



# ASICamera2 软件开发包

版本:2, 9  
2023. 1. 11

本手册版权供振旺光电公司所有，公司有权利随时变更本手册的内容而无须声明。



## 目录

1 介绍 .....	4
2 枚举类型和结构体的定义 .....	4
2.1 typedef enum ASI_BAYER_PATTERN .....	4
2.2 typedef enum ASI_IMG_TYPE .....	5
2.3 typedef enum ASI_GUIDE_DIRECTION .....	5
2.4 typedef enum ASI_FLIP_STATUS .....	5
2.5 typedef enum ASI_CAMERA_MODE .....	5
2.6 typedef enum ASI_ERROR_CODE .....	5
2.7 typedef enum ASI_BOOL .....	6
2.8 typedef struct _ASI_CAMERA_INFO .....	6
2.9 typedef enum ASI_CONTROL_TYPE .....	6
2.10 typedef struct _ASI_CONTROL_CAPS .....	7
2.11 typedef enum ASI_EXPOSURE_STATUS .....	7
2.12 typedef struct _ASI_ID .....	7
2.13 typedef struct _ASI_SUPPORTED_MODE .....	8
3 函数定义 .....	8
3.1 ASIGetNumOfConnectedCameras .....	8
3.2 ASIGetCameraProperty .....	8
3.3 ASIOpenCamera .....	8
3.4 ASIInitCamera .....	8
3.5 ASICloseCamera .....	8
3.6 ASIGetNumOfControls .....	9
3.7 ASIGetControlCaps .....	9
3.8 ASIGetControlValue .....	9
3.9 ASISetControlValue .....	9
3.10 ASISetROIFormat .....	9
3.11 ASIGetROIFormat .....	10
3.12 ASISetStartPos .....	10
3.13 ASIGetStartPos .....	10
3.14 ASIGetDroppedFrames .....	10
3.15 ASIEnableDarkSubtract .....	10
3.16 ASIDisableDarkSubtract .....	11
3.17 ASIStartVideoCapture .....	11
3.18 ASIStopVideoCapture .....	11
3.19 ASIGetVideoData .....	11
3.20 ASIPulseGuideOn .....	11
3.21 ASIPulseGuideOff .....	11
3.22 ASIStartExposure .....	11
3.23 ASIStopExposure .....	11
3.24 ASIGetExpStatus .....	11
3.25 ASIGetDataAfterExp .....	12
3.26 ASIGetID .....	12
3.27 ASISetID .....	12
3.28 ASIGetProductIDs .....	12
3.29 ASICameraCheck .....	12
3.30 ASIGetSDKVersion .....	12
3.31 ASIGetCameraSupportMode .....	12
3.32 ASIGetCameraMode .....	12
3.33 ASISetCameraMode .....	13
3.34 ASISendSoftTrigger .....	13
4 建议的调用顺序 .....	13



---

4.1 初始化.....	13
4.2 读取和设置控件的值 .....	13
4.3 相机模式.....	13
4.4 图像捕捉.....	13
4.5 关闭相机.....	14



## 修改记录

时间	版本	备注
2023.1.11	2.9	修复文档中的小错误 优化了部分函数的说明 增加了新 API 的说明
2018.8.8	2.8	增加 API 外触发相关
2018.5.3	2.7	增加 API ASIGetSDKVersion 增加 BitDepth 到 ASI_CAMERA_INFO
2017.9.1	2.6	删除 ASI_AUTO_MAX_EXP_MS, ASI_AUTO_MAX_EXP 单位改成毫秒
2017.6.26	2.4	修改 ASIGetVideoData: iWaitms
2017.5.2	2.3	修改 ASIGetCameraProperty
2017.4.12	2.2	修改内容
2017.2.24	2.1	增加 ASI_CONTROL_TYPE: ASI_AUTO_MAX_EXP_MS
2016.12.9	2.0	增加 ASI_CONTROL_TYPE: ASI_ANTI_DEW_HEATER 增加 ASIGetProductIDs
2016.9.19	1.3	增加 ASI_CONTROL_TYPE: ASI_PATTERN_ADJUS 等 增加 ASIInitCamera

