



QHYCCD LabVIEW 说明文档

V1.0

目录

1. 项目介绍	3
2. 函数介绍	4
2.1. InitQHYCCDSDKResource	4
2.2. ReleaseQHYCCDSDKResource	5
2.3. ScanQHYCCDCameraNumber	5
2.4. GetQHYCCDCameraID	6
2.5. GetQHYCCDCameraReadModeNumber	6
2.6. GetQHYCCDCameraReadModeName	7
2.7. SetQHYCCDCameraWorkMode	7
2.8. SetQHYCCDCameraFormat	8
2.9. SetQHYCCDCameraBinMode	9
2.10. GetQHYCCDCameraChipInfo	10
2.11. GetQHYCCDCameraEffectiveArea	11
2.12. GetQHYCCDCameraOverScanArea	12
2.13. GetQHYCCDCameraMemLength	13
2.14. SetQHYCCDCameraROI	13
2.15. SetQHYCCDCameraExpTime	14
2.16. SetQHYCCDCameraGain	15
2.17. SetQHYCCDCameraOffset	16
2.18. SetQHYCCDCameraTraffic	16
2.19. SetQHYCCDCameraGainR	17
2.20. SetQHYCCDCameraGainG	17
2.21. SetQHYCCDCameraGainB	18
2.22. SetQHYCCDCameraBrightness	19
2.23. SetQHYCCDCameraContrast	19
2.24. SetQHYCCDCameraGamma	20
2.25. GetQHYCCDCameraOneSingleFrame	21
2.26. CancelQHYCCDCameraSingleCapture	22
2.27. BeginQHYCCDCameraLiveCapture	23
2.28. GetQHYCCDCameraOneLiveFrame	23
2.29. StopQHYCCDCameraLiveCapture	25
2.30. GetQHYCCDCameraTriggerInterfaceNumber	25
2.31. GetQHYCCDCameraTriggerInterfaceName	26
2.32. SetQHYCCDCameraTriggerInterface	27
2.33. GetQHYCCDCameraTriggerModeNumber	27
2.34. GetQHYCCDCameraTriggerModeName	28
2.35. SetQHYCCDCameraTriggerMode	29
2.36. SetQHYCCDCameraTriggerOnOff	29
2.37. SetQHYCCDCameraTrigerInOnly	30
2.38. SetQHYCCDCameraTrigerOutOnly	31



2.39. EnableQHYCCDCameraTriggerOut	31
2.40. SetQHYCCDCameraBurstModeOnOff	32
2.41. SetQHYCCDCameraBurstModeStartEnd	33
2.42. SetQHYCCDCameraBurstModePatchNumber	34
2.43. SetQHYCCDCameraBurstModeCapture	34
2.44. SetQHYCCDCameraTargetTemperature	35
2.45. GetQHYCCDCameraCurTemperature	35

1. 项目介绍

此工程围绕 QHYCCD SDK (即 qhyccd.dll 库文件) 进行开发, 由于 QHYCCD SDK 直接使用 USB 设备句柄控制相机, 而在 LabVIEW 中使用句柄进行操作不像 C/C++ 中那样方便, 因此开发此工程, 旨在提供简单实用的函数接口, 以实现 LabVIEW 控制 QHYCCD 相机设备的需求。此外, 使用此工程可避免部分兼容性问题, 如无法识别 QHYCCD SDK 库文件或 QHYCCD SDK 函数等问题。

关于 QHYCCD SDK 函数具体细节可以参考 SDK API 文档, 下载位置:
https://www.qhyccd.cn/sdk_demo/。

SDK API手册

记录 SDK API 说明的手册, 包含常用函数的使用方式及一些简单的示例代码。

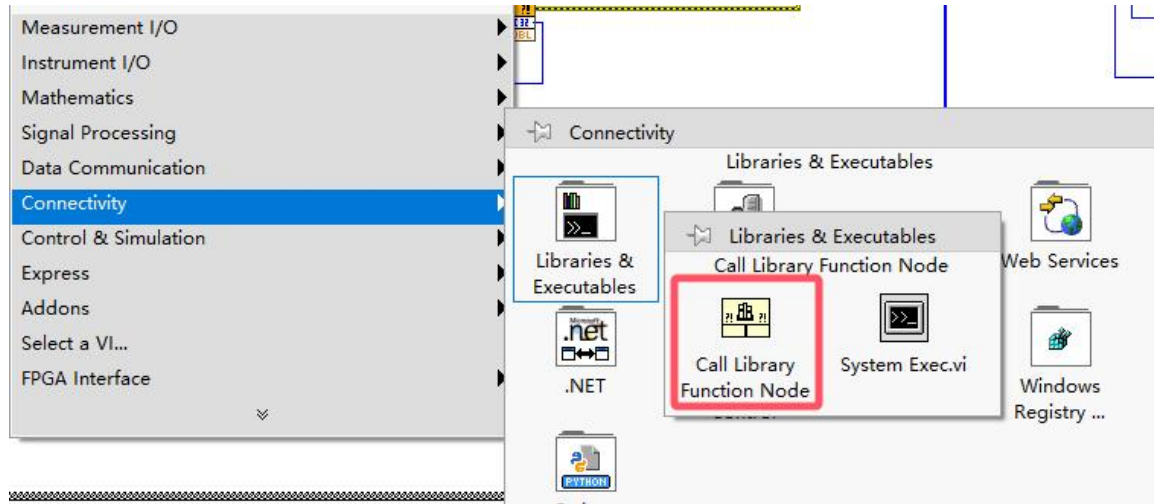
[中文版下载地址](#)

[英文版下载地址](#)

2. 函数介绍

此部分内容将介绍如何在 LabVIEW 中引入 C++库函数节点，以及各个函数的使用说明，另外需要注意的是，引入 C++库函数时，Library name or path 项不是固定路径，需根据实际情况自行修改，本手册内路径相关展示内容仅作参考。

引入 C++库函数节点的具体操作为，打开 LabVIEW 后面板，在空白处右键，依次选择 Connectivity→Libraries & Executables→Call Library Function Node。



2.1. InitQHYCCDSDKResource

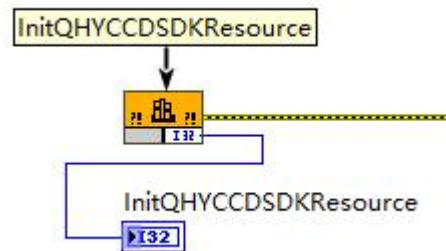
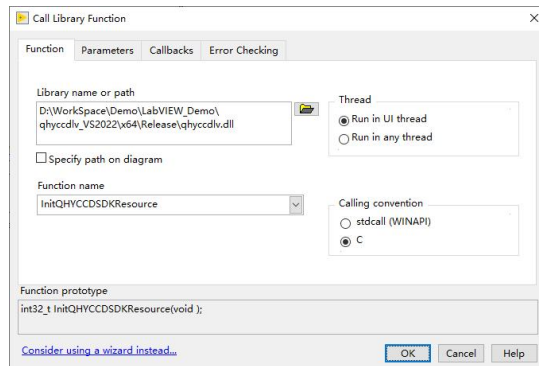
函数原型：

```
int32_t InitQHYCCDSDKResource();
```

函数说明：

初始化 QHYCCD SDK 资源，在程序开始时执行，每次运行程序仅需执行一次，执行成功时返回值为 0，失败时返回值为-1，调用的 QHYCCD SDK 函数为 InitQHYCCDResource。

LabVIEW 实例：



2.2. ReleaseQHYCCDSDKResource

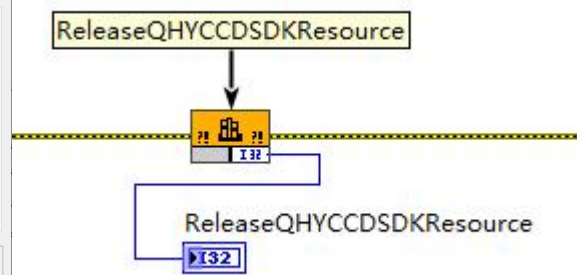
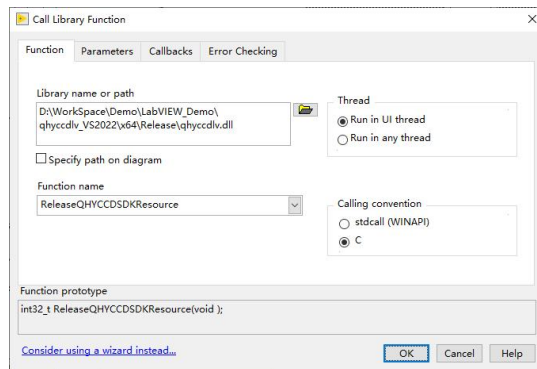
函数原型:

```
int32_t ReleaseQHYCCDSDKResource();
```

函数说明:

释放 QHYCCD SDK 资源，在程序结束时执行，每次运行程序仅需执行一次，执行成功时返回值为 0，失败时返回值为-1，调用的 QHYCCD SDK 函数为 CloseQHYCCD、ReleaseQHYCCDResource。

LabVIEW 实例:



2.3. ScanQHYCCDCameraNumber

函数原型:

```
int32_t ScanQHYCCDCameraNumber(
    int32_t* number
);
```

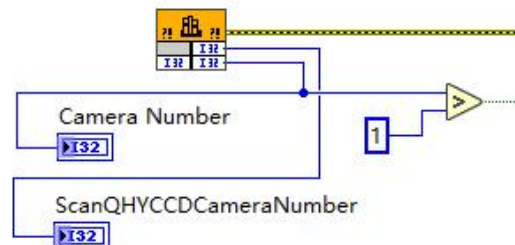
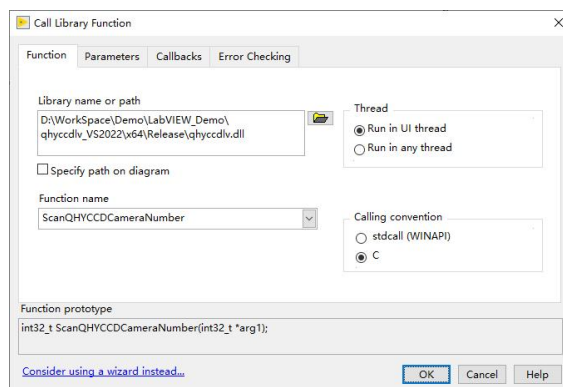
函数说明:

扫描连接到电脑上的相机数量，并获取所有相机的 ID 和设备句柄，可根据数量判断是否继续下一步操作，执行成功时返回值为 0，失败时返回值为-1，调用的 QHYCCD SDK 函数包括 ScanQHYCCD、GetQHYCCDId、OpenQHYCCD。

参数说明:

number: 返回相机数量。

LabVIEW 实例:



2.4. GetQHYCCDCameraID

函数原型:

```
int32_t GetQHYCCDCameraID(
    int32_t camIndex,
    char *camID);
```

函数说明:

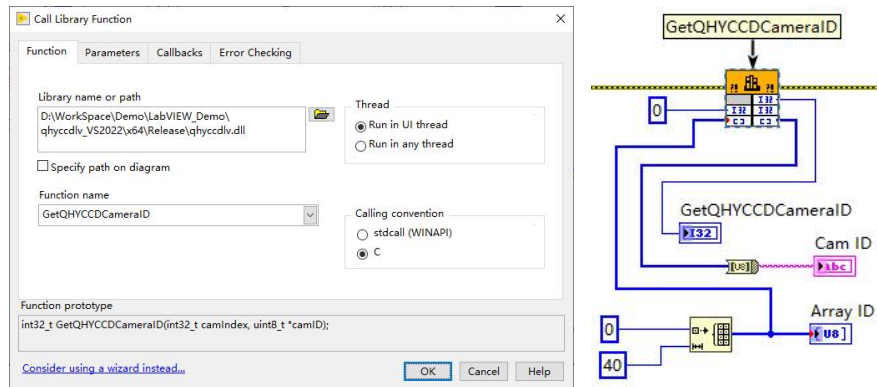
获取相机的 ID，执行成功时返回值为 0，失败时返回值为-1，相机 ID 为相机型号及序列号组成的一串字符，其中序列号为每个相机独有的固定标识，此函数未调用 QHYCCD SDK 的函数，只是将 ScanQHYCCDCameraNumber 函数运行时存储的相机 ID 数据读出来。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

camID: 返回相机的 ID 字符串。

LabVIEW 实例:



2.5. GetQHYCCDCameraReadModeNumber

函数原型:

```
int32_t GetQHYCCDCameraReadModeNumber(
    int32_t camIndex,
    int32_t *number
);
```

函数说明:

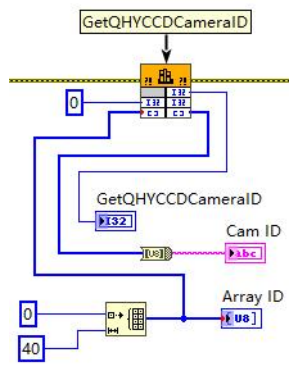
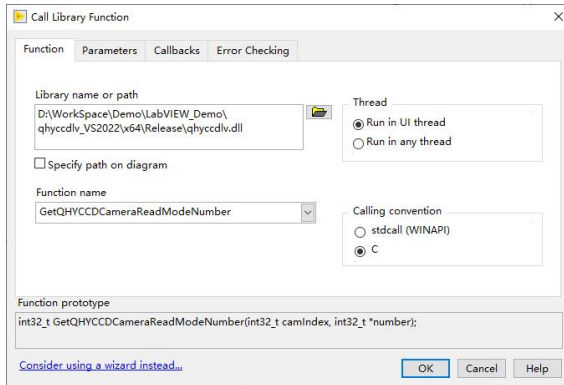
获取相机读出模式的数量，所有相机都至少有一个读出模式，即标准模式，执行成功时返回值为 0，失败时返回值为-1，调用的 QHYCCD SDK 函数为 GetQHYCCDNumberOfReadModes。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

number: 返回读出模式的数量。

LabVIEW 实例:



2.6. GetQHYCCDCameraReadModeName

函数原型:

```
int32_t GetQHYCCDCameraReadModeName(
    int32_t camIndex,
    int32_t modeIndex,
    char* name
);
```

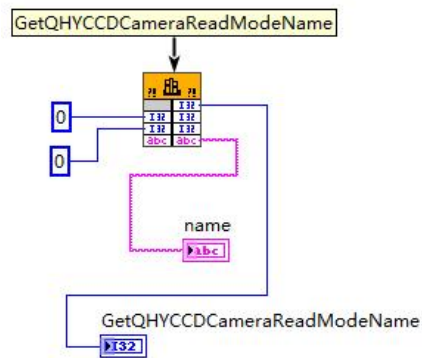
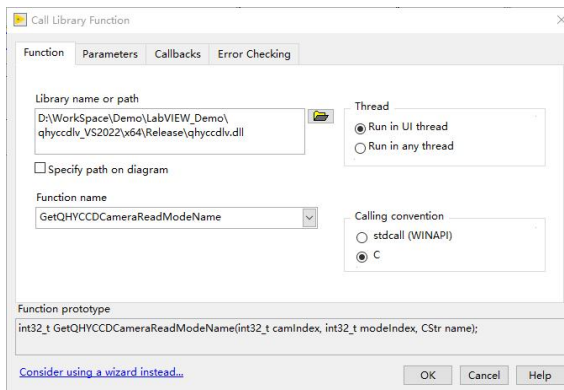
函数说明:

获取各个读出模式的名称，执行成功时返回值为 0，失败时返回值为-1，调用的 QHYCCD SDK 函数包括 GetQHYCCDReadModeName。

参数说明:

- camIndex: 目标相机的索引，从 0 开始计数。
- modeIndex: 读出模式的索引，从 0 开始计数。
- name: 返回读出模式的名称。

LabVIEW 实例:



2.7. SetQHYCCDCameraWorkMode

函数原型:


```
int32_t SetQHYCCDCameraWorkMode(
    int32_t camIndex,
    int32_t readmode,
    int32_t streammode
);
```

函数说明：

设置相机的工作模式，包括读出模式设置和数据流模式（单帧/连续模式）设置，设置工作模式后会初始化相机寄存器状态并清除之前的参数设置，如曝光时间、增益等，切换工作模式参数时需要重新设置曝光时间等参数，执行成功时返回值为 0，失败时返回值为-1，调用的 QHYCCD SDK 函数包括 SetQHYCCDReadMode、SetQHYCCDStreamMode、InitQHYCCD。

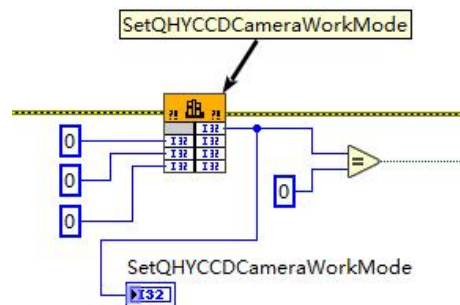
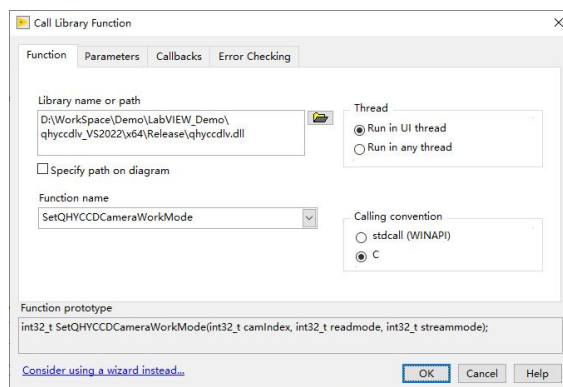
参数说明：

camIndex: 目标相机的索引，从 0 开始计数。

readmode: 读出模式参数。

streammode: 流模式参数，单帧模式设置为 0，连续模式设置为 1。

LabVIEW 实例：



2.8. SetQHYCCDCameraFormat

函数原型：

```
int32_t SetQHYCCDCameraFormat(
    int32_t camIndex,
    int32_t bits,
    int32_t color
);
```

函数说明：

设置相机输出的图像数据格式，可设置格式有黑白八位格式图像数据（RAW8/MONO8）、黑白 16 位格式图像数据（RAW16/MONO16）和彩色 8 位格式图像数据（RGB24），设置相机工作模式后可以执行此函数，切换数据格式时需要先调用工作模式设置函数，然后再设置图像数据格式及曝光时间等参数，执行成功时返回值为 0，失败时返回值为-1，调用的 QHYCCD SDK 函数为 SetQHYCCDParam。

参数说明：

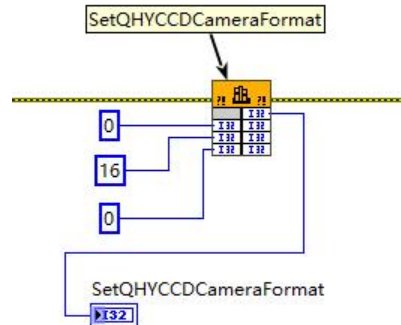
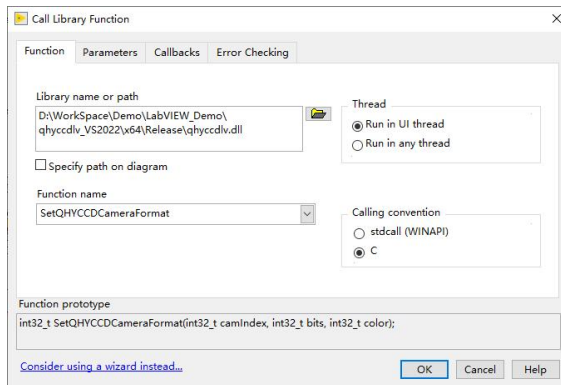
camIndex: 目标相机的索引, 从 0 开始计数。

bits: 图像位数, 设置参数为 8 或 16, 分别对应 8 位图像和 16 位图像。

color: 设置黑白或彩色图像, 0 为黑白图像, 1 为彩色图像, 仅彩色型号相机可以设置此参数, 黑白相机设置参数不生效。

bits 设置 8, color 设置 0 时为黑白 8 位图像, bits 设置 16, color 设置 0 时为黑白 16 位图像, bits 设置 8, color 设置 1 时为彩色 8 位图像。

LabVIEW 实例:



2.9. SetQHYCCDCameraBinMode

函数原型:

```
int32_t SetQHYCCDCameraBinMode(
    int32_t camIndex,
    int32_t bin
);
```

函数说明:

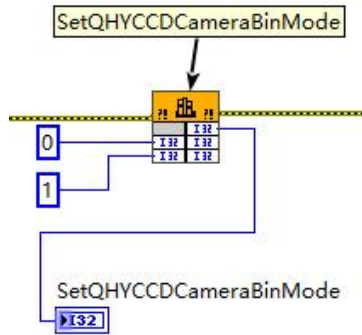
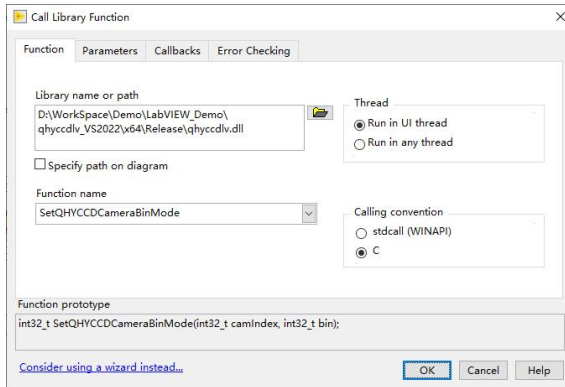
设置图像像素合并模式, 此功能为软件合并, 不会减少图像传输时间以及提升帧率, 设置相机工作模式后可以执行此函数, 切换合并模式时需要先调用工作模式设置函数, 然后再设置合并模式及曝光时间等参数, 执行成功时返回值为 0, 失败时返回值为-1, 调用的 QHYCCD SDK 函数包括 SetQHYCCDBinMode、SetQHYCCDResolution。

参数说明:

camIndex: 目标相机的索引, 从 0 开始计数。

bin: 像素合并的倍数, 1 为 1x1 BIN, 2 为 2x2 BIN, 3 为 3x3BIN, 4 为 4x4 BIN。

LabVIEW 实例:



2.10. GetQHYCCDCameraChipInfo

函数原型:

```
int32_t GetQHYCCDCameraChipInfo(
    int32_t camIndex,
    double* chipw,
    double* chiph,
    int32_t * imagew,
    int32_t * imageh,
    double* pixelw,
    double* pixelh,
    int32_t * bpp
);
```

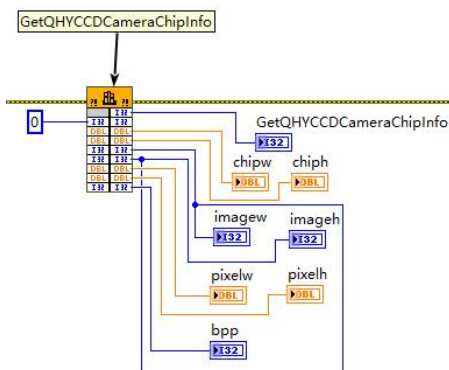
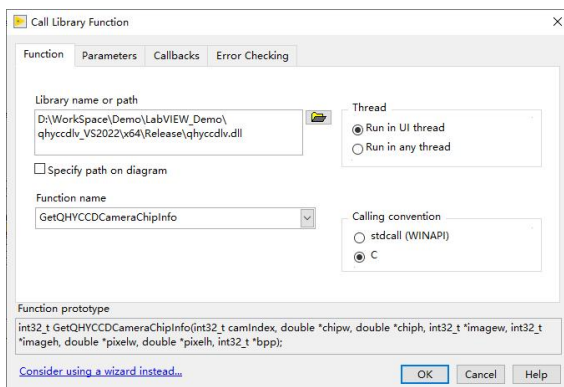
函数说明:

获取相机硬件相关参数信息，包括芯片物理尺寸，单位为毫米，图像像素尺寸，像素物理尺寸，单位为微米，不同工作模式下，获取的参数值可能会发生变化，执行成功时返回 0，失败时返回 -1，调用的 QHYCCD SDK 函数为 GetQHYCCDChipInfo。

参数说明:

- camIndex: 目标相机的索引，从 0 开始计数。
- chipw: 芯片的物理宽度，单位为毫米。
- chiph: 芯片的物理高度，单位为毫米。
- imagew: 图像的宽度，单位为像素个数。
- imageh: 图像高度，单位为像素个数。
- pixelw: 像素的物理宽度，单位为微米。
- pixelh: 像素的物理高度，单位为微米。
- bpp: 图像数据的位数，此参数返回值不固定，设置位数时会返回相应值。

LabVIEW 实例:



2.11. GetQHYCCDCameraEffectiveArea

函数原型:

```
int32_t GetQHYCCDCameraEffectiveArea(
    int32_t camIndex,
    int32_t * startx,
    int32_t * starty,
    int32_t * sizex,
    int32_t * sizey
);
```

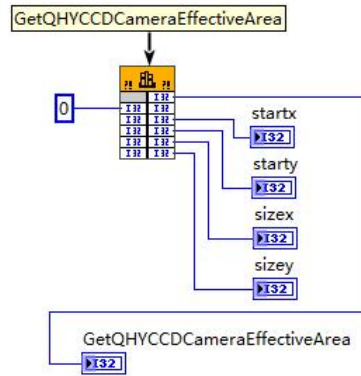
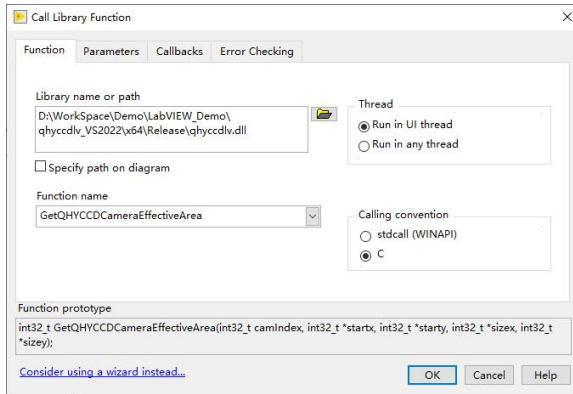
函数说明:

获取图像有效区域位置及尺寸信息，通常在设置合并模式之后执行此函数，函数执行成功时返回 0，失败时返回-1，不同 BIN 模式或工作模式下，得到的参数值会有变化，调用的 QHYCCD SDK 函数为 GetQHYCCDEffectiveArea。

参数说明:

- camIndex: 目标相机的索引，从 0 开始计数。
- startx: 有效区域起始位置的 x 坐标，以图像左上角为参考点。
- starty: 有效区域起始位置的 y 坐标，以图像左上角为参考点。
- sizex: 有效区域的宽度。
- sizey: 有效区域的高度。

LabVIEW 实例:



2.12. GetQHYCCDCameraOverScanArea

函数原型:

```
int32_t GetQHYCCDCameraOverScanArea(
    int32_t camIndex,
    int32_t * startx,
    int32_t * starty,
    int32_t * sizex,
    int32_t * sizey
);
```

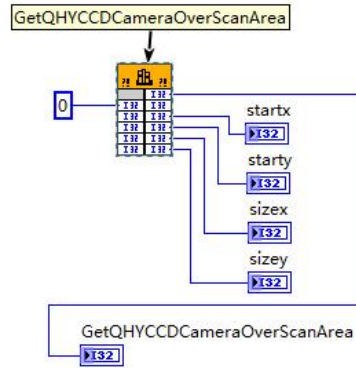
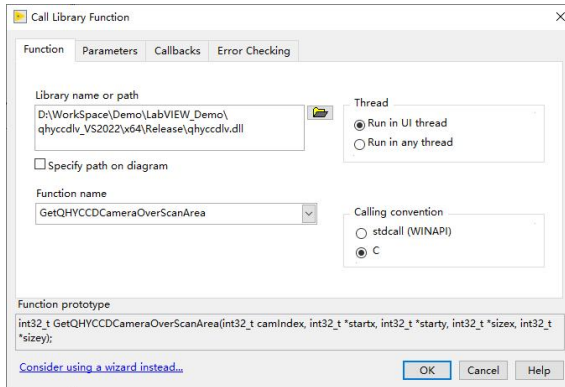
函数说明:

获取图像过扫区位置及尺寸信息，通常在设置合并模式之后执行此函数，函数执行成功时返回 0，失败时返回-1，不同 BIN 模式或工作模式下，得到的参数值会有变化，调用的 QHYCCD SDK 函数为 GetQHYCCDOverScanArea。

参数说明:

- camIndex: 目标相机的索引，从 0 开始计数。
- startx: 过扫区域起始位置的 x 坐标，以图像左上角为参考点。
- starty: 过扫区域起始位置的 y 坐标，以图像左上角为参考点。
- sizex: 过扫区域的宽度。
- sizey: 过扫区域的高度。

LabVIEW 实例:



2.13. GetQHYCCDCameraMemLength

函数原型:

```
int32_t GetQHYCCDCameraMemLength(
    int32_t camIndex,
    int32_t * length
);
```

函数说明:

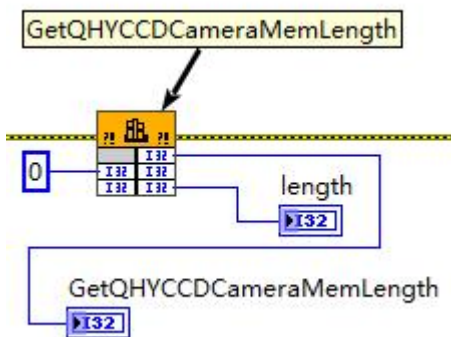
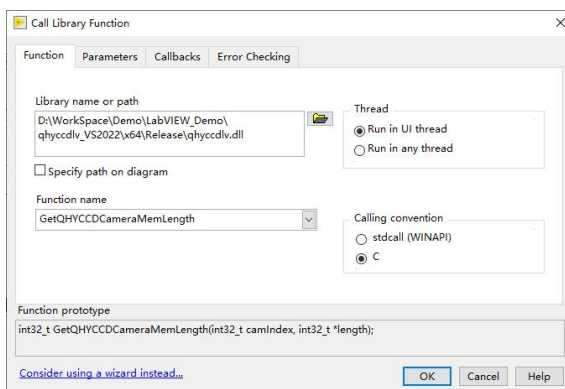
获取相机图像数据内存长度，函数获取的数据长度会比实际的多，计算公式为(imagw+100) * (imageh+100) * 4，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 GetQHYCCDMemoryLength。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

length: 返回的图像数据长度。

LabVIEW 实例:



2.14. SetQHYCCDCameraROI

函数原型:

```
int32_t SetQHYCCDCameraROI(
```

```
int32_t camIndex,
int32_t startx,
int32_t starty,
int32_t sizex,
int32_t sizey
);
```

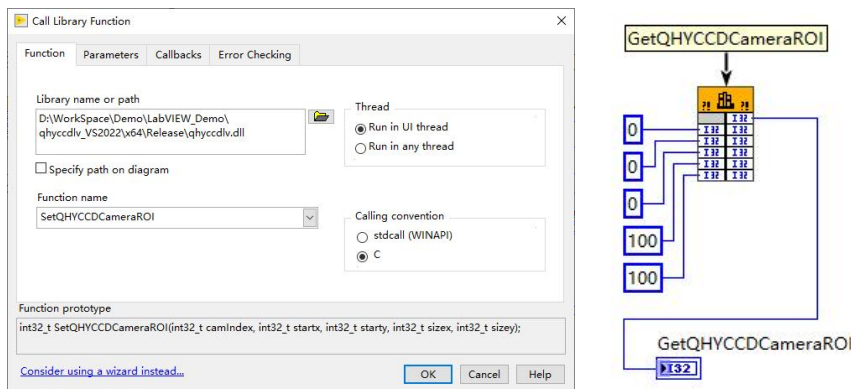
函数说明:

设置图像 ROI，部分相机支持垂直方向的硬件裁剪，设置垂直方向的图像裁剪可以减少数据传输量，增加连续模式图像帧率，若相机不支持硬件裁剪则默认使用软件裁剪，此时并不会提高图像帧率，通常在设置合并模式之后执行此函数，且需注意 ROI 范围参数不能超出图像实际尺寸，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 SetQHYCCDResolution。

参数说明:

- camIndex: 目标相机的索引，从 0 开始计数。
- startx: ROI 区域起始位置的 x 坐标，以图像左上角为参考点。
- starty: ROI 区域起始位置的 y 坐标，以图像左上角为参考点。
- sizex: ROI 区域的宽度。
- sizey: ROI 区域的高度。

LabVIEW 实例:



2.15. SetQHYCCDCameraExpTime

函数原型:

```
int32_t SetQHYCCDCameraExpTime(
    int32_t camIndex,
    int32_t unit,
    int32_t time
);
```

函数说明:

设置相机拍摄时的曝光时间，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 SetQHYCCDParam。

