

高德红外SGP_SDK开发手册(C++版)

版权声明

版权所有 武汉高德红外股份有限公司2022。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本
文档内容的部分或全部，并不得以任何形式传播

技术支持

武汉高德红外股份有限公司

咨询热线：4008 822 866 （周一至周五 8:40-17:30）

售后服务专线：027-81298738 （周一至周五 8:40-17:30）

微信公众号：GuideSensmart

官方微博：@高德红外

E-mail：marketing@guide-infrared.com

地址：湖北省武汉市东湖开发区黄龙山南路6号

邮编：430205

版权所有©武汉高德红外股份有限

公司

概述

欢迎使用SGP_SDK开发手册，本文档详细描述了开发包中各个函数实现功能、接口及其函数之间的调用关系和示例实现。

本开发包主要包含业务操作和设备管理两大部分：

业务操作

实时预览、字符叠加、拍照、视频录制、温度获取、报警等功能。

设备管理

设备重启、密码管理、设备参数配置(系统通用参数配置、报警参数配置、图像配置、视频配置、网络配置、电机配置、融合配置)等功能。

Windows下，SDK包括的文件有

功能库	SgpApi.h	SDK对外头文件
	SgpParam.h	结构体定义头文件
	SgpApi.lib	lib功能库
	SgpApi.dll	功能库
依赖库	PocoJSON.dll	
	PocoFoundation.dll	
	avcodec-58.dll	
	avformat-58.dll	
	avutil-56.dll	
	libcrypto-1_1.dll	
	swresample-3.dll	

	swscale-5.dll	
	PocoNet.dll	
	PocoUtil.dll	
	PocoXML.dll	
配置 文件	log_conf.properties	
Linux下，SDK包括的文件有		
功能 库	SgpApi.h	SDK对外头文件
	SgpParam.h	结构体定义头文件
	libSgpApi.so	功能库
依赖 库	libavcodec.so	
	libva.so	
	libva-x11.so	
	libavdevice.so	
	libavfilter.so	
	libavformat.so	
	libavutil.so	
	libva-drm.so	
	ibcrypto.so	
	libPocoFoundation.so	
	libPocoJSON.so	
	libPocoNet.so	
	libPocoUtil.so	
	libPocoXML.so	
	libpostproc.so	
	libc.so	
	libssl.so	
	libswresample.so	
	libswscale.so	
	libx264.so.164	

配置文件	log_conf.properties	
	sgp_sdk.log	SDK生成的日志文件、日志文件按照文件大小自动备份 定期自动删除
<p>本SDK的功能库和依赖库都是必须，缺少依赖库会导致某些功能运行异常。sgp_sdk.log为SDK日志文件。</p>		

版权所有©武汉高德红外股份有限

公司

SDK支持系统
<p>Windows 32/64位网络SDK：</p> <p>Windows 10/Windows 8/Windows 7以及Windows Server 2012/2008</p> <p>x86 Linux 32/64位设备网络SDK：</p> <p>已测系统：CentOS 7、Redhat、Ubuntu 12、Ubuntu 14、Ubuntu 16、Ubuntu 18、Ubuntu 20</p> <p>Arm Linux 32/64位设备网络SDK：</p> <p>需提供交叉编译环境，定制化编译</p>

开发说明	
Window 开发环境	<p>SDK:</p> <p>根据平台选择32位或者64位SDK，SDK包可单放在某个文件夹下或者将SDK包内容拷贝到应用程序exe同级别目录下。</p> <p>Demo:</p> <p>GuiderApiTest_IPT工程文件可用QT或者VS2015打开。</p>
Linux 开发环境	<p>SDK:</p> <p>根据平台选择32位或者64位SDK，解压命令tar -xzf lib.tar.gz。开使用的库在SDK\lib\目录下。SDK包可单放在某个文件夹下或者将SDK包内容拷贝到应用程序同级别目录下。</p> <p>Demo:</p> <p>main.cpp为C语言编写测试示例，可供参考。</p> <p>GuiderApiTest为QT开发的DEMO，为了能够正常运行，运行Demo需安装QT5.7，</p> <p>从QT官网下载qt-opensource-linux-x64-5.7.0.run。安装完成后设置环境变量。QTDIR改成实际安装路径。</p> <pre>export QTDIR=/home/Qt5.7.0 export PATH=\$QTDIR/5.7/gcc_64/bin:\$PATH export PATH=\$QTDIR/Tools/QtCreator/bin:\$PATH export LD_LIBRARY_PATH=\$QTDIR/5.7/gcc_64/lib:\$LD_LIBRARY_PATH</pre> <p>运行Demo</p> <ol style="list-style-type: none"> 1、拷贝lib.tar.gz到linux某目录，解压tar -xzf lib.tar.gz。 2、设置共享库路径。 <p>用root权限vi /etc/ld.so.conf，在"include ld.so.conf.d/*.conf"下方增加lib路径。</p>

	再执行/sbin/ldconfig -v使之生效。 3、关闭防火墙。 systemctl stop firewalld.service 4、切换到普通用户执行./GuiderApiTest或者双击GuiderApiTest运行程序。
--	---

版权所有©武汉高德红外股份有限

公司

修订记录

发布时间	版本号	修订内容
2022.03	V1.1.11	新增
2022.04	V1.1.14	增加电子变倍接口
2022.05	V1.1.15	修改SGP_PORT_INFO等几个结构体声明
2022.06	V1.1.16	增加注册外部告警回调函数、恢复出厂设置函数，增加部分结构体变量
2022.07	V1.1.19	在字符串叠加函数中增加IPT640M机芯显示左上、左下选项
2022.08	V1.2.1	1、增加三个拍照函数 SGP_GetScreenCaptureCache、 SGP_GetHeatMapCache、 SGP_GetFirHeatMapCache
2023.07	V1.2.9	1、增加通过SDK获取Y16数据 (SGP_GetY16) 进行测温SGP_GetTempMatrixEx 2.拍照

		等接口缺少对文件路径的校验 3. 增加获取测温点数组对应温度值接口 SGP_GetTempPoints 4. 增加火灾报警回调函数 SGP_RegisterFireAlarmCallback 5. 增加自动调焦回调函数 SGP_RegisterAutoFocusCallback
2023.09.25	V1.2.10	1、增加alarm_interal字段到分析对象结构体SGP_RULE中
2023.10.11	V1.2.14	1、增加show_type字段到分析对象结构体SGP_RULE中
2023.12.22	V1.2.15	1、增加注册对象温差告警回调函数 SGP_RegisterObjTempAlarmCallback , 增加对象温差结构体SGP_OBJTEMPALARMNOTIFY 2、对SGP_GetTempPoints、 SGP_SetTempPoints、 SGP_GetMatrixTempPoints、 SGP_GetPointTemp增加备注，指出接口的使用注意事项
2024.01.26	V1.2.16	1、更新 SGP_SetTempPoints 接口使用备注信息

公司

流程说明：

创建实例会返回实例序号值，之后的所有函数都需要用到这个序号值。

操作任何业务之前首先要登陆平台

创建实例 ([SGP_InitDevice](#)) : 生成一个SGPSDK的独立实例，之后所有函数操作都作用于这个实例。可多次调用此函数生成多个实例，多个实例之间的操作互不影响。

登陆平台 ([SGP_Login](#)) : 实现用户登录服务器功能。平台中设置用户为复用时，在可以同时间多次登陆。

报警业务：布控之后如果有报警触发，则会有报警事件触发。详见报警业务。

登出平台 ([SGP_Logout](#)) : 登出平台

销毁实例 ([SGP_UnInitDevice](#)) :

公司

流程说明：

创建实例会返回实例序号值，之后的所有函数都需要用到这个序号值。

操作任何业务之前首先要登陆到平台

创建实例 ([SGP_InitDevice](#)) : 生成一个SGPSDK的独立实例，之后所有函数操作都作用于这个实例。可多次调用此函数生成多个实例，多个实例之间的操作互不影响。

登陆平台 ([SGP_Login](#)) : 实现用户登录服务器功能。平台中设置用户为复用时，在可以同时间多次登陆。

拍照业务：调用[SGP_GetHeatMap](#)函数获得jpeg国网格式热图，调用[SGP_GetFirHeatMap](#)获得FIR格式国网格式热图。

登出平台 ([SGP_Logout](#)) : 登出平台

销毁实例 ([SGP_UnInitDevice](#)) :

版权所有©武汉高德红外股份有限

公司

连接设备接口

接口描述	功能描述	功能详细描述
SGP_InitDevice	初始化一个设备对象	
SGP_UnInitDevice	释放设备对象	
SGP_Login	用户登录	
SGP_Logout	用户登出	
SGP_GetGeneralInfo	获取通用信息	
SGP_ChangePassword	修改密码	
SGP_ResetPassword	重置密码	

版权所有©武汉高德

智感科技有限公司

初始化一个设备对象SGP_InitDevice

选项：	说明
描述：	初始化一个设备对象
详细描述：	
函数：	SGP_HANDLE SGP_InitDevice();

参数:	[in] 无
返回值:	成功 返回设备对象句柄
备注:	首次调用的函数，与 SGP_UnInitDevice 成对使用。同一台设备最多支持20路访问
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象句柄。 */ void Init() { SGP_HANDLE handle = 0; handle = SGP_InitDevice(); if (handle) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

版权所有©武汉高德红外股份有限

公司

释放设备对象SGP_UnInitDevice

选项：	说明
描述：	释放设备对象
详细描述：	
函数：	<code>void SGP_UnInitDevice(SGP_HANDLE handle);</code>
参数：	handle [in] 设备对象句柄
返回值：	无
备注：	与 SGP_InitDevice 成对使用
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象句柄。 **/ void Uninit() { SGP_UnInitDevice(handle); }</pre>

版权所有©武汉高德红外股份有限

公司

用户登录SGP_Login

选项:	说明
描述:	用户登录
详细描述:	需要登录成功以后才能访问其他接口
函数:	<pre>int SGP_Login(SGP_HANDLE handle, const char *server, const char *username, const char *password, int port);</pre>
参数:	<p>handle</p> <p>[in] 设备对象句柄</p> <p>server</p> <p>[in] 设备ip地址，设备ip初始默认为“192.168.1.168”</p> <p>username</p> <p>[in] 用户名，管理员账号默认为“admin”</p> <p>password</p> <p>[in] 密码，admin账号密码默认为“admin123”，明文方式</p> <p>port</p> <p>[int] 端口，端口默认为80</p>
返回值:	成功返回 SGP_OK ，失败返回 错误码
备注:	需要登录成功以后才能访问其他接口，与 SGP_Logout 成对使用

使用示例:	<pre> /** 示例中部分类、变量、函数的解释： 1, handle 设备对象句柄。 */ void Init() { SGP_HANDLE handle = 0; handle = SGP_InitDevice(); if (handle) { const char *server = "192.168.1.168"; const char *username = "admin"; const char *password = "admin123"; int port = 80; int ret = SGP_Login(handle, server, username, password, port); if (ret == SGP_OK) { //登录成功，TODO..... } else { SGP_UnInitDevice(handle) //登录失败，TODO..... } } else { //失败，TODO..... } } </pre>
-------	---

用户登出SGP_Logout

选项：	说明
描述：	用户登出
详细描述：	
函数：	<code>int SGP_Logout(SGP_HANDLE handle);</code>
参数：	<code>handle</code> [in] 设备对象句柄
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	与 SGP_Login 成对使用
使用示例：	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象句柄。 */ void Init() { int ret = SGP_Logout(handle); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

公司

修改密码SGP_ChangePassword

选项：	说明
描述：	修改密码
详细描述：	
函数：	<pre>int SGP_ChangePassword(SGP_HANDLE handle, const char *username, const char *oldpassword, const char *newpassword);</pre>
参数：	<pre>handle [in] 设备对象句柄 username [in] 登录用户名 oldpassword</pre>

	<div>[in] 旧密码 明文 newpassword [in] 新密码 明文</div>
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	
使用示例:	<div><pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象句柄。 */ void Init() { const char* username = "admin"; const char* oldpassword = "admin123"; const char* newpassword = "admin567"; int ret = SGP_ChangePassword(m_handle, username, oldpassword, newpassword) ; if (ret ==SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre></div>

公司

获取通用信息SGP_GetGeneralInfo

选项:	说明
描述:	获取通用信息
详细 描述:	
函数:	<code>int SGP_GetGeneralInfo(SGP_HANDLE handle, SGP_GENERAL_INFO *output);</code>
参数:	<code>handle</code> [in] 设备对象句柄 <code>output</code> [out] 输出信息，获取的通用信息
返回 值:	成功返回SGP_OK，失败返回 错误码
备注:	结构体变量在使用前先初始化
使用 示例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象句柄。 **/ void Init() { SGP_GENERAL_INFO info; memset(&info, 0x00, sizeof(SGP_GENERAL_INFO));</pre>

	<pre>int ret = SGP_GetGeneralInfo(handle,&info); if (ret ==SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>
--	--