## 1 介绍

这个软件开发包（SDK）描述了一组可以用来操作 ASI 系列相机的函数，通过 C、C++、C #等开发工具调用，适用于 x86 或 x64 的 Windows，Linux，和 OSX 操作系统，以及 ARM 平台的 Linux。

头文件：ASICamera2.H

windows 下的导入库和动态库：ASICamera2.lib，ASICamera2.dll

在 Linux 的动态库和静态库：ASICamera2.so，ASICamera2.a

OSX 下的动态库和静态库：ASICamera2.dylib，ASICamera2.a

安装方法：

在 Windows 中，下载后解压 zip 文件到任何目录，并添加 DLL 的路径的系统环境变量，有时需要注销并重新登录。还可以将 DLL 置于包含应用程序可执行文件的文件夹中。

## 2 枚举类型和结构体的定义

2.1 typedef enum ASI\_BAYER\_PATTERN

```
{  
    ASI_BAYER_RG=0,  
    ASI_BAYER_BG,  
    ASI_BAYER_GR,
```



```
    ASI_BAYER_GB  
}ASI_BAYER_PATTERN;  
    Bayer滤镜类型
```

## 2.2 typedef enum ASI\_IMG\_TYPE

```
{  
    ASI_IMG_RAW8 = 0, // 每个像素8位  
    ASI_IMG_RGB24, // 每个像素包含RGB, 共3个字节 (仅彩色相机)  
    ASI_IMG_RAW16, // 每个像素2个字节  
    ASI_IMG_Y8, // 黑白模式, 每个像素一个字节 (仅彩色相机)  
    ASI_IMG_END = -1  
}ASI_IMG_TYPE;  
    图像格式
```

## 2.3 typedef enum ASI\_GUIDE\_DIRECTION

```
{  
    ASI_GUIDE_NORTH=0,  
    ASI_GUIDE_SOUTH,  
    ASI_GUIDE_EAST,  
    ASI_GUIDE_WEST  
}ASI_GUIDE_DIRECTION;  
    导星方向
```

## 2.4 typedef enum ASI\_FLIP\_STATUS

```
{  
    ASI_FLIP_NONE = 0, // 不翻转  
    ASI_FLIP_HORIZ, // 水平翻转  
    ASI_FLIP_VERT, // 竖直翻转  
    ASI_FLIP_BOTH, // 水平+竖直翻转
```

```
}ASI_FLIP_STATUS;
```

图像翻转

## 2.5 typedef enum ASI\_CAMERA\_MODE

```
{  
    ASI_MODE_NORMAL = 0,  
    ASI_MODE_TRIG_SOFT_EDGE,  
    ASI_MODE_TRIG_RISE_EDGE,  
    ASI_MODE_TRIG_FALL_EDGE,  
    ASI_MODE_TRIG_SOFT_LEVEL,  
    ASI_MODE_TRIG_HIGH_LEVEL,  
    ASI_MODE_TRIG_LOW_LEVEL,  
    ASI_MODE_END = -1
```

```
}ASI_CAMERA_MODE;
```

相机模式

## 2.6 typedef enum ASI\_ERROR\_CODE

```
{  
    ASI_SUCCESS = 0, // 操作成功  
    ASI_ERROR_INVALID_INDEX, //非法的序号  
    ASI_ERROR_INVALID_ID, //非法的ID  
    ASI_ERROR_INVALID_CONTROL_TYPE, //非法的ControlType
```



```
ASI_ERROR_CAMERA_CLOSED, //相机没有打开
ASI_ERROR_CAMERA_REMOVED, //相机已经移除
ASI_ERROR_INVALID_PATH, //没找到文件
ASI_ERROR_INVALID_FILEFORMAT,
ASI_ERROR_INVALID_SIZE, //错误的格式
ASI_ERROR_INVALID_IMGTYPE, //不支持的格式
ASI_ERROR_OUTOF_BOUNDARY, //尺寸错误
ASI_ERROR_TIMEOUT, //超时
ASI_ERROR_INVALID_SEQUENCE, //调用顺序错误
ASI_ERROR_BUFFER_TOO_SMALL, //缓冲不够大
ASI_ERROR_VIDEO_MODE_ACTIVE,
ASI_ERROR_EXPOSURE_IN_PROGRESS,
ASI_ERROR_GENERAL_ERROR, //其他错误
ASI_ERROR_END
}ASI_ERROR_CODE;
    错误代码
```

