL1	积分方法信号 1
		IKP_CC_SOURCE_VAL_INTEGRATI ON_SIGNAL2	积分方法信号 2
		IKP_CC_SOURCE_VAL_LOW	低电平信号
		IKP_CC_SOURCE_VAL_HIGH	高电平信号
		IKP_CC_SOURCE_VAL_SOFTWARE	软件触发源
		IKP_CC_SOURCE_VAL_GENERAL_I NPUT1	通用输入信号 1
		IKP_CC_SOURCE_VAL_GENERAL_I NPUT2	通用输入信号 2
		IKP_CC_SOURCE_VAL_BOARD_SY NC1	多卡同步信号 1
		IKP_CC_SOURCE_VAL_BOARD_SY NC2	多卡同步信号 2
		IKP_CC_SOURCE_VAL_STROBE	闪光信号
GetInfo	hDev	设备句柄	
	uType	IKP_CC4_SOURCE (0x10000018)	
	npValue	获取 CL 采集卡 CC4 信号源	
说明	本参数表示 CL 采集卡 CC4 信号源。		

IKP_BOARD_TRIGGER_MODE 采集卡触发模式

属性	Get	Set	vlcf parameter
	支持	支持	GRAB_MODE
SetInfo	hDev	设备句柄	
	uType	IKP_BOARD_TRIGGER_MODE (0x1000001c)	
	nValue	设置采集卡触发模式	
		IKP_BOARD_TRIGGER_MODE_V AL_INNER	自由运行模式
		IKP_BOARD_TRIGGER_MODE_V AL_OUTTER	外部触发模式
		IKP_BOARD_TRIGGER_MODE_V AL_CXP_LEVEL_VALID	CXP 采集卡电平有效
GetInfo	hDev	设备句柄	
	uType	IKP_BOARD_TRIGGER_MODE (0x1000001c)	
	npValue	获取采集卡触发模式	
说明	本参数表示采集卡采集图像数据过程中的信号触发模式。		

IKP_BOARD_TRIGGER_SOURCE 采集卡触发信号源

属性	Set	Get	vlcf parameter
	支持	支持	GRAB_TRIGGER_SOURCE
SetInfo	hDev	设备句柄	
	uType	IKP_BOARD_TRIGGER_SOURCE (0x1000001d)	
	nValue	设置采集卡外部触发时的触发信号源	
		IKP_BOARD_TRIGGER_SOURCE_VAL_G ENERAL_INPUT1	通用输入信号 1
		IKP_BOARD_TRIGGER_SOURCE_VAL_G ENERAL_INPUT2	通用输入信号 2
		IKP_BOARD_TRIGGER_SOURCE_VAL_S HAFT_ENCODER1	编码器信号
		IKP_BOARD_TRIGGER_SOURCE_VAL_B OARD_SYNC1	多卡同步信号 1
		IKP_BOARD_TRIGGER_SOURCE_VAL_B OARD_SYNC2	多卡同步信号 2
		IKP_BOARD_TRIGGER_SOURCE_VAL_I INNER_TRIGGER	内部触发信号
		IKP_BOARD_TRIGGER_SOURCE_VAL_S OFTWARE	软件触发信号
		IKP_BOARD_TRIGGER_SOURCE_VAL_I NTEGRATION_SIGNAL_1	积分信号 1
		IKP_BOARD_TRIGGER_SOURCE_VAL_I	积分信号 2

		NTEGRATION_SIGNAL_2		
GetInfo	hDev	设备句柄		
	uType	IKP_BOARD_TRIGGER_SOURCE (0x1000001d)		
	npValue	获取采集卡外部触发时的触发信号源		
说明	本参数表示采集卡外部触发时的触发信号源，只有当IKP_BOARD_TRIGGER_MODE设置为外部触发时本参数才有效。			

IKP_GENERAL_INPUT1_SAMPLE_MODE CL 采集卡通用输入信号 1 采样模式【已弃用】

属性	Get	Set	vlfw parameter	
	支持	支持	GENERAL_INPUT1_SAMPLE_MODE	
SetInfo	hDev	设备句柄		
	uType	IKP_GENERAL_INPUT1_SAMPLE_MODE (0x1000001e)		
	nValue	设置 CL 采集卡通用输入信号 1 采样模式		
		IKP_GENERAL_INPUT_SAMPLE_MODE_VAL_ACTIVE_HIGH	高电平	
		IKP_GENERAL_INPUT_SAMPLE_MODE_VAL_ACTIVE_LOW	低电平	
		IKP_GENERAL_INPUT_SAMPLE_MODE_VAL_RISING_EDGE	上升沿	
		IKP_GENERAL_INPUT_SAMPLE_MODE_VAL_FALLING_EDGE	下降沿	
GetInfo	hDev	设备句柄		
	uType	IKP_GENERAL_INPUT1_SAMPLE_MODE (0x1000001e)		
	npValue	获取 CL 采集卡通用输入信号 1 采样模式		
说明	本参数表示 CL 采集卡通用输入信号 1 采样模式【已弃用】。			

IKP_GENERAL_INPUT1_PROTECT_MODE CL 采集卡通用输入信号 1 脉冲保护模式【已弃用】

属性	Get	Set	vlfw parameter	
	支持	支持	GENERAL_INPUT1_PROTECT_MODE	
SetInfo	hDev	设备句柄		
	uType	IKP_GENERAL_INPUT1_PROTECT_MODE (0x1000001f)		
	nValue	设置 CL 采集卡通用输入信号 1 脉冲保护模式		
		IKP_GENERAL_INPUT_PROTECT_MODE_VAL_NOT_USE	不保护	
		IKP_GENERAL_INPUT_PROTECT_MODE_VAL_DELETE	删除	
		IKP_GENERAL_INPUT_PROTECT_MODE VAL MEMORIZE	记录	

GetInfo	hDev	设备句柄
	uType	IKP_GENERAL_INPUT1_PROTECT_MODE (0x1000001f)
	npValue	获取 CL 采集卡通用输入信号 1 脉冲保护模式
说明	本参数表示 CL 采集卡通用输入信号 1 脉冲保护模式【已弃用】。	

IKP_GENERAL_INPUT1_MINIMUM_INTERVAL CL 采集卡通用输入信号 1 最小脉冲间隔【已弃用】

属性	Get	Set	vlcf parameter
	支持	支持	GENERAL_INPUT1_MINIMUM_INTERVAL
SetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_INPUT1_PROTECT_MODE (0x10000020)	
	nValue	设置 CL 采集卡输入信号 1 最小脉冲间隔,取值范围为 1 ~ 40000000	
GetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_INPUT1_PROTECT_MODE (0x10000020)	
	npValue	获取 CL 采集卡通用输入信号 1 最小脉冲间隔	
说明	本参数表示 CL 采集卡通用输入信号 1 最小脉冲间隔【已弃用】。		

IKP_GENERAL_INPUT2_SAMPLE_MODE CL 采集卡通用输入信号 2 采样模式【已弃用】

属性	Get	Set	vlcf parameter	
	支持	支持	GENERAL_INPUT2_SAMPLE_MODE	
SetInfo	hDev	设备句柄		
	uType	IKP_GENERAL_INPUT2_SAMPLE_MODE (0x10000021)		
	nValue	设置 CL 采集卡通用输入信号 2 采样模式		
		IKP_GENERAL_INPUT_SAMPLE_MODE_VAL_ACTIVE_HIGH	高电平	
		IKP_GENERAL_INPUT_SAMPLE_MODE_VAL_ACTIVE_LOW	低电平	
		IKP_GENERAL_INPUT_SAMPLE_MODE_VAL_RISING_EDGE	上升沿	
		IKP_GENERAL_INPUT_SAMPLE_MODE_VAL_FALLING_EDGE	下降沿	
GetInfo	hDev	设备句柄		
	uType	IKP_GENERAL_INPUT2_SAMPLE_MODE (0x10000021)		
	npValue	获取 CL 采集卡通用输入信号 2 采样模式		
说明	本参数表示 CL 采集卡通用输入信号 2 采样模式【已弃用】。			

IKP_GENERAL_INPUT2_PROTECT_MODE CL 采集卡通用输入信号 2 脉冲保护模式【已弃用】

属性	Get	Set	vlfw parameter	
	支持	支持	GENERAL_INPUT2_PROTECT_MODE	
SetInfo	hDev	设备句柄		
	uType	IKP_GENERAL_INPUT2_PROTECT_MODE (0x10000022)		
	nValue	设置 CL 采集卡通用输入信号 2 脉冲保护模式		
		IKP_GENERAL_INPUT_PROTECT_MODE_VAL_NOT_USE	不保护	
		IKP_GENERAL_INPUT_PROTECT_MODE_VAL_DELETE	删除	
		IKP_GENERAL_INPUT_PROTECT_MODE_VAL_MEMORIZE	记录	
GetInfo	hDev	设备句柄		
	uType	IKP_GENERAL_INPUT2_PROTECT_MODE (0x10000022)		
	npValue	获取 CL 采集卡通用输入信号 2 脉冲保护模式		
说明	本参数表示 CL 采集卡通用输入信号 2 脉冲保护模式【已弃用】。			

IKP_GENERAL_INPUT2_MINIMUM_INTERVAL CL 采集卡通用输入信号 2 最小脉冲间隔【已弃用】

属性	Get	Set	vlc parameter
	支持	支持	GENERAL_INPUT2_MINIMUM_INTERVAL
SetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_INPUT2_PROTECT_MODE (0x10000023)	
	nValue	设置 CL 采集卡输入信号 2 最小脉冲间隔,取值范围为 1 ~ 40000000	
GetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_INPUT2_PROTECT_MODE (0x10000023)	
	npValue	获取 CL 采集卡通用输入信号 2 最小脉冲间隔	
说明	本参数表示 CL 采集卡通用输入信号 2 最小脉冲间隔【已弃用】。		

IKP_SHAFT_ENCODER1_PULSE_DROP 编码器分频系数

属性	Get	Set	vlfw parameter
	支持	支持	SHAFTENCODER1_PULSE_DROP
SetInfo	hDev	设备句柄	
	uType	IKP_SHAFT_ENCODER1_PULSE_DROP (0x10000024)	
	nValue	设置编码器脉冲分频系数	
GetInfo	hDev	设备句柄	
	uType	IKP SHAFT ENCODER1 PULSE DROP (0x10000024)	

	npValue	获取编码器脉冲分频系数
说明	本参数表示编码器脉冲分频系数。	

IKP_SHAFT_ENCODER1_PROTECT_MODE CL 采集卡编码器脉冲保护模式【已弃用】

属性	Get	Set	vlcf parameter	
	支持	支持	SHAFTENCODER1_PROTECT_MODE	
SetInfo	hDev	设备句柄		
	uType	IKP_SHAFT_ENCODER1_PROTECT_MODE (0x10000025)		
	nValue	设置 CL 采集卡编码器脉冲保护模式		
		IKP_SHAFT_ENCODER_PROTECT_MODE_V AL_NOT_USE	不保护	
		IKP_SHAFT_ENCODER_PROTECT_MODE_V AL_DELETE	删除	
IKP_SHAFT_ENCODER_PROTECT_MODE_V AL_MEMORIZE		记录		
GetInfo	hDev	设备句柄		
	uType	IKP_SHAFT_ENCODER1_PROTECT_MODE (0x10000025)		
	npValue	获取 CL 采集卡编码器脉冲保护模式		
说明	本参数表示 CL 采集卡编码器脉冲保护模式【已弃用】。			

IKP_SHAFT_ENCODER1_MINIMUM_INTERVAL CL 采集卡编码器最小脉冲间隔【已弃用】

属性	Get	Set	vlcf parameter
	支持	支持	SHAFT_ENCODER_MINIMUM_INTERVAL
SetInfo	hDev	设备句柄	
	uType	IKP_SHAFT_ENCODER1_MINIMUM_INTERVAL (0x10000026)	
	nValue	设置 CL 采集卡编码器最小脉冲间隔，取值范围为 1 ~ 40000000	
GetInfo	hDev	设备句柄	
	uType	IKP_SHAFT_ENCODER1_MINIMUM_INTERVAL (0x10000026)	
	npValue	获取 CL 采集卡编码器最小脉冲间隔	
说明	本参数表示 CL 采集卡编码器最小脉冲间隔【已弃用】。		

IKP_INTEGRATION_TRIGGER_SOURCE 积分触发信号源

属性	Get	Set	vlcf parameter
	支持	支持	INTEGRATION_TRIGGER_SOURCE

SetInfo	hDev	设备句柄	
	uType	IKP_INTEGRATION_TRIGGER_SOURCE (0x10000027)	
	nValue	设置积分控制方法触发信号源，有效取值范围如下	
		IKP_INTEGRATION_TRIGGER_SOURCE_VAL_INTERNAL	采集卡内部触发信号
		IKP_INTEGRATION_TRIGGER_SOURCE_VAL_GENERAL_INPUT1	外部通用输入信号 1
		IKP_INTEGRATION_TRIGGER_SOURCE_VAL_GENERAL_INPUT2	外部通用输入信号 2
		IKP_INTEGRATION_TRIGGER_SOURCE_VAL_SHAFT_ENCODER1	编码器信号
		IKP_INTEGRATION_TRIGGER_SOURCE_VAL_BOARD_SYNC1	多卡同步信号 1
		IKP_INTEGRATION_TRIGGER_SOURCE_VAL_BOARD_SYNC2	多卡同步信号 2
		IKP_INTEGRATION_TRIGGER_SOURCE_VAL_SOFTWARE	软件触发信号
GetInfo	hDev	设备句柄	
	uType	IKP_INTEGRATION_TRIGGER_SOURCE (0x10000027)	
	npValue	获取积分控制方法触发信号源	
说明	本参数表示积分控制方法触发信号源。		

IKP_INTEGRATION_TRIGGER_FREQUENCY 积分触发信号频率

属性	Get	Set	vlcf parameter
	支持	支持	INTERNAL_TRIGGER_FREQUENCY
SetInfo	hDev	设备句柄	
	uType	IKP_INTEGRATION_TRIGGER_FREQUENCY (0x10000028)	
	nValue	当 IKP_INTEGRATION_TRIGGER_SOURCE 设置为采集卡内部触发时，本参数指明内部触发信号的频率，取值范围为 1Hz ~ 500000Hz	
GetInfo	hDev	设备句柄	
	uType	IKP_INTEGRATION_TRIGGER_FREQUENCY (0x10000028)	
	npValue	获取内部触发信号的频率	
说明	当 IKP_INTEGRATION_TRIGGER_SOURCE 设置为采集卡内部触发时，本参数指明内部触发信号的频率。		