版权所有©武汉高德红外股份有限

公司

重置密码SGP_ResetPassword

选项:	说明
描述:	重置密码
详细描述:	
函数:	<pre>int SGP_ResetPassword(SGP_HANDLE handle, const char *username);</pre>
参数:	<pre>handle [in] 设备对象句柄</pre>

	<code>username</code> [in] 登录用户名
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象句柄。 */ void Init() { const char* username = "admin"; int ret = SGP_ResetPassword(m_handle, username); if (ret ==SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

接口描述	功能描述	功能详细描述
SGP_GetPointTemp	获取单点温度	
SGP_GetAnalyticObjectsTemp	获取分析对象温度	
SGP_GetImageTemps	获取温度矩阵	
SGP_GetTempMatrix	获取温度矩阵 (医疗机芯有效)	从设备端获取温度矩阵
SGP_SetTempPoints	设置测温点索引数组	
SGP_GetTempPoints	获取测温点温度数组	
SGP_GetMatrixTempPoints	获取指定矩形区域温度矩阵	
SGP_GetTempMatriRotation	Y16旋转	
SGP_GetTempMatriRotationEx	温度矩阵旋转	
SGP_GetY16	获取Y16	
SGP_StopY16	停止获取Y16	

SGP_GetTempMatrixEx	Y16输出 温度矩阵	
SGP_GetMeasureTempInfo	获取校 温信息	

版权所有©武汉高德

智感科技有限公司

获取单点温度SGP_GetPointTemp

选 项：	说明
描 述：	获取单点温度
详 细描 述：	
函 数：	<pre>int SGP_GetPointTemp(SGP_HANDLE handle, int x, int y, float *output);</pre>
参 数：	<pre>handle [in] 传入设备对象 x [in] 横坐标，范围在1到图像宽度之间 y</pre>

	<p>[in] 纵坐标，范围在1到图像高度之间</p> <p>output</p> <p>[out] 点温度</p>
返回值：	成功返回 SGP_OK ，失败返回 错误码
备注：	1、要确保传入的动态数组output对象不为空，否则可能会引发异常
使用示例：	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { int x = 100; int y = 100; float output=0.0f; int ret = SGP_GetPointTemp(handle,x,y,&output); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

获取分析对象温度SGP_GetAnalyticObjectsTemp

选项:	说明
描述:	获取分析对象温度
详细描述:	
函数:	<pre>int SGP_GetAnalyticObjectsTemp(SGP_HANDLE handle, SGP_ANALYTIC_TEMPS *output);</pre>
参数:	<pre>handle [in] 传入设备对象 output [out] 输出信息</pre>
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	结构体变量在使用前先初始化
使用示例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_ANALYTIC_TEMPS array;</pre>

	<pre> memset(&array, 0x00, sizeof(SGP_ANALYTIC_TEMPS)); int ret = SGP_GetAnalyticObjectsTemp(handle,&array); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>
--	--

版权所有©武汉高德红外股份有限

公司

获取温度矩阵SGP_GetImageTemps

选 项：	说明
描 述：	获取温度矩阵
详 细 描 述 ：	
函 数：	int SGP_GetImageTemps(SGP_HANDLE handle,

	<pre>float *output, int length, int type);</pre>
参 数:	<pre>handle [in] 传入设备对象 output [out] 输出温度矩阵 length [in] output大小 type [in] 0为推流红外分辨率，1为设备红外原始分辨率</pre>
返 回 值:	成功返回 SGP_OK ，失败返回 错误码
备 注:	1、温度矩阵值为float类型，4个字节长度，调用SGP_GetGeneralInfo函数获取红外模组宽和红外模组高，第三个参数length值传入红外模组宽*红外模组高*4
使 用 示 例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_GENERAL_INFO info; memset(&info, 0x00, sizeof(info)); int ret = SGP_GetGeneralInfo(handle,&info); if (ret == SGP_OK) { int heigth = info.ir_model_h; int width = info.ir_model_w; int length = heigth*width; int type = 1;</pre>

```
float *output = (float *)calloc(length,
sizeof(float));
if(output!=NULL)
{
    ret =
SGP_GetImageTemps(handle,output,length*4,type);
    if (ret == SGP_OK )
    {
        //成功 , TODO.....
    }
    else
    {
        //失败 , TODO.....
    }
}
free(output);
output=NULL;
}
```

版权所有©武汉高德红外股份有限

公司

获取温度矩阵 (医疗机芯有效-仅ZU13A支持)

SGP_GetTempMatrix

选项：	说明
描述：	获取温度矩阵 (医疗机芯有效)
详细描述：	医疗机芯有效
函数：	int SGP_GetTempMatrix(SGP_HANDLE handle,

	SGP_TEMPCALLBACK callback, void *pUser);
参数:	handle [in] 传入设备对象 callback [in] 注册温度矩阵回调函数 (Short数据 , 温度*100倍) pUser [in] 回调函数入参
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ static void TempCall(short *temp, int w, int h, void *pUser) { MainWindow *pDlg = (MainWindow *)pUser; //TODO..... } void MainWindow::Init() { int ret = SGP_GetTempMatrix(handle, TempCall, this); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>

	}
	}

回调函数描述		
函数名称	<pre>typedef void(*SGP_TEMPCALLBACK)(short *temp, int w, int h, void *pUser);</pre>	
功能描述	温度矩阵回调函数	
参数说明	temp	输出参数，温度矩阵数据
	w	输出参数，温度矩阵宽
	h	输出参数，温度矩阵高
	pUser	输出参数
返回值	无	

版权所有©武汉高德红外股份有限

公司

设置测温点索引数组 SGP_SetTempPoints

选项：	说明
描述：	设置测温点索引数组 (该接口当前仅ZU08D支持)
详细描述：	
函数：	<pre>int SGP_SetTempPoints (SGP_HANDLE handle, int *index , int length, int type);</pre>
参数：	<p>handle</p> <p> [in] 传入设备对象</p> <p>index</p> <p> [in] 传入测温点索引数组</p> <p>length</p> <p> [in] index大小</p> <p>type</p> <p> [in] 0为推流红外分辨率，1为设备红外原始分辨率</p>
返回值：	成功返回 SGP_OK ，失败返回 错误码
备注：	<p>1、传入测温点的索引（将二维图像矩阵转换为一维矩阵），测温索引数组大小即为测温点个数。调用此接口后，设备端会缓存此数组。</p> <p>2、实际传入的测温点不能超过当前设备的分辨率宽高乘积的最大值，否则，得到的数据也没有实际意义的。</p> <p>3、调用SGP_GetTempPoints即可返回对应索引的温度值，无需反复设置。</p> <p>4、要确保传入的测温点索引数组的长度和length的大小一致，否则可能会引发异常</p> <p>5、如果需要请求全图温度矩阵，建议调用SGP_GetImageTemps接口。</p>
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。</pre>

	<pre> **/ const int pointNum = 100; void Init() { //开辟长度为100的数组，并填充100个点的数组索引进行赋值 int *index = (int *)malloc(pointNum * sizeof(int)); for (int i = 0; i < pointNum; i++) { index[i] = i * 100; } int ret = SGP_SetTempPoints(m_handle, index, pointNum, 1); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } free(index); index =NULL; } </pre>

版权所有©武汉高德红外股份有限

公司

获取测温点温度数组 SGP_GetTempPoints

选项：	说明
描述：	获取测温点温度数组（该接口当前仅ZU08D支持）
详细描述：	
函数：	<pre>int SGP_GetTempPoints (SGP_HANDLE handle, float *output , int length, int type);</pre>
参数：	<p>handle</p> <p>[in] 传入设备对象</p> <p>output</p> <p>[out] 输出测温点数组对应的温度值数组</p> <p>length</p> <p>[in] output大小</p> <p>type</p> <p>[in] 0为推流红外分辨率，1为设备红外原始分辨率</p>
返回值：	成功返回 SGP_OK ，失败返回 错误码
备注：	<p>1、温度矩阵值为float类型，4个字节长度</p> <p>2、第三个参数length值传入SGP_SetTempPoints接口的索引数组长度</p> <p>3、要确保传入的测温点数组的长度和length的大小一致，否则可能会引发异常</p>
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ const int pointNum = 100; void Init() { //开辟长度为100的数组，并填充100个点的数组索引进行赋值</pre>

	<pre>int *index = (int *)malloc(pointNum * sizeof(int)); for (int i = 0; i < pointNum; i++) { index[i] = i * 100; } int ret = SGP_SetTempPoints(m_handle, index, pointNum, 1); if (ret == SGP_OK) { float *temp = (float *)malloc(pointNum * sizeof(float)); memset(temp, 0, pointNum * sizeof(float)); if(temp != NULL) { int ret = SGP_GetTempPoints(m_handle, temp, pointNum, 1); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } } free(temp); temp =NULL; } free(index); index = nullptr; }</pre>

获取指定矩形区域温度矩阵 SGP_GetMatrixTempPoints

选项：	说明
描述：	获取指定矩形区域温度矩阵（该接口当前仅ZU08D支持）
详细描述：	
函数：	<pre>int SGP_GetMatrixTempPoints(SGP_HANDLE handle, float *output , int length, const SGP_RECT &input);</pre>
参数：	<pre>handle [in] 传入设备对象 output [out] 输出测温点数组对应的温度值数组 length [in] output大小 input [in] 输入参数，指定的矩形区域</pre>
返回值：	成功返回 SGP_OK ，失败返回 错误码
备注：	1、要确保测温点数组对应数组的长度和length的大小一致，否则可能会引发异常
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { float *output = new float[100]; SGP_RECT rect; rect.x = 0; rect.y = 0; rect.w = 20;</pre>

	<pre>rect.h = 5; const int length = 100; int ret = SGP_GetMatrixTempPoints(m_handle, output, length, rect); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } delete[] output; output = nullptr; }</pre>

版权所有©武汉高德红外股份有限

公司

温度矩阵旋转 SGP_GetTempMatriRotation

选项：	说明
描述：	温度矩阵旋转

详细描述：	从设备端获取温度矩阵
函数：	<pre>int SGP_GetTempMatriRotation(SGP_HANDLE handle, short *dst, short *src, int w, int h, int rotation);</pre>
参数：	<p>handle [in] 传入设备对象</p> <p>dst [out] 输出旋转后的温度矩阵</p> <p>src [in] 输入需要旋转的温度矩阵</p> <p>w [in] 输入src的宽</p> <p>h [in] 输入src的高</p> <p>rotation [in] 0：旋转90，1：旋转180°，2：旋转270°</p>
返回值：	成功返回 SGP_OK ，失败返回 错误码
备注：	温度矩阵值为short类型，2个字节长度。

注 :	
使 用 示 例 :	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { //通过SGP_GetTempMatrix回调的温度矩阵传入接口 SGP_GetTempMatriRotation 获取旋转后的温度矩阵。 int rotation = 1; ret = SGP_GetTempMatriRotation(handle,dst,src,width,height,rotation); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>

公司

版权所有©武汉高德红外股份有限

温度矩阵旋转 SGP_GetTempMatriRotationEx

选	说明
---	----

项 :	
描 述 :	温度矩阵旋转
详 细 描 述 :	从设备端获取温度矩阵
函 数 :	<pre>int SGP_GetTempMatriRotationEx(SGP_HANDLE handle, float* dst, float* src, int w, int h, int rotation);</pre>
参 数 :	<p>handle [in] 传入设备对象</p> <p>dst [out] 输出旋转后的温度矩阵</p> <p>src [in] 输入需要旋转的温度矩阵</p> <p>w [in] 输入src的宽</p> <p>h [in] 输入src的高</p> <p>rotation [in] 0: 旋转90°, 1: 旋转180°, 2: 旋转270°</p>
返	成功返回 SGP_OK , 失败返回 错误码

返回值：	
备注：	温度矩阵值为short类型，2个字节长度。
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { //通过SGP_GetTempMatrix回调的温度矩阵传入接口 SGP_GetTempMatriRotationEx 获取旋转后的温度矩阵。 int rotation = 1; ret = SGP_GetTempMatriRotationEx(handle,dst,src,width,height,rotation); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

获取Y16 SGP_GetY16

选项：	说明
描述：	获取Y16数据
详细描述：	设备端传输Y16数据进行回调
函数：	<pre>int SGP_GetY16(SGP_HANDLE handle, SGP_Y16CALLBACK callback, void *pUser);</pre>
参数：	<pre>handle [in] 传入设备对象 callback [in] 回调函数地址 pUser [in] 回调函数入参</pre>
返回值：	成功返回 SGP_OK ，失败返回 错误码
备注：	1、要先打开视频流 2、单台机芯设备建议仅拉取一路Y16数据，否则会影响性能、帧率等
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 2, 以QT界面库为例 **/ static void GetY16Data(short *y16,int length, void *pUser) { MainWindow *pDlg = (MainWindow *)pUser;</pre>

	<pre>//TODO..... } void MainWindow::Init() { SGP_GetY16(handle, GetY16Data, this); //TODO..... }</pre>
--	--