## 2.7 typedef enum ASI\_BOOL

```
{
    ASI_FALSE = 0,
    ASI_TRUE
}ASI_BOOL;
    真假
```

## 2.8 typedef struct \_ASI\_CAMERA\_INFO

```
{
    char Name[64]; //相机名称
    int CameraID; //相机ID，用来区分不同的相机
    long MaxHeight; //最大高度
    long MaxWidth; // 最大宽度
    ASI_BOOL IsColorCam; //是否是彩色相机
    ASI_BAYER_PATTERN BayerPattern; //Bayer滤镜类型
    int SupportedBins[16]; //支持的bin数组，以0结束
    ASI_IMG_TYPE SupportedVideoFormat[8]; //支持的图像格式数组，以ASI_IMG_END结束
    double PixelSize; //像素尺寸(um)
    ASI_BOOL MechanicalShutter; //是否支持机械快门
    ASI_BOOL ST4Port; //是否有ST4
    ASI_BOOL IsCoolerCam; //是否冷冻相机
    ASI_BOOL IsUSB3Host; //是否工作为USB3.0
    ASI_BOOL IsUSB3Camera; //是否是USB3相机
    float ElecPerADU; //系统增益
    int BitDepth; //sensor的ADC实际深度
    char Unused[20];
} ASI_CAMERA_INFO;
    相机信息
```

## 2.9 typedef enum ASI\_CONTROL\_TYPE

```
{
    ASI_GAIN = 0, //增益
```



```
ASI_EXPOSURE, //曝光时间(微秒)
ASI_GAMMA, //gamma (范围1到100, 默认50)
ASI_WB_R, //白平衡的红色分量
ASI_WB_B, //白平衡的蓝色分量
ASI_BRIGHTNESS, //偏移
ASI_BANDWIDTHOVERLOAD, //占总的带宽的百分比
ASI_OVERCLOCK, //超频
ASI_TEMPERATURE, //温度 (x10)
ASI_FLIP, //image flip
ASI_AUTO_MAX_GAIN, //自动调节时的最大增益
ASI_AUTO_MAX_EXP, //自动调节时的, 单位是微秒
ASI_AUTO_MAX_BRIGHTNESS, //自动调节时的目标亮度
ASI_HARDWARE_BIN, //硬件合并
ASI_HIGH_SPEED_MODE, //高速模式
ASI_COOLER_POWER_PERC, //制冷功率(仅冷冻相机)
ASI_TARGET_TEMP, //sensor's target temperature(仅冷冻相机), 不需除以10
ASI_COOLER_ON, //打开制冷 (仅冷冻相机)
ASI_MONO_BIN, //
ASI_PATTERN_ADJUST, //只有1600 黑白相机支持
ASI_ANTI_DEW_HEATER, //保护玻璃加热
} ASI_CONTROL_TYPE;
控制类型
```

## 2.10 typedef struct \_ASI\_CONTROL\_CAPS

```
{
    char Name[64]; //控制类型名称, 比如"Gain" "Exposure"...
    char 描述[128]; //描述
    long MaxValue; //最大值
    long MinValue; //最小值
    long DefaultValue; //默认值
    ASI_BOOL IsAutoSupported; //是否支持自动调节
    ASI_BOOL IsWritable; //能否写入
    ASI_CONTROL_TYPE ControlType; //控制类型ID
    char Unused[32];
} ASI_CONTROL_CAPS;
控制类型的内容
```