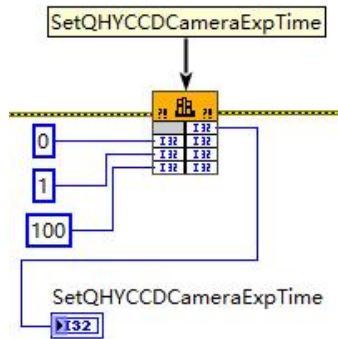
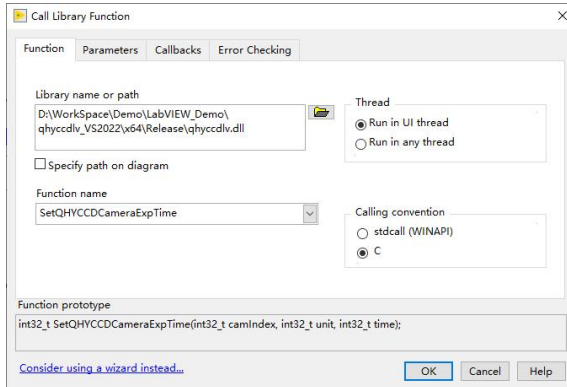
参数说明:

camIndex: 目标相机的索引, 从 0 开始计数。

unit: 时间单位, 参数为 0 时代表微秒, 为 1 时代表毫秒, 为 2 时代表秒。

time: 时间长度值。

LabVIEW 实例:



2.16. SetQHYCCDCameraGain

函数原型:

```
int32_t SetQHYCCDCameraGain(
    int32_t camIndex,
    int32_t gain
);
```

函数说明:

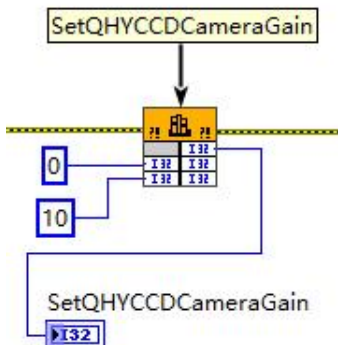
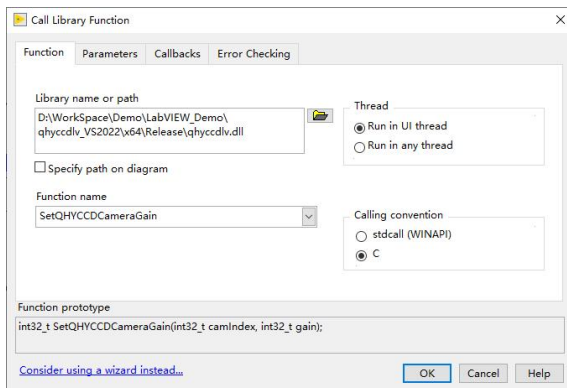
设置相机增益, 设置相机工作模式后可以执行此函数, 函数执行成功时返回 0, 失败时返回-1, 调用的 QHYCCD SDK 函数为 SetQHYCCDParam。

参数说明:

camIndex: 目标相机的索引, 从 0 开始计数。

gain: 增益参数值。

LabVIEW 实例:



2.17. SetQHYCCDCameraOffset

函数原型:

```
int32_t SetQHYCCDCameraOffset(
    int32_t camIndex,
    int32_t offset
);
```

函数说明:

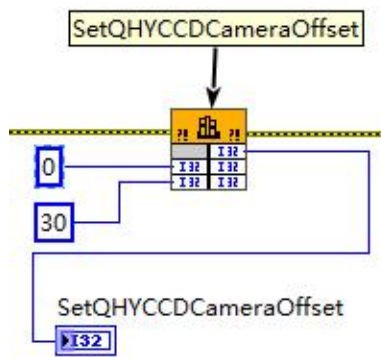
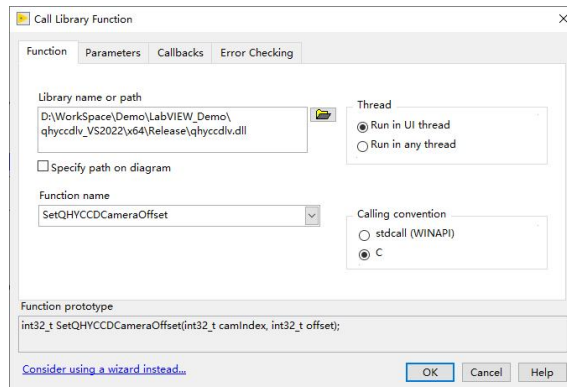
设置相机 offset，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 SetQHYCCDParam。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

offset: offset 参数值。

LabVIEW 实例:



2.18. SetQHYCCDCameraTraffic

函数原型:

```
int32_t SetQHYCCDCameraTraffic(
    int32_t camIndex,
    int32_t traffic
);
```

函数说明:

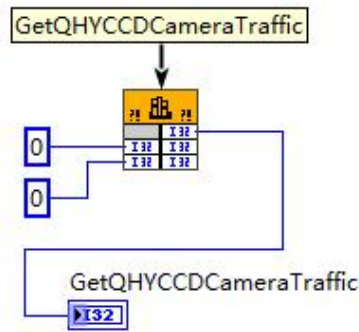
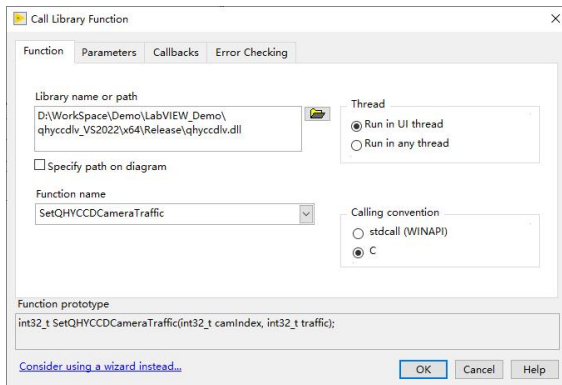
设置相机 traffic，通过此参数可以调节连续模式下的帧率，traffic 值越小帧率越高，设置 0 时为最高，需要注意的是，此参数不会使上位机软件显示的帧率保持某一固定值，而是一个会波动的变化值，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 SetQHYCCDParam。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

traffic: traffic 参数值。

LabVIEW 实例:



2.19. SetQHYCCDCameraGainR

函数原型:

```
int32_t SetQHYCCDCameraGainR(
    int32_t camIndex,
    int32_t gainR
);
```

函数说明:

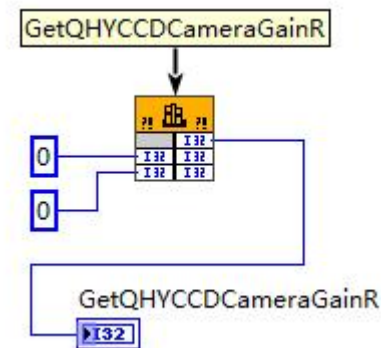
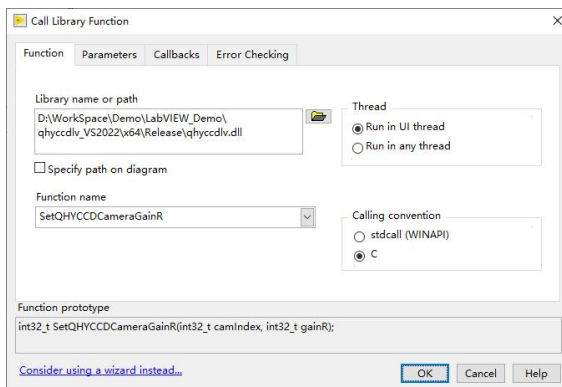
设置彩色相机 R 通道增益，仅彩色型号相机支持此设置，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 SetQHYCCDParam。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

gainR: R 通道增益参数值。

LabVIEW 实例:



2.20. SetQHYCCDCameraGainG

函数原型:

```
int32_t SetQHYCCDCameraGainG(
    int32_t camIndex,
```

```
int32_t gainG
);
```

函数说明:

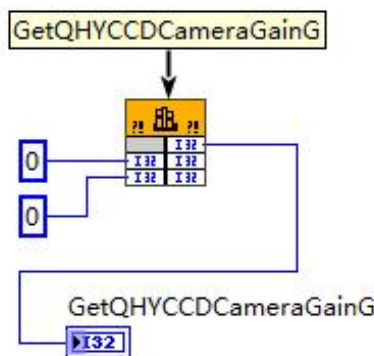
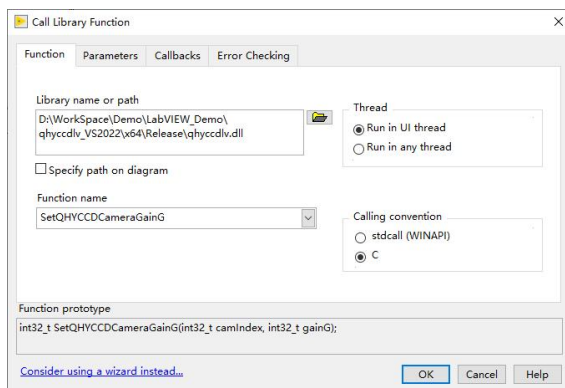
设置彩色相机 G 通道增益，仅彩色型号相机支持此设置，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 SetQHYCCDParam。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

gainG: G 通道增益参数值。

LabVIEW 实例:



2.21. SetQHYCCDCameraGainB

函数原型:

```
int32_t SetQHYCCDCameraGainB(
    int32_t camIndex,
    int32_t gainB
);
```

函数说明:

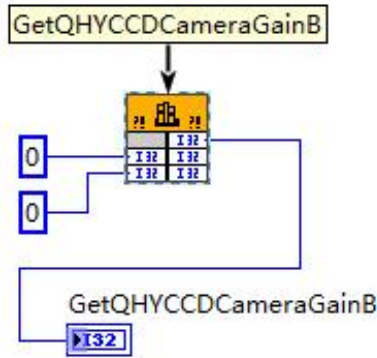
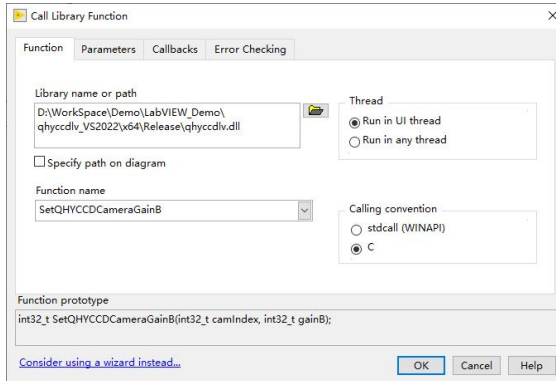
设置彩色相机 B 通道增益，仅彩色型号相机支持此设置，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 SetQHYCCDParam。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

gainB: B 通道增益参数值。

LabVIEW 实例:



2.22. SetQHYCCDCameraBrightness

函数原型:

```
int32_t SetQHYCCDCameraBrightness(
    int32_t camIndex,
    double brightness
);
```

函数说明:

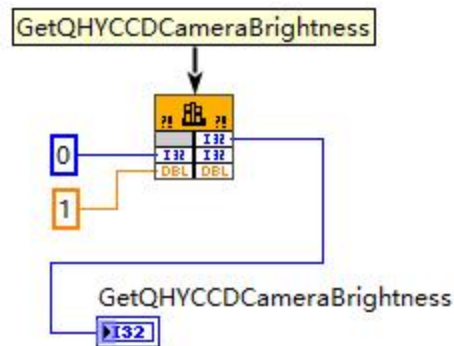
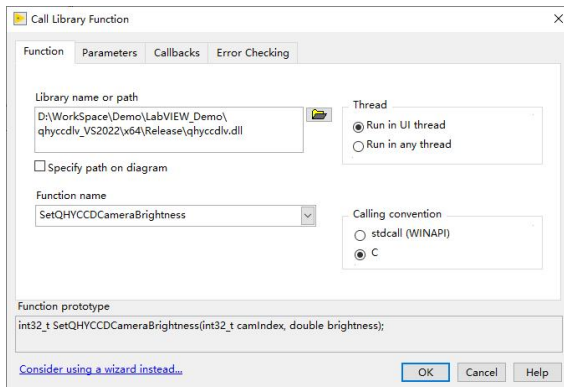
设置连续模式图像亮度，设置值越大图像越亮，反之越暗，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 SetQHYCCDParam。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

brightness: 亮度参数值，默认值为 1.0。

LabVIEW 实例:



2.23. SetQHYCCDCameraContrast

函数原型:

```
int32_t SetQHYCCDCameraContrast(
    int32_t camIndex,
```

```
double contrast
);
```

函数说明:

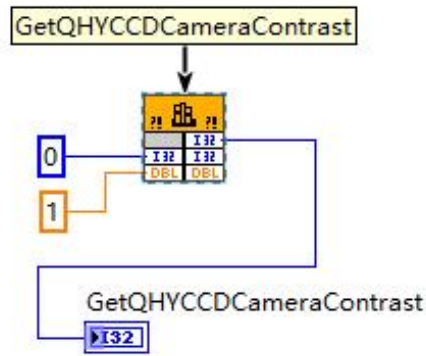
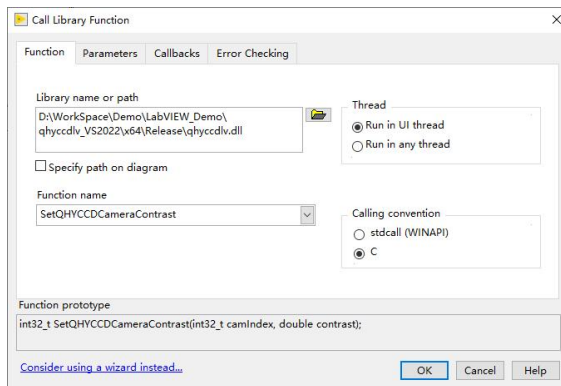
设置连续模式图像对比度，设置值越大，对比度越高，反之越低，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回 -1，调用的 QHYCCD SDK 函数为 SetQHYCCDParam。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

contrast: 对比度参数值，默认值为 1.0。

LabVIEW 实例:



2.24. SetQHYCCDCameraGamma

函数原型:

```
int32_t SetQHYCCDCameraGamma(
    int32_t camIndex,
    double gamma
);
```

函数说明:

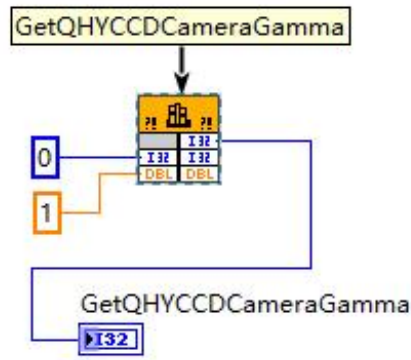
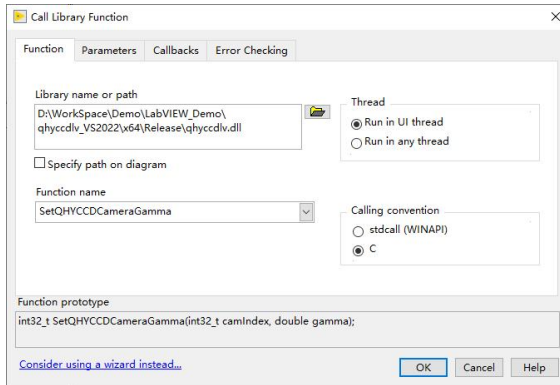
设置连续模式图像 gamma 值，设置值越大图像越亮，反之越暗，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回 -1，调用的 QHYCCD SDK 函数为 SetQHYCCDParam。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

gamma: gamma 参数值，默认值为 0。

LabVIEW 实例:



2.25. GetQHYCCDCameraOneSingleFrame

函数原型:

```
int32_t GetQHYCCDCameraOneSingleFrame(
    int32_t camIndex,
    int32_t * ImgW,
    int32_t * ImgH,
    int32_t * ImgBpp,
    int32_t * ImgChannel,
    uint32_t * ImgData
);
```

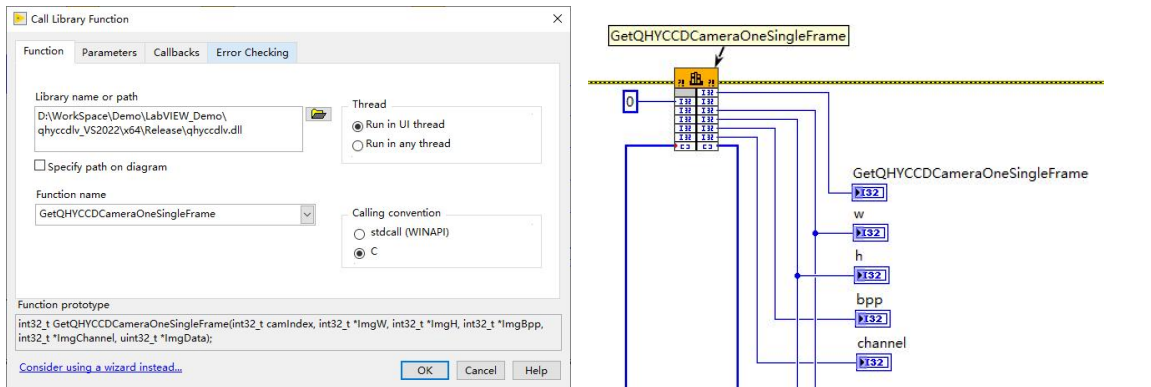
函数说明:

控制相机进行拍摄并获取一帧图像数据，函数为阻塞运行，长曝光模式时软件卡顿为正常现象，拍摄完成后会固定按照从左到右从上到下的顺序读取芯片像素数据，并依次存放到一维数组中，单帧模式数据格式通常固定为黑白十六位模式，此时两个数组元素存储一个像素的数据，偶数索引数组元素存储低位数据，奇数索引数组元素存储高位数据，原始像素值计算方式为高位数据*256+低位数据，例如 ImgData[0]、ImgData[1]存储第一个像素的数据，像素值为 ImgData[1]*256+ImgData[0]，设置参数后可以执行此函数，函数执行成功时返回 0，失败时返回 -1，调用的 QHYCCD SDK 函数包括 ExpQHYCCDSingleFrame、GetQHYCCDSingleFrame。

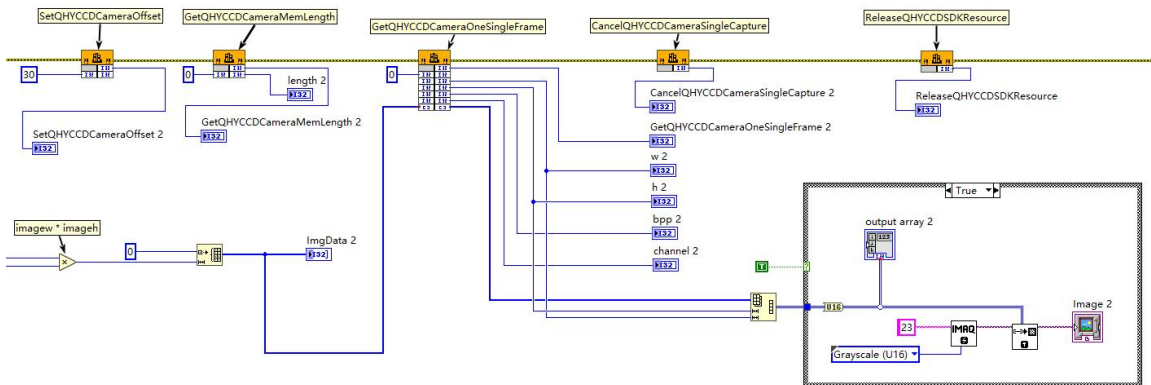
参数说明:

- camIndex: 目标相机的索引，从 0 开始计数。
- ImgW: 返回获取到的图像宽度。
- ImgH: 返回获取到的图像高度。
- ImgBpp: 返回获取到的图像位数，单帧模式通常为 16。
- ImgChannel: 返回获取到的图像通道数，黑白图像固定为 1，彩色图像固定为 3。
- ImgData: 返回获取到的图像数据。

LabVIEW 实例:



完整显示图像实例：



2.26. CancelQHYCCDCameraSingleCapture

函数原型：

```
int32_t CancelQHYCCDCameraSingleCapture(
    int32_t camIndex
);
```

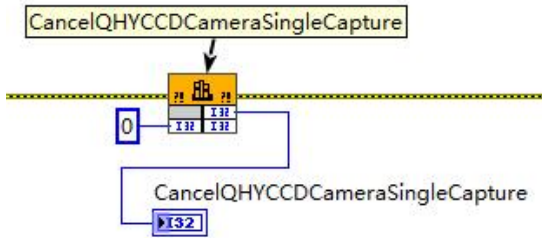
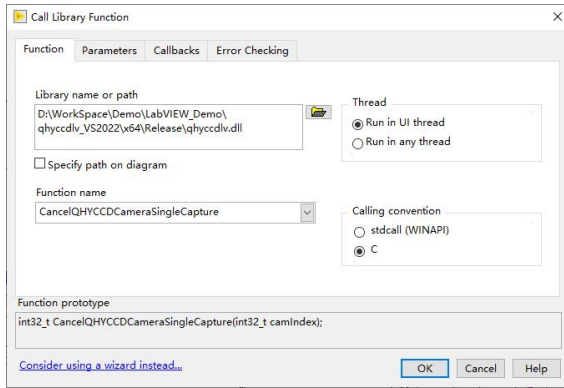
函数说明：

取消相机单帧拍摄操作且不会读出图像数据，执行成功时返回 0，失败时返回 1，调用的 QHYCCD SDK 函数为 CancelQHYCCDExposingAndReadout。

参数说明：

camIndex: 目标相机的索引，从 0 开始计数。

LabVIEW 实例：



2.27. BeginQHYCCDCameraLiveCapture

函数原型:

```
int32_t BeginQHYCCDCameraLiveCapture(
    int32_t camIndex
);
```

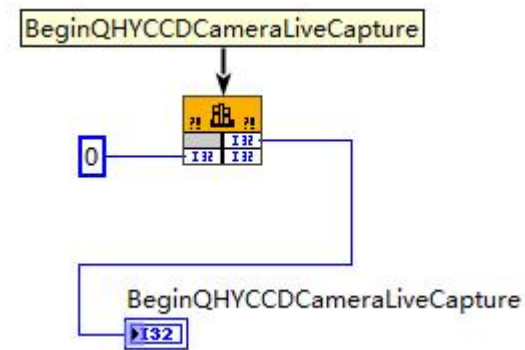
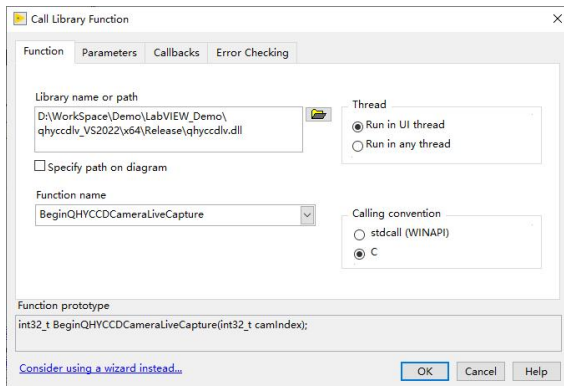
函数说明:

开始连续模式拍摄，调用此函数后 QHYCCD SDK 内部线程会自动获取图像数据，执行成功时返回 0，失败时返回-1，设置参数后可执行此函数，调用的 QHYCCD SDK 函数为 BeginQHYCCDLive。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

LabVIEW 实例:



2.28. GetQHYCCDCameraOneLiveFrame

函数原型:

```
int32_t GetQHYCCDCameraOneLiveFrame(
    int32_t camIndex,
    int32_t * ImgW,
    int32_t * ImgH,
```



```
int32_t * ImgBpp,
int32_t * ImgChannel,
uint32_t * ImgData
);
```

函数说明:

从 QHYCCD SDK 线程中获取一帧图像数据，函数为阻塞运行，长曝光模式时软件卡顿为正常现象，拍摄完成后会固定按照从左到右从上到下的顺序读取芯片像素数据，并依次存放到一个一维数组中，黑白八位模式下一个数组元素存储一个像素的数据，黑白十六位模式下两个数组元素存储一个像素的数据，偶数索引数组元素存储低位数据，奇数索引数组元素存储高位数据，原始像素值计算方式为高位数据*256+低位数据，例如 ImgData[0]、ImgData[1]存储第一个像素的数据，像素值为 $\text{ImgData}[1]*256+\text{ImgData}[0]$ ，八位彩色模式下三个数组元素存储一个像素数据，分别存储 B、G、R 三个通道的图像数据，开始连续模式拍摄后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 GetQHYCCDLiveFrame。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

ImgW: 返回获取到的图像宽度。

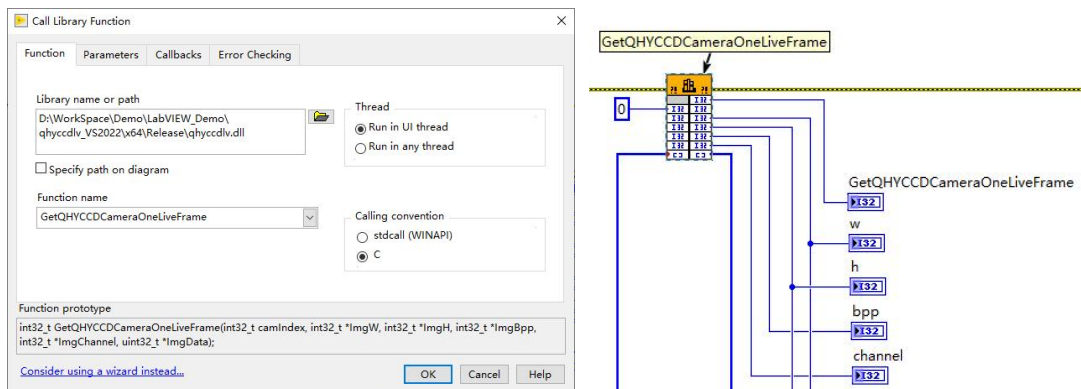
ImgH: 返回获取到的图像高度。

ImgBpp: 返回获取到的图像位数。

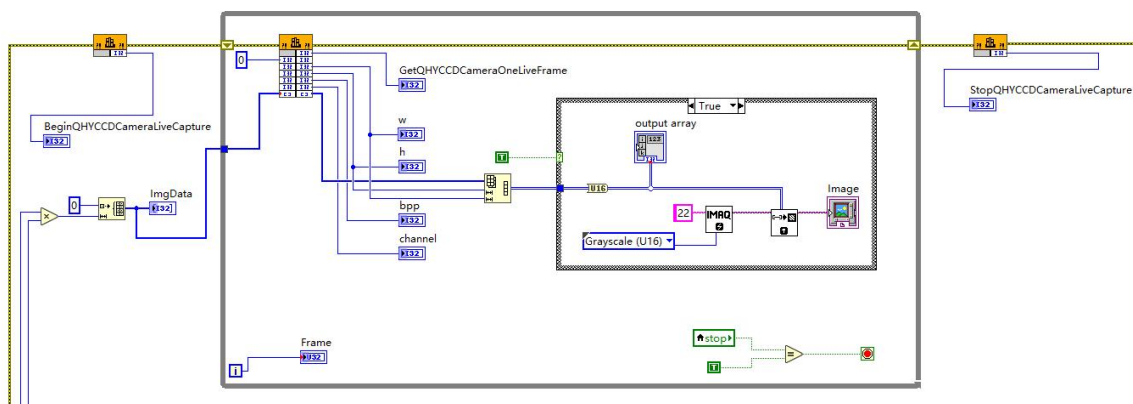
ImgChannel: 返回获取到的图像通道数，黑白图像固定为 1，彩色图像固定为 3。

ImgData: 返回获取到的图像数据。

LabVIEW 实例:



完整显示图像实例:



2.29. StopQHYCCDCameraLiveCapture

函数原型:

```
int32_t StopQHYCCDCameraLiveCapture(
    int32_t camIndex
);
```

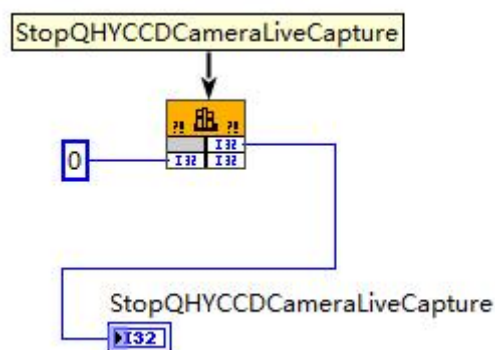
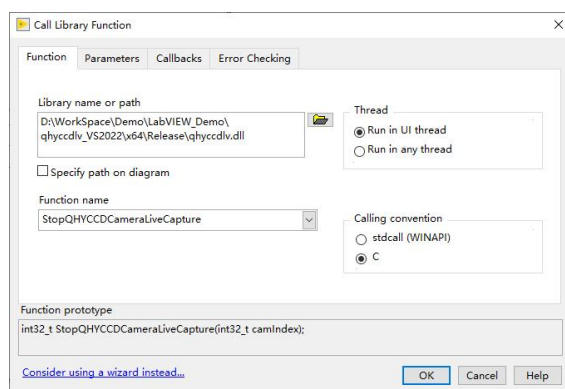
函数说明:

停止连续模式拍摄，结束 QHYCCD SDK 内部线程，执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 StopQHYCCDLive。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

LabVIEW 实例:



2.30. GetQHYCCDCameraTriggerInterfaceNumber

函数原型:

```
int32_t GetQHYCCDCameraTriggerInterfaceNumber(
    int32_t camIndex,
    int32_t *number
)
```

);

函数说明：

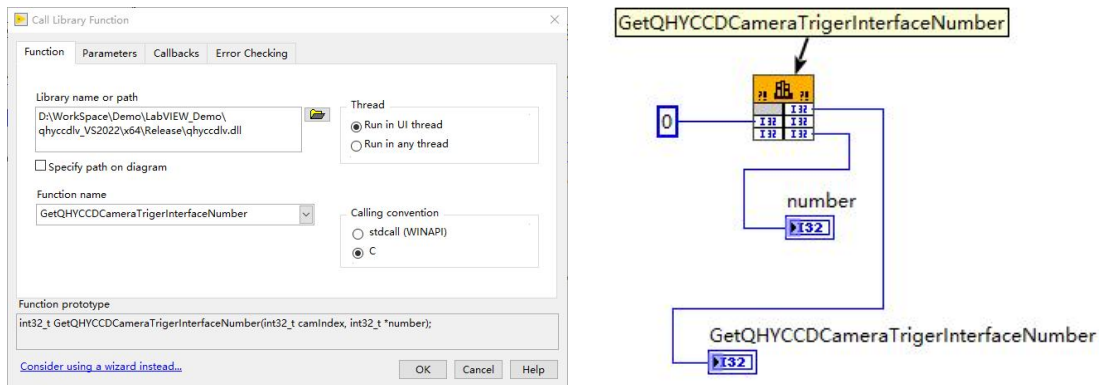
获取相机支持的硬件触发接口数量，部分相机带有多个硬件触发接口，如 SMA 或 GPIO，可以根据需求进行选择，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 GetQHYCCDTrigerInterfaceNumber。

参数说明：

camIndex: 目标相机的索引，从 0 开始计数。

number: 返回触发接口的数量。

LabVIEW 实例：



2.31. GetQHYCCDTrigerInterfaceName

函数原型：

```
int32_t GetQHYCCDTrigerInterfaceName(
    int32_t camIndex,
    int32_t interfaceIndex,
    char *name
);
```

函数说明：

获取硬件触发接口的名称，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 GetQHYCCDTrigerInterfaceName。

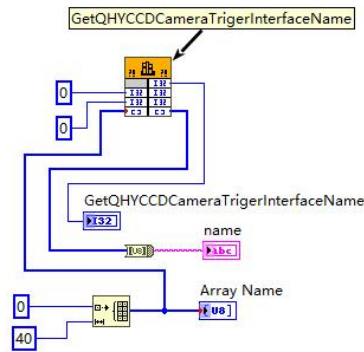
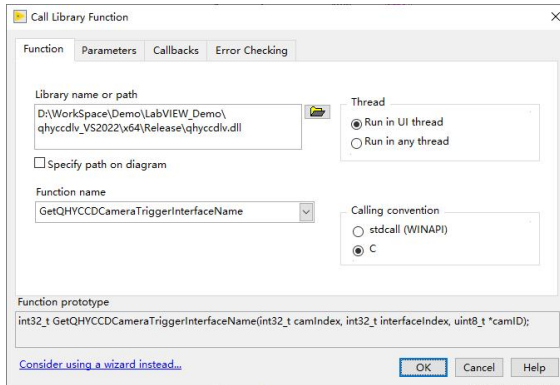
参数说明：

camIndex: 目标相机的索引，从 0 开始计数。

interfaceIndex: 触发接口的索引，从 0 开始计数。

name: 返回触发接口的名称。

LabVIEW 实例：



2.32. SetQHYCCDCameraTriggerInterface

函数原型:

```
int32_t SetQHYCCDCameraTriggerInterface(
    int32_t camIndex,
    int32_t i
);
```

函数说明:

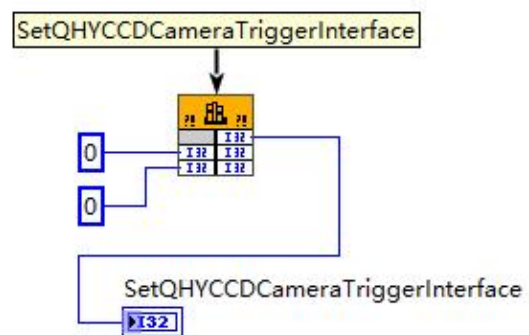
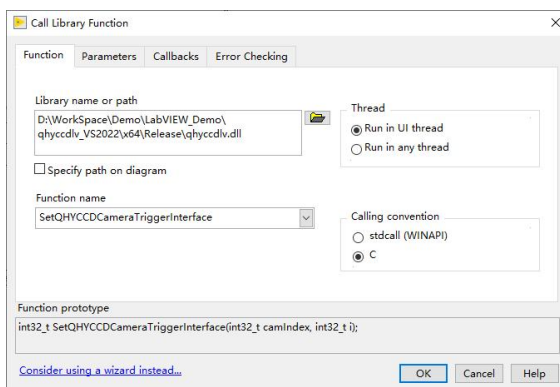
设置相机外触发功能硬件触发接口，部分相机带有多个硬件触发接口，如 SMA 或 GPIO，此时需要人为设置要使用的触发接口，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 SetQHYCCDTrigerInterface。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

i: 触发接口。

LabVIEW 实例:



2.33. GetQHYCCDCameraTriggerModeNumber

函数原型:

```
int32_t GetQHYCCDCameraTriggerModeNumber(
```

```
int32_t camIndex,
int32_t *number
);
```

函数说明:

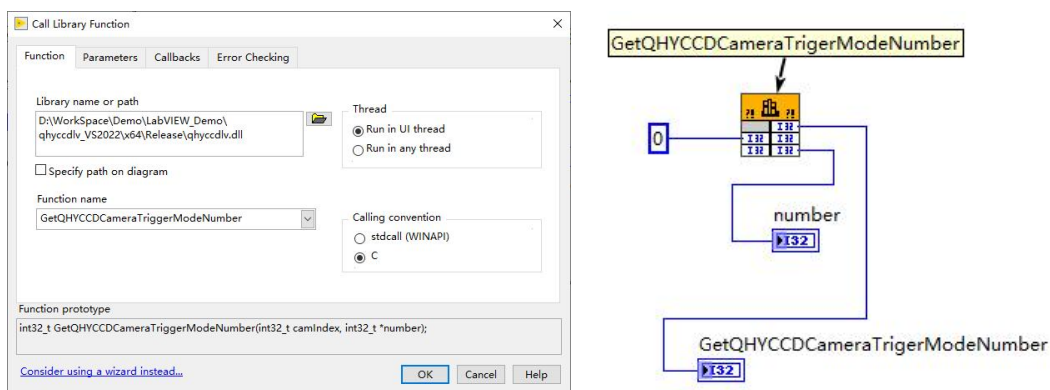
获取相机支持的触发模式数量，部分相机支持多个触发模式，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回 -1，调用的 QHYCCD SDK 函数为 GetQHYCCDTrigerModeNumber。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

number: 返回触发模式数量。

LabVIEW 实例:



2.34. GetQHYCCDTrigerModeName

函数原型:

```
int32_t GetQHYCCDTrigerModeName(
    int32_t camIndex,
    int32_t modeIndex,
    char *name
);
```

函数说明:

获取触发模式的名称，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回 -1，调用的 QHYCCD SDK 函数为 GetQHYCCDTrigerModeName。

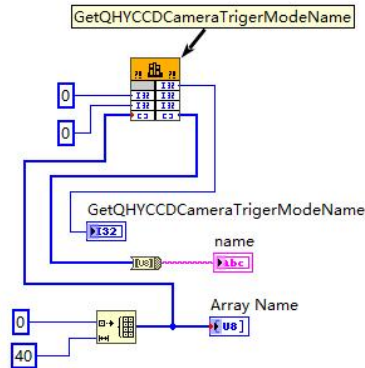
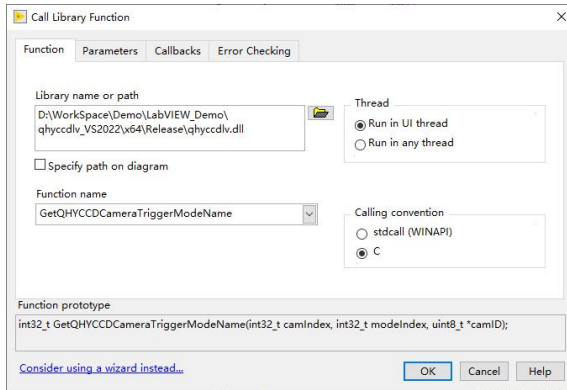
参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

modeIndex: 触发模式的索引，从 0 开始计数。

name: 返回触发模式的名称。

LabVIEW 实例:



2.35. SetQHYCCDCameraTriggerMode

函数原型:

```
int32_t SetQHYCCDCameraTriggerMode(
    int32_t camIndex,
    int32_t mode
);
```

函数说明:

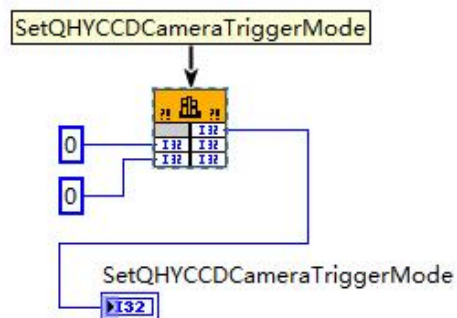
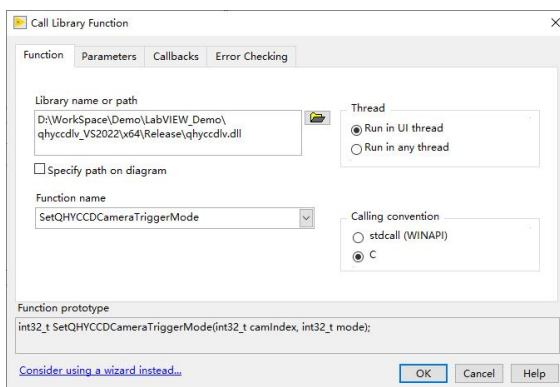
设置相机触发模式，部分相机支持多种触发模式，此时需要人为设置模式，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 SetQHYCCDTriggerMode。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

mode: 触发模式。

LabVIEW 实例:



2.36. SetQHYCCDCameraTriggerOnOff

函数原型:

```
int32_t SetQHYCCDCameraTriggerOnOff(
```

```
int32_t camIndex,
int32_t onoff
);
```

函数说明:

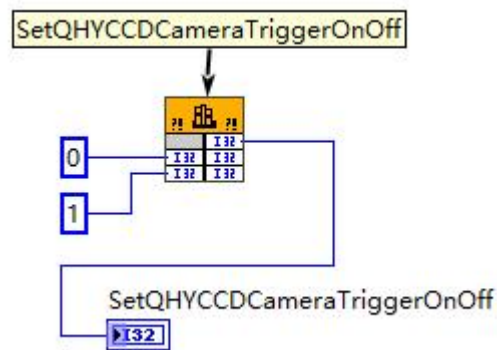
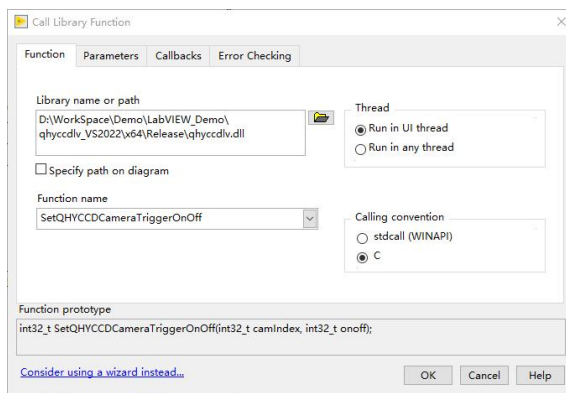
设置触发功能开启和关闭，开启触发功能时，相机将受硬件触发信号控制进行拍摄，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 SetQHYCCDTrigerFunction。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

onoff: 1 为开启，0 为关闭。

LabVIEW 实例:



2.37. SetQHYCCDCameraTriggerInOnly

函数原型:

```
int32_t SetQHYCCDCameraTriggerInOnly(
    int32_t camIndex,
    int32_t onoff
);
```

函数说明:

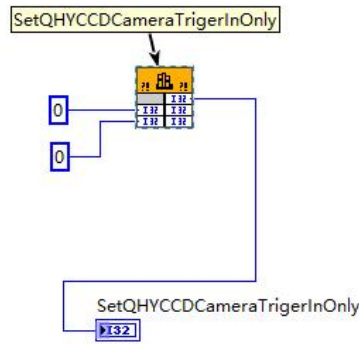
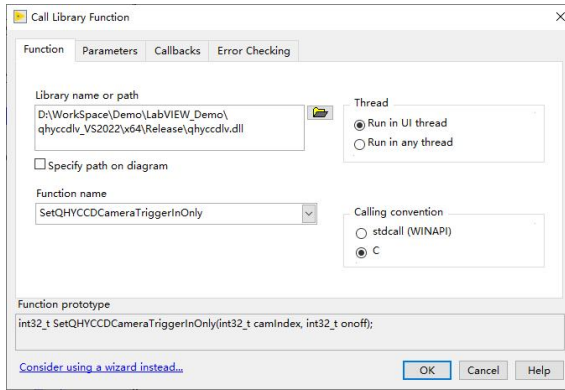
设置触发输入功能开启或关闭，此函数只设置触发输入功能，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回 -1，调用的 QHYCCD SDK 函数为 GetQHYCCDTrigerInOnly。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

onoff: 设置触发输入开启或关闭，1 为开启，0 为关闭。

LabVIEW 实例:



2.38. SetQHYCCDCameraTrigerOutOnly

函数原型:

```
int32_t SetQHYCCDCameraTrigerOutOnly(
    int32_t camIndex,
    int32_t onoff
);
```

函数说明:

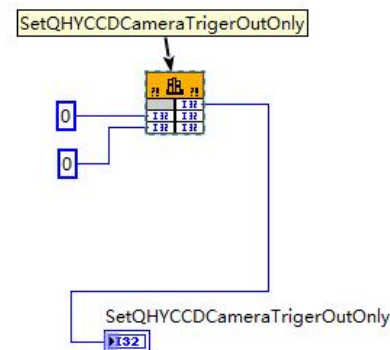
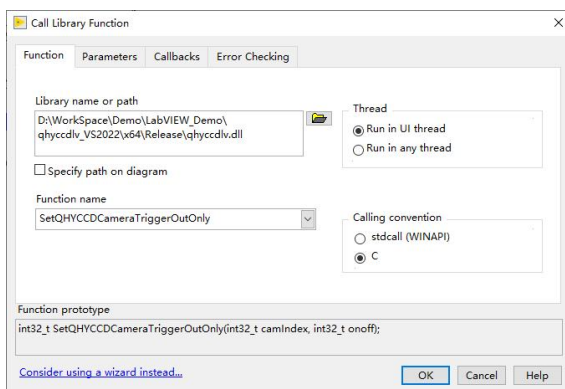
设置触发输出功能开启或关闭，此函数只设置触发输出功能，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回 -1，调用的 QHYCCD SDK 函数为 SetQHYCCDTrigerOutOnly。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

onoff: 设置触发输出开启或关闭，1 为开启，0 为关闭。

LabVIEW 实例:



2.39. EnableQHYCCDCameraTrigerOut

函数原型:

```
int32_t EnableQHYCCDCameraTrigerOut(
```



```
int32_t camIndex
);
```