IKP_STROBE_TRIGGER_SOURCE 闪光触发信号源

属性	Get	Set	vlcf parameter
	支持	支持	STROBE_TRIGGER_SOURCE

SetInfo	hDev	设备句柄	
	uType	IKP_STROBE_TRIGGER_SOURCE (0x10000029)	
	nValue	设置闪光控制触发信号源，有效范围如下	
		IKP_STROBE_TRIGGER_SOURCE_VAL_INTERNAL	内部触发信号
		IKP_STROBE_TRIGGER_SOURCE_VAL_GENERAL_INPUT1	通用输入信号 1
		IKP_STROBE_TRIGGER_SOURCE_VAL_GENERAL_INPUT2	通用输入信号 2
		IKP_STROBE_TRIGGER_SOURCE_VAL_SHAFT_ENCODER1	编码器信号
		IKP_STROBE_TRIGGER_SOURCE_VAL_BOARD_SYNC1	板间同步信号 1
		IKP_STROBE_TRIGGER_SOURCE_VAL_BOARD_SYNC2	板间同步信号 2
		IKP_STROBE_TRIGGER_SOURCE_VAL_FRAME_TRANSFER_END	帧传输结束信号
		IKP_STROBE_TRIGGER_SOURCE_VAL_SOFTWARE	软件触发信号
		IKP_STROBE_TRIGGER_SOURCE_VAL_FRAME_TRANSFER_START	帧传输开始信号
GetInfo	hDev	设备句柄	
	uType	IKP_STROBE_TRIGGER_SOURCE (0x10000029)	
	npValue	获取闪光控制触发信号源	
说明	本参数指明闪光控制触发信号源。		

IKP_BOARD_SYNC_OUTPUT1_SOURCE CL 采集卡多卡同步输出 1 信号源

属性	Get	Set	vlcf parameter
	支持	支持	BOARD_SYNC_OUTPUT1_SOURCE
SetInfo	hDev	设备句柄	
	uType	IKP_BOARD_SYNC_OUTPUT1_SOURCE (0x1000002a)	
	nValue	设置 CL 采集卡多卡同步 1 信号源，有效范围如下	
		IKP_BOARD_SYNC_OUTPUT_SOURCE_VAL_NOT_USE	不使用
		IKP_BOARD_SYNC_OUTPUT_SOURCE_VAL_INTERNAL	内部触发信号 1
		IKP_BOARD_SYNC_OUTPUT_SOURCE_VAL_GENERAL_INPUT1	通用输入信号 1
		IKP_BOARD_SYNC_OUTPUT_SOURCE_VAL_GENERAL_INPUT2	通用输入信号 2
		IKP_BOARD_SYNC_OUTPUT_SOURCE_VAL_ENCODER	编码器信号

		AL_SHAFT_ENCODER1	
		IKP_BOARD_SYNC_OUTPUT_SOURCE_V AL_INTEGRATION_SIGNAL1	积分控制信号 1
		IKP_BOARD_SYNC_OUTPUT_SOURCE_V AL_INTEGRATION_SIGNAL2	积分控制信号 2
		IKP_BOARD_SYNC_OUTPUT_SOURCE_V AL_STROBLE_SIGNAL1	闪光控制信号 1
		IKP_BOARD_SYNC_OUTPUT_SOURCE_V AL_BOARD_SYNC1	多卡同步信号 1
		IKP_BOARD_SYNC_OUTPUT_SOURCE_V AL_BOARD_SYNC2	多卡同步信号 2
GetInfo	hDev	设备句柄	
	uType	IKP_BOARD_SYNC_OUTPUT1_SOURCE (0x1000002a)	
	npValue	获取 CL 采集卡多卡同步 1 信号源	
说明	本参数指明 CL 采集卡多卡同步 1 信号源。		

IKP_BOARD_SYNC_OUTPUT2_SOURCE CL 采集卡多卡同步输出 2 信号源

属性	Get	Set	vlc parameter
	支持	支持	BOARD_SYNC_OUTPUT2_SOURCE
SetInfo	hDev	设备句柄	
	uType	IKP_BOARD_SYNC_OUTPUT2_SOURCE (0x1000002b)	
	nValue	设置 CL 采集卡多卡同步 2 信号源，有效范围如下	
		IKP_BOARD_SYNC_OUTPUT_SOURCE_V AL_NOT_USE	不使用
		IKP_BOARD_SYNC_OUTPUT_SOURCE_V AL_INTERNAL	内部触发信号 1
		IKP_BOARD_SYNC_OUTPUT_SOURCE_V AL_GENERAL_INPUT1	通用输入信号 1
		IKP_BOARD_SYNC_OUTPUT_SOURCE_V AL_GENERAL_INPUT2	通用输入信号 2
		IKP_BOARD_SYNC_OUTPUT_SOURCE_V AL_SHAFT_ENCODER1	编码器信号
		IKP_BOARD_SYNC_OUTPUT_SOURCE_V AL_INTEGRATION_SIGNAL1	积分控制信号 1
		IKP_BOARD_SYNC_OUTPUT_SOURCE_V AL_INTEGRATION_SIGNAL2	积分控制信号 2
		IKP_BOARD_SYNC_OUTPUT_SOURCE_V AL_STROBLE_SIGNAL1	闪光控制信号 1
		IKP_BOARD_SYNC_OUTPUT_SOURCE_V AL_BOARD_SYNC1	板间同步信号 1
		IKP_BOARD_SYNC_OUTPUT_SOURCE_V	板间同步信号 2

		AL_BOARD_SYNC2	
GetInfo	hDev	设备句柄	
	uType	IKP_BOARD_SYNC_OUTPUT2_SOURCE (0x1000002b)	
	npValue	获取 CL 采集卡多卡同步 2 信号源	
说明	本参数指明 CL 采集卡多卡同步 2 信号源。		

IKP_GENERAL_OUTPUT1_SOURCE 通用输出信号 1 信号源

属性	Get	Set	vlcf parameter
	支持	支持	GENERAL_OUTPUT1_SOURCE
SetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_OUTPUT1_SOURCE (0x1000002c)	
	nValue	设置通用输出信号 1 的信号源，有效范围如下	
		IKP_GENERAL_OUTPUT_SOURCE_V AL_INTERNAL	内部触发信号 1
		IKP_GENERAL_OUTPUT_SOURCE_V AL_GENERAL_INPUT1	通用输入信号 1
		IKP_GENERAL_OUTPUT_SOURCE_V AL_GENERAL_INPUT2	通用输入信号 2
		IKP_GENERAL_OUTPUT_SOURCE_V AL_SHAFT_ENCODER1	编码器信号
		IKP_GENERAL_OUTPUT_SOURCE_V AL_INTEGRATION_SIGNAL1	积分控制信号 1
		IKP_GENERAL_OUTPUT_SOURCE_V AL_INTEGRATION_SIGNAL2	积分控制信号 2
		IKP_GENERAL_OUTPUT_SOURCE_V AL_STROBLE_SIGNAL1	闪光控制信号
		IKP_GENERAL_OUTPUT_SOURCE_V AL_SOFTWARE	软件触发信号
GetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_OUTPUT1_SOURCE (0x1000002c)	
	npValue	获取通用输出信号 1 的信号源	
说明	本参数指明通用输出信号 1 的信号源。		

IKP_GENERAL_OUTPUT2_SOURCE 通用输出信号 2 信号源

属性	Get	Set	vlcf parameter
	支持	支持	GENERAL_OUTPUT2_SOURCE
SetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_OUTPUT2_SOURCE (0x1000002d)	
	nValue	设置通用输出信号 2 的信号源，有效范围如下	

		IKP_GENERAL_OUTPUT_SOURCE_VAL_INTERNAL	内部触发信号 1
		IKP_GENERAL_OUTPUT_SOURCE_VAL_GENERAL_INPUT1	通用输入信号 1
		IKP_GENERAL_OUTPUT_SOURCE_VAL_GENERAL_INPUT2	通用输入信号 2
		IKP_GENERAL_OUTPUT_SOURCE_VAL_SHAFT_ENCODER1	编码器信号
		IKP_GENERAL_OUTPUT_SOURCE_VAL_INTEGRATION_SIGNAL1	积分控制信号 1
		IKP_GENERAL_OUTPUT_SOURCE_VAL_INTEGRATION_SIGNAL2	积分控制信号 2
		IKP_GENERAL_OUTPUT_SOURCE_VAL_STROBLE_SIGNAL1	闪光控制信号
		IKP_GENERAL_OUTPUT_SOURCE_VAL_SOFTWARE	软件触发信号
GetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_OUTPUT2_SOURCE (0x1000002d)	
	npValue	获取通用输出信号 2 的信号源	
说明	本参数指明通用输出信号 2 的信号源。		

IKP_INTEGRATION_METHOD 积分控制方法索引

属性	Get	Set	vpcf parameter	
	支持	支持	INTEGRATION_METHOD	
SetInfo	hDev	设备句柄		
	uType	IKP_INTEGRATION_METHOD (0x1000002e)		
	nValue	设置积分控制方法索引，有效范围如下		
		INTEGRATION_METHOD_VAL_1	积分控制方法 1	
		INTEGRATION_METHOD_VAL_2	积分控制方法 2	
		INTEGRATION_METHOD_VAL_3	积分控制方法 3	
		INTEGRATION_METHOD_VAL_4	积分控制方法 4	
GetInfo	hDev	设备句柄		
	uType	IKP_INTEGRATION_METHOD (0x1000002e)		
	npValue	获取积分控制方法索引		
说明	本参数指明积分控制方法索引。			

IKP_INTEGRATION_PARAM1 积分控制方法参数 1

属性	Get	Set	vpcf parameter
	支持	支持	INTEGRATION_PARAM1

SetInfo	hDev	设备句柄
	uType	IKP_INTEGRATION_PARAM1 (0x1000002f)
	nValue	设置积分控制方法参数 1，取值范围为 0 ~ 25264513μs
GetInfo	hDev	设备句柄
	uType	IKP_INTEGRATION_PARAM1 (0x1000002f)
	npValue	获取积分控制方法参数 1
说明	本参数指明积分控制方法参数 1。	

IKP_INTEGRATION_PARAM2 积分控制方法参数 2

属性	Get	Set	vlcf parameter
	支持	支持	INTEGRATION_PARAM2
SetInfo	hDev	设备句柄	
	uType	IKP_INTEGRATION_PARAM2 (0x10000030)	
	nValue	设置积分控制方法参数 2，取值范围为 0 ~ 25264513μs	
GetInfo	hDev	设备句柄	
	uType	IKP_INTEGRATION_PARAM2 (0x10000030)	
	npValue	获取积分控制方法参数 2	
说明	本参数指明积分控制方法参数 2。		

IKP_INTEGRATION_PARAM3 积分控制方法参数 3

属性	Get	Set	vlcf parameter
	支持	支持	INTEGRATION_PARAM3
SetInfo	hDev	设备句柄	
	uType	IKP_INTEGRATION_PARAM3 (0x10000031)	
	nValue	设置积分控制方法参数 3，取值范围为 0 ~ 25264513μs	
GetInfo	hDev	设备句柄	
	uType	IKP_INTEGRATION_PARAM3 (0x10000031)	
	npValue	获取积分控制方法参数 3	
说明	本参数指明积分控制方法参数 3。		

IKP_INTEGRATION_PARAM4 积分控制方法参数 4

属性	Get	Set	vlcf parameter
	支持	支持	INTEGRATION_PARAM4
SetInfo	hDev	设备句柄	
	uType	IKP_INTEGRATION_PARAM4 (0x10000032)	
	nValue	设置积分控制方法参数 4，取值范围为 0 ~ 25264513μs	
GetInfo	hDev	设备句柄	

	uType	IKP_INTEGRATION_PARAM4 (0x10000031)
	npValue	获取积分控制方法参数 4
说明	本参数指明积分控制方法参数 4。	

IKP_INTEGRATION_POLARITY1 积分控制方法信号 1 极性

属性	Get	Set	vlcf parameter	
	支持	支持	INTEGRATION_POLARITY1	
SetInfo	hDev	设备句柄		
	uType	IKP_INTEGRATION_POLARITY1 (0x10000033)		
	nValue	设置积分控制方法信号 1 极性，有效范围如下		
		IKP_INTEGRATION_POLARITY_VAL_HI GH		正极性
		IKP_INTEGRATION_POLARITY_VAL_L OW		负极性
GetInfo	hDev	设备句柄		
	uType	IKP_INTEGRATION_POLARITY1 (0x10000033)		
	npValue	获取积分控制方法信号 1 极性		
说明	本参数指明积分控制方法信号 1 极性。			

IKP_INTEGRATION_POLARITY2 积分控制方法信号 2 极性

属性	Get	Set	vlcf parameter	
	支持	支持	INTEGRATION_POLARITY2	
SetInfo	hDev	设备句柄		
	uType	IKP_INTEGRATION_POLARITY2 (0x10000034)		
	nValue	设置积分控制方法信号 2 极性，有效范围如下		
		IKP_INTEGRATION_POLARITY_VAL_HI GH		正极性
		IKP_INTEGRATION_POLARITY_VAL_L OW		负极性
GetInfo	hDev	设备句柄		
	uType	IKP_INTEGRATION_POLARITY2 (0x10000034)		
	npValue	获取积分控制方法信号 2 极性		
说明	本参数指明积分控制方法信号 2 极性。			

IKP_STROBE_METHOD 闪光控制方法索引

属性	Get	Set	vlcf parameter
	支持	支持	STROBE_METHOD
SetInfo	hDev	设备句柄	

	uType	IKP_STROBE_METHOD (0x10000035)	
	nValue	设置闪光控制方法索引，有效范围如下	
		IKP_STROBE_METHOD_VAL_1	闪光控制方法 1
		IKP_STROBE_METHOD_VAL_2	闪光控制方法 2
		IKP_STROBE_METHOD_VAL_3	闪光控制方法 3
		IKP_STROBE_METHOD_VAL_4	闪光控制方法 4
GetInfo	hDev	设备句柄	
	uType	IKP_STROBE_METHOD (0x10000035)	
	npValue	获取闪光控制方法索引	
说明	本参数指明闪光控制方法索引。		

IKP_STROBE_PARAM1 闪光控制方法参数 1

属性	Get	Set	vlcf parameter
	支持	支持	STROBE_PARAM1
SetInfo	hDev	设备句柄	
	uType	IKP_STROBE_PARAM1 (0x10000036)	
	nValue	设置闪光控制方法参数 1，取值范围为 0 ~ 429496729μs	
GetInfo	hDev	设备句柄	
	uType	IKP_STROBE_PARAM1 (0x10000036)	
	npValue	获取闪光控制方法参数 1	
说明	本参数指明闪光控制方法参数 1。		