注意: ASI\_TEMPERATURE的最大和最小值需要除以10

## 2.11 typedef enum ASI\_EXPOSURE\_STATUS

```
{
    ASI_EXP_IDLE = 0, //就绪, 可以开始曝光
    ASI_EXP_WORKING, //正在曝光
    ASI_EXP_SUCCESS, //曝光成功可以读取数据
    ASI_EXP_FAILED, //曝光失败
} ASI_EXPOSURE_STATUS;
单张曝光时候的状态
```

## 2.12 typedef struct \_ASI\_ID

```
{
```



```
    unsigned char id[8];
}ASI_ID;
    可以写入到相机存储器的8位ID
2.13 typedef struct _ASI_SUPPORTED_MODE
{
    ASI_CAMERA_MODE SupportedCameraMode[16];// 这个数组将用来存储从SDK中返回的
    当前相机支持的模式
}ASI_SUPPORTED_MODE;
    返回相机支持的模式
```

## 3 函数定义

### 3.1 ASIGetNumOfConnectedCameras

语法: int ASIGetNumOfConnectedCameras()

用处: 得到连接的ASI相机个数

### 3.2 ASIGetCameraProperty

语法: ASI\_ERROR\_CODE ASIGetCameraProperty(ASI\_CAMERA\_INFO \*pASICameraInfo, int iCameraIndex)

用处: 得到指定序号（以0开始）的相机信息

描述:

ASI\_CAMERA\_INFO \*pASICameraInfo: 指向相机信息结构体的指针

int iCameraIndex: 相机序号

示例:

```
int iNumofConnectCameras = ASIGetNumOfConnectedCameras();
ASI_CAMERA_INFO **ppASICameraInfo = (ASI_CAMERA_INFO
**)malloc(sizeof(ASI_CAMERA_INFO *)*iNumofConnectCameras);
for(int i = 0; i < iNumofConnectCameras; i++)
{
    ppASICameraInfo[i] = (ASI_CAMERA_INFO *)malloc(sizeof(ASI_CAMERA_INFO ));
    ASIGetCameraProperty(ppASICameraInfo[i], i);
}
```

注意:

可以在ASIOpenCamera之前调用

### 3.3 ASIOpenCamera

语法: ASI\_ERROR\_CODE ASIOpenCamera(int iCameraID)

用处: 打开指定ID的相机。 这不会影响正在捕捉的相机，这是操作相机的第一步

### 3.4 ASIInitCamera

语法: ASI\_ERROR\_CODE ASIInitCamera (int iCameraID)

用处: 初始化指定 ID 的相机，此 API 只影响您要初始化的相机，不会影响其他相机。这应该是第二个调用函数。

### 3.5 ASICloseCamera

语法: ASI\_ERROR\_CODE ASICloseCamera(int iCameraID)

用处: 关闭指定 ID 的相机使其资源被释放。这应该是最后一个调用的函数。





### 3.6 ASIGetNumOfControls

语法: `ASI_ERROR_CODE ASIGetNumOfControls(int iCameraID, int * piNumberOfControls)`

用处: 得到指定ID相机的控制类型的数量

### 3.7 ASIGetControlCaps

语法: `ASI_ERROR_CODE ASIGetControlCaps(int iCameraID, int iControlIndex, ASI_CONTROL_CAPS * pControlCaps)`

用处: 得到特定序号的控制类型的内容

描述:

int iCameraID: 相机ID

int iControlIndex: control index

ASI\_CONTROL\_CAPS \* pControlCaps: pointer to control capacity

注意: iControlIndex 是控件序号, 不是ControlType

### 3.8 ASIGetControlValue

语法: `ASI_ERROR_CODE ASIGetControlValue (int iCameraID, ASI_CONTROL_TYPE ControlType, long *pIValue, ASI_BOOL *pbAuto)`