回调函数描述		
函数名称	typedef void(*SGP_Y16CALLBACK)(short *y16, int length, void *pUser);	
功能描述	传输Y16数据的回调函数	
参数说明	y16	输出参数，Y16和参数行数据
参数说明	length	输出参数，Y16和参数行数据的字节数
参数说明	pUser	输出参数
返回值	无	

公司

停止获取Y16 SGP_StopY16

选项：	说明
描述：	停止获取Y16数据
详细描述：	设备端停止传输Y16数据
函数：	<code>int SGP_StopY16(SGP_HANDLE handle);</code>
参数：	handle [in] 传入设备对象
返回值：	无
备注：	
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 2, 以QT界面库为例 **/ void MainWindow::Init() { SGP_StopY16(handle); //TODO..... }</pre>

公司

Y16输出温度矩阵SGP_GetTempMatrixEx

选项:	说明
描述:	Y16输出温度矩阵
详细描述:	设备端传输Y16数据，由SDK调用测温库进行测温
函数数:	<pre>int SGP_GetTempMatrixEx(SGP_HANDLE handle, float*dst, short *src, int w, int h);</pre>
参数数:	<p>handle</p> <p>[in] 传入设备对象</p> <p>dst</p> <p>[out] 输出温度矩阵</p> <p>src</p> <p>[in] 输入需要的Y16数据</p> <p>w</p> <p>[in] 输入src的宽</p> <p>h</p> <p>[in] 输入src的高</p>
返回值:	成功返回 SGP_OK ，失败返回 错误码

备注:	温度矩阵值为short类型，2个字节长度。
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { //通过SGP_GetY16回调的Y16传入接口 SGP_GetTempMatrixEx 获取温度矩阵。 ret = SGP_GetTempMatrixEx(handle,dst,src,width,height); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

版权所有©武汉高德红外股份有限

公司

获取校温信息 SGP_GetMeasureTempInfo

选项:	说明
描述:	获取校温信息

详细描述：	
函数：	<pre>int SGP_GetMeasureTempInfo(SGP_HANDLE handle, SGP_MEASURE_TEMP_INFO &output);</pre>
参数：	<pre>handle [in] 传入设备对象 output [out] 输出校温信息</pre>
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { // 获取校温信息 SGP_MEASURE_TEMP_INFO output; ret = SGP_GetMeasureTempInfo(handle,output); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>

	}

版权所有©武汉高德红外股份有限

公司

拍照接口

接口描述	功能描述	功能详细描述
SGP_GetScreenCapture	获取屏幕截图	
SGP_GetScreenCaptureCache	获取屏幕截图 (非文件)	
SGP_GetHeatMap	获取热图	
SGP_GetHeatMapCache	获取热图 (非文件)	
SGP_GetFirHeatMap	获取高压热图	
SGP_GetFirHeatMapCache	获取高压热图 (非文件)	

版权所有©武汉高德

智感科技有限公司

获取屏幕截图SGP_GetScreenCapture

选项：	说明
描述：	获取屏幕截图
详细描述：	图片不带温度
函数：	<pre>int SGP_GetScreenCapture(SGP_HANDLE handle, SGP_IMAGE_TYPE type, const char *input);</pre>
参数：	<p>handle</p> <p>[in] 传入设备对象</p> <p>type</p> <p>[in] 图片类型，1可见光，2红外图片</p> <p>input</p> <p>[in] 保存文件路径+文件名+.jpg</p>
返回值：	成功返回 SGP_OK ，失败返回 错误码
备注：	
使用示例：	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { SGP_IMAGE_TYPE type = SGP_IR_IMAGE; char path[] = "./screenpic.jpg"; int ret = SGP_GetScreenCapture(handle,type ,path); if (ret == SGP_OK) { //成功，TODO..... } else</pre>

	<pre> { //失败，TODO..... } }</pre>
--	---

版权所有©武汉高德红外股份有限

公司

获取屏幕截图(非文件)SGP_GetScreenCaptureCache

选项:	说明
描述:	获取屏幕截图(非文件)
详细描述:	图片不带温度
函数数:	<pre>int SGP_GetScreenCaptureCache(SGP_HANDLE handle, SGP_IMAGE_TYPE type, char *input, int input_length, int *output_length);</pre>
参数数:	<pre>handle [in] 传入设备对象</pre>

	<p>type</p> <p>[in] 图片类型, 1可见光, 2红外图片</p> <p>input</p> <p>[in] 外部分配, 获取图片二进制数据</p> <p>input_length</p> <p>[in] input大小</p> <p>output_length</p> <p>[out] 图片二进制数据值大小</p>
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释: * 1, handle 设备对象。 */ void Init() { int input_length = 1024*1024*10; int output_length = 0; SGP_IMAGE_TYPE type = SGP_IR_IMAGE; char *input= (char *)calloc(input_length, sizeof(char)); if(input!=NULL) { int ret=SGP_GetScreenCaptureCache(handle,type,input,input_length, &output_length); if (ret == SGP_OK) { //成功, TODO..... } else { //失败, TODO..... } } }</pre>

	<pre>} free(input); input=NULL; }</pre>
--	---

公司

获取热图SGP_GetHeatMap

选项：	说明
描述：	获取热图
详细描述：	热图文件格式为国网格式
函数：	<code>int SGP_GetHeatMap(SGP_HANDLE handle, const char *input);</code>
参数：	handle [in] 传入设备对象 input [in] 保存文件路径+文件名+.jpg
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	获取的红外热图满足DLT 664-2016带电红外设备诊断应用规范对jpeg格式要求， 格式参考 表1

使用示例 :	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 */ void Init() { const char path[] = "./screenpic.jpg"; int ret = SGP_GetHeatMap (handle,path); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>
-----------	---

表1

版权所有©武汉高德红外股份有限	
公司	
获取热图(非文件)SGP_GetHeatMapCache	
选	说明

项 :	
描 述 :	获取热图(非文件)
详 细 描 述 :	热图文件格式为国网格式
函 数 :	<pre>int SGP_GetHeatMapCache(SGP_HANDLE handle, char *input, int input_length, int *output_length);</pre>
参 数 :	<pre>handle [in] 传入设备对象 input [in] 外部分配, 获取图片二进制数据 input_length [in] input大小 output_length [out] 图片二进制数据大小</pre>
返 回 值 :	成功返回 SGP_OK , 失败返回 错误码
备	获取的红外热图满足DLT 664-2016带电红外设备诊断应用规范对jpeg格式要求

注 :	格式参考 表1
使 用 示 例 :	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { int input_length = 1024*1024*10; int output_length = 0; char *input= (char *)calloc(input_length, sizeof(char)); if(input!=NULL) { int ret = SGP_GetHeatMapCache(handle,input,input_length,&output_length) ; if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } } free(input); input=NULL; }</pre>

获取高压热图SGP_GetFirHeatMap

选项：	说明
描述：	获取高压热图
详细描述：	
函数：	<code>int SGP_GetFirHeatMap(SGP_HANDLE handle, const char *input);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>input</code> [in] 保存文件路径+文件名+.fir
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	高压红外热图满足Q/GDW 12164-2021 变电站远程智能巡视系统技术规范中附录A 文件格式定义 A.1 红外图谱文件，格式参考 表1
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { const char path[] = "./screenpic.fir"; int ret = SGP_GetFirHeatMap(handle,path); if (ret == SGP_OK) { //成功 , TODO..... } else</pre>

	<pre>{ //失败，TODO..... }</pre>
--	---

表1

版权所有©武汉高德红外股份有限

公司

获取高压热图(非文件)SGP_GetFirHeatMapCache

选项:	说明
描述:	获取高压热图(非文件)
详细描述:	热图文件格式为高压国网格式
函数:	<pre>int SGP_GetFirHeatMapCache (SGP_HANDLE handle, char *input, int input_length, int *output_length);</pre>

参 数:	handle [in] 传入设备对象 input [in] 外部分配，获取图片二进制数据 input_length [in] input大小 output_length [out] 图片二进制数据大小
返 回 值:	成功返回 SGP_OK ，失败返回 错误码
备 注:	高压红外热图满足Q/GDW 12164-2021 变电站远程智能巡视系统技术规范中附录A 文件格式定义 A.1 红外图谱文件，格式参考 表1
使 用 示 例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { int input_length = 1024*1024*10; int output_length = 0; char *input= (char *)calloc(input_length, sizeof(char)); if(input!=NULL) { int ret = SGP_GetFirHeatMapCache (handle,input,input_length,&output_length); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } } }</pre>


```
    }  
    }  
    free(input);  
    input=NULL;  
}
```

版权所有©武汉高德红外股份有限

公司

成像接口

接口描述	功能描述	功能详细描述
SGP_OpenIrVideo	开启红外视频	
SGP_OpenVlVideo	开启可见光视频	
SGP_CloseIrVideo	关闭红外视频	
SGP_CloseVlVideo	关闭可见光视频	
SGP_GetIrImageInfo	获取成像参数	
SGP_SetIrImageInfo	设置成像参数	

版权所有©武汉高德

智感科技有限公司

开启红外视频SGP_OpenIrVideo

选项：	说明
描述：	开启红外视频
详细描述：	
函数：	<code>int SGP_OpenIrVideo(SGP_HANDLE handle, SGP_RTSPCALLBACK callback, void *pUser);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>callback</code> [in] 注册图像回调函数 (RGB24数据) <code>pUser</code> [in] void* 可传入窗口句柄指针
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1 , handle 设备对象。 **/ /** 示例中部分类、变量、函数的解释： 1 , handle 设备对象。 2 , 以QT界面库为例 **/ static void GetIrRtsp(unsigned char *outdata, int w, int h, void *pUser) {</pre>

	<pre>MainWindow *pDlg = (MainWindow *)pUser; //TODO..... } void MainWindow::Init() { SGP_OpenIrVideo(handle, GetIrRtsp, this); //TODO..... }</pre>
--	--

回调函数描述		
函数名称	typedef void(*SGP_RTSPCALLBACK)(unsigned char *outdata, int w, int h, void *pUser);	
功能描述	图像数据回调函数	
参数说明	outdata	输出参数 图像数据
	w	输出参数 图像宽度
	h	输出参数 图像高度
	pUser	输出参数
返回值	无	

公司

开启可见光视频SGP_OpenVlVideo

选项：	说明
描述：	开启可见光视频
详细描述：	
函数：	<pre>int SGP_OpenVlVideo(SGP_HANDLE handle, SGP_RTSPCALLBACK callback, void *pUser);</pre>
参数：	<pre>handle [in] 传入设备对象 callback [in] 注册图像回调函数 (RGB24数据) pUser [in] void* 可传入窗口句柄指针</pre>
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/</pre>

	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 2, 以QT界面库为例 **/ static void GetVIRtsp(unsigned char *outdata, int w, int h, void *ptr) { MainWindow *pDlg = (MainWindow *)ptr; //TODO..... } void MainWindow::Init() { SGP_OpenVlVideo(handle, GetVIRtsp, this); //TODO..... }</pre>
--	--

版权所有©武汉高德红外股份有限

公司

关闭红外视频SGP_CloseIrVideo

选项：	说明
描述：	关闭红外视频
详细描述：	

函数：	<code>void SGP_CloseIrVideo(SGP_HANDLE handle);</code>
参数：	<code>handle</code> [in] 传入设备对象
返回值：	无
备注：	退出登录会自动关闭视频流
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_CloseIrVideo(handle); //TODO..... }</pre>

版权所有©武汉高德红外股份有限

公司

关闭可见光视频SGP_CloseVlVideo

选项：	说明
描述：	关闭可见光视频
详细描述：	

函数：	<code>void SGP_CloseVlVideo (SGP_HANDLE handle);</code>
参数：	<code>handle</code> [in] 传入设备对象
返回值：	无
备注：	退出登录会自动关闭视频流
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_CloseVlVideo (handle); //TODO..... }</pre>

版权所有©武汉高德红外股份有限

公司

获取成像参数 SGP_GetIrImageInfo

选项：	说明
描述：	获取成像参数
详细描述：	

函数:	<code>int SGP_GetIrImageInfo(SGP_HANDLE handle, SGP_IR_IMAGE_INFO * output);</code>
参数:	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 传出降噪参数信息
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	
使用示例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ /** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 2, 以QT界面库为例 **/ void MainWindow::Init() { SGP_IR_IMAGE_INFO output; SGP_GetIrImageInfo(handle, &output); //TODO..... }</pre>

版权所有©武汉高德红外股份有限

公司

设置成像参数 SGP_SetIrImageInfo

选项：	说明
描述：	设置成像参数
详细描述：	
函数：	<pre>int SGP_SetIrImageInfo(SGP_HANDLE handle, SGP_IR_IMAGE_INFO_ENUM input, int value);</pre>
参数：	<p>handle</p> <p>[in] 传入设备对象</p> <p>input</p> <p>[in] 传入成像参数枚举类型</p> <p>value</p> <p>[in] 传入参数值</p>
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ /** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 2, 以QT界面库为例 **/ void MainWindow::Init() { SGP_SetIrImageInfo(handle, SGP_3D_FLAG, 4); //TODO..... }</pre>

--	--

版权所有©武汉高德红外股份有限

公司

录像接口

接口描述	功能描述	功能详细描述
SGP_Record	录制视频	控制设备端录像，文件录制在红外设备上
SGP_StartRecord	开始录制	文件录制在本地电脑
SGP_StopRecord	停止录制	文件录制在本地电脑

版权所有©武汉高德

智感科技有限公司

录制视频SGP_Record

选项：	说明
描述：	录制视频
详细描述	控制设备端录像，录制的视频文件存放在红外设备中

述 :	
函 数:	<pre>int SGP_Record(SGP_HANDLE handle, int subtype, int record_stream);</pre>
参 数:	<pre>handle [in] 传入设备对象 subtype [in] 1:开始录制 2:停止录制 record_stream [in] 1:单光可见光; 2:单光红外; 3:双光, 同 时录制</pre>
返 回 值:	成功返回 SGP_OK , 失败返回 错误码
备 注:	
使 用 示 例:	<pre>/** 示例中部分类、变量、函数的解释: 1, handle 设备对象。 **/ void Init() { int record = 1; int record_stream = 2; int ret = SGP_Record(handle, subtype, record_stream); if (ret == SGP_OK) { //成功, TODO..... } else { //失败, TODO..... } }</pre>

	}
}	

版权所有©武汉高德红外股份有限

公司

开始录制SGP_StartRecord

选项:	说明
描述:	开始录制
详细描述:	录制文件存放在本地
函数:	<pre>int SGP_StartRecord(SGP_HANDLE handle, SGP_VIDEO_TYPE type, const char *input, SGP_RECORDCALLBACK callback, void *pUser);</pre>
参数:	<pre>handle [in] 传入设备对象 type [in] 录像类型,1可见光录像,2红外录像 input</pre>

	<p>[in] 保存文件路径+文件名+.mp4</p> <p>callback</p> <p>[in] 录像状态回调函数，例如自动停止录像</p> <p>pUser</p> <p>[in] 回调函数入参</p>
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	与 SGP_StopRecord 成对使用，如果是红外录像，需要先调用接口 SGP_OpenIrVideo 打开红外视频，才能录像，如果是可见光本地录像，需要先调用接口 SGP_OpenVlVideo 打开可见光视频，才能进行可见光本地录像。
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ static void RecordCall(int state, void *pUser) { MainWindow *pDlg = (MainWindow *)pUser; //TODO..... } static void GetIrRtsp(unsigned char *outdata, int w, int h, void *pUser) { MainWindow *pDlg = (MainWindow *)pUser; //TODO..... } void Init() { SGP_VIDEO_TYPE type = SGP_IR_VIDEO; const char path[] = "./record.mp4";</pre>

```
int ret = SGP_OpenIrVideo(handle, GetIrRtsp,
this);
if (ret == SGP_OK )
{
    Sleep(5000); //打开视频后，间隔5s后开始录制视频
    int ret =
SGP_StartRecord(handle,type,path,RecordCall,this);
    if (ret == SGP_OK )
    {
        //成功，TODO.....
    }
    else
    {
        //失败，TODO.....
    }
    SGP_StopRecord(handle,type);
}
SGP_CloseIrVideo(handle);
}
```

回调函数描述		
函数名称	typedef void(*SGP_RECORDCALLBACK)(int state, void *pUser);	
功能描述	本地录像回调函数	
参数说明	state	输出参数,1：开始录制 2：录 制中 3：停止录制
	pUser	输出参数

返回值	无	
-----	---	--

版权所有©武汉高德红外股份有限

公司

停止录制SGP_StopRecord

选项:	说明
描述:	停止录制
详细描述:	停止本地录制
函数:	<code>void SGP_StopRecord(SGP_HANDLE handle, SGP_VIDEO_TYPE type);</code>
参数:	<code>handle</code> [in] 传入设备对象 <code>type</code> [in] 录像类型,1可见光录像,2红外录像

返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	与 SGP_StartRecord 成对使用
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ static void RecordCall(int state, void *pUser) { MainWindow *pDlg = (MainWindow *)pUser; //TODO..... } static void GetIrRtsp(unsigned char *outdata, int w, int h, void *pUser) { MainWindow *pDlg = (MainWindow *)pUser; //TODO..... } void Init() { SGP_VIDEO_TYPE type = SGP_IR_VIDEO; const char path[] = "./record.mp4"; int ret = SGP_OpenIrVideo(handle, GetIrRtsp, this); if (ret == SGP_OK) { sleep(5000); //打开视频后，间隔5s后开始录制视频 int ret = SGP_StartRecord(handle, type, path, RecordCall, this); if (ret == SGP_OK) { //成功，TODO..... } else {</pre>


```
        //失败，TODO.....
    }
    SGP_StopRecord(handle,type);
}
SGP_CloseIrVideo(handle);
}
```

公司

版权所有©武汉高德红外股份有限

操控设备接口

接口描述	功能描述	功能详细描述
SGP_SynchroTime	同步系统时间	
SGP_RebootSystem	系统重启	
SGP_ClearData	清理数据	
SGP_DoShutter	快门操作	
SGP_SetThermometryFlag	设置全局测温开关	
SGP_GetThermometryParam	获取全局测温参数	
SGP_SetThermometryParam	设置全局测温参数	
SGP_SetColorBar	设置色带号	
SGP_SetColorBarShow	设置色带显示	

SGP_SetTempShowMode	设置温度显示类型	
SGP_SetFocus	调焦	
SGP_GetMotorPosition	获取电机位置	
SGP_SetRange	切换测温范围	
SGP_SetStringShow	设置字符串叠加	
SGP_GetThermometryRule	获取分析对象	
SGP_AddThermometryRule	添加分析对象	
SGP_UpdateThermometryRule	更新分析对象	
SGP_DeleteThermometryRule	删除分析对象	
SGP_DeleteAllThermometryRule	删除全部分析对象	
SGP_SetThermometryRuleShowMode	设置分析对象温度显示类型	
SGP_GetIrImageEffectParam	获取红外图像效果参数	
SGP_SetIrImageEffectParam	设置红外图像效果参数	
SGP_GetVlImageEffectParam	获取可见光图像效果参数	
SGP_SetVlImageEffectParam	设置可见光图像效果参数	
SGP_GetImageFusion	获取图像融合	
SGP_SetImageFusion	设置图像融合	
SGP_GetNetInfo	获取网络信息	
SGP_SetNetInfo	设置网络信息	
SGP_GetPortInfo	获取端口信息	
SGP_SetPortInfo	设置端口信息	

SGP_GetShieldArea	获取屏蔽区域	
SGP_SetShieldArea	设置屏蔽区域	
SGP_GetColdHotTrace	获取全局温度告警	
SGP_SetColdHotTrace	设置全局温度告警	
SGP_GetTempAlarm	获取分析对象告警	
SGP_SetTempAlarm	设置分析对象告警	
SGP_GetVideoParam	获取视频参数	
SGP_SetVideoParam	设置视频参数	
SGP_GetVersionInfo	获取系统版本信息	
SGP_GetNetException	获取网络异常	
SGP_SetNetException	设置网络异常	
SGP_GetAccessViolation	获取非法访问	
SGP_SetAccessViolation	设置非法访问	
SGP_GetEmilInfo	获取邮件信息	
SGP_SetEmilInfo	设置邮件信息	
SGP_GetFillLight	获取补光灯信息	
SGP_SetFillLight	设置补光灯信息	
SGP_GetInfraredMode	获取融合状态	
SGP_SetInfraredMode	设置融合状态	
SGP_GetSilentMode	获取蜂鸣器状态	
SGP_SetSilentMode	设置蜂鸣器状态	
SGP_GetRecordInfo	获取录制信息	
SGP_SetRecordInfo	设置录制信息	

SGP_SetElectronicMagnification	设置电子变倍	
SGP_GetAlarmInput	获取报警输入	
SGP_SetAlarmInput	设置报警输入	
SGP_FactoryReset	恢复出厂设置	
SGP_CommandSend	透传RS485数据 查询	

版权所有©武汉高德

智感科技有限公司

同步系统时间SGP_SynchroTime

选项：	说明
描述：	同步系统时间
详细描述：	
函数：	<pre>int SGP_SynchroTime(SGP_HANDLE handle, const char *datetime);</pre>
参数：	<p>handle [in] 传入设备对象</p> <p>datetime [in] 同步时间,时间格式为"2020-05-21 12:22:33"</p>

返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	红外设备自身不带电池，此函数实现将电脑时间同步到红外设备上
使用示例：	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { const char* datetime = "2022-03-18 12:22:33"; int ret = SGP_SynchroTime(handle,datetime); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

版权所有©武汉高德红外股份有限

公司

系统重启SGP_RebootSystem

选项：	说明
-----	----

描述：	系统重启
详细描述：	
函数：	<code>int SGP_RebootSystem(SGP_HANDLE handle);</code>
参数：	handle [in] 传入设备对象
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { int ret = SGP_RebootSystem(handle); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

清理数据SGP_ClearData

选项：	说明
描述：	清理数据
详细描述：	用于清理红外设备缓存数据
函数：	<code>int SGP_ClearData(SGP_HANDLE handle);</code>
参数：	handle [in] 传入设备对象
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { int ret = SGP_ClearData(handle); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

公司

快门操作SGP_DoShutter

选项：	说明
描述：	快门操作
详细描述：	
函数：	<code>int SGP_DoShutter(SGP_HANDLE handle, SGP_SHUTTER_ENUM type);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>type</code> [in] 快门操作类型
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_SHUTTER_ENUM type = SGP_SHUTTER; int ret = SGP_DoShutter(handle,type); if (ret == SGP_OK) {</pre>

	<pre> //成功，TODO..... } else { //失败，TODO..... } }</pre>
--	---

版权所有©武汉高德红外股份有限

公司

设置全局测温开关SGP_SetThermometryFlag

选项 :	说明
描述 :	设置全局测温开关
详细描述 :	
函数 :	<pre>int SGP_SetThermometryFlag(SGP_HANDLE handle, int input);</pre>
参数 :	<pre>handle [in] 传入设备对象 input [in] 0关闭 1开启</pre>
返回值 :	成功返回 SGP_OK ，失败返回 错误码

备注：	
使用示例：	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { int flag = 0; int ret = SGP_SetThermometryFlag(handle,flag); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

版权所有©武汉高德红外股份有限

公司

获取全局测温参数SGP_GetThermometryParam

选项：	说明
描述：	获取全局测温参数

详细描述:	
函数:	<code>int SGP_GetThermometryParam(SGP_HANDLE handle, SGP_THERMOMETRY_PARAM *output);</code>
参数:	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 输出信息
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	结构体变量在使用前先初始化
使用示例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_THERMOMETRY_PARAM info; memset(&info, 0x00, sizeof(SGP_THERMOMETRY_PARAM)); int ret = SGP_GetThermometryParam(handle,&info); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>

公司

设置全局测温参数SGP_SetThermometryParam

选项：	说明
描述：	设置全局测温参数
详细描述：	
函数：	<code>int SGP_SetThermometryParam(SGP_HANDLE handle, SGP_THERMOMETRY_PARAM input);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>input</code> [in] 输入信息
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	使用前先调用 SGP_GetThermometryParam 获取全局测温参数，然后再修改测温参数， 结构体变量在使用前先初始化

使 用示 例：	<pre> /** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 */ void Init() { //先获取全局测温参数，再设置参数。 SGP_THERMOMETRY_PARAM info; memset(&info, 0x00, sizeof(SGP_THERMOMETRY_PARAM)); int ret = SGP_GetThermometryParam(handle,&info); if (ret == SGP_OK) { parm.dist = 5; //修改测温距离为5 米 ret = SGP_SetThermometryParam(handle,info); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } } } </pre>
---------------	--

设置色带号SGP_SetColorBar

选项：	说明
描述：	设置色带号
详细描述：	
函数：	<code>int SGP_SetColorBar(SGP_HANDLE handle, int input);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>input</code> [in] 色带号1~26
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { int colorbar = 2; int ret = SGP_SetColorBar(handle,colorbar); if (ret == SGP_OK) { //success, TODO..... } else { //fail, TODO } }</pre>

--	--

版权所有©武汉高德红外股份有限

公司

设置色带显示SGP_SetColorBarShow

选项:	说明
描述:	设置色带显示
详细描述:	
函数:	<pre>int SGP_SetColorBarShow(SGP_HANDLE handle, int input);</pre>
参数:	<pre>handle [in] 传入设备对象 input [in] 0关闭 1开启</pre>
返回值:	成功返回 SGP_OK , 失败返回 错误码

备注:	
使用示例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 */ void Init() { int showcolorbar = 1; int ret = SGP_SetColorBarShow(handle,showcolorbar); if (ret == SGP_OK) { //success, TODO..... } else { //fail, TODO } }</pre>

版权所有©武汉高德红外股份有限

公司

设置温度显示类型SGP_SetTempShowMode

选项:	说明
描述:	设置温度显示类型

详细描述：	
函数：	<pre>int SGP_SetTempShowMode(SGP_HANDLE handle, int input);</pre>
参数：	<p>handle</p> <p>[in] 传入设备对象</p> <p>input</p> <p>[in] 温度显示方式：1 最高温 2 最低温 3 平均温 4 最高温 + 最低温 5 最高温 + 平均温 6 平均温 + 最低温 7 最高温 + 最低温 + 平均温 8不显示</p>
返回值：	成功返回 SGP_OK ，失败返回 错误码
备注：	
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { int showtype = 1; int ret = SGP_SetTempShowMode(handle, showtype); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

	}
--	---

版权所有©武汉高德红外股份有限

公司

调焦SGP_SetFocus

选项：	说明
描述：	调焦
详细描述：	
函数：	<pre>int SGP_SetFocus(SGP_HANDLE handle, SGP_FOCUS_TYPE type, int value);</pre>
参数：	<p>handle</p> <p> [in] 传入设备对象</p> <p>type</p> <p> [in] 操作类型</p> <p>value</p> <p> [in] 电机位置值0~750，当type传入SGP_FOCUS_PLACE有效</p>
返回值：	成功返回 SGP_OK ，失败返回 错误码
备注：	电机位置值范围0~750

使用 示例：	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { SGP_FOCUS_TYPE type = SGP_FOCUS_AUTO; int value =0; int ret = SGP_SetFocus(handle,type,value); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>
-----------	---

版权所有©武汉高德红外股份有限

公司

获取电机位置SGP_GetMotorPosition

选项 :	说明
描述 :	获取电机位置

详细 描述 :	
函数 :	<code>int SGP_GetMotorPosition(SGP_HANDLE handle, int *output);</code>
参数 :	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 电机位置
返回 值:	成功返回 SGP_OK , 失败返回 错误码
备注 :	
使用 示例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { int value =0; int ret = SGP_GetMotorPosition(handle,&value); if (ret == SGP_OK) { //成功, TODO..... } else { //失败, TODO..... } }</pre>

公司

切换测温范围SGP_SetRange

选项：	说明
描述：	切换测温范围
详细描述：	
函数：	<code>int SGP_SetRange(SGP_HANDLE handle, int input);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>input</code> [in] 0~2，部分设备只有1个档位，目前最多有3个档位
返回值：	成功返回 SGP_OK ，失败返回 错误码
备注：	调用 SGP_GetGeneralInfo 获取设备支持的档位， 例如对IPT640M，0表示-20℃~150℃,1表示100℃~350℃， 2表示100℃~550℃（如果设备支持则包含2）
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_GENERAL_INFO info;</pre>

	<pre> memset(&info, 0x00, sizeof(SGP_GENERAL_INFO)); int ret = SGP_GetGeneralInfo(handle,&info); if (ret ==SGP_OK) { int range = info.range_num; ret = SGP_SetRange(handle,range -1); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } } }</pre>
--	--

设置字符串叠加SGP_SetStringShow

选项：	说明
描述：	设置字符串叠加
详细 描述：	
函数：	int SGP_SetStringShow(SGP_HANDLE handle, int type,

	<code>const char *input);</code>
参数:	<div>handle</div> <div>[in] 传入设备对象</div> <div>type</div> <div>[in] 是否使用字符叠加 其他机型1:关闭; 2, 4, 5:右下; 3:右上</div> <div>IPT640M 1:关闭;</div> <div>2:左上; 3:右上; 4:左下; 5:右下</div> <div>IPM630 1:关闭;</div> <div>5:右下</div> <div>IPT430M 1:关闭;</div> <div>5:右下</div> <div>input</div> <div>[in] 需要显示的字符串</div>
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { int type = 3; const char* dateshow= "1号设备终端"; int ret = SGP_SetStringShow(handle,type ,dateshow); if (ret == SGP_OK) { //成功，TODO..... } }</pre>

	<pre>else { //失败，TODO..... } }</pre>
--	--

版权所有©武汉高德红外股份有限

公司

获取分析对象SGP_GetThermometryRule

选项：	说明
描述：	获取分析对象
详细描述：	
函数：	<pre>int SGP_GetThermometryRule(SGP_HANDLE handle, SGP_RULE_ARRAY *output);</pre>
参数：	<pre>handle [in] 传入设备对象 output [out] 全部分析对象</pre>
返回值：	成功返回 SGP_OK ，失败返回 错误码

备注:	结构体变量在使用前先初始化
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { SGP_RULE_ARRAY array; memset(&array, 0x00, sizeof(SGP_RULE_ARRAY)); int ret = SGP_GetThermometryRule(handle,&array); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>

版权所有©武汉高德红外股份有限

公司

添加分析对象SGP_AddThermometryRule

选项:	说明
-----	----

描述:	添加分析对象
详细描述:	
函数:	<code>int SGP_AddThermometryRule(SGP_HANDLE handle, SGP_RULE input);</code>
参数:	<code>handle</code> [in] 传入设备对象 <code>input</code> [in] 分析对象类型
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	结构体变量在使用前先初始化
使用示例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_RULE rulePoint; memset(&rule, 0x00, sizeof(SGP_RULE)); rulePoint.alarm_condition = 1; rulePoint.alarm_flag = 1; rulePoint.alarm_time = 30; rulePoint.alarm_type = 1; rulePoint.avg_temp = 30; rulePoint.flag = 1; rulePoint.high_temp = 35;</pre>

```

rulePoint.low_temp = 28;
rulePoint.points_num = 1; //点个数是1
rulePoint.points[0].x = 200;
rulePoint.points[0].y = 200;
strcpy(rulePoint.rule_name, "点1");
rulePoint.show_location = 1;
rulePoint.temp_mod = 1;

rulePoint.type = 1; //类型是1
rulePoint.atmo_trans = 0.9;
rulePoint.dist = 2;
rulePoint.emiss = 0.8;
rulePoint.emiss_mode = 1;
rulePoint.humi = 80;
rulePoint.opti_trans = 1;
rulePoint.ref_temp = 25;

int ret =
SGP_AddThermometryRule(handle,rulePoint);
if (ret == SGP_OK )
{
    //成功，TODO.....
}
else
{
    //失败，TODO.....
}
}

```

版权所有©武汉高德红外股份有限

公司

更新分析对象SGP_UpdateThermometryRule

选项：	说明
描述：	更新分析对象
详细描述：	
函数：	<pre>int SGP_UpdateThermometryRule(SGP_HANDLE handle, SGP_RULE input);</pre>
参数：	<pre>handle [in] 传入设备对象 input [in] 分析对象</pre>
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	先调用 SGP_GetThermometryRule 函数获取,再更新, 结构体变量在使用前先初始化

使用示例：

```
/**
 示例中部分类、变量、函数的解释：
 1, handle 设备对象。
 **/
void Init()
{
    SGP_RULE_ARRAY array;
    memset(&array, 0x00, sizeof(SGP_RULE_ARRAY));
    int ret = SGP_GetThermometryRule(handle, &array);
    if (ret == SGP_OK )
    {
        if(array.rule_num>0)
        {
            SGP_RULE rule;
            memset(&rule, 0, sizeof(SGP_RULE));
            memcpy(&rule, &array.rule[0], sizeof(SGP_RULE)); //取第一组分析对象值

            rule.alarm_condition = 1;
            rule.alarm_flag = 1;
            rule.alarm_time = 10;
            int ret =
SGP_UpdateThermometryRule(handle, rule);
            if (ret == SGP_OK )
            {
                //成功，TODO.....
            }
            else
            {
                //失败，TODO.....
            }
        }
        //成功，TODO.....
    }
    else
    {
        //失败，TODO.....
    }
}
```

公司

删除分析对象SGP_DeleteThermometryRule

选项:	说明
描述:	删除分析对象
详细描述:	
函数:	<code>int SGP_DeleteThermometryRule(SGP_HANDLE handle, int input);</code>
参数:	<code>handle</code> [in] 传入设备对象 <code>input</code> [in] 分析对象id
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	
使用:	<code>/** 示例中部分类、变量、函数的解释:</code>

示 例:	<pre> 1,handle 设备对象。 **/ void Init() { SGP_RULE_ARRAY array; memset(&array, 0x00, sizeof(SGP_RULE_ARRAY)); int ret = SGP_GetThermometryRule(handle,&array); if (ret == SGP_OK) { if(array.rule_num>0) { int ret = SGP_DeleteThermometryRule(handle,array.rule[0].id); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } } //成功 , TODO..... } else { //失败 , TODO..... } } </pre>
---------	--

删除全部分析对象SGP_DeleteAllThermometryRule

选项:	说明
描述:	删除全部分析对象
详细描述:	
函数:	<code>int SGP_DeleteAllThermometryRule(SGP_HANDLE handle);</code>
参数:	handle [in] 传入设备对象
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	
使用示例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { int ret = SGP_DeleteAllThermometryRule(handle); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

	}
}	

版权所有©武汉高德红外股份有限

公司

设置分析对象温度显示类型

SGP_SetThermometryRuleShowMode

选项:	说明
描述:	设置分析对象温度显示类型
详细描述:	
函数:	<code>int SGP_SetThermometryRuleShowMode(SGP_HANDLE handle, int input);</code>
参数:	<code>handle</code> [in] 传入设备对象 <code>input</code> [in] 对象温度显示:1最高温;2最低温;3平均温;4仅名称;5不显示

返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	前提是有分析对象，设置分析对象温度显示类型才会生效。
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { int showtype = 1; int ret = SGP_SetThermometryRuleShowMode(handle, showtype); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

获取红外图像效果参数SGP_GetIrImageEffectParam

选项:	说明
描述:	获取红外图像效果参数
详细描述:	
函数:	<pre>int SGP_GetIrImageEffectParam(SGP_HANDLE handle, SGP_IAMGE_EFFECT_PARAM_IR_CONFIG *output);</pre>
参数:	<pre>handle [in] 传入设备对象 output [out] 输出信息</pre>
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	结构体变量在使用前先初始化
使用示例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_IAMGE_EFFECT_PARAM_IR_CONFIG info; memset(&info, 0x00, sizeof(info)); int ret = SGP_GetIrImageEffectParam(handle,&info); if (ret == SGP_OK) {</pre>

```
        //成功 , TODO.....
    }
    else
    {
        //失败 , TODO.....
    }
}
```

版权所有©武汉高德红外股份有限

公司

设置红外图像效果参数SGP_SetIrImageEffectParam

选项:	说明
描述:	设置红外图像效果参数
详细描述:	
函数数:	int SGP_SetIrImageEffectParam(SGP_HANDLE handle, SGP_IR_IMAGE_EFFECT_ENUM type, int value);
参数数:	handle [in] 传入设备对象

	<div>type</div> <div>[in] 参数类型</div> <div>value</div> <div>[in] 参数值</div>
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { SGP_IR_IMAGE_EFFECT_ENUM type = SGP_IR_ROTATE; int value = 1; int ret = SGP_SetIrImageEffectParam(handle, type, value); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

获取可见光图像效果参数SGP_GetVlImageEffectParam

选项:	说明
描述:	获取可见光图像效果参数
详细描述:	此函数适用于带可见光的双光设备
函数:	<pre>int SGP_GetVlImageEffectParam(SGP_HANDLE handle, SGP_IAMGE_EFFECT_PARAM_VL_CONFIG *output);</pre>
参数:	<pre>handle [in] 传入设备对象 output [out] 输出值</pre>
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	结构体变量在使用前先初始化
使用示例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_IAMGE_EFFECT_PARAM_VL_CONFIG info;</pre>

```
        memset(&info, 0x00,
sizeof(SGP_IAMGE_EFFECT_PARAM_VL_CONFIG));
        int ret =
SGP_GetVlImageEffectParam(handle,&info);
        if (ret == SGP_OK )
        {
                //成功 , TODO.....
        }
        else
        {
                //失败 , TODO.....
        }
}
```

版权所有©武汉高德红外股份有限

公司

设置可见光图像效果参数SGP_SetVlImageEffectParam

选 项：	说明
描 述：	设置可见光图像效果参数
详 细 描 述 ：	此函数适用于带可见光的双光设备
函 数：	int SGP_SetVlImageEffectParam(SGP_HANDLE handle,

	<code>SGP_VL_IMAGE_EFFECT_ENUM type, int value);</code>
参 数:	<code>handle</code> [in] 传入设备对象 <code>type</code> [in] 参数类型 <code>value</code> [in] 参数值
返 回 值:	成功返回 SGP_OK , 失败返回 错误码
备 注:	
使 用 示 例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { SGP_VL_IMAGE_EFFECT_ENUM type = SGP_VL_BRIGHTNESS; int value = 50; int ret = SGP_SetVlImageEffectParam(handle,type,value); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>