IKP_STROBE_PARAM2 闪光控制方法参数 2

属性	Get	Set	vlcf parameter
	支持	支持	STROBE_PARAM2
SetInfo	hDev	设备句柄	
	uType	IKP_STROBE_PARAM2 (0x10000037)	
	nValue	设置闪光控制方法参数 2，取值范围为 0 ~ 429496729μs	
GetInfo	hDev	设备句柄	
	uType	IKP_STROBE_PARAM2 (0x10000037)	
	npValue	获取闪光控制方法参数 2	
说明	本参数指明闪光控制方法参数 2。		

IKP_STROBE_PARAM3 闪光控制方法参数 3

属性	Get	Set	vlcf parameter
	支持	支持	STROBE_PARAM3
SetInfo	hDev	设备句柄	

	uType	IKP_STROBE_PARAM3 (0x10000038)
	nValue	设置闪光控制方法参数 3，取值范围为 0 ~ 429496729μs
	hDev	设备句柄
GetInfo	uType	IKP_STROBE_PARAM3 (0x10000038)
	npValue	获取闪光控制方法参数 3
说明	本参数指明闪光控制方法参数 3。	

IKP_STROBE_PARAM4 闪光控制方法参数 4

属性	Get	Set	vlcf parameter
	支持	支持	STROBE_PARAM4
SetInfo	hDev	设备句柄	
	uType	IKP_STROBE_PARAM4 (0x10000039)	
	nValue	设置闪光控制方法参数 4，取值范围为 0 ~ 429496729μs	
GetInfo	hDev	设备句柄	
	uType	IKP_STROBE_PARAM4 (0x10000039)	
	npValue	获取闪光控制方法参数 4	
说明	本参数指明闪光控制方法参数 4。		

IKP_STROBE_POLARITY 闪光控制方法信号极性

属性	Get	Set	vlcf parameter	
	支持	支持	STROBE_POLARITY1	
SetInfo	hDev	设备句柄		
	uType	IKP_STROBE_POLARITY (0x1000003a)		
	nValue	设置闪光控制方法信号极性，有效范围如下		
		IKP_STROBE_POLARITY_VAL_HIGH		正极性
		IKP_STROBE_POLARITY_VAL_LOW		负极性
GetInfo	hDev	设备句柄		
	uType	IKP_STROBE_POLARITY1 (0x1000003a)		
	npValue	获取闪光控制方法信号极性		
说明	本参数指明闪光控制方法信号极性。			

IKP_GENERAL_OUTPUT1_POLARITY 通用输出信号 1 极性

属性	Get	Set	vlcf parameter
	支持	支持	GENERALOUTPUT1_POLARITY
SetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_OUTPUT1_POLARITY (0x1000003b)	
	nValue	设置通用输出信号 1 信号极性，有效范围如下	

		IKP_GENERAL_OUTPUT_POLARITY_VAL_SAME_TO_SOURCE	与信号源极性相同
		IKP_GENERAL_OUTPUT_POLARITY_VAL_OPPOSITE_TO_SOURCE	与信号源极性相反
GetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_OUTPUT1_POLARITY (0x1000003b)	
	npValue	获取通用输出信号 1 极性	
说明	本参数指明通用输出信号 1 极性。		

IKP_GENERAL_OUTPUT1_DELAY CL 采集卡通用输出信号 1 输出延时【已弃用】

属性	Get	Set	vlcf parameter
	支持	支持	GENERALOUTPUT1_DELAY
SetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_OUTPUT1_DELAY (0x1000003c)	
	nValue	设置 CL 采集卡通用输出信号 1 的输出延时，取值范围为 0 ~ 429496729μs	
GetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_OUTPUT1_DELAY (0x1000003c)	
	npValue	获取 CL 采集卡通用输出信号 1 的输出延时	
说明	本参数指明 CL 采集卡通用输出信号 1 的输出延时。		

IKP_GENERAL_OUTPUT2_POLARITY 通用输出信号 2 极性

属性	Get	Set	vlc parameter	
	支持	支持	GENERALOUTPUT2_POLARITY	
SetInfo	hDev	设备句柄		
	uType	IKP_GENERAL_OUTPUT2_POLARITY (0x1000003d)		
	nValue	设置通用输出信号 2 信号极性，有效范围如下		
		IKP_GENERAL_OUTPUT_POLARITY_VAL_SAME_TO_SOURCE	与信号源极性相同	
		IKP_GENERAL_OUTPUT_POLARITY_VAL_OPPOSITE_TO_SOURCE	与信号源极性相反	
GetInfo	hDev	设备句柄		
	uType	IKP_GENERAL_OUTPUT2_POLARITY (0x1000003d)		
	npValue	获取通用输出信号 2 极性		
说明	本参数指明通用输出信号 2 极性。			

IKP_GENERAL_OUTPUT2_DELAY CL 采集卡通用输出信号 2 输出延时【已弃用】

属性	Get	Set	vlcf parameter
----	-----	-----	----------------

	支持	支持	GENERALOUTPUT2_DELAY
SetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_OUTPUT2_DELAY (0x1000003e)	
	nValue	设置 CL 采集卡通用输出信号 2 的输出延时，取值范围为 0 ~ 429496729μs	
GetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_OUTPUT2_DELAY (0x1000003e)	
	npValue	获取 CL 采集卡通用输出信号 2 的输出延时	
说明	本参数指明 CL 采集卡通用输出信号 2 的输出延时。		

IKP_GENERAL_INPUT1_TRIGGER_MODE 通用输入信号 1 触发模式

属性	Get	Set	vlfw parameter	
	支持	支持	GENERALINPUT1_TRIGGER_MODE	
SetInfo	hDev	设备句柄		
	uType	IKP_GENERAL_INPUT1_TRIGGER_MODE (0x1000003f)		
	nValue	设置通用输入信号 1 的触发模式，有效值如下		
		IKP_GENERAL_INPUT_TRIGGER_MODE_V AL_EDGE		边沿触发
		IKP_GENERAL_INPUT_TRIGGER_MODE_V AL_LEVEL		电平触发
GetInfo	hDev	设备句柄		
	uType	IKP_GENERAL_INPUT1_TRIGGER_MODE (0x1000003f)		
	npValue	获取通用输入信号 1 的触发模式		
说明	本参数表示通用输入信号 1 的触发模式。			

IKP_GENERAL_INPUT2_TRIGGER_MODE 通用输入信号 2 触发模式

属性	Get	Set	vlfw parameter	
	支持	支持	GENERALINPUT2_TRIGGER_MODE	
SetInfo	hDev	设备句柄		
	uType	IKP_GENERAL_INPUT2_TRIGGER_MODE (0x10000040)		
	nValue	设置通用输入信号 2 的触发模式，有效值如下		
		IKP_GENERAL_INPUT_TRIGGER_MODE_V AL_EDGE		边沿触发
		IKP_GENERAL_INPUT_TRIGGER_MODE_V AL_LEVEL		电平触发
GetInfo	hDev	设备句柄		
	uType	IKP_GENERAL_INPUT2_TRIGGER_MODE (0x10000040)		
	npValue	获取通用输入信号 2 的触发模式		
说明	本参数设置/获取通用输入信号 2 的触发模式。			

IKP_BOARD_SYNC1_TRIGGER_MODE CL 采集卡多卡同步信号 1 触发模式

属性	Get	Set	vlfw parameter	
	支持	支持	BOARD_SYNC1_TRIGGER_MODE	
SetInfo	hDev	设备句柄		
	uType	IKP_BOARD_SYNC1_TRIGGER_MODE (0x10000041)		
	nValue	设置 CL 采集卡多卡同步信号 1 触发模式，有效值如下		
		IKP_BOARD_SYNC_TRIGGER_MODE_V AL_EDGE		边沿触发
		IKP_BOARD_SYNC_TRIGGER_MODE_V AL_LEVEL		电平触发
GetInfo	hDev	设备句柄		
	uType	IKP_BOARD_SYNC1_TRIGGER_MODE (0x10000041)		
	npValue	获取 CL 采集卡多卡同步信号 1 触发模式		
说明	本参数表示 CL 采集卡多卡同步信号 1 触发模式。			

IKP_BOARD_SYNC2_TRIGGER_MODE CL 采集卡多卡同步信号 2 触发模式

属性	Get	Set	vlfw parameter	
	支持	支持	BOARD_SYNC2_TRIGGER_MODE	
SetInfo	hDev	设备句柄		
	uType	IKP_BOARD_SYNC2_TRIGGER_MODE (0x10000042)		
	nValue	设置 CL 采集卡多卡同步信号 2 触发模式，有效值如下		
		IKP_BOARD_SYNC_TRIGGER_MODE_V AL_EDGE		边沿触发
		IKP_BOARD_SYNC_TRIGGER_MODE_V AL_LEVEL		电平触发
GetInfo	hDev	设备句柄		
	uType	IKP_BOARD_SYNC2_TRIGGER_MODE (0x10000042)		
	npValue	获取 CL 采集卡多卡同步信号 2 触发模式		
说明	本参数表示 CL 采集卡多卡同步信号 2 触发模式。			

IKP_SHAFT_ENCODER1_CHANNEL 编码器 A/B 通道选择

属性	Get	Set	vlfw parameter	
	支持	支持	SHAFT_ENCODER_CHANNEL	
SetInfo	hDev	设备句柄		
	uType	IKP_SHAFT_ENCODER1_CHANNEL (0x10000043)		
	nValue	选择编码器 A/B 通道，有效值如下		
		IKP_SHAFT_ENCODER_CHANNEL_VAL_A		A 通道

		IKP_SHAFT_ENCODER_CHANNEL_VAL_B	B 通道
GetInfo	hDev	设备句柄	
	uType	IKP_SHAFT_ENCODER1_CHANNEL (0x10000043)	
	npValue	获取编码器当前使用通道	
说明	本参数表示编码器使用通道。		

IKP_SHAFT_ENCODER1_MULTIPLY_FACTOR 编码器倍频系数

属性	Get	Set	vlfw parameter	
	支持	支持	SHAFT_ENCODER_MULTIPLY_FACTOR	
SetInfo	hDev	设备句柄		
	uType	IKP_SHAFT_ENCODER1_MULTIPLY_FACTOR (0x10000044)		
	nValue	选择编码器倍频系数，有效值如下		
		IKP_SHAFT_ENCODER_MULTIPLY_FACTOR_VAL_1	1 倍频	
		IKP_SHAFT_ENCODER_MULTIPLY_FACTOR_VAL_2	2 倍频	
		IKP_SHAFT_ENCODER_MULTIPLY_FACTOR_VAL_4	4 倍频	
		IKP_SHAFT_ENCODER_MULTIPLY_FACTOR_VAL_8	8 倍频	
		IKP_SHAFT_ENCODER_MULTIPLY_FACTOR_VAL_16	16 倍频	
		IKP_SHAFT_ENCODER_MULTIPLY_FACTOR_VAL_32	32 倍频	
hDev		设备句柄		
uType	IKP_SHAFT_ENCODER1_MULTIPLY_FACTOR (0x10000044)			
npValue	获取编码器倍频系数			
说明	本参数表示编码器倍频系数。			

IKP_PCB_VERSION CL 采集卡 PCB 版本号

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_PCB_VERSION (0x10000045)	
	npValue	获取 CL 采集卡 PCB 版本号	
说明	无		

IKP_LVAL_FILTER CL 采集卡 LVAL 滤波控制

属性	Get	Set	vlfw parameter	
	支持	支持	LINE_VALID_FILTER	
SetInfo	hDev	设备句柄		
	uType	IKP_LVAL_FILTER (0x10000046)		
	nValue	使能 CL 采集卡 LVAL 滤波功能，有效值如下		
		IKP_LVAL_FILTER_DISABLE	禁用	
		IKP_LVAL_FILTER_ENABLE	启用	
GetInfo	hDev	设备句柄		
	uType	IKP_LVAL_FILTER (0x10000046)		
	npValue	获取 CL 采集卡 LVAL 滤波功能是否启用		
说明	使能滤波情况下，CL 采集卡会对 LVAL 脉冲宽度进行检测，若 LVAL 脉宽小于一定阈值则认为是毛刺，此时不启动数据结束。滤波操作适用于外部存在强电磁干扰情况，正常采集环境不需要使用该功能。			

IKP_FRAME_TRANSFER_PERIOD CL 采集卡面阵图像 DMA 传输周期数

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_FRAME_TRANSFER_PERIOD (0x10000047)	
	npValue	获取 CL 采集卡面阵图像 DMA 传输周期数	
说明	无		

IKP_LINE_TRANSFER_PERIOD CL 采集卡线阵图像 DMA 传输周期数

属性	Get	Set	vlfw parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_LINE_TRANSFER_PERIOD (0x10000048)	
	npValue	获取 CL 采集卡线阵图像 DMA 传输周期数	
说明	无		

IKP_FPGA_EXTERNAL_TRIGGER_TIMEOUT CL 采集卡外触发超时系数

属性	Get	Set	vlfw parameter	
	支持	支持	EXTERNAL_LINE_TIMEOUT_FACTOR	

SetInfo	hDev	设备句柄
	uType	IKP_FPGA_EXTERNAL_TRIGGER_TIMEOUT (0x10000049)
	nValue	选择 CL 采集卡外部触发超时系数，取值范围为 8 ~ 1024
GetInfo	hDev	设备句柄
	uType	IKP_FPGA_EXTERNAL_TRIGGER_TIMEOUT (0x10000049)
	npValue	获取 CL 采集卡外部触发超时系数
说明	CL 采集卡工作在外部触发模式下，当外部触发信号不良导致图像采集失败时，可以通过增加该参数值保证图像采集的正常运行。	

IKP_IMAGE_OFFSET_X 相机 Tap 水平偏移

属性	Get	Set	vlcf parameter
	支持	支持	IMAGE_OFFSET_X
SetInfo	hDev	设备句柄	
	uType	IKP_IMAGE_OFFSET_X (0x10000050)	
	nValue	设置相机 TAP 水平偏移	
GetInfo	hDev	设备句柄	
	uType	IKP_IMAGE_OFFSET_X (0x10000050)	
	npValue	获取相机 TAP 水平偏移	
说明	本参数表示相机 Tap 水平偏移，对于相机输出的实际数据，采集卡会根据当前 Tap 排列方式截取每个 Tap 区域最开始的若干像素。 如果用户希望获取图像的感兴趣区域，请使用 IKP_IMAGE_ROI_X 参数。		