用处: 得到指定ID相机的某控制类型的值

描述:

int iCameraID: 相机ID

ASI\_CONTROL\_TYPE ControlType: 控制类型

long \*pIValue: 指向当前值的指针

ASI\_BOOL \*pbAuto: 指针, 表示是否自动调节

### 3.9 ASISetControlValue

语法: `ASI_ERROR_CODE ASISetControlValue(int iCameraID, ASI_CONTROL_TYPE ControlType, long IValue, ASI_BOOL bAuto)`

用处: 设置指定ID相机的控制类型的值

描述:

int iCameraID: 相机ID

ASI\_CONTROL\_TYPE ControlType: 控制类型

long IValue: 要设置的值

ASI\_BOOL bAuto: 是否要自动调节

注意: (1) 当设置为自动调节(bAuto=ASI\_TRUE), IValue应该设置为当前值

(2) 自动曝光和自动增益仅在Video模式下有效(即, 当您通过调用ASIGetVideoData来获取图像时), 在Snap模式下是无效的(即, 当您通过ASIGetDataAfterExp来获取图像时)

### 3.10 ASISetROIFormat

语法: `ASI_ERROR_CODE ASISetROIFormat(int iCameraID, int iWidth, int iHeight, int iBin, ASI_IMG_TYPE Img_type)`

用处: 设置感兴趣的区域 (ROI) 尺寸, 像素合并, 以及图像格式

描述:

int iCameraID: 相机ID

int iWidth: 图像宽度

int iHeight: 图像高度

int iBin: NxN 像素合并值



ASI\_IMG\_TYPE Img\_type: 图像格式

注意: 要确保 iWidth%8=0, iHeight%2=0。对于 USB2.0 相机 ASI120, 确保 iWidth\*iHeight%1024=0, 否则不能设置成功。

### 3.11 ASIGetROIFormat

语法: ASI\_ERROR\_CODE ASIGetROIFormat(int iCameraID, int \*piWidth, int \*piHeight, int \*piBin, ASI\_IMG\_TYPE \*pImg\_type)

用处: 得到感兴趣的区域 (ROI) 尺寸, 像素合并, 以及图像格式

描述:

int iCameraID: 相机ID

int \*piWidth: 图像宽度

int \*piHeight: 图像高度

int \*piBin: 像素合并值

ASI\_IMG\_TYPE \*pImg\_type: 图像格式

### 3.12 ASISetStartPos

语法: ASI\_ERROR\_CODE ASISetStartPos(int iCameraID, int iStartX, int iStartY)

用处: 设置ROI的起始位置

描述:

int iCameraID: 相机ID

int iStartX: x轴的起始位置

int iStartY: y轴的起始位置

注意: 起始位置是相对于像素合并后的图像。ASISetROIFormat 后 ROI 会变成中央区域, 调用此函数使之回到原来的位置。

### 3.13 ASIGetStartPos

语法: ASI\_ERROR\_CODE ASIGetStartPos(int iCameraID, int \*piStartX, int \*piStartY)

用处: 得到ROI的起始位置

描述:

int iCameraID: 相机ID

int \*piStartX: x轴的起始位置

int \*piStartY: y轴的起始位置

注意: 起始位置是相对于像素合并后的图像

### 3.14 ASIGetDroppedFrames

语法: ASI\_ERROR\_CODE ASIGetDroppedFrames(int iCameraID, int \*piDropFrames)

用处: 得到视频模式时的丢帧数量, 每次开始视频清零

### 3.15 ASIEnableDarkSubtract

语法: ASI\_ERROR\_CODE ASIEnableDarkSubtract(int iCameraID, char \*pcBMPPath)

用处: 减暗场

描述:

int iCameraID: 相机ID

char \* pcBMPPath: 暗场图片的路径(.bmp)

注意: 暗场图像是由相机的 direct show 驱动得到, 位于捕获应用程序的菜单“video capture filter”->“ROI 和其他”页



### 3.16 ASIDisableDarkSubtract

语法: `ASI_ERROR_CODE ASIDisableDarkSubtract(int iCameraID)`

用处: 取消减暗场

### 3.17 ASIStartVideoCapture

语法: `ASI_ERROR_CODE ASIStartVideoCapture(int iCameraID)`

用处: 开始连续曝光

### 3.18 ASIStopVideoCapture

语法: `ASI_ERROR_CODE ASIStopVideoCapture(int iCameraID)`

用处: 停止连续曝光

### 3.19 ASIGetVideoData

语法: `ASI_ERROR_CODE ASIGetVideoData(int iCameraID, unsigned char* pBuffer, long lBuffSize, int iWaitms)`

用处: 调用 `ASIStartVideoCapture` 后, 开始连续图像捕捉, 如果调用两次, 并不能获得两次相同的帧。

描述:

`unsigned char* pBuffer`: 指向图像缓冲区的指针

`long lBuffSize`: 缓冲区的大小

`int iWaitms`: 等待时间, 单位是毫秒, -1 是无限等待

注意: 如果读取速度不够快的话, 新得到的帧会被丢弃, 最好建立一个环形缓冲区。

`bufSize` 的字节数: RAW8 和 Y8, `bufSize >= image_width*image_height`, RAW16, `bufSize >= image_width*image_height *2`, RGB24, `bufSiz >= image_width*image_height *3`

建议的 `iWaitms` 值: `exposure_time*2 + 500` 毫秒

### 3.20 ASIPulseGuideOn

语法: `ASI_ERROR_CODE ASIPulseGuideOn(int iCameraID, ASI_GUIDE_DIRECTION direction)`

用处: 发送 ST4 导星信号开始导星, 仅带有 ST4 口的相机支持

注意: 之后必须调用 `ASIPulseGuideOff` 来停止导星

### 3.21 ASIPulseGuideOff

语法: `ASI_ERROR_CODE ASIPulseGuideOff(int iCameraID, ASI_GUIDE_DIRECTION direction)`

用处: 发送 ST4 导星信号停止导星, 仅带有 ST4 口的相机支持

### 3.22 ASIStartExposure

语法: `ASI_ERROR_CODE ASIStartExposure(int iCameraID)`

用处: 开始单张曝光

### 3.23 ASIStopExposure

语法: `ASI_ERROR_CODE ASIStopExposure(int iCameraID)`

用处: 中止曝光

注意: 如果中止曝光后的曝光状态是 `ASI_EXP_SUCCESS`, 你仍可以读取图像

### 3.24 ASIGetExpStatus

语法: `ASI_ERROR_CODE ASIGetExpStatus(int iCameraID, ASI_EXPOSURE_STATUS *pExpStatus)`

用处: 得到单张曝光的状态

注意: 开始单张曝光后, 需要连续地读取曝光状态



### 3.25 ASIGetDataAfterExp

语法: `ASI_ERROR_CODE ASIGetDataAfterExp(int iCameraID, unsigned char* pBuffer, long lBuffSize)`

用处: 单张曝光成功后读取图像

描述:

int iCameraID: 相机 ID

unsigned char\* pBuffer: 指向图像缓冲的指针

long lBuffSize: 缓冲的大小

注意: lBuffSize 参考 ASIGetVideoData ()

### 3.26 ASIGetID

语法: `ASI_ERROR_CODE ASIGetID(int iCameraID, ASI_ID* pID)`

用处: 得到储存在相机flash里的ID, 仅USB3.0相机支持

### 3.27 ASISetID

语法: `ASI_ERROR_CODE ASISetID(int iCameraID, ASI_ID ID)`

用处: 写入ID到相机flash里, 仅USB3.0相机支持

### 3.28 ASIGetProductIDs

语法: `int ASIGetProductIDs(int* pPIDs)`

用处: 得到所有支持的相机PID, 先使pPIDs为0, 得到长度, 然后分配内存再次读取PID

注意: 本函数在未来版本中可能会取消, 建议使用ASICameraCheck代替

描述:

int\* pPIDs: 指向 PIDs数组的指针

返回值: 数组长度。

### 3.29 ASICameraCheck

语法: `ASI_BOOL ASICameraCheck(int iVID, int iPID)`

用处: 检查一个设备是否是ASI相机

描述:

int iVID: 设备的VID

int iPID: 设备的PID

返回值: 如果设备是ASI相机, 返回ASI\_TRUE, 否则返回ASI\_FALSE

### 3.30 ASIGetSDKVersion

语法: `ASICAMERA_API char* ASIGetSDKVersion()`

用处: 得到SDK的版本字符串

### 3.31 ASIGetCameraSupportMode

语法: `ASI_ERROR_CODE ASIGetCameraSupportMode(int iCameraID, ASI_SUPPORTED_MODE* pSupportedMode)`

用处: 取得当前相机支持的所有相机模式

### 3.32 ASIGetCameraMode

语法: `ASI_ERROR_CODE ASIGetCameraMode(int iCameraID, ASI_CAMERA_MODE* mode)`

用处: 得到当前相机所处的相机模式



### 3.33 ASISetCameraMode

语法: `ASI_ERROR_CODE ASISetCameraMode(int iCameraID, ASI_CAMERA_MODE mode)`

用处: 设置一个模式到当前相机

### 3.34 ASISendSoftTrigger

语法: `ASI_ERROR_CODE ASISendSoftTrigger(int iCameraID, ASI_BOOL bStart)`

用处: 发出一个软件模拟的外部触发信号。当参数bStart为ASI\_TRUE的时候, 开始曝光。对于边沿触发模式, 不需要再发送bStart为ASI\_FALSE的结束信号, 系统会自动复位为假。对于电平触发, 必须要通过发送bStart为ASI\_FALSE来作为曝光的结束信号。

## 4 建议的调用顺序

### 4.1 初始化

得到连接的相机的数量--> `ASIGetNumOfConnectedCameras`

得到相机的信息, 包括ID、名称、分辨率等, ID是不会改变的--> `ASIGetCameraProperty`

打开相机 --> `ASIOpenCamera` (注意: 此SDK可以操作多个相机, 通过CameraID区分)

初始化--> `ASIInitCamera`

得到控制类型的数量--> `ASIGetNumOfControls`

得到每个控制类型的信息--> `ASIGetControlCaps`

设置图像尺寸和格式--> `ASISetROIFormat`

设置ROI的起始位置--> `ASISetStartPos`

### 4.2 读取和设置控件的值

`ASIGetControlValue`

`ASISetControlValue` //曝光时也可以操作, 但在触发模式下, 不能设置曝光。

### 4.3 相机模式

首先, 使用位于 `ASI_CAMERA_INFO` 结构体中的IsTriggerCam变量来判断相机是否支持多种模式。如果不支持, 那么就没有必要调用这些模式相关的函数。

取得相机支持的模式--> `ASIGetCameraSupportMode`

设置模式 --> `ASISetCameraMode`

取得当前相机模式--> `ASIGetCameraMode`

### 4.4 图像捕捉

有两种曝光模式: 视频模式和单张模式。视频模式时图像是连续采集的, 单张模式时每次只捕捉一张图片

如果相机工作在触发模式, 只能使用视频模式来采集图像。

#### ● 视频模式

开始视频捕捉--> `ASIStartVideoCapture`

读取图像--> `ASIGetVideoData`

停止捕捉--> `ASIStopVideoCapture`

建议在单独的线程里对图像操作:

```
while(1)
{
    if(ASIGetVideoData == ASI_SUCCESS)
    {
```

```
    ...  
  }  
}
```

- 单张曝光

ASISStartExposure

```
while(1)  
{  
    ASIGetExpStatus(&status)  
    ...  
}
```

中止曝光: ASISStopExposure

```
if(status == ASI_EXP_SUCCESS)//如果曝光成功则读取图片  
    ASIGetDataAfterExp
```

#### 4.5 关闭相机

ASICloseCamera//释放资源