公司

获取图像融合SGP_GetImageFusion

选项：	说明
描述：	获取图像融合
详细描述：	此函数适用于带可见光的双光设备
函数：	<code>int SGP_GetImageFusion(SGP_HANDLE handle, SGP_IMAGE_FUSION *output);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 输出信息
返回值：	成功返回SGP_OK, 失败返回 错误码
备注：	结构体变量在使用前先初始化
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_IMAGE_FUSION info; memset(&info, 0x00, sizeof(SGP_IMAGE_FUSION));</pre>

	<pre>int ret = SGP_GetImageFusion(handle,&info); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>
--	---

版权所有©武汉高德红外股份有限

公司

设置图像融合SGP_SetImageFusion

选项：	说明
描述：	设置图像融合
详细 描述：	此函数适用于带可见光的双光设备
函数：	<code>int SGP_SetImageFusion(SGP_HANDLE handle, SGP_IMAGE_FUSION input);</code>
参数：	<div>handle</div> <div>[in] 传入设备对象</div> <div>input</div> <div>[in] 输入信息</div>

返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	先调用 SGP_GetImageFusion 函数获取,再设置 , 结构体变量在使用前先初始化
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { SGP_IMAGE_FUSION info; memset(&info, 0x00, sizeof(SGP_IMAGE_FUSION)); int ret = SGP_GetImageFusion(handle,&info); if (ret == SGP_OK) { info.percent = 50; ret = SGP_SetImageFusion(handle,info); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } } else { //失败 , TODO..... } }</pre>

公司

获取网络信息SGP_GetNetInfo

选项：	说明
描述：	获取网络信息
详细描述：	
函数：	<code>int SGP_GetNetInfo(SGP_HANDLE handle, SGP_NET_INFO *output);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 输出信息
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	结构体变量在定义后先初始化
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_NET_INFO info; memset(&info, 0x00, sizeof(SGP_NET_INFO)); int ret = SGP_GetNetInfo(handle,&info);</pre>

	<pre>if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>
--	---

版权所有©武汉高德红外股份有限

公司

设置网络信息SGP_SetNetInfo

选项：	说明
描述：	设置网络信息
详细描述：	
函数：	<code>int SGP_SetNetInfo(SGP_HANDLE handle, SGP_NET_INFO input);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>input</code> [in] 输入信息
返回值：	成功返回 SGP_OK , 失败返回 错误码

备注：	先调用 SGP_GetNetInfo 函数,再设置，结构体变量在定义后需要先初始化
使用示例：	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { SGP_NET_INFO info; memset(&info, 0x00, sizeof(SGP_NET_INFO)); int ret = SGP_GetNetInfo(handle,&info); if (ret == SGP_OK) { info.card = 1; ret = SGP_SetNetInfo(handle,info); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } } else { //失败，TODO..... } }</pre>

公司

获取端口信息SGP_GetPortInfo

选项：	说明
描述：	获取端口信息
详细描述：	
函数：	<code>int SGP_GetPortInfo(SGP_HANDLE handle, SGP_PORT_INFO *output);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 输出信息
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	结构体变量在使用前先初始化
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_PORT_INFO info; memset(&info, 0x00, sizeof(SGP_PORT_INFO)); int ret = SGP_GetPortInfo(handle,&info); if (ret == SGP_OK) {</pre>

	<pre> //成功 , TODO..... } else { //失败 , TODO..... } }</pre>
--	---

版权所有©武汉高德红外股份有限

公司

设置端口信息SGP_SetPortInfo

选项：	说明
描述：	设置端口信息
详细描述：	
函数：	<code>int SGP_SetPortInfo(SGP_HANDLE handle, SGP_PORT_INFO input);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>input</code> [in] 输入信息
返回值：	成功返回 SGP_OK , 失败返回 错误码

备注：	先调用 SGP_GetPortInfo 函数再设置，结构体变量在使用前先初始化
使用示例：	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { SGP_PORT_INFO info; memset(&info, 0x00, sizeof(SGP_PORT_INFO)); int ret = SGP_GetPortInfo(handle,&info); if (ret == SGP_OK) { info.max_connectios = 10; ret = SGP_SetPortInfo(handle,info); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } } else { //失败，TODO..... } }</pre>

获取屏蔽区域SGP_GetShieldArea

选项：	说明
描述：	获取屏蔽区域
详细描述：	
函数：	<code>int SGP_GetShieldArea(SGP_HANDLE handle, SGP_SHIELD_AREA_INFO *output);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 输出信息
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	结构体变量在使用前先初始化
使用示例：	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { SGP_SHIELD_AREA_INFO info; memset(&info, 0x00, sizeof(SGP_SHIELD_AREA_INFO)); int ret = SGP_GetShieldArea(handle,&info); if (ret == SGP_OK) { //成功 , TODO..... } }</pre>

	<pre> } else { // 失败，TODO..... } }</pre>
--	---

版权所有©武汉高德红外股份有限

公司

设置屏蔽区域SGP_SetShieldArea

选项：	说明
描述：	设置屏蔽区域
详细描述：	
函数：	<code>int SGP_SetShieldArea(SGP_HANDLE handle, SGP_SHIELD_AREA_INFO input);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>input</code> [in] 输入信息
返回值：	成功返回 SGP_OK ，失败返回 错误码
备注：	先调用 SGP_GetShieldArea 函数，再设置，结构体变量在使用前先初始化
使用示例：	<code>/**</code>

示例中部分类、变量、函数的解释：

1, handle 设备对象。

```
*/  
void Init()  
{  
    SGP_SHIELD_AREA_INFO info;  
    memset(&info, 0x00,  
sizeof(SGP_SHIELD_AREA_INFO));  
    int ret = SGP_GetShieldArea(handle,&info);  
    if (ret == SGP_OK )  
    {  
        info.rect_num = 2;  
  
        info.rect[0].x = 100;  
        info.rect[0].y = 100;  
        info.rect[0].w = 50;  
        info.rect[0].h = 50;  
        info.rect[1].x = 200;  
        info.rect[1].y = 200;  
        info.rect[1].w = 50;  
        info.rect[1].h = 50;  
  
        ret = SGP_SetShieldArea(handle,info);  
        if (ret == SGP_OK )  
        {  
            //成功 , TODO.....  
        }  
        else  
        {  
            //失败 , TODO.....  
        }  
    }  
    else  
    {  
        //失败 , TODO.....  
    }  
}
```

公司

获取全局温度告警SGP_GetColdHotTrace

选项 :	说明
描述 :	获取全局温度告警
详细 描述 :	
函数 :	<code>int SGP_GetColdHotTrace(SGP_HANDLE handle, SGP_COLD_HOT_TRACE_INFO *output);</code>
参数 :	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 输出信息
返回 值:	成功返回 SGP_OK , 失败返回 错误码
备注 :	结构体变量在使用前先初始化
使用 示例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_COLD_HOT_TRACE_INFO info; memset(&info, 0x00, sizeof(SGP_COLD_HOT_TRACE_INFO));</pre>

	<pre>int ret = SGP_GetColdHotTrace(handle,&info); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>
--	--

版权所有©武汉高德红外股份有限

公司

设置全局温度告警SGP_SetColdHotTrace

选项 :	说明
描述 :	设置全局温度告警
详细 描述 :	
函数 :	<pre>int SGP_SetColdHotTrace(SGP_HANDLE handle, SGP_COLD_HOT_TRACE_INFO input);</pre>
参数 :	<pre>handle [in] 传入设备对象 input</pre>

	[in] 输入信息
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	先调用 SGP_GetColdHotTrace 函数, 再设置, 结构体变量在使用前先初始化
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { SGP_COLD_HOT_TRACE_INFO info; memset(&info, 0x00, sizeof(SGP_COLD_HOT_TRACE_INFO)); int ret = SGP_GetColdHotTrace(handle, &info); if (ret == SGP_OK) { info.alarm_out_delay = 10; ret = SGP_SetColdHotTrace(handle, info); if (ret == SGP_OK) { //成功, TODO..... } else { //失败, TODO..... } } else { //失败, TODO..... } }</pre>

公司

获取分析对象告警SGP_GetTempAlarm

选项：	说明
描述：	获取分析对象告警
详细描述：	
函数：	<code>int SGP_GetTempAlarm(SGP_HANDLE handle, SGP_TEMP_ALARM_INFO *output);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 输出信息
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	结构体变量在使用前先初始化
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_TEMP_ALARM_INFO info; memset(&info, 0x00, sizeof(SGP_TEMP_ALARM_INFO));</pre>

	<pre>int ret = SGP_GetTempAlarm(handle,&info); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>
--	--

公司

设置分析对象告警SGP_SetTempAlarm

选项：	说明
描述：	设置分析对象告警
详细描述：	
函数：	<code>int SGP_SetTempAlarm(SGP_HANDLE handle, SGP_TEMP_ALARM_INFO input);</code>
参数：	<div>handle</div> <div>[in] 传入设备对象</div> <div>input</div> <div>[in] 输入信息</div>

返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	先调用 SGP_GetTempAlarm 函数,再设置, 结构体变量在使用前先初始化
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { SGP_TEMP_ALARM_INFO info; memset(&info, 0x00, sizeof(SGP_TEMP_ALARM_INF)); int ret = SGP_GetTempAlarm(handle,&info); if (ret == SGP_OK) { info.audio_flag = 1; ret = SGP_SetTempAlarm(handle,info); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } } else { //失败 , TODO..... } }</pre>

公司

获取视频参数SGP_GetVideoParam

选项：	说明
描述：	获取视频参数
详细描述：	
函数：	<code>int SGP_GetVideoParam(SGP_HANDLE handle, SGP_VIDEO_PARAM_ENUM type, SGP_VIDEO_PARAM *output);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>type</code> [in] 视频类别 <code>output</code> [out] 输出信息
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	结构体变量在使用前先初始化
使用示例：	<code>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。</code>

```

    **/
void Init()
{
    SGP_VIDEO_PARAM_ENUM type = SGP_IR;
    SGP_VIDEO_PARAM info;
    memset(&info, 0x00,
sizeof(SGP_VIDEO_PARAM));
    int ret =
SGP_GetVideoParam(handle,type,&info);
    if (ret == SGP_OK )
    {

        //成功 , TODO.....
    }
    else
    {

        //失败 , TODO.....
    }
}

```

版权所有©武汉高德红外股份有限

公司

设置视频参数SGP_SetVideoParam

选 项：	说明
描 述：	设置视频参数
详 细描 述：	

函数:	<pre>int SGP_SetVideoParam(SGP_HANDLE handle, SGP_VIDEO_PARAM_ENUM type, SGP_VIDEO_PARAM input);</pre>
参数:	<pre>handle [in] 传入设备对象 type [in] 视频类别 input [in] 参数值</pre>
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	调用 SGP_GetVideoParam 函数,再设置, 结构体变量在使用前先初始化。设置帧率后必须把i帧间隔和码流同步设置, 否则可能会设置失败。建议添加i帧间隔和码流设置
使用示例:	<pre>/** 示例中部分类、变量、函数的解释: 1, handle 设备对象。 **/ void Init() { SGP_GENERAL_INFO info; memset(&info, 0x00, sizeof(SGP_GENERAL_INFO)); int ret = SGP_GetGeneralInfo(handle,&info); if (ret == SGP_OK) { SGP_VIDEO_PARAM_ENUM type = SGP_IR; SGP_VIDEO_PARAM param; memset(&param, 0x00, sizeof(SGP_VIDEO_PARAM));</pre>

	<pre> ret = SGP_GetVideoParam(handle,type,&param); param.fps = 25; param.bit_size = info.ir_model_w *ir_model_h *1.5*info.fps*8/18; param.gop_size = info.fps*2; ret = SGP_SetVideoParam(handle,type,param); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } } else { //失败 , TODO..... } }</pre>
--	--

版权所有©武汉高德红外股份有限

公司

获取系统版本信息SGP_GetVersionInfo

选项：	说明
描述：	获取系统版本信息

详细描述：	
函数：	<code>int SGP_GetVersionInfo(SGP_HANDLE handle, SGP_VERSION_INFO *output);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 版本信息
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	结构体变量在使用前先初始化
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_VERSION_INFO info; memset(&info, 0x00, sizeof(SGP_VERSION_INFO)); int ret = GetVersionInfo(handle,&info); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

公司

获取网络异常SGP_GetNetException

选项 :	说明
描述 :	获取网络异常
详细 描述:	
函数 :	<code>int SGP_GetNetException(SGP_HANDLE handle, SGP_NET_EXCEPTION_INFO *output);</code>
参数 :	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 输出信息
返回 值:	成功返回 SGP_OK , 失败返回 错误码
备注 :	结构体变量在使用前先初始化
使用 示例:	<pre>/** 示例中部分类、变量、函数的解释： 1 , handle 设备对象。 **/ void Init() { SGP_NET_EXCEPTION_INFO info; memset(&info, 0x00, sizeof(SGP_NET_EXCEPTION_INFO));</pre>

	<pre>int ret = SGP_GetNetException(handle,&info); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>
--	--

版权所有©武汉高德红外股份有限

公司

设置网络异常SGP_SetNetException

选项 :	说明
描述 :	设置网络异常
详细 描述 :	
函数 :	<code>int SGP_SetNetException(SGP_HANDLE handle, SGP_NET_EXCEPTION_INFO input);</code>
参数 :	<code>handle</code> [in] 传入设备对象 <code>input</code> [in] 输入信息

返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	先调用 SGP_GetNetException 函数, 再设置, 结构体变量在使用前先初始化
使用示例:	<pre> /** * 示例中部分类、变量、函数的解释: * 1, handle 设备对象。 */ void Init() { SGP_NET_EXCEPTION_INFO info; memset(&info, 0x00, sizeof(SGP_NET_EXCEPTION_INFO)); int ret = SGP_GetNetException(handle, &info); if (ret == SGP_OK) { info.audio_flag = 1; ret = SGP_SetNetException(handle, info); if (ret == SGP_OK) { //成功, TODO..... } else { //失败, TODO..... } } else { //失败, TODO..... } } </pre>

公司

获取非法访问SGP_GetAccessViolation

选项:	说明
描述:	获取非法访问
详细描述:	
函数:	<code>int SGP_GetAccessViolation(SGP_HANDLE handle, SGP_ACCESS_VIOLATION_INFO *output);</code>
参数:	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 输出信息
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	结构体变量在使用前先初始化
使用示例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_ACCESS_VIOLATION_INFO info;</pre>

	<pre> memset(&info, 0x00, sizeof(SGP_ACCESS_VIOLATION_INFO)); int ret = SGP_GetAccessViolation(handle,&info); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>
--	---

版权所有©武汉高德红外股份有限

公司

设置非法访问SGP_SetAccessViolation

选项 :	说明
描述 :	设置非法访问
详细 描述 :	
函数 :	int SGP_SetAccessViolation(SGP_HANDLE handle, SGP_ACCESS_VIOLATION_INFO input);
参数 :	handle [in] 传入设备对象

	<div>input</div> <div>[in] 输入信息</div>
返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	先调用 SGP_GetAccessViolation 函数,再设置,结构体变量在使用前先初始化
使用示例:	<div>/**</div> <div>示例中部分类、变量、函数的解释:</div> <div>1, handle 设备对象。</div> <div>**/</div> <div>void Init()</div> <div>{</div> <div> SGP_ACCESS_VIOLATION_INFO info;</div> <div> memset(&info, 0x00,</div> <div>sizeof(SGP_ACCESS_VIOLATION_INF));</div> <div> int ret =</div> <div>GetAccessViolation(handle,&info);</div> <div> if (ret == SGP_OK)</div> <div> {</div> <div> info.audio_flag = 0;</div> <div> ret =</div> <div>SGP_SetAccessViolation(handle,info);</div> <div> if (ret == SGP_OK)</div> <div> {</div> <div> //成功, TODO.....</div> <div> }</div> <div> } else</div> <div> {</div> <div> //失败, TODO.....</div> <div> }</div> <div> }</div> <div>else</div> <div>{</div> <div> //失败, TODO.....</div> <div>}</div> <div>}</div>

公司

获取邮件信息SGP_GetEmilInfo

选项：	说明
描述：	获取邮件信息
详细描述：	
函数：	<code>int SGP_GetEmilInfo(SGP_HANDLE handle, SGP_EMAIL_INFO *output);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 输出信息
返回值：	成功返回SGP_OK, 失败返回 错误码
备注：	结构体变量在使用前先初始化
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_EMAIL_INFO info; memset(&info, 0x00, sizeof(SGP_EMAIL_INFO));</pre>

	<pre>int ret = SGP_GetEmilInfo(handle,&info); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>
--	--

版权所有©武汉高德红外股份有限

公司

设置邮件信息SGP_SetEmilInfo

选项：	说明
描述：	设置邮件信息
详细描述：	
函数：	int SGP_SetEmilInfo(SGP_HANDLE handle, SGP_EMAIL_INFO input);
参数：	handle [in] 传入设备对象 input [in] 输入信息

返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	先调用 SGP_GetEmilInfo 函数,再设置,结构体变量在使用前先初始化
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { SGP_EMAIL_INFO info; memset(&info, 0x00, sizeof(SGP_EMAIL_INFO)); int ret = SGP_GetEmilInfo(handle,&info); if (ret == SGP_OK) { info.alarm = 0; ret = SGP_SetEmilInfo(handle,info); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } } else { //失败 , TODO..... } }</pre>

公司

获取补光灯信息SGP_GetFillLight

选项：	说明
描述：	获取补光灯信息
详细描述：	此函数适用于带可见光的双光设备
函数：	<code>int SGP_GetFillLight(SGP_HANDLE handle, SGP_FILL_LIGHT_INFO *output);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 输出信息
返回值：	成功返回SGP_OK, 失败返回错误码
备注：	结构体变量在使用前先初始化
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_FILL_LIGHT_INFO info; memset(&info, 0x00, sizeof(SGP_FILL_LIGHT_INFO)); int ret = SGP_GetFillLight(handle,&info); if (ret == SGP_OK)</pre>

	<pre> { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>
--	---

公司

设置补光灯信息SGP_SetFillLight

选项：	说明
描述：	设置补光灯信息
详细描述：	此函数适用于带可见光的双光设备
函数：	<code>int SGP_SetFillLight(SGP_HANDLE handle, SGP_FILL_LIGHT_INFO input);</code>
参数：	<div>handle</div> <div>[in] 传入设备对象</div> <div>input</div> <div>[in] 输入信息</div>
返回值：	成功返回 SGP_OK , 失败返回 错误码

备注:	先调用 SGP_GetFillLight 函数,再设置,结构体变量在使用前先初始化
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释: * 1, handle 设备对象。 */ void Init() { SGP_FILL_LIGHT_INFO info; memset(&info, 0x00, sizeof(SGP_FILL_LIGHT_INFO)); int ret = SGP_GetFillLight(handle, &info); if (ret == SGP_OK) { info.brightness= 50; ret = SGP_SetFillLight(handle, info); if (ret == SGP_OK) { //成功, TODO..... } else { //失败, TODO..... } } else { //失败, TODO..... } }</pre>

公司

获取融合状态SGP_GetInfraredMode

选项 :	说明
描述 :	获取融合状态
详细描述 :	此函数适用于带可见光的双光设备
函数 :	<code>int SGP_GetInfraredMode(SGP_HANDLE handle, int *output);</code>
参数 :	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 输出信息，mode红外模式：0:单光红外； 1:双光红外
返回值 :	成功返回 SGP_OK ，失败返回 错误码
备注 :	
使用 示例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { int mode = 0; int ret = SGP_GetInfraredMode(handle, &mode);</pre>

	<pre>if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } }</pre>
--	---

设置融合状态SGP_SetInfraredMode

选项：	说明
描述：	设置融合状态
详细描述：	此函数适用于带可见光的双光设备
函数：	<code>int SGP_SetInfraredMode(SGP_HANDLE handle, int input);</code>
参数：	<div>handle [in] 传入设备对象</div> <div>input [in] mode红外模式： 0:单光红外； 1:双光红外</div>

返回值:	成功返回 SGP_OK , 失败返回 错误码
备注:	
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { int mode = 1; int ret = SGP_SetInfraredMode(handle,mode); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

版权所有©武汉高德红外股份有限

公司

获取蜂鸣器状态SGP_GetSilentMode

选项:	说明
-----	----

描述 :	获取蜂鸣器状态
详细描述 :	
函数 :	<code>int SGP_GetSilentMode(SGP_HANDLE handle, int *output);</code>
参数 :	<code>handle</code> [in] 传入设备对象 <code>output</code> [out] 0:非静音; 1:静音
返回值 :	成功返回 SGP_OK , 失败返回 错误码
备注 :	
使用 示例:	<pre>/** * 示例中部分类、变量、函数的解释: * 1, handle 设备对象. */ void Init() { int silent= 0; int ret = SGP_GetSilentMode(handle,&silent); if (ret == SGP_OK) { //成功, TODO..... } else { //失败, TODO..... } }</pre>

公司

设置蜂鸣器状态SGP_GetSilentMode

选项：	说明
描述：	设置蜂鸣器状态
详细描述：	
函数：	<code>int SGP_SetSilentMode(SGP_HANDLE handle, int input);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>input</code> [in] 0:非静音；1:静音
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { int input= 0; int ret = SGP_SetSilentMode(handle,input); if (ret == SGP_OK)</pre>

	<pre>{ //成功，TODO..... } else { //失败，TODO..... } }</pre>
--	---

版权所有©武汉高德红外股份有限

公司

获取录制信息SGP_GetRecordInfo

选项：	说明
描述：	获取录制信息
详细 描述：	
函数：	<pre>int SGP_GetRecordInfo(SGP_HANDLE handle, SGP_RECORD_INFO *output);</pre>
参数：	<pre>handle [in] 传入设备对象 output [out] 输出信息</pre>
返回 值：	成功返回 SGP_OK ，失败返回 错误码

备注：	结构体变量在使用前先初始化
使用示例：	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { SGP_RECORD_INFO info; memset(&info, 0x00, sizeof(SGP_RECORD_INFO)); int ret = SGP_GetRecordInfo(handle,&info); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

设置录制信息SGP_SetRecordInfo

选项：	说明
描述：	设置录制信息

详细 描述：	
函数：	<code>int SGP_SetRecordInfo(SGP_HANDLE handle, SGP_RECORD_INFO input);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>input</code> [in] 录制信息
返回 值：	成功返回 SGP_OK , 失败返回 错误码
备注：	先调用 SGP_GetRecordInfo 函数,再设置,结构体 变量在使用前先初始化
使用 示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { SGP_RECORD_INFO info; memset(&info, 0x00, sizeof(SGP_RECORD_INFO)); int ret = SGP_GetRecordInfo(handle,&info); if (ret == SGP_OK) { info.record_interval = 1; ret = SGP_SetRecordInfo(handle,info); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } } }</pre>

	<pre> } } else { //失败，TODO..... } }</pre>
--	--

版权所有©武汉高德红外股份有限

公司

设置电子变倍 SGP_SetElectronicMagnification

选项:	说明
描述:	设置电子变倍，只对主码流有效
详细描述:	
函数数:	<code>int SGP_SetElectronicMagnification(SGP_HANDLE handle, SGP_VIDEO_PARAM_ENUM type, int magnification);</code>
参数数:	<code>handle</code> [in] 传入设备对象 <code>type</code>

	<p>[in] 视频类型值</p> <p>input</p> <p>[in] 1：红外原始，可见光原始 2：红外2倍，可见光4倍 3：红外3倍，可见光16倍</p>
返回值：	成功返回 SGP_OK ，失败返回 错误码
备注：	
使用示例：	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { int magnification = 2; SGP_VIDEO_PARAM_ENUM type = 3; int ret = SGP_SetElectronicMagnification(handle,type,magnification); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

获取报警输入 SGP_GetAlarmInput

选项：	说明
描述：	获取报警输入
详细描述：	
函数：	<pre>int SGP_GetAlarmInput(SGP_HANDLE handle, SGP_ALARM_INPUT_INFO *output);</pre>
参数：	<pre>handle [in] 传入设备对象 output [out] 输出信息</pre>
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	
使用示例：	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ void Init() { SGP_ALARM_INPUT_INFO info; memset(&info, 0x00, sizeof(SGP_ALARM_INPUT_INFO)); int ret = SGP_GetAlarmInput(handle, &info); if (ret == SGP_OK) { //成功, TODO..... } else</pre>

	<pre>{ // 失败，TODO..... } }</pre>
--	--

版权所有©武汉高德红外股份有限

公司

设置报警输入 SGP_SetAlarmInput

选项：	说明
描述：	设置报警输入
详细描述：	
函数：	<pre>int SGP_SetAlarmInput(SGP_HANDLE handle, SGP_ALARM_INPUT_INFO input);</pre>
参数：	<pre>handle [in] 传入设备对象 input [in] 报警信息</pre>
返回值：	成功返回 SGP_OK ，失败返回 错误码
备注：	
使用示例：	<pre>/** * 示例中部分类、变量、函数的解释：</pre>

	<pre> 1,handle 设备对象。 **/ void Init() { SGP_ALARM_INPUT_INFO info; memset(&info, 0x00, sizeof(SGP_ALARM_INPUT_INFO)); int ret = SGP_GetAlarmInput(handle,&info); if (ret == SGP_OK) { info.flag = 1; ret = SGP_SetAlarmInput(handle,info); if (ret == SGP_OK) { //成功 , TODO..... } else { //失败 , TODO..... } } else { //失败 , TODO..... } } </pre>
--	---

版权所有©武汉高德红外股份有限

公司

恢复出厂设置 SGP_FactoryReset

选项：	说明
描述：	恢复出厂设置
详细描述：	
函数：	<code>int SGP_FactoryReset(SGP_HANDLE handle);</code>
参数：	handle [in] 传入设备对象
返回值：	成功返回 SGP_OK , 失败返回 错误码
备注：	
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { int ret = SGP_FactoryReset(handle); if (ret == SGP_OK) { //成功，TODO..... } else { //失败，TODO..... } }</pre>

公司

透传RS485数据查询 SGP_CommandSend

选项：	说明
描述：	透传RS485数据查询
详细描述：	
函数：	<code>int SGP_CommandSend(SGP_HANDLE handle, const char *data);</code>
参数：	<code>handle</code> [in] 传入设备对象 <code>data</code> [in] 传入查询指令 例如： "05030000006705A4"
返回值：	成功返回SGP_OK, 失败返回错误码
备注：	
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ void Init() { const char* data= "05030000006705A4"; int ret = SGP_CommandSend(handle,data); if (ret == SGP_OK) {</pre>

	<pre> //成功，TODO..... } else { //失败，TODO..... } }</pre>
--	---

版权所有©武汉高德红外股份有限

公司

注册温度告警回调函数SGP_RegisterTempAlarmCallback

选项:	说明
描述:	注册温度告警回调函数
详细描述:	
函数:	<code>void SGP_RegisterTempAlarmCallback(SGP_HANDLE handle, SGP_TEMPALARMCALLBACK callback, void *pUser);</code>

参 数:	<code>handle</code> <code>[in]</code> 传入设备对象 <code>callback</code> <code>[in]</code> 回调函数地址 <code>pUser</code> <code>[in]</code> 回调函数传入参数,例如QT,可以传入this 指针
返 回 值:	无
备 注:	
使 用 示 例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 2, 以QT界面库为例 **/ static void TempAlarm(SGP_TEMPALARMNOTIFY notify, void *pUser) { MainWindow *pDlg = (MainWindow *)pUser; printf("获取的高温温度是%f\n", notify.high_temp); printf("获取的低温温度是%f\n", notify.low_temp); printf("获取的平均温度是%f\n", notify.avg_temp); printf("获取的报警类型是%d\n", notify.temp_flag); //TODO..... }</pre>

```
void MainWindow::Init()  
{  
    SGP_RegisterTempAlarmCallback(handle,  
TempAlarm, this);  
    //TODO.....  
}
```

回调函数描述		
函数名称	typedef void(*SGP_TEMPALARMCALLBACK)(SGP_TEMPALARMNOTIFY notify, void *pUser);	
功能描述	温度告警回调函数	
参数说明	notify	输出参数
	pUser	输出参数
返回值	无	

注册对象温差告警回调函数
SGP_RegisterObjTempAlarmCallback

选项:	说明
描述:	注册对象温差告警回调函数
详细描述:	
函数:	<code>void SGP_RegisterObjTempAlarmCallback(SGP_HANDLE handle, SGP_OBJTEMPALARMCALLBACK callback, void *pUser);</code>
参数:	<code>handle</code> [in] 传入设备对象 <code>callback</code> [in] 回调函数地址 <code>pUser</code> [in] 回调函数传入参数,例如QT,可以传入this指针
返回值:	无
备注:	
使用示例:	<code>/** 示例中部分类、变量、函数的解释: 1, handle 设备对象。 2, 以QT界面库为例 **/</code>

```
static void
TempAlarm(SGP_OBJTEMPALARMNOTIFY notify,
void *pUser)
{
    MainWindow *pDlg = (MainWindow *)pUser;
    printf("获取的分析对象1的温度是%f\n",
notify.fTemp1);
    printf("获取的分析对象2的温度是%f\n",
notify.fTemp2);
    printf("获取的分析对象1、2的温差值是%f\n",
notify.fTempDiff);
    printf("获取的对象温差比较的判断条件是%d\n",
notify.iTempFlag);
    //TODO.....
}
void MainWindow::Init()
{
    SGP_RegisterObjTempAlarmCallback(handle,
TempAlarm, this);
    //TODO.....
}
```

回调函数描述		
函数名称	typedef void(*SGP_OBJTEMPALARMCALLBACK)(SGP_OBJTEMPALARMNOTIFY notify, void *pUser);	
功能描述	对象温差告警回调函数	
参数说明	notify	输出参数

	pUser	输出参数
返回值	无	

版权所有©武汉高德红外股份有限

公司

注册内存已满回调函数SGP_RegisterMemoryFullCallback

选项:	说明
描述:	注册内存已满回调函数
详细描述:	
函数:	<code>void SGP_RegisterMemoryFullCallback(SGP_HANDLE handle, SGP_MEMORYFULLCALLBACK callback, void *pUser);</code>
参数:	<code>handle</code> [in] 传入设备对象 <code>callback</code> [in] 回调函数地址 <code>pUser</code>

	[in] 回调函数入参
返回值:	无
备注:	
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 * 2, 以QT界面库为例 */ static void MemoryFull(SGP_MEMORYFULLNOTIFY notify ,void *pUser) { MainWindow *pDlg = (MainWindow *)pUser; printf("总存储是%dM\n", notify.total); printf("可用大小%dM\n", notify.free); printf("报警阈值%dM\n", notify.limit); //TODO..... } void MainWindow::Init() { SGP_RegisterMemoryFullCallback(handle, MemoryFull, this); //TODO..... }</pre>

函数名称	typedef void(*SGP_MEMORYFULLCALLBACK)(SGP_MEMORYFULLNOTIFY notify, void *pUser);	
功能描述	内存已满回调函数	
参数说明	notify	输出参数
	pUser	输出参数
返回值	无	

版权所有©武汉高德红外股份有限

公司

注册存储故障回调函数
SGP_RegisterStorageErrorCallback

选项:	说明
描述:	注册存储故障回调函数
详细描述	

述 :	
函 数:	<code>void SGP_RegisterStorageErrorCallback(SGP_HANDLE handle, SGP_STORAGEERRORCALLBACK callback, void *pUser);</code>
参 数:	<code>handle</code> [in] 传入设备对象 <code>callback</code> [in] 回调函数地址 <code>pUser</code> [in] 回调函数入参
返 回 值:	无
备 注:	
使 用 示 例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 2, 以QT界面库为例 **/ static void StorageError(void *pUser) { MainWindow *pDlg = (MainWindow *)pUser; printf("Storage Error\n"); //TODO..... } void MainWindow::Init() { SGP_RegisterStorageErrorCallback(handle, StorageError, this); //TODO..... }</pre>

回调函数描述		
函数名称	typedef void(*SGP_STORAGEERRORCALLBACK)(void *pUser);	
功能描述	存储故障回调函数	
参数说明	pUser	输出参数
返回值	无	

版权所有©武汉高德红外股份有限

公司

注册推流异常回调函数SGP_RegisterRtspErrorCallback

选项：	说明
描述：	注册推流异常回调函数

详细描述：	
函数：	<pre>void SGP_RegisterRtspErrorCallback(SGP_HANDLE handle, SGP_RTSPERRORCALLBACK callback, void *pUser);</pre>
参数：	<pre>handle [in] 传入设备对象 callback [in] 回调函数地址 pUser [in] 回调函数入参</pre>
返回值：	无
备注：	
使用示例：	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 2, 以QT界面库为例 **/ static void RtspError(int type,void *pUser) { MainWindow *pDlg = (MainWindow *)pUser; printf("类型是%d\n", type); //TODO.....</pre>

```
}  
void MainWindow::Init()  
{  
    SGP_RegisterRtspErrorCallback(handle,  
RtspError, this);  
    //TODO.....  
}
```

回调函数描述		
函数名称	typedef void(*SGP_RTSPERRORCALLBACK)(int type, void *pUser);	
功能描述	推流异常回调函数	
参数说明	type	输出参数
参数说明	pUser	输出参数
返回值	无	

注册非法访问回调函数

SGP_RegisterAccessViolationCallback

选项:	说明
描述:	注册非法访问回调函数
详细描述:	
函数:	<code>void SGP_RegisterAccessViolationCallback(SGP_HANDLE handle, SGP_ACCESSVIOLATIONCALLBACK callback, void *pUser);</code>
参数:	<div>handle</div> <div>[in] 传入设备对象</div> <div>callback</div> <div>[in] 回调函数地址</div> <div>pUser</div> <div>[in] 回调函数入参</div>
返回值:	无
备注:	
使用:	<div>/**</div> <div>示例中部分类、变量、函数的解释：</div> <div>1, handle 设备对象。</div>

示

例:

```
    **/  
static void  
AccessViolation(SGP_ACCESSVIOLATIONNOTIFY  
notify, void *pUser)  
{  
    MainWindow *pDlg = (MainWindow *)pUser;  
    printf("异常登录用户名是%s\n", notify.user);  
    printf("异常登录IP是%s\n", notify.ip);  
    printf("异常登录时间是%s\n", notify.time);  
  
    //TODO.....  
}  
void MainWindow::Init()  
{  
    SGP_RegisterAccessViolationCallback(handle,  
AccessViolation, this);  
    //TODO.....  
}
```

回调函数描述		
函数名称	typedef void(*SGP_ACCESSVIOLATIONCALLBACK)(SGP_ACCESSVIOLATIONNOTIFY notify, void *pUser);	
功能描述	非法访问回调函数	
参数说明	notify	输出参数
	pUser	输出参数

返回值	无	
-----	---	--

版权所有©武汉高德红外股份有限

公司

注册网络异常回调函数
SGP_RegisterNetworkErrorCallback

选项:	说明
描述:	注册网络异常回调函数
详细描述:	
函数:	<pre>void SGP_RegisterNetworkErrorCallback(SGP_HANDLE handle, SGP_NETWORKERRORCALLBACK callback, void *pUser);</pre>
参数:	<pre>handle [in] 传入设备对象 callback [in] 回调函数地址 pUser</pre>

	[in] 回调函数入参
返回值:	无
备注:	
使用示例:	<pre>/** * 示例中部分类、变量、函数的解释： * 1, handle 设备对象。 */ static void NetworkError(SGP_NETWORKERRORNOTIFY notify, void *pUser) { MainWindow *pDlg = (MainWindow *)pUser; printf("类型是%d\n", notify.type); printf("IP是%s\n", notify.ip); //TODO..... } void MainWindow::Init() { SGP_RegisterNetworkErrorCallback(handle, NetworkError, this); //TODO..... }</pre>

回调函数描述		
函数名称	typedef void(*SGP_NETWORKERRORCALLBACK)(SGP_NETWORKERRORNOTIFY notify, void *pUser);	
功能描述	网络异常回调函数	
参数说明	notify	
	pUser	
返回值	无	

版权所有©武汉高德红外股份有限

公司

注册外部告警回调函数 SGP_RegisterAlarmInputCallback

选项:	说明
描述:	注册外部告警回调函数
详细描述	

述 :	
函 数:	<code>void SGP_RegisterAlarmInputCallback(SGP_HANDLE handle, SGP_ALARMINPUTCALLBACK callback, void *pUser);</code>
参 数:	<code>handle</code> [in] 传入设备对象 <code>callback</code> [in] 回调函数地址 <code>pUser</code> [in] 回调函数入参
返 回 值:	无
备 注:	
使 用 示 例:	<pre>/** 示例中部分类、变量、函数的解释： 1 , handle 设备对象。 **/ static void AlarmInput(SGP_ALARMINPUTCALLBACK notify, void *ptr) { MainWindow *pDlg = (MainWindow *)ptr; printf("报警时间是%s\n", notify.time); printf("红外JPEG图片得BASE64格式是%s\n", notify.ir_image_content); printf("可见光录像地址是%s\n", notify.vl_video_url); printf("红外录像地址是%s\n", notify.ir_video_url);</pre>

```
//TODO.....  
}  
void MainWindow::Init()  
{  
    SGP_RegisterAlarmInputCallback(handle,  
AlarmInput, this);  
    //TODO.....  
}
```

回调函数描述		
函数名称	typedef void(*SGP_ALARMINPUTCALLBACK)(SGP_ALARMINPUTNOTIFY notify, void *pUser);	
功能描述	外部告警回调函数	
参数说明	notify	
	pUser	
返回值	无	

公司

注册火警报警回调函数 SGP_RegisterFireAlarmCallback

选项:	说明
描述:	注册火警报警回调函数
详细描述:	
函数:	<code>void SGP_RegisterFireAlarmCallback(SGP_HANDLE handle, SGP_FIREALARMCALLBACK callback, void *pUser);</code>
参数:	<code>handle</code> [in] 传入设备对象 <code>callback</code> [in] 回调函数地址 <code>pUser</code> [in] 回调函数入参
返回值:	无
备注:	

使用示例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 */ static void AlarmInput(SGP_FIRE_ALARM notify, void *ptr) { MainWindow *pDlg = (MainWindow *)ptr; printf("报警时间是%s\n", notify.time); printf("红外JPEG图片得BASE64格式是%s\n", notify.ir_image_url); printf("可见光录像地址是%s\n", notify.vl_video_url); printf("红外录像地址是%s\n", notify.ir_video_url); //TODO..... } void MainWindow::Init() { SGP_RegisterFireAlarmCallback(handle, AlarmInput, this); //TODO..... }</pre>