IKP_GENERAL_INPUT1_POLARITY 通用输入信号 1 极性

属性	Get	Set	vlcf parameter
	支持	支持	GENERAL_INPUT1_POLARITY
SetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_INPUT1_POLARITY (0x10000051)	
	nValue	设置通用输入信号 1 的极性，具体含义如下	
		IKP_GENERAL_INPUT1_TRIGGER_MODE 选择边沿触发：	
		IKP_GENERAL_INPUT_POLARITY_V AL_HIGH	上升沿触发
		IKP_GENERAL_INPUT_POLARITY_V AL_LOW	下降沿触发
		IKP_GENERAL_INPUT1_TRIGGER_MODE 选择电平触发：	
		IKP_GENERAL_INPUT_POLARITY_V AL_HIGH	高电平触发
		IKP_GENERAL_INPUT_POLARITY_V	低电平触发

		AL_LOW	
GetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_INPUT1_POLARITY (0x10000051)	
	npValue	获取通用输入信号 1 的极性，具体含义见上文	
说明	本参数表示通用输入信号 1 的极性。		

IKP_GENERAL_INPUT1_MIN_WIDTH 通用输入信号 1 最小脉冲宽度

属性	Get	Set	vlcf parameter
	支持	支持	GENERAL_INPUT1_MIN_WIDTH
SetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_INPUT1_MIN_WIDTH (0x10000052)	
	nValue	设置通用输入信号 1 的最小脉冲宽度，最小值 50ns	
GetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_INPUT1_MIN_WIDTH (0x10000052)	
	npValue	获取通用输入信号 1 的最小脉冲宽度，最小值 50ns	
说明	本参数表示通用输入信号 1 的最小脉冲宽度。当实际输入信号的脉冲宽度大于该值认为是有效触发信号，否则丢弃。		

IKP_GENERAL_INPUT2_POLARITY 通用输入信号 2 极性

属性	Get	Set	vlcf parameter
	支持	支持	GENERAL_INPUT2_POLARITY
SetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_INPUT2_POLARITY (0x10000053)	
	nValue	设置通用输入信号 2 的极性，具体含义如下	
		IKP_GENERAL_INPUT2_TRIGGER_MODE 选择边沿触发：	
		IKP_GENERAL_INPUT_POLARITY_VA L_HIGH	上升沿触发
		IKP_GENERAL_INPUT_POLARITY_VA L_LOW	下降沿触发
		IKP_GENERAL_INPUT2_TRIGGER_MODE 选择电平触发：	
		IKP_GENERAL_INPUT_POLARITY_VA L_HIGH	高电平触发
		IKP_GENERAL_INPUT_POLARITY_VA L_LOW	低电平触发
GetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_INPUT2_POLARITY (0x10000053)	
	npValue	获取通用输入信号 2 的极性，具体含义见上文。	
说明	本参数表示通用输入信号 2 的极性。		

IKP_GENERAL_INPUT2_MIN_WIDTH 通用输入信号 2 最小脉冲宽度

属性	Get	Set	vlcf parameter
	支持	支持	GENERAL_INPUT2_MIN_WIDTH
SetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_INPUT2_MIN_WIDTH (0x10000054)	
	nValue	设置通用输入信号 2 的最小脉冲宽度，最小值 50ns	
GetInfo	hDev	设备句柄	
	uType	IKP_GENERAL_INPUT2_MIN_WIDTH (0x10000054)	
	npValue	获取通用输入信号 2 的最小脉冲宽度，最小值 50ns	
说明	本参数表示通用输入信号 2 的最小脉冲宽度。当实际输入信号的脉冲宽度大于该值认为是有效触发信号，否则丢弃。		

IKP_CAMERA_PIXEL_CLOCK CL 相机时钟频率

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_CAMERA_PIXEL_CLOCK (0x10000055)	
	npValue	获取 CL 相机时钟频率，单位 MHz	
说明	获取 CL 相机的时钟频率，单位 MHz。		

IKP_PCIE_KERNEL_BLOCK_SIZE CL 采集卡 PCIe 内部缓冲区块大小

属性	Get	Set	vlc parameter	
	支持	支持	无	
SetInfo	hDev	设备句柄		
	uType	IKP_PCIE_KERNEL_BLOCK_SIZE (0x10000056)		
	nValue	CL 采集卡 PCIe 内部缓冲区大小，有效值如下		
		IKP_PCIE_KERNEL_BLOCK_SIZE_VAL_4M	4MB	
		IKP_PCIE_KERNEL_BLOCK_SIZE_VAL_8M	8MB	
		IKP_PCIE_KERNEL_BLOCK_SIZE_VAL_12M	12MB	
		IKP_PCIE_KERNEL_BLOCK_SIZE_VAL_16M	16MB	
GetInfo	hDev	设备句柄		
	uType	IKP_PCIE_KERNEL_BLOCK_SIZE (0x10000056)		
	npValue	CL 采集卡 PCIe 内部缓冲区大小，具体含义见上文		
说明	本参数用来设置 CL 采集卡 PCIe 内核缓冲区大小，默认值为 4M，提高该参数有助于减少丢帧数据的发送。			

IKP_SOFTWARE_TRIGGER_WIDTH 软件触发脉冲宽度

属性	Get	Set	vlcf parameter
	支持	支持	SOFTWARE_TRIGGER_WIDTH
SetInfo	hDev	设备句柄	
	uType	IKP_SOFTWARE_TRIGGER_WIDTH (0x10000057)	
	nValue	设置软件触发的脉冲宽度，单位μs	
GetInfo	hDev	设备句柄	
	uType	IKP_SOFTWARE_TRIGGER_WIDTH (0x10000057)	
	npValue	获取软件触发的脉冲宽度，单位μs	
说明	当用户选择软件触发作为 CC1~CC4 的触发源时，设置该值可以改变软件触发信号的脉冲宽度，软件触发的脉冲宽度必须小于软件触发周期且大于 0。		

IKP_SOFTWARE_TRIGGER_PERIOD CL 采集卡软件触发脉冲周期

属性	Get	Set	vlcf parameter
	支持	支持	SOFTWARE_TRIGGER_PERIOD
SetInfo	hDev	设备句柄	
	uType	IKP_SOFTWARE_TRIGGER_PERIOD (0x10000058)	
	nValue	设置 CL 采集卡软件触发的脉冲周期，单位μs	
GetInfo	hDev	设备句柄	
	uType	IKP_SOFTWARE_TRIGGER_PERIOD (0x10000058)	
	npValue	获取 CL 采集卡软件触发的脉冲周期，单位μs	
说明	当用户选择软件触发作为 CC1~CC4 的触发源时，设置该值可以改变软件触发信号的脉冲周期，软件触发的脉冲宽度必须小于软件触发周期且大于 0。		

IKP_SOFTWARE_TRIGGER_COUNT CL 采集卡软件触发次数

属性	Get	Set	vlcf parameter
	支持	支持	SOFTWARE_TRIGGER_COUNT
SetInfo	hDev	设备句柄	
	uType	IKP_SOFTWARE_TRIGGER_COUNT (0x10000059)	
	nValue	设置 CL 采集卡软件触发脉冲的次数，该值必须大于 0	
GetInfo	hDev	设备句柄	
	uType	IKP_SOFTWARE_TRIGGER_COUNT (0x10000059)	
	npValue	获取 CL 采集卡软件触发脉冲的次数，该值必须大于 0	
说明	当用户选择软件触发作为 CC1~CC4 的触发源时，设置该值可以改变软件触发信号的脉冲次数，比如设置该值为 1，则采集卡会产生一个脉冲信号从而导致相机输出一张完整的图像数据。		

IKP_SOFTWARE_TRIGGER_START CL 采集卡开始软件触发

属性	Get	Set	vlcf parameter
	支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_SOFTWARE_TRIGGER_START (0x10000060)	
	nValue	该值总是 1	
GetInfo	hDev	设备句柄	
	uType	IKP_SOFTWARE_TRIGGER_START (0x10000060)	
	npValue	该值为 1 时，CL 采集卡开始软件触发	
说明	当用户选择软件触发作为 CC1~CC4 的触发源时，每次设置该值后采集卡都会根据当前的软件触发脉冲宽度、周期和次数产生触发信号。		

IKP_SOFTWARE_TRIGGER_DELAY 软件触发初始延迟时间

属性	Get	Set	vlcf parameter
	支持	支持	SOFTWARE_TRIGGER_DELAY
SetInfo	hDev	设备句柄	
	uType	IKP_SOFTWARE_TRIGGER_DELAY (0x10000061)	
	nValue	设置软件触发脉冲的初始延迟时间	
GetInfo	hDev	设备句柄	
	uType	IKP_SOFTWARE_TRIGGER_DELAY (0x10000061)	
	npValue	获取软件触发脉冲的初始延迟时间	
说明	当用户选择软件触发作为 CC1~CC4 的触发源时，设置该值可以改变软件触发信号的初始延迟时间，即用户开始软件触发和触发信号实际产生的时间间隔。		

IKP_SOFTWARE_TRIGGER_POLARITY 软件触发极性

属性	Get	Set	vlcf parameter	
	支持	支持	SOFTWARE_TRIGGER_POLARITY	
SetInfo	hDev	设备句柄		
	uType	IKP_SOFTWARE_TRIGGER_POLARITY (0x10000062)		
	nValue	设置软件触发脉冲极性		
		IKP_SOFTWARE_TRIGGER_POLARITY_V AL_HIGH	正脉冲，平时为低电平， 有效时为高电平	
IKP_SOFTWARE_TRIGGER_POLARITY_V AL_LOW		负脉冲，平时为高电平， 有效时为低电平		
GetInfo	hDev	设备句柄		
	uType	IKP_SOFTWARE_TRIGGER_POLARITY (0x10000062)		
	npValue	获取软件触发脉冲极性		
说明	当用户选择软件触发作为 CC1~CC4 的触发源时，设置该值可以改变软件触发信号的			

脉冲极性，该值为 0 表示正脉冲，即平时为低电平有效时为高电平；该值为 1 表示负脉冲，即平时为高电平，有效时为低电平。

IKP_GRAB_STATUS 采集卡采集状态

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_GRAB_STATUS (0x10000063)	
	npValue	获取采集卡采集状态	
说明	该状态为 0 指示采集卡当前未采集图像，该状态为 1 指示采集卡当前正在采集图像。		

IKP_CHECK_FRAME_VALID_SIGNAL CL 采集卡是否校验 FVAL 信号

属性	Get	Set	vlcf parameter
	支持	支持	CHECK_FVAL_SIGNAL
SetInfo	hDev	设备句柄	
	uType	IKP_CHECK_FRAME_VALID_SIGNAL (0x10000064)	
	nValue	0: CL 采集卡在采集面阵图像过程中不校验 Frame valid 信号 1: CL 采集卡在采集面阵图像过程中校验 Frame valid 信号	
GetInfo	hDev	设备句柄	
	uType	IKP_CHECK_FRAME_VALID_SIGNAL (0x10000064)	
	npValue	具体信息见上文	
说明	本参数仅对应于面阵相机有效，当用户设置本参数为 0 时，CL 采集卡将失去帧自动校准功能。CL 采集卡默认校验相机 Frame valid 信号。		

IKP_PIXEL_CLOCK_POLLING_TIME CL 相机像素时钟轮询时间

属性	Get	Set	vlcf parameter
	支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_PIXEL_CLOCK_POLLING_TIME (0x10000065)	
	nValue	设置 CL 相机像素时钟轮询时间，默认值 1000ms	
GetInfo	hDev	设备句柄	
	uType	IKP_PIXEL_CLOCK_POLLING_TIME (0x10000065)	
	npValue	获取 CL 相机像素时钟轮询时间，默认值 1000ms	
说明	本参数表示 CL 相机像素时钟轮询时间，用户可以通过注册 IKEvent_PixelClock 和 IKEvent No PixelClock 回调函数来判断相机是否接通电源。		

IKP_SOFTWARE_TRIGGER_SYNC_MODE CL 采集卡软件触发同步模式

属性	Get	Set	vlcf parameter
	支持	支持	SOFTWARE_TRIGGER_SYNC_MODE
SetInfo	hDev	设备句柄	
	uType	IKP_SOFTWARE_TRIGGER_SYNC_MODE (0x10000066)	
	nValue	设置 CL 采集卡软件触发的同步模式	
		IKP_SOFTWARE_TRIGGER_SYNC_MODE_VAL_DISABLE	禁用软件触发同步机制, 此时软件触发信号的周期由 IKP_SOFTWARE_TRIGGER_PERIOD 参数控制
		IKP_SOFTWARE_TRIGGER_SYNC_MODE_VAL_ENABLE	使能软件触发同步机制, 此时软件触发信号的周期由相机帧率决定, 每当相机输出完整的一帧图像后, 应用程序会立即产生一个软件触发信号, 使相机继续产生下一帧图像
GetInfo	hDev	设备句柄	
	uType	IKP_SOFTWARE_TRIGGER_SYNC_MODE (0x10000066)	
	npValue	获取 CL 采集卡软件触发的同步模式, 具体见上文。	
说明	本参数用来禁用/使能软件触发同步机制。		

IKP_HARDWARE_TRIGGER_GENERAL_INPUT1_DELAY 外部触发通用输入信号 1 延迟

属性	Get	Set	vlcf parameter
	支持	支持	HARDWARE_TRIGGER_GENERAL_INPUT1_DELAY
SetInfo	hDev	设备句柄	
	uType	IKP_HARDWARE_TRIGGER_GENERAL_INPUT1_DELAY (0x10000067)	
	nValue	设置通用输入信号 1 延迟，单位μs	
GetInfo	hDev	设备句柄	
	uType	IKP_HARDWARE_TRIGGER_GENERAL_INPUT_1_DELAY (0x10000067)	
	npValue	获取通用输入信号 1 延迟，单位μs	
说明	本参数表示通用输入信号 1 延迟。当用户选择通用输入信号 1 作为采集卡触发信号时，调整该值可以改变信号产生和实际采集图像时刻之间的间隔。		

IKP_HARDWARE_TRIGGER_GENERAL_INPUT2_DELAY 外部触发通用输入信号 2 延迟

属性	Get	Set	vlcf parameter
----	-----	-----	----------------

	支持	支持	HARDWARE_TRIGGER_GENERAL_INPUT2_DELAY
SetInfo	hDev	设备句柄	
	uType	IKP_HARDWARE_TRIGGER_GENERAL_INPUT2_DELAY (0x10000068)	
	nValue	设置通用输入信号 2 延迟，单位μs	
GetInfo	hDev	设备句柄	
	uType	IKP_HARDWARE_TRIGGER_GENERAL_INPUT_2_DELAY (0x10000068)	
	npValue	获取通用输入信号 2 延迟，单位μs	
说明	本参数表示通用输入信号 2 延迟。当用户选择通用输入信号 2 作为采集卡触发信号时，调整该值可以改变信号产生和实际采集图像时刻之间的间隔。		

IKP_IMAGE_ROI_X CL 采集卡图像感兴趣区域水平偏移

属性	Get	Set	vlcf parameter
	支持	支持	IMAGE_ROI_OFFSET_X
SetInfo	hDev	设备句柄	
	uType	IKP_IMAGE_ROI_X (0x10000069)	
	nValue	设置 CL 采集卡图像 ROI 水平偏移	
GetInfo	hDev	设备句柄	
	uType	IKP_IMAGE_ROI_X (0x10000069)	
	npValue	获取 CL 采集卡图像 ROI 水平偏移	
说明	本参数表示图像 ROI 区域的水平偏移，本参数必须是 16 的整数倍。		

IKP_SHAFT_ENCODER1_MIN_WIDTH CL 采集卡编码器输入信号最小脉冲宽度

属性	Get	Set	vlc parameter
	支持	支持	SHAFT_ENCODER1_MIN_WIDTH
SetInfo	hDev	设备句柄	
	uType	IKP_SHAFT_ENCODER1_MIN_WIDTH (0x10000070)	
	nValue	设置 CL 采集卡编码器输入信号最小脉冲宽度，最小值 50ns	
GetInfo	hDev	设备句柄	
	uType	IKP_SHAFT_ENCODER1_MIN_WIDTH (0x10000070)	
	npValue	获取 CL 采集卡编码器输入信号最小脉冲宽度，最小值 50ns	
说明	本参数表示 CL 采集卡编码器输入信号的最小脉冲宽度。当实际输入信号的脉冲宽度大于该值认为是有效触发信号，否则丢弃。		

IKP_SHAFT_ENCODER1_VALID_DIRECTION CL 采集卡编码器输入信号有效方向控制

属性	Get	Set	vlcf parameter
	支持	支持	SHAFT_ENCODER1_VALID_DIRECTION

SetInfo	hDev	设备句柄	
	uType	IKP_SHAFT_ENCODER1_VALID_DIRECTION (0x10000071)	
	nValue	设置 CL 采集卡编码器输入信号有效方向控制	
		IKP_SHAFT_ENCODER1_VALID_DIR_BOTH	正向转动或反向转动均视为有效输入信号
		IKP_SHAFT_ENCODER1_VALID_DIR_FORWARD	正向转动视为有效输入信号
	IKP_SHAFT_ENCODER1_VALID_DIR_BACKWARD	反向转动视为有效输入信号	
GetInfo	hDev	设备句柄	
	uType	IKP_SHAFT_ENCODER1_VALID_DIRECTION (0x10000071)	
	npValue	获取 CL 采集卡编码器输入信号有效方向控制	
说明	本参数表示 CL 采集卡编码器输入信号有效方向控制。		
	当本参数为 IKP_SHAFT_ENCODER1_VALID_DIR_BOTH 时，编码器正向和反向输入信号均被视为有效信号，采集卡会根据该信号产生触发信号。		
	当本参数为 IKP_SHAFT_ENCODER1_VALID_DIR_FORWARD 时，仅编码器正向信号被视为有效信号，反向信号会被采集卡丢弃。		
	当本参数为 IKP_SHAFT_ENCODER1_VALID_DIR_BACKWARD 时，仅编码器反向信号被视为有效信号，正向信号会被采集卡丢弃。		

IKP_SHAFT_ENCODER1_REVERSE_COMPENSATION CL 采集卡编码器输入信号反向输入补偿功能

属性	Get	Set	vllcf parameter	
	支持	支持	SHAFT_ENCODER1_REVERSE_COMPENSATION	
SetInfo	hDev	设备句柄		
	uType	IKP_SHAFT_ENCODER1_REVERSE_COMPENSATION (0x10000072)		
	nValue	IKP_SHAFT_ENCODER1_REVERSE_C OMPENSATION_OFF	禁用编码器反向输入信号补偿功能	
		IKP_SHAFT_ENCODER1_REVERSE_C OMPENSATION_ON	启用编码器反向输入信号补偿功能	
GetInfo	hDev	设备句柄		
	uType	IKP_SHAFT_ENCODER1_REVERSE_COMPENSATION (0x10000072)		
	npValue	具体见上文。		
说明	本参数表明 CL 采集卡编码器输入信号反向补偿功能。 当 IKP_SHAFT_ENCODER1_VALID_DIRECTION 参数选择正转有效或者反转有效时，可以使能该参数来进行反向补偿。反向补偿的效果如下： 选择编码器正转输入有效且使能反向补偿功能时，若编码器正转 100mm，然后反转 10mm，接着正转 50mm，则采集卡最终输出(100-10+50)mm 的图像，对于反转时候回退的距离，采集卡会内部记录保证不重复出图。			

IKP_FRAME_SIZE_64_LOW 每帧图像低 32 位大小

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_FRAME_SIZE_64_LOW (0x10000073)	
	npValue	获取每帧图像低 32 位大小	
说明	本参数表示每帧图像低 32 位大小。		

IKP_FRAME_SIZE_64_HIGH 每帧图像高 32 位大小

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_FRAME_SIZE_64_HIGH (0x10000074)	
	npValue	获取每帧图像高 32 位大小	
说明	本参数表示每帧图像高 32 位大小。		

IKP_BOARD_TRIGGER_OUTTER_MODE_FRAME_COUNT CL 采集卡外触发模式触发帧数

属性	Get	Set	vlcf parameter
	支持	支持	GRABBER_OUTTER_MODE_FRAME_COUNT
SetInfo	hDev	设备句柄	
	uType	IKP_BOARD_TRIGGER_OUTTER_MODE_FRAME_COUNT (0x10000075)	
	nValue	设置 CL 采集卡外触发模式触发帧数	
GetInfo	hDev	设备句柄	
	uType	IKP_BOARD_TRIGGER_OUTTER_MODE_FRAME_COUNT (0x10000075)	
	npValue	获取 CL 采集卡外触发模式触发帧数	
说明	本参数表示采集卡外触发模式触发帧数，即每接收到一个外触发信号，采集卡所采集的图像的帧数。		

IKP_SHAFT_ENCODER1_QUAD_FREQUENCY_SOURCE_TYPE CL 采集卡编码器 4 倍频以上信号来源

属性	Get	Set	vLcf parameter	
	支持	支持	SHAFT_ENCODER_QUAD_FREQUENCY_SOURCE_TYPE PE	
SetInfo	hDev	设备句柄		
	uType	IKP_SHAFT_ENCODER1_QUAD_FREQUENCY_SOURCE_TYPE (0x10000076)		
	nValue	设置 CL 采集卡编码器 4 倍频以上信号来源		
		IKP_SHAFT_ENCODER1_QUAD_FREQUENCY_SOURCE_TYPE_VAL_DOUBLE_CHANNEL ANNEL		A 或 B 单通道
		IKP_SHAFT_ENCODER1_QUAD_FREQUENCY_SOURCE_TYPE_VAL_SINGLE_CHANNEL NNEL		A 和 B 双通道
GetInfo	hDev	设备句柄		
	uType	IKP_SHAFT_ENCODER1_QUAD_FREQUENCY_SOURCE_TYPE (0x10000076)		
	npValue	获取 CL 采集卡编码器 4 倍频以上信号来源		
说明	本参数表示编码器 4 倍频以上信号来源。当编码器倍频为 1 或 2 时，只能通过 A 或 B 单通道传输信号；当编码器倍频为 4 及以上时，既可以通过 A 或 B 单通道传输信号，又可以通过 A、B 双通道一起传输信号。			

IKP_CURRENT_BUFFER_INDEX 当前缓冲区帧索引

属性	Get	Set	vlc parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_CURRENT_BUFFER_INDEX (0x10000077)	
	npValue	获取当前缓冲区帧索引	
说明	本参数表示当前缓冲区帧索引，必须在回调函数中获取该参数。		

IKP_FPGA_SERIAL_NUMBER 采集卡序列号

属性	Get	Set	vlc parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_FPGA_SERIAL_NUMBER (0x20000001)	
	npValue	获取采集卡序列号	

说明	采集卡序列号，该参数是 32 位无符号整数类型。
----	--------------------------

IKP_PCIE_LINK_STATE 采集卡 PCIe 链路状态

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_PCIE_LINK_STATE (0x20000002)	
	npValue	获取 PCIe 链路状态	
说明	该参数指示当前采集卡的 PCIe 链路状态，当采集卡 PCIe 链路速度无法满足图像传输速度会导致采集丢帧。		

IKP_PCIE_SPEED_MISS_REQUIREMENT CL 采集卡 PCIe 链路速度不足导致的丢帧次数

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_PCIE_SPEED_MISS_REQUIREMENT (0x20000003)	
	npValue	获取 CL 采集卡 PCIe 链路速度不足导致的丢帧次数	
说明	该参数指示 CL 采集卡 PCIe 链路速度不足导致的丢帧次数。		

IKP_PCI_CONFIGURATION 采集卡 PCIe 配置空间

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_PCI_CONFIGURATION (0x20000004)	
	npValue	采集卡 PCIe 配置空间	
说明	该参数指示采集卡 PCIe 配置空间。		

IKP_HARDWARE_TRIGGER_GENERAL_INPUT_DELAY_MODE CL 采集卡通用输入信号延迟模式

属性	Get	Set	vlcf parameter
	支持	支持	HARDWARE_TRIGGER_GENERAL_INPUT_DELAY_MODE
SetInfo	hDev	设备句柄	

	uType	IKP_HARDWARE_TRIGGER_GENERAL_INPUT_DELAY_MODE (0x20000005)	
	nValue	设置 CL 采集卡外触发通用输入信号延迟模式	
		IKP_HARDWARE_TRIGGER_GENERAL_INPUT_DELAY_MODE_IN_LINES	行延迟
		IKP_HARDWARE_TRIGGER_GENERAL_INPUT_DELAY_MODE_IN_US	时间延迟
GetInfo	hDev	设备句柄	
	uType	IKP_HARDWARE_TRIGGER_GENERAL_INPUT_DELAY_MODE (0x20000005)	
	npValue	获取 CL 采集卡外触发通用输入信号延迟模式，具体信息见上文	
说明	本参数表示 CL 采集卡通用输入信号延迟模式，分为行延迟模式和时间延迟模式。		

IKP_HARDWARE_TRIGGER_GENERAL_INPUT1_DELAY_IN_LINES CL 采集卡通用输入信号 1 行延迟

属性	Get	Set	vlcf parameter
	支持	支持	HARDWARE_TRIGGER_GENERAL_INPUT1_DELAY_LINES
SetInfo	hDev	设备句柄	
	uType	IKP_HARDWARE_TRIGGER_GENERAL_INPUT1_DELAY_IN_LINES (0x20000006)	
	nValue	设置 CL 采集卡通用输入信号 1 行延迟大小，取值范围为 0~65535	
GetInfo	hDev	设备句柄	
	uType	IKP_HARDWARE_TRIGGER_GENERAL_INPUT1_DELAY_IN_LINES (0x20000006)	
	npValue	获取 CL 采集卡通用输入信号 1 行延迟大小	
说明	本参数表示 CL 采集卡通用输入信号 1 行延迟大小。		

IKP_HARDWARE_TRIGGER_GENERAL_INPUT2_DELAY_IN_LINES CL 采集卡通用输入信号 2 行延迟

属性	Get	Set	vlcf parameter
	支持	支持	HARDWARE_TRIGGER_GENERAL_INPUT2_DELAY_LINES
SetInfo	hDev	设备句柄	
	uType	IKP_HARDWARE_TRIGGER_GENERAL_INPUT2_DELAY_IN_LINES (0x20000007)	
	nValue	设置 CL 采集卡通用输入信号 2 行延迟大小，取值范围为 0~65535	
GetInfo	hDev	设备句柄	

	uType	IKP_HARDWARE_TRIGGER_GENERAL_INPUT2_DELAY_IN_LINES (0x20000007)
	npValue	获取 CL 采集卡通用输入信号 2 行延迟大小
说明	本参数表示 CL 采集卡通用输入信号 2 行延迟大小。	

IKP_EVENT_INPUT_INTERNAL_COUNT 采集卡内部信号计数器

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_EVENT_INPUT_INTERNAL_COUNT (0x20000008)	
	npValue	获取从采集卡读取的内部信号的个数	
说明	本参数表示从采集卡读取的内部信号的个数。		

IKP_EVENT_INPUT_GENERAL_1_COUNT 采集卡通用输入信号 1 计数器

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_EVENT_INPUT_GENERAL_1_COUNT (0x20000009)	
	npValue	获取从采集卡读取的通用输入信号 1 的信号个数	
说明	本参数表示从采集卡读取的通用输入信号 1 的信号个数。		

IKP_EVENT_INPUT_GENERAL_2_COUNT 采集卡通用输入信号 2 计数器

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_EVENT_INPUT_GENERAL_2_COUNT (0x2000000a)	
	npValue	获取从采集卡读取的通用输入信号 2 的信号个数	
说明	本参数表示从采集卡读取的通用输入信号 2 的信号个数。		

IKP_EVENT_INPUT_SHAFT_ENCODER_A_COUNT 采集卡编码器 A 通道信号计数器

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_EVENT_INPUT_SHAFT_ENCODER_A_COUNT (0x2000000b)	
	npValue	获取从采集卡读取的编码器 A 通道的信号个数	
说明	本参数表示从采集卡读取的编码器 A 通道的信号个数。		

IKP_EVENT_INPUT_SHAFT_ENCODER_B_COUNT 采集卡编码器 B 通道信号计数器

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_EVENT_INPUT_SHAFT_ENCODER_B_COUNT (0x2000000c)	
	npValue	获取从采集卡读取的编码器 B 通道的信号个数	
说明	本参数表示从采集卡读取的编码器 B 通道的信号个数。		

IKP_EVENT_INPUT_BOARD_SYNC_IN_1_COUNT 采集卡板间同步信号 1 计数器

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_EVENT_INPUT_BOARD_SYNC_IN_1_COUNT (0x2000000d)	
	npValue	获取从采集卡读取的板间同步信号 1 的信号个数	
说明	本参数表示从采集卡读取的板间同步信号 1 的信号个数。		

IKP_EVENT_INPUT_INTEGRATION_SIG_1_COUNT 采集卡积分控制信号 1 计数器

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_EVENT_INPUT_INTEGRATION_SIG_1_COUNT (0x2000000e)	

	npValue	获取从采集卡读取的积分控制信号 1 的信号个数
说明	本参数表示从采集卡读取的积分控制信号 1 的信号个数。	

IKP_EVENT_INPUT_INTEGRATION_SIG_2_COUNT 采集卡积分控制信号 2 计数器

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_EVENT_INPUT_INTEGRATION_SIG_2_COUNT (0x2000000f)	
	npValue	获取从采集卡读取的积分控制信号 2 的信号个数	
说明	本参数表示从采集卡读取的积分控制信号 2 的信号个数。		

IKP_RCV_MORE_DATA_IN_TRIGGER_MODE CL 采集卡行间超时情况下是否丢弃已采集的未完成帧的数据

属性	Get	Set	vlcf parameter
	支持	支持	RCV_MORE_DATA_IN_TRIGGER_MODE
SetInfo	hDev	设备句柄	
	uType	IKP_RCV_MORE_DATA_IN_TRIGGER_MODE (0x20000010)	
	nValue	设置 CL 采集卡行间超时情况下是否丢弃已采集的未完成帧的数据	
GetInfo	hDev	设备句柄	
	uType	IKP_RCV_MORE_DATA_IN_TRIGGER_MODE (0x20000010)	
	npValue	获取 CL 采集卡行间超时情况下是否丢弃已采集的未完成帧的数据	
说明	本参数表示 CL 采集卡行间超时情况下是否丢弃已采集的未完成帧的数据。		

IKP_DISABLE_IO_EVENT 禁用 IO 事件

属性	Get	Set	vlcf parameter
	支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_DISABLE_IO_EVENT (0x20000011)	
	nValue	设置是否禁用 IO 事件	
GetInfo	hDev	设备句柄	
	uType	IKP_DISABLE_IO_EVENT (0x20000011)	
	npValue	获取是否禁用 IO 事件	
说明	本参数表示是否禁用 IO 事件。		

IKP_PUBLIC_VERSION_SUPPORT 是否支持公版库

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_PUBLIC_VERSION_SUPPORT (0x20000012)	
	npValue	0	不支持公版库
		1	支持公版库
说明	本参数表示是否支持公版库。		

IKP_SHAFT_ENCODER1_REVERSE_COMPENSATION_LIMIT CL 采集卡编码器反转补偿最大值

属性	Get	Set	vlcf parameter
	支持	支持	SHAFT_ENCODER1_REVERSE_COMPENSATION
SetInfo	hDev	设备句柄	
	uType	IKP_SHAFT_ENCODER1_REVERSE_COMPENSATION_LIMIT (0x20000013)	
	nValue	设置 CL 采集卡编码器反转补偿最大值	
GetInfo	hDev	设备句柄	
	uType	IKP_SHAFT_ENCODER1_REVERSE_COMPENSATION_LIMIT (0x20000013)	
	npValue	获取 CL 采集卡编码器反转补偿最大值	
说明	本参数表示 CL 采集卡编码器反转补偿最大值，超过该值后不再补偿编码器反转信号。		

IKP_SHAFT_ENCODER1_CLOCK_DUTY_COMPENSATION_TYPE CL 采集卡行信号占空比补偿功能类型

属性	Get	Set	vlcf parameter
	支持	支持	SHAFT_ENCODER1_CLOCK_DUTY_TYPE
SetInfo	hDev	设备句柄	
	uType	IKP_SHAFT_ENCODER1_CLOCK_DUTY_COMPENSATION_TYPE (0x20000014)	
	nValue	设置 CL 采集卡行信号占空比补偿功能类型	
		IKP_SHAFT_ENCODER1_CLOCK_DUTY_COMPENSATION_RISING_EDGE	上升沿
		IKP_SHAFT_ENCODER1_CLOCK_DUTY_COMPENSATION_FALLING_EDGE	下降沿
GetInfo	hDev	设备句柄	

	uType	IKP_SHAFT_ENCODER1_CLOCK_DUTY_COMPENSATION_TYPE (0x20000014)
	npValue	获取 CL 采集卡行信号占空比补偿功能类型，具体信息见上文
说明	本参数表示 CL 采集卡行信号占空比补偿功能。当采集卡同时使用 AB 相触发且占空比不是 50%时，采集卡行频会发生波动。为了消除行频波动，可以设置此选项。当占空比小于 50%时选择 Falling，当占空比大于 50%时选择 Rising。	

IKP_SHAFT_ENCODER1_CLOCK_DUTY_COMPENSATION_WIDTH CL 采集卡行信号占空比不是 50%时的补偿宽度

属性	Get	Set	vlcf parameter
	支持	支持	SHAFT_ENCODER1_CLOCK_DUTY_WIDTH
SetInfo	hDev	设备句柄	
	uType	IKP_SHAFT_ENCODER1_CLOCK_DUTY_COMPENSATION_WIDTH (0x20000015)	
	nValue	设置 CL 采集卡行信号占空比不是 50%时的补偿宽度	
GetInfo	hDev	设备句柄	
	uType	IKP_SHAFT_ENCODER1_CLOCK_DUTY_COMPENSATION_WIDTH (0x20000015)	
	npValue	获取 CL 采集卡行信号占空比不是 50%时的补偿宽度	
说明	本参数表示 CL 采集卡行信号占空比不是 50%时的补偿宽度，计算公式如下：Clock Duty Compensation Width(ns)= （0.5 - 占空比）*Period 其中 Period 为触发信号行频对应的周期，单位 ns。		

IKP_CL_VALID_COLUMN CL 采集卡有效列

属性	Get	Set	vlcf parameter
	支持	支持	VALID_COLUMN
SetInfo	hDev	设备句柄	
	uType	IKP_CL_VALID_COLUMN (0x20000016)	
	nValue	设置 CL 采集卡有效列	
		IKP_CL_VALID_COLUMN_ALL	所有列
		IKP_CL_VALID_COLUMN_EVEN	偶数列
GetInfo	npValue	IKP_CL_VALID_COLUMN_ODD	奇数列
		hDev	设备句柄
		uType	IKP_CL_VALID_COLUMN (0x20000016)
说明	本参数表示 CL 采集卡有效列，抽取所有列或偶数列或奇数列进行图像输出。设置奇/偶数列时需要注意行分辨率减半。		

IKP_CL_SIGNAL_ENHANCE_MODE CL 采集卡信号增强模式

属性	Get	Set	vlfw parameter	
	支持	支持	SIGNAL_ENHANCE_MODE	
SetInfo	hDev	设备句柄		
	uType	IKP_CL_SIGNAL_ENHANCE_MODE (0x20000017)		
	nValue	设置是否开启 CL 采集卡信号增强模式		
		IKP_CL_SIGNAL_ENHANCE_MODE_OFF	关闭	
		IKP_CL_SIGNAL_ENHANCE_MODE_ON	开启	
GetInfo	hDev	设备句柄		
	uType	IKP_CL_SIGNAL_ENHANCE_MODE (0x20000017)		
	npValue	获取是否开启 CL 采集卡信号增强模式，具体信息见上文		
说明	本参数表示是否开启 CL 采集卡信号增强模式，一般保持默认设置，主要用于改善 Plus 采集卡在远距离传输时（如使用 7 米、10 米线缆）的误码现象。需要注意的是 Plus 采集卡在短距离传输时（如使用 3 米、5 米线缆）要关闭信号增强模式，其他情况默认开启。			

IKP_CL_LONG_DISTANCE_TRANSMISSION K6 采集卡远距离传输功能

属性	Get	Set	vlfw parameter	
	支持	支持	LONG_DISTANCE_TRANSMISSION	
SetInfo	hDev	设备句柄		
	uType	IKP_CL_LONG_DISTANCE_TRANSMISSION (0x20000018)		
	nValue	IKP_CL_LONG_DISTANCE_TRANSMISSION_OFF	传输线缆小于等于 7 米时关闭该功能	
		IKP_CL_LONG_DISTANCE_TRANSMISSION_ON	传输线缆大于 7 米时开启该功能	
GetInfo	hDev	设备句柄		
	uType	IKP_CL_LONG_DISTANCE_TRANSMISSION (0x20000018)		
	npValue	具体信息见上文		
说明	针对 K6 采集卡的功能。当传输线缆小于等于 7 米时关闭该功能，否则开启该功能。			

IKP_FPGA_SERIAL_NUMBER_HIGH 采集卡序列号高 32 位

属性	Get	Set	vlfw parameter
	支持	不支持	FPGA_SERIAL_NUMBER_HIGH
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_FPGA_SERIAL_NUMBER_HIGH (0x20000019)	
	npValue	获取采集卡序列号高 32 位。	
说明	采集卡序列号共 64 位，本参数用于获取采集卡序列号高 32 位数据。		

IKP_CXP_TEST_IMAGE CXP 采集卡测试图像

属性	Get	Set	vlfw parameter	
	支持	支持	CXP_TEST_IMAGE	
SetInfo	hDev	设备句柄		
	uType	IKP_CXP_TEST_IMAGE (0x30000001)		
	nValue	设置 CXP 采集卡测试图像		
		IKP_CXP_TEST_IMAGE_OFF		关闭测试图像功能
		IKP_CXP_TEST_IMAGE_1		测试图像 1
		IKP_CXP_TEST_IMAGE_2		测试图像 2
GetInfo	hDev	设备句柄		
	uType	IKP_CXP_TEST_IMAGE (0x30000001)		
	npValue	获取 CXP 采集卡测试图像，具体信息见上文		
说明	本参数表示 CXP 采集卡测试图像。			

IKP_CXP_TRIGGER_OUTPUT_SELECTOR CXP 采集卡触发输出选择器

属性	Get	Set	vlfw parameter	
	支持	支持	CXP_TRIGGER_OUTPUT_SELECTOR	
SetInfo	hDev	设备句柄		
	uType	IKP_CXP_TRIGGER_OUTPUT_SELECTOR (0x30000002)		
	nValue	设置 CXP 采集卡触发输出信号		
		IKP_CXP_TRIGGER_OUTPUT_NO		关闭触发输出功能
		IKP_CXP_TRIGGER_OUTPUT_INTEGRATION_SIGNAL1		积分信号 1
		IKP_CXP_TRIGGER_OUTPUT_INTEGRATION_SIGNAL2		积分信号 2
GetInfo	hDev	设备句柄		
	uType	IKP_CXP_TRIGGER_OUTPUT_SELECTOR (0x30000002)		
	npValue	获取 CXP 采集卡触发输出信号，具体信息见上文		
说明	本参数表示 CXP 采集卡触发输出信号。			

IKP_LAST_FRAME_INDEX 最新帧索引

属性	Get	Set	vlcw parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_LAST_FRAME_INDEX (0x30000003)	
	npValue	获取图像采集最新帧索引	
说明	本参数表示采集卡最近一次采集完成的图像的帧索引。		

IKP_CXP_VOLTAGE_SUPPLY_STATUS CXP 采集卡电压源状态

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_VOLTAGE_SUPPLY_STATUS (0x30000004)	
	npValue	获取当前选中通道的电压供电状态	
说明	本参数表示 CXP 采集卡电压源状态。		

IKP_CXP_POWER_SWITCH CXP 采集卡 PoCXP 功能开关

属性	Get	Set	vlcf parameter
	支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_POWER_SWITCH (0x30000005)	
	nValue	设置 CXP 采集卡当前选中通道的 PoCXP 功能开关	
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_POWER_SWITCH (0x30000005)	
	npValue	获取 CXP 采集卡当前选中通道的 PoCXP 功能开关	
说明	本参数表示 CXP 采集卡当前选中通道的 PoCXP 功能开关。		

IKP_CXP_POWER_STATUS CXP 采集卡电源状态

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_POWER_STATUS (0x30000006)	
	npValue	获取 CXP 采集卡电源状态	
说明	本参数表示 CXP 采集卡电源状态。Off 表示当前选中的通道 PoCXP 功能关闭，或未插接外部供电线；Detecting 表示当前选中的通道 PoCXP 功能打开，但未找到匹配的 PoCXP 受电设备；On 表示当前选中的通道正在为设备提供 PoCXP 供电；Ocp 表示当前选中的通道发生了过流保护事件，需要进行恢复。		

IKP_CXP_SENSE_CURRENT CXP 采集卡当前选中通道的供电电流

属性	Get	Set	vlcf parameter
	支持	不支持	无

SetInfo	本参数不支持手动设定	
GetInfo	hDev	设备句柄
	uType	IKP_CXP_SENSE_CURRENT (0x30000007)
	npValue	获取 CXP 采集卡当前选中通道的供电电流
说明	本参数表示 CXP 采集卡当前选中通道的供电电流。	

IKP_CXP_BUS_VOLTAGE CXP 采集卡当前选中通道的供电电压

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_BUS_VOLTAGE (0x30000008)	
	npValue	获取 CXP 采集卡当前选中通道的供电电压	
说明	本参数表示 CXP 采集卡当前选中通道的供电电压。		

IKP_CXP_RESET_OCP CXP 采集卡过流保护重置

属性	Get	Set	vlcf parameter
	不支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_RESET_OCP (0x30000009)	
	nValue	设置 CXP 采集卡过流保护重置	
GetInfo	本参数不支持获取		
说明	本参数表示 CXP 采集卡过流保护重置。		

IKP_CXP_SENSE_VOLTAGE_HIGH CXP 采集卡检测状态监测电压上限

属性	Get	Set	vlcf parameter
	支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_SENSE_VOLTAGE_HIGH (0x3000000a)	
	nValue	设置 CXP 采集卡检测状态监测电压上限	
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_SENSE_VOLTAGE_HIGH (0x3000000a)	
	npValue	获取 CXP 采集卡检测状态监测电压上限	
说明	本参数表示 CXP 采集卡检测状态监测电压上限，默认值 4400（5.5V）。		

IKP_CXP_SENSE_VOLTAGE_LOW CXP 采集卡检测状态监测电压下限

属性	Get	Set	vlcf parameter
	支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_SENSE_VOLTAGE_LOW (0x3000000b)	
	nValue	设置 CXP 采集卡检测状态监测电压下限	
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_SENSE_VOLTAGE_LOW (0x3000000b)	
	npValue	获取 CXP 采集卡检测状态监测电压下限	
说明	本参数表示 CXP 采集卡检测状态监测电压下限，默认值 800（1V）。		

IKP_CXP_SUPPLY_VOLTAGE_LOW CXP 采集卡供电状态监测电压下限

属性	Get	Set	vlcf parameter
	支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_SUPPLY_VOLTAGE_LOW (0x3000000c)	
	nValue	设置 CXP 采集卡供电状态监测电压下限	
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_SUPPLY_VOLTAGE_LOW (0x3000000c)	
	npValue	获取 CXP 采集卡供电状态监测电压下限	
说明	本参数表示 CXP 采集卡 CXP 采集卡供电状态监测电压下限，默认值 800(1V)。		

IKP_CXP_SUPPLY_CURRENT_LOW CXP 采集卡供电状态监测电流下限

属性	Get	Set	vlcf parameter
	支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_SUPPLY_CURRENT_LOW (0x3000000d)	
	nValue	设置 CXP 采集卡供电状态监测电流下限	
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_SUPPLY_CURRENT_LOW (0x3000000d)	
	npValue	获取 CXP 采集卡供电状态监测电流下限	
说明	本参数表示 CXP 采集卡 CXP 采集卡供电状态监测电流下限，默认值 320（8mA）。		

IKP_CXP_FPGA_FRAME_TIMEOUT CXP 采集卡超时时间【已弃用】

属性	Get	Set	vlcf parameter
	支持	支持	无

SetInfo	hDev	设备句柄
	uType	IKP_CXP_FPGA_FRAME_TIMEOUT (0x3000000e)
	nValue	设置 CXP 采集卡超时时间
GetInfo	hDev	设备句柄
	uType	IKP_CXP_FPGA_FRAME_TIMEOUT (0x3000000e)
	npValue	获取 CXP 采集卡超时时间
说明	本参数表示 CXP 采集卡超时时间。	

IKP_CXP_FPGA_FRAME_TIMEOUT_MULTIPLE CXP 采集卡超时时间【已弃用】

属性	Get	Set	vlcf parameter
	支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_FPGA_FRAME_TIMEOUT_MULTIPLE (0x3000000f)	
	nValue	设置 CXP 采集卡超时时间	
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_FPGA_FRAME_TIMEOUT_MULTIPLE (0x3000000f)	
	npValue	获取 CXP 采集卡超时时间	
说明	本参数表示 CXP 采集卡超时时间。		

IKP_CXP_CRC_ERROR_COUNT CXP 采集卡 CRC 错误个数

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_CRC_ERROR_COUNT (0x30000010)	
	npValue	获取 CXP 采集卡 CRC 错误个数	
说明	本参数表示 CXP 采集卡 CRC 错误个数。		

IKP_CXP_PoCXP_CHANNEL CXP 采集卡 PoCXP 通道

属性	Get	Set	vlcf parameter
	支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_PoCXP_CHANNEL (0x30000011)	
	nValue	设置 CXP 采集卡 PoCXP 通道	
GetInfo	hDev	设备句柄	

	uType	IKP_CXP_PoCXP_CHANNEL (0x30000011)
	npValue	获取 CXP 采集卡 PoCXP 通道
说明	本参数表示 CXP 采集卡 PoCXP 通道。	

IKP_CXP_PoCXP_LOCKED_VOLTAGE CXP 采集卡切换至高压前检测到的 PoCXP 电压值

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_PoCXP_LOCKED_VOLTAGE (0x30000012)	
	npValue	获取 CXP 采集卡切换至高压前检测到的 PoCXP 电压值	
说明	本参数表示 CXP 采集卡切换至高压前检测到的 PoCXP 电压值。		

IKP_CXP_SHAFT_ENCODER_DEBOUNCE CXP 采集卡编码器输入信号去抖窗口长度

属性	Get	Set	vlcf parameter
	支持	支持	SHAFT_ENCODER_DEBOUNCE
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_SHAFT_ENCODER_DEBOUNCE (0x30000013)	
	nValue	设置 CXP 采集卡编码器输入信号去抖窗口长度，取值范围为 0~65535	
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_SHAFT_ENCODER_DEBOUNCE (0x30000013)	
	npValue	获取 CXP 采集卡编码器输入信号去抖窗口长度	
说明	本参数表示 CXP 采集卡编码器输入信号去抖窗口长度，为了减少毛刺信号对于相机的影响。去抖窗口越大，滤除信号的脉冲宽度也越大。		

IKP_CXP_SHAFT_ENCODER_TICK_MODE CXP 采集卡编码器计数器模式

属性	Get	Set	vlcf parameter
	支持	支持	SHAFT_ENCODER_TICK_MODE
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_SHAFT_ENCODER_TICK_MODE (0x30000014)	
	nValue	设置 CXP 采集卡编码器计数器模式	
		IKP_CXP_SHAFT_ENCODER_TICK_MODE_FOLLOW_DIRECTION	正向递增反向递减
		IKP_CXP_SHAFT_ENCODER_TICK_MODE_ANY_DIRECTION	任意方向均递增

GetInfo	hDev	设备句柄
	uType	IKP_CXP_SHAFT_ENCODER_TICK_MODE (0x30000014)
	npValue	获取 CXP 采集卡编码器计数器模式
说明	本参数表示 CXP 采集卡编码器计数器模式。	

IKP_CXP_SHAFT_ENCODER_TICK_MAX CXP 采集卡编码器的计数器最大值

属性	Get	Set	vlcf parameter
	支持	支持	SHAFT_ENCODER_TICK_MAX
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_SHAFT_ENCODER_TICK_MAX (0x30000015)	
	nValue	设置 CXP 采集卡编码器的计数器最大值，取值范围为 0~65535	
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_SHAFT_ENCODER_TICK_MAX (0x30000015)	
	npValue	获取 CXP 采集卡编码器的计数器最大值	
说明	本参数表示 CXP 采集卡编码器的计数器最大值。		

IKP_CXP_SHAFT_ENCODER_TICK_RESET CXP 采集卡编码器计数器重置

属性	Get	Set	vlcf parameter
	不支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_SHAFT_ENCODER_TICK_RESET (0x30000016)	
	nValue	设置 CXP 采集卡编码器计数器重置	
GetInfo	本参数不支持获取		
说明	本参数表示 CXP 采集卡编码器计数器重置。		

IKP_CXP_SHAFT_ENCODER_TICK_COUNT CXP 采集卡编码器计数器当前值

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_SHAFT_ENCODER_TICK_COUNT (0x30000017)	
	npValue	获取 CXP 采集卡编码器计数器当前值	
说明	本参数表示 CXP 采集卡编码器计数器当前值。		

IKP_CXP_SHAFT_ENCODER_REVERSE_MODE CXP 采集卡编码器工作模式

属性	Get	Set	vlcf parameter
	支持	支持	SHAFT_ENCODER_TICK_REVERSE_MODE
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_SHAFT_ENCODER_REVERSE_MODE (0x30000018)	
	nValue	设置 CXP 采集卡编码器工作模式	
		IKP_CXP_SHAFT_ENCODER_REVERSE_MODE_FORWARD_ONLY	仅正向
		IKP_CXP_SHAFT_ENCODER_REVERSE_MODE_ANY_DIRECTION	任意方向
		IKP_CXP_SHAFT_ENCODER_REVERSE_MODE_BACKWARD_ONLY	仅反向
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_SHAFT_ENCODER_REVERSE_MODE (0x30000018)	
	npValue	获取 CXP 采集卡编码器工作模式	
说明	本参数表示 CXP 采集卡编码器工作模式。		

IKP_CXP_SHAFT_ENCODER_REVERSE_MAX CXP 采集卡编码器反向计数器最大值

属性	Get	Set	vlcf parameter
	支持	支持	SHAFT_ENCODER_TICK_REVERSE_MAX
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_SHAFT_ENCODER_REVERSE_MAX (0x30000019)	
	nValue	设置 CXP 采集卡编码器反向计数器最大值，取值范围为 0~65535	
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_SHAFT_ENCODER_REVERSE_MAX (0x30000019)	
	npValue	获取 CXP 采集卡编码器反向计数器最大值	
说明	本参数表示 CXP 采集卡编码器反向计数器最大值。		

IKP_CXP_SHAFT_ENCODER_REVERSE_RESET CXP 采集卡编码器反向计数器重置

属性	Get	Set	vlcf parameter
	不支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_SHAFT_ENCODER_REVERSE_RESET (0x30000020)	
	nValue	设置 CXP 采集卡编码器反向计数器重置	

GetInfo	本参数不支持获取
说明	本参数表示 CXP 采集卡编码器反向计数器重置。

IKP_CXP_SHAFT_ENCODER_REVERSE_COUNT CXP 采集卡编码器反向计数器当前值

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_SHAFT_ENCODER_REVERSE_COUNT (0x30000021)	
	npValue	获取 CXP 采集卡编码器反向计数器当前值	
说明	本参数表示 CXP 采集卡编码器反向计数器当前值。		

IKP_CXP_GENERAL_OUTPUT1_THRESHOLD CXP 采集卡通用输出信号 1 阈值

属性	Get	Set	vlcf parameter	
	支持	支持	GENERAL_OUTPUT1_THRESHOLD	
SetInfo	hDev	设备句柄		
	uType	IKP_CXP_GENERAL_OUTPUT1_THRESHOLD (0x30000022)		
	nValue	设置 CXP 采集卡通用输出信号 1 阈值		
		IKP_GENERAL_OUTPUT_THRESHOLD_3 V3	3.3V	
		IKP_GENERAL_OUTPUT_THRESHOLD_2 4V	24V	
GetInfo	hDev	设备句柄		
	uType	IKP_CXP_GENERAL_OUTPUT1_THRESHOLD (0x30000022)		
	npValue	获取 CXP 采集卡通用输出信号 1 阈值		
说明	本参数表示 CXP 采集卡通用输出信号 1 阈值。			

IKP_CXP_GENERAL_OUTPUT2_THRESHOLD CXP 采集卡通用输出信号 2 阈值

属性	Get	Set	vlcf parameter
	支持	支持	GENERAL_OUTPUT2_THRESHOLD
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_GENERAL_OUTPUT2_THRESHOLD (0x30000023)	
	nValue	设置 CXP 采集卡通用输出信号 2 阈值	
		IKP_GENERAL_OUTPUT_THRESHOLD_3 V3	3.3V
		IKP_GENERAL_OUTPUT_THRESHOLD_2 4V	24V

		4V	
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_GENERAL_OUTPUT2_THRESHOLD (0x30000023)	
	npValue	获取 CXP 采集卡通用输出信号 2 阈值	
说明	本参数表示 CXP 采集卡通用输出信号 2 阈值。		

IKP_CXP_FIRMWARE_TYPE CXP 采集卡固件类型

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_FIRMWARE_TYPE (0x30000024)	
	npValue	获取 CXP 采集卡固件类型	
		IKP_CXP_FIRMWARE_TYPE_1_C AMERA	1 拖 1 类型
		IKP_CXP_FIRMWARE_TYPE_2_C AMERA	1 拖 2 类型
说明	本参数表示 CXP 采集卡固件类型。		

IKP_CXP_TEMPERATURE CXP 采集卡温度

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_TEMPERATURE (0x30000025)	
	npValue	获取 CXP 采集卡温度	
说明	本参数表示 CXP 采集卡温度。		

IKP_CXP_GENERAL_INPUT_THRESHOLD CXP 采集卡通用输入信号阈值

属性	Get	Set	vlcf parameter
	支持	支持	GENERAL_INPUT_THRESHOLD
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_GENERAL_INPUT_THRESHOLD (0x30000026)	
	nValue	设置 CXP 采集卡通用输入信号阈值	
		IKP_CXP_GENERAL_INPUT_THRESHOLD_5 V	5V
		IKP_CXP_GENERAL_INPUT_THRESHOLD_3 V	3V

GetInfo	hDev	设备句柄
	uType	IKP_CXP_GENERAL_INPUT_THRESHOLD (0x30000026)
	npValue	获取 CXP 采集卡通用输入信号阈值
说明	本参数表示 CXP 采集卡通用输入信号阈值。	

IKP_CXP_GENERAL_OUTPUT1_SOURCE_CHANNEL CXP 采集卡通用输出信号 1 通道

属性	Get	Set	vlfw parameter	
	支持	支持	GENERAL_OUTPUT1_SOURCE_CHANNEL	
SetInfo	hDev	设备句柄		
	uType	IKP_CXP_GENERAL_OUTPUT1_SOURCE_CHANNEL (0x30000027)		
	nValue	设置 CXP 采集卡通用输出信号 1 通道		
		IKP_CXP_GENERAL_OUTPUT_SOURCE_C HANNEL_A	通道 A	
		IKP_CXP_GENERAL_OUTPUT_SOURCE_C HANNEL_B	通道 B	
		IKP_CXP_GENERAL_OUTPUT_SOURCE_C HANNEL_C	通道 C	
		IKP_CXP_GENERAL_OUTPUT_SOURCE_C HANNEL_D	通道 D	
GetInfo	hDev	设备句柄		
	uType	IKP_CXP_GENERAL_OUTPUT1_SOURCE_CHANNEL (0x30000027)		
	npValue	获取 CXP 采集卡通用输出信号 1 通道		
说明	本参数表示 CXP 采集卡通用输出信号 1 通道。			

IKP_CXP_GENERAL_OUTPUT2_SOURCE_CHANNEL CXP 采集卡通用输出信号 2 通道

属性	Get	Set	vlfw parameter	
	支持	支持	GENERAL_OUTPUT2_SOURCE_CHANNEL	
SetInfo	hDev	设备句柄		
	uType	IKP_CXP_GENERAL_OUTPUT2_SOURCE_CHANNEL (0x30000028)		
	nValue	设置 CXP 采集卡通用输出信号 2 通道		
		IKP_CXP_GENERAL_OUTPUT_SOURCE_C HANNEL_A	通道 A	
		IKP_CXP_GENERAL_OUTPUT_SOURCE_C HANNEL_B	通道 B	
IKP_CXP_GENERAL_OUTPUT_SOURCE_C HANNEL C		通道 C		

		IKP_CXP_GENERAL_OUTPUT_SOURCE_C HANNEL_D	通道 D
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_GENERAL_OUTPUT2_SOURCE_CHANNEL (0x30000028)	
	npValue	获取 CXP 采集卡通用输出信号 2 通道	
说明	本参数表示 CXP 采集卡通用输出信号 2 通道。		

IKP_CXP_GENERAL_INPUT1_TYPE CXP 采集卡通用输入信号 1 模式

属性	Get	Set	vlcf parameter	
	支持	支持	GENERAL_INPUT1_TYPE	
SetInfo	hDev	设备句柄		
	uType	IKP_CXP_GENERAL_INPUT1_TYPE (0x30000029)		
	nValue	设置 CXP 采集卡通用输入信号 1 模式		
		IKP_CXP_GENERAL_INPUT_TYPE_DIFFERENTIAL		差分
		IKP_CXP_GENERAL_INPUT_TYPE_SINGLE_END		单端
GetInfo	hDev	设备句柄		
	uType	IKP_CXP_GENERAL_INPUT1_TYPE (0x30000029)		
	npValue	获取 CXP 采集卡通用输入信号 1 模式		
说明	本参数表示 CXP 采集卡通用输入信号 1 模式。			

IKP_CXP_GENERAL_INPUT2_TYPE CXP 采集卡通用输入信号 2 模式

属性	Get	Set	vlcf parameter
	支持	支持	GENERAL_INPUT2_TYPE
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_GENERAL_INPUT2_TYPE (0x3000002a)	
	nValue	设置 CXP 采集卡通用输入信号 2 模式	
		IKP_CXP_GENERAL_INPUT_TYPE_DIFFERENTIAL	差分
		IKP_CXP_GENERAL_INPUT_TYPE_SINGLE_END	单端
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_GENERAL_INPUT2_TYPE (0x3000002a)	
	npValue	获取 CXP 采集卡通用输入信号 2 模式	
说明	本参数表示 CXP 采集卡通用输入信号 2 模式。		

IKP_CXP_TRANSFER_CHANNEL_SELECTOR CXP 采集卡传输通道选择器

属性	Get	Set	vlcfc parameter
	支持	支持	无
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_TRANSFER_CHANNEL_SELECTOR (0x3000002b)	
	nValue	设置 CXP 采集卡传输通道	
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_TRANSFER_CHANNEL_SELECTOR (0x3000002b)	
	npValue	获取 CXP 采集卡传输通道	
说明	本参数表示 CXP 采集卡传输通道。		

IKP_CXP_CHANNEL_LOST_COUNT CXP 传输包丢失数量

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_CHANNEL_LOST_COUNT (0x3000002c)	
	npValue	获取 CXP 传输包丢失数量	
说明	本参数表示 CXP 传输包丢失数量。		

IKP_CXP_CHANNEL_ERROR_COUNT CXP 传输包错误数量

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_CHANNEL_ERROR_COUNT (0x3000002d)	
	npValue	获取 CXP 传输包错误数量	
说明	本参数表示 CXP 传输包错误数量。		

IKP_TRIGGER_FRAME_ACTIVE_MODE 帧触发不定长采集模式

属性	Get	Set	vlfw parameter	
	支持	支持	FRAME_ACTIVE_MODE	
SetInfo	hDev	设备句柄		
	uType	IKP_TRIGGER_FRAME_ACTIVE_MODE (0x3000002e)		
	nValue	设置帧触发不定长采集模式		
		IKP_TRIGGER_FRAME_ACTIVE_MODE_OFF		关闭

		IKP_TRIGGER_FRAME_ACTIVE_MODE_ON	开启
GetInfo	hDev	设备句柄	
	uType	IKP_TRIGGER_FRAME_ACTIVE_MODE (0x3000002e)	
	npValue	获取帧触发不定长采集模式	
说明	本参数表示帧触发不定长采集模式。		

IKP_JPEG_COMPRESS_ENABLE 使能 JPEG 压缩模式

属性	Get	Set	vlc parameter	
	支持	支持	JPEG_COMPRESS_ENABLE	
SetInfo	hDev	设备句柄		
	uType	IKP_JPEG_COMPRESS_ENABLE (0x3000002f)		
	nValue	设置 JPEG 压缩模式		
		IKP_JPEG_COMPRESS_ENABLE_OFF		关闭
		IKP_JPEG_COMPRESS_ENABLE_ON		开启
GetInfo	hDev	设备句柄		
	uType	IKP_JPEG_COMPRESS_ENABLE (0x3000002f)		
	npValue	获取 JPEG 压缩模式		
说明	本参数表示 JPEG 压缩模式。			

IKP_JPEG_COMPRESS_QUALITY JPEG 压缩质量

属性	Get	Set	vlc parameter
	支持	支持	JPEG_COMPRESS_QUALITY
SetInfo	hDev	设备句柄	
	uType	IKP_JPEG_COMPRESS_QUALITY (0x30000030)	
	nValue	设置 JPEG 压缩质量，取值范围为 1~100	
GetInfo	hDev	设备句柄	
	uType	IKP_JPEG_COMPRESS_QUALITY (0x30000030)	
	npValue	获取 JPEG 压缩质量	
说明	本参数表示 JPEG 压缩质量，数值越大，图像压缩质量越好，传输数据量也越大。		

IKP_CXP_FRAME_BURST_COUNT CXP 采集卡接受一个触发信号后产生的给相机的连续信号的个数

属性	Get	Set	vlcf parameter
	支持	支持	FRAME_BURST_COUNT
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_FRAME_BURST_COUNT (0x30000031)	
	nValue	设置 CXP 采集卡接受一个触发信号后产生的给相机的连续信号的个	

		数，取值范围为 1 ~ 2147483647
GetInfo	hDev	设备句柄
	uType	IKP_CXP_FRAME_BURST_COUNT (0x30000031)
	npValue	获取 CXP 采集卡接受一个触发信号后产生的给相机的连续信号的个数
说明	本参数表示 CXP 采集卡接受一个触发信号后产生的给相机的连续信号的个数。	

IKP_CXP_FRAME_BURST_PERIOD CXP 采集卡产生连续信号的周期

属性	Get	Set	vlcf parameter
	支持	支持	FRAME_BURST_PERIOD
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_FRAME_BURST_PERIOD (0x30000032)	
	nValue	设置 CXP 采集卡产生连续信号的周期	
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_FRAME_BURST_PERIOD (0x30000032)	
	npValue	获取 CXP 采集卡产生连续信号的周期	
说明	本参数表示 CXP 采集卡产生连续信号的周期。该参数应该大于相机的最小帧周期，否则可能会出现丢失触发信号的现象。		

IKP_CXP_CHANNEL_CRC_ERROR_COUNT CXP 采集卡通道 CRC 错误发生次数

属性	Get	Set	vlcf parameter
	支持	不支持	无
SetInfo	本参数不支持手动设定		
GetInfo	hDev	设备句柄	
	uType	IKP_CXP_CHANNEL_CRC_ERROR_COUNT (0x30000033)	
	npValue	获取 CXP 采集卡通道 CRC 错误发生次数	
说明	本参数表示 CXP 采集卡通道 CRC 错误发生次数。		

IKP_CXP_TRIG_EDGE_MODE CXP 采集卡边沿触发模式

属性	Get	Set	vlcf parameter
	支持	支持	TRIG_EDGE_MODE
SetInfo	hDev	设备句柄	
	uType	IKP_CXP_TRIG_EDGE_MODE (0x30000034)	
	nValue	设置 CXP 采集卡边沿触发模式	
		IKP_CXP_TRIG_EDGE_MODE_DOUBLE_EDGE	双边
GetInfo	hDev	IKP_CXP_TRIG_EDGE_MODE_RISE_ONLY	单边
		设备句柄	

	uType	IKP_CXP_TRIG_EDGE_MODE (0x30000034)
	npValue	获取 CXP 采集卡边沿触发模式
说明	本参数表示 CXP 采集卡边沿触发模式。	

IKP_CXP_DATA_PACKED_TRANSFER CXP 采集卡压缩传输模式

属性	Get	Set	vlfw parameter	
	支持	支持	DATA_PACKED_TRANSFER	
SetInfo	hDev	设备句柄		
	uType	IKP_CXP_DATA_PACKED_TRANSFER (0x30000035)		
	nValue	设置 CXP 采集卡压缩传输模式		
		IKP_CXP_DATA_PACKED_TRANSFER_OFF	关闭	
		IKP_CXP_DATA_PACKED_TRANSFER_ON	开启	
GetInfo	hDev	设备句柄		
	uType	IKP_CXP_DATA_PACKED_TRANSFER (0x30000035)		
	npValue	获取 CXP 采集卡压缩传输模式		
说明	本参数表示 CXP 采集卡压缩传输模式。			

IKP_CXP_TRIG_LEVEL 光口采集卡传输给相机的触发包电平极性

属性	Get	Set	vlfw parameter	
	支持	支持	CXP_TRIG_LEVEL	
SetInfo	hDev	设备句柄		
	uType	IKP_CXP_TRIG_LEVEL (0x30000036)		
	nValue	设置光口采集卡传输给相机的触发包电平极性		
		IKP_CXP_TRIG_LEVEL_ALL_LOW		低电平
		IKP_CXP_TRIG_LEVEL_ALL_HIGH		高电平
		IKP_CXP_TRIG_LEVEL_GENERAL_INPUT1		通用输入信号 1 同步
		IKP_CXP_TRIG_LEVEL_GENERAL_INPUT2		通用输入信号 2 同步
GetInfo	hDev	设备句柄		
	uType	IKP_CXP_TRIG_LEVEL (0x30000036)		
	npValue	获取光口采集卡传输给相机的触发包电平极性		
说明	本参数表示光口采集卡传输给相机的触发包电平极性。			

9 附录

9.1 设备类型常量列表

定义值	十六进制值	含义
IKBoardUnknown	0xFFFFFFFF	未定义的设备
IKBoardALL	0x00000000	所有设备
IKBoardUSB30	0x00000001	USB3.0 接口设备
IKBoardPCIE	0x00000002	PCIe 接口设备

9.2 回调事件常量列表

定义值	十六进制值	含义
IKEvent_GrabStart	0x00000000	连续采集开始时触发该回调函数
IKEvent_FrameReady	0x00000001	在连续采集图像的过程中，图像缓冲区有一帧或者超过一帧的完整图像被采集完毕时触发该回调函数。
IKEvent_GrabStop	0x00000002	停止连续采集图像时触发该回调函数
IKEvent_FrameLost	0x00000003	图像缓冲区溢出时触发该回调函数
IKEvent_TimeOut	0x00000004	图像采集超时触发该回调函数
IKEvent_PixelClock	0x00000005	相机时钟信号有效时调用该回调函数
IKEvent_No_PixelClock	0x00000006	相机时钟信号无效时调用该回调函数
IKEvent_External_Trigger_Ignored	0x00000007	当采集卡工作在外触发模式，外部触发信号被采集卡丢失时触发该回调函数。该回调函数可能在外触发频率快于采集帧率的情况下发生。
IKEvent_CXP_Over_Current_Protection	0x00000008	当 CXP 采集卡发生过流保护时调用该函数。
IKEvent_CXP_CRC_Check_Error	0x00000009	当 CXP 采集卡发生 CRC 错误时调用该函数。
IKEvent_CXP_Transfer_Error	0x0000000a	当 CXP 采集卡发生传输错误时调用该函数。
IKEvent_Line_Timeout	0x0000000b	当采集卡发生行间超时时触发该回调函数
IKEvent_CXP_Incomplete_Frame	0x0000000c	当 CXP 采集卡收到未完成帧时触发该回调函数。
IKEvent_GrabLine	0x00100000+n	当第 n 行数据从相机传入采集卡中时触发该回调函数，此时图像缓冲区内尚未存在有效的图像数据。仅适用于线扫描相机（该功能已弃用）。
IKEvent_GrabLineEnd	0x00200000+n	当第 n 行数据从采集卡传入内存时触发该回调函数。仅适用于线扫描相机。

IKEvent_INPUT_FALLING_EDGE	0x00400000+n	当采集卡检测外部 IO 信号 n 的下降沿时触发该回调函数，n 的取值参考 IKapRegisterCallback() 函数说明。
IKEVENT_INPUT_RISING_EDGE	0x00800000+n	当采集卡检测外部 IO 信号 n 的上升沿时触发该回调函数，n 的取值参考 IKapRegisterCallback() 函数说明。
IKEvent_Start_Of_Frame	0x01000000	当采集卡开始一帧图像数据时触发该回调函数。 注意： 只有 CameraLink 采集卡在帧触发模式下，此功能才使能。
IKEvent_End_Of_Frame	0x02000000	当采集卡结束一帧图像数据时触发该回调函数（该功能已弃用）。
IKEvent_End_Of_Transfer	0x04000000+n	当第 n 个缓冲区数据传输完成时触发该回调函数，缓冲区计数从 0 开始。

注意：IKEvent_INPUT_FALLING_EDGE 和 IKEVENT_INPUT_RISING_EDGE 不能同时注册。

9.3 错误码类型常量列表

定义值	十六进制值	含义
IKStatus_Success	0x00000001	操作成功
IKStatus_BoardNotFound	0x00000002	设备未发现
IKStatus_AllocMemoryFail	0x00000003	申请内存失败
IKStatus_InvalidParameter	0x00000004	无效的参数
IKStatus_OpenBoardFail	0x00000005	打开设备失败
IKStatus_TimeOut	0x00000006	操作超时
IKStatus_WinError	0x00000007	系统错误
IKStatus_BoardNotOpen	0x00000008	设备未打开
IKStatus_ConfigFilePathInvalid	0x00000009	配置文件路径无效
IKStatus_ConfigParameterInvalid	0x0000000a	配置参数无效
IKStatus_ZLP	0x0000000b	USB3.0 零长度包
IKStatus_ThreadUnsetup	0x0000000c	线程无法建立
IKStatus_ThreadExist	0x0000000d	当前线程已存在
IKStatus_CameraUnsupport	0x0000000e	相机类型不支持
IKStatus_XMLFileLoadFail	0x0000000f	载入 XML 文件失败
IKStatus_NodeNotExist	0x00000010	XML 文件节点不存在
IKStatus_WriteSerialFail	0x00000011	写串口失败
IKStatus_CommandNotExist	0x00000012	串口指令不存在
IKStatus_CommandCollision	0x00000013	串口指令冲突
IKStatus_CommandMissRequirement	0x00000014	串口指令不符合要求
IKStatus_CommandNotAllowed	0x00000015	当前不允许串口指令
IKStatus_CommandSyntaxError	0x00000016	串口指令格式错误

IKStatus_NodeTypeMismatch	0x00000017	XML 节点不匹配
IKStatus_FeatureUnSupport	0x00000018	不支持该相机特征
IKStatus_CommandResultNoExit	0x00000019	相机执行结果无意义
IKStatus_CLRegPathNotFound	0x0000001a	CamaraLinkPath 不存在
IKStatus_CLDLLNotFound	0x0000001b	CLallserial.dll 不存在
IKStatus_CameraNotFound	0x0000001c	相机未发现
IKStatus_BufferTooSmall	0x0000001d	传入缓冲区域太小
IKStatus_BaudrateNotSupport	0x0000001e	不支持该波特率
IKStatus_CameraInUse	0x0000001f	相机正在使用中
IKStatus_FPGA_EraseFlashFail	0x00000020	擦除 FPGA 失败
IKStatus_FPGA_CheckFail	0x00000021	FPGA 下载过程校验失败
IKStatus_BoardNotBindingCOM	0x00000022	当前设备未绑定串口
IKStatus_ReadRegFail	0x00000023	读取相机状态寄存器失败
IKStatus_Invalid_Mutex	0x00000024	无效的设备锁句柄
IKStatus_Mutex_Locked	0x00000025	设备已经被占用
IKStatus_Invalid_Handle	0x00000026	无效的设备句柄
IKStatus_Set_Info_Error	0x00000027	设置参数值无效或者错误
IKStatus_Grab_Pending	0x00000028	当前有正在执行的图像采集过程
IKStatus_Insufficient_Resource	0x00000029	系统没有足够的资源
IKStatus_Grab_Abort	0x0000002a	放弃当前采集
IKStatus_Need_Reboot	0x0000002b	系统出现致命的错误， 必须重启计算机才能恢复正常
IKStatus_Need_Restart	0x0000002c	采集卡出现错误，需要重新打开采集卡设备才能恢复正常
IKStatus_Not_Implement	0x0000002d	功能未实现