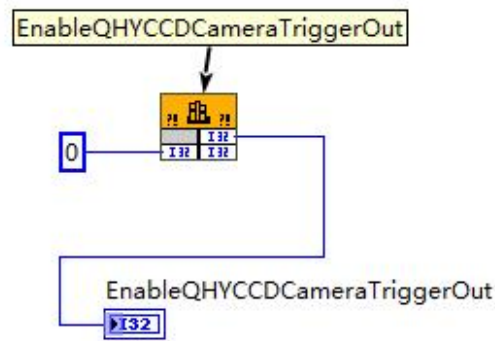
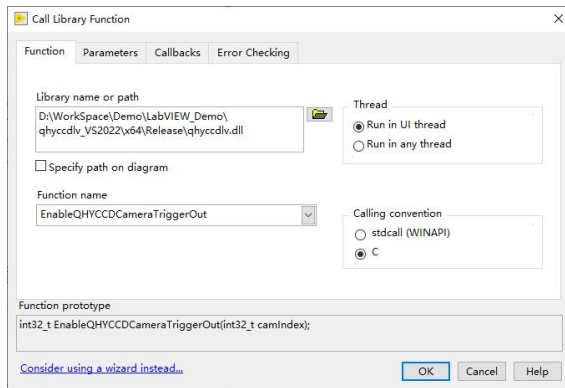
函数说明:

使能触发输出功能，此函数设置只启用触发输出功能，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 EnableQHYCCDTrigerOut。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

LabVIEW 实例:



2.40. SetQHYCCDCameraBurstModeOnOff

函数原型:

```
int32_t SetQHYCCDCameraBurstModeOnOff(
    int32_t camIndex,
    int32_t onoff
);
```

函数说明:

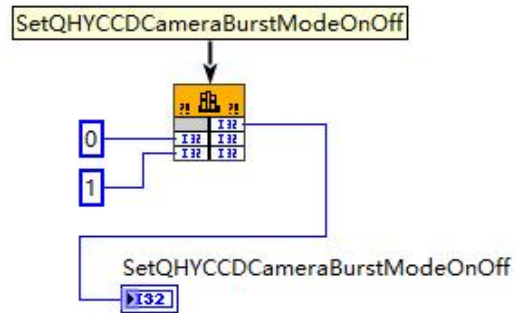
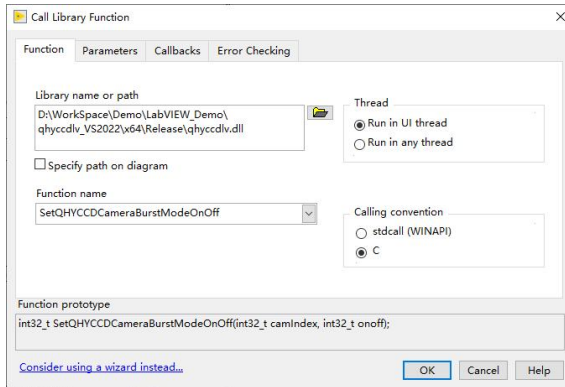
设置 burst 模式开启和关闭，此模式仅在连续模式下可以开启，开启 Burst 模式后相机进入 IDLE 状态，此状态下可以通过发送软件指令控制相机拍摄指定数量的图像，此功能可以代替单帧模式更快速地获取图像数据，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 EnableQHYCCDBurstMode。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

onoff: 1 为开启，0 为关闭。

LabVIEW 实例:



2.41. SetQHYCCDCameraBurstModeStartEnd

函数原型:

```
int32_t SetQHYCCDCameraBurstModeStartEnd(
    int32_t camIndex,
    int32_t start,
    int32_t end
);
```

函数说明:

设置 Burst 模式图像开始和结束帧序号,拍摄时将输出开始和结束帧中间的图像,设置 start、end 分别为 1、3,拍摄时相机将输出帧序号为 2 的图像,设置相机工作模式后可以执行此函数,函数执行成功时返回 0,失败时返回-1,调用的 QHYCCD SDK 函数为 SetQHYCCDBurstStartEnd。

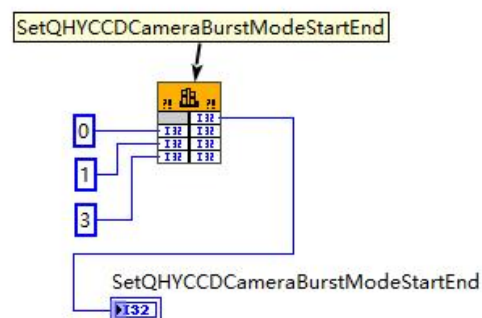
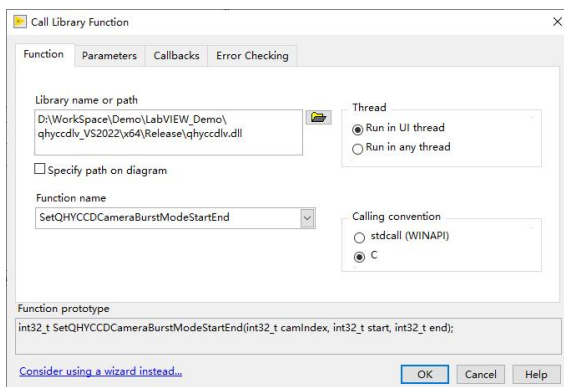
参数说明:

camIndex: 目标相机的索引,从 0 开始计数。

start: 开始帧的序号。

end: 结束帧的序号。

LabVIEW 实例:



2.42. SetQHYCCDCameraBurstModePatchNumber

函数原型:

```
int32_t SetQHYCCDCameraBurstModePatchNumber(
    int32_t camIndex,
    int32_t number
);
```

函数说明:

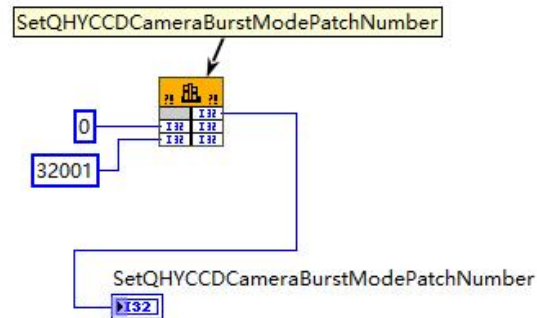
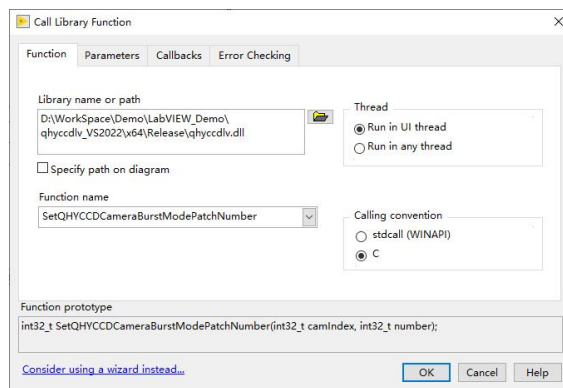
设置 Burst 模式补充数据包的长度，Burst 模式下若相机缓存中数据过少，有可能会造成图像数据无法完整读出，此时可以通过此函数使图像数据可以被正常读出，通常设置 32001 以内的数值，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，执行失败时返回-1，调用的 QHYCCD SDK 函数为 SetQHYCCDBurstModePatchNumber。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

number: 补充数据包的长度。

LabVIEW 实例:



2.43. SetQHYCCDCameraBurstModeCapture

函数原型:

```
int32_t SetQHYCCDCameraBurstModeCapture(
    int32_t camIndex
);
```

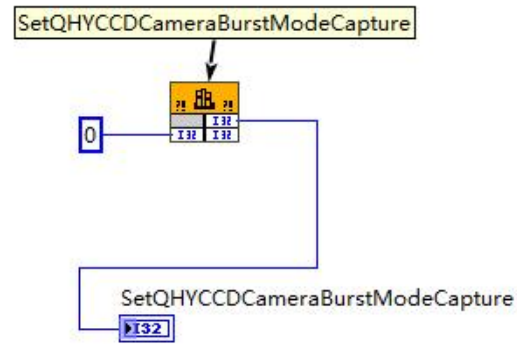
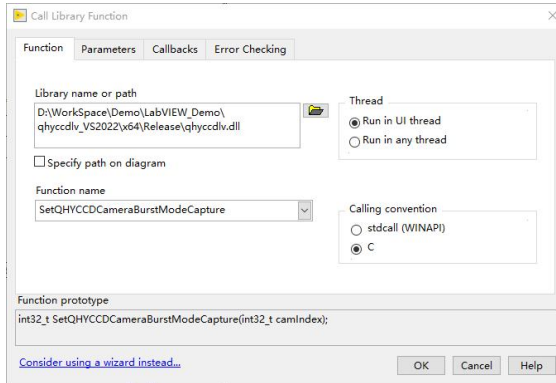
函数说明:

Burst 模式，调用此函数可以发送指令，控制相机拍摄图像，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数包括 SetQHYCCDBurstIDLE、ReleaseQHYCCDBurstIDLE。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

LabVIEW 实例:



2.44. SetQHYCCDCameraTargetTemperature

函数原型:

```
int32_t SetQHYCCDCameraTargetTemperature(
    int32_t camIndex,
    double temp
);
```

函数说明:

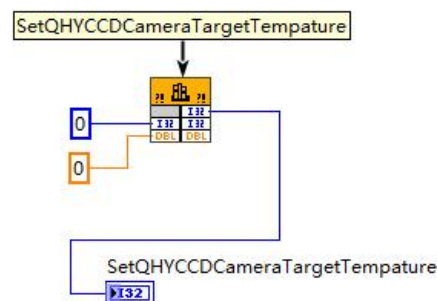
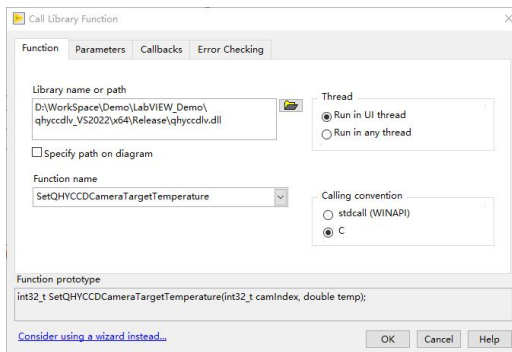
设置制冷器目标温度，设置完成后相机将自动调节制冷器功率，使相机维持目标温度，需要注意的是相机制冷能力有上限，一般制冷范围为低于环境温度 30~35℃，若设置的目标温度与环境温度相差过大，会超出相机制冷能力范围，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 为 SetQHYCCDParam。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

temp: 目标温度。

LabVIEW 实例:



2.45. GetQHYCCDCameraCurTemperature

函数原型:

```
int32_t GetQHYCCDCameraCurTemperature(
```

```
int32_t camIndex,  
double * temp  
);
```

函数说明:

获取相机当前温度，此函数为单次获取，若要实现显示温度变化，需持续调用函数获取温度，设置相机工作模式后可以执行此函数，函数执行成功时返回 0，失败时返回-1，调用的 QHYCCD SDK 函数为 GetQHYCCDParam。

参数说明:

camIndex: 目标相机的索引，从 0 开始计数。

temp: 相机当前温度。

LabVIEW 实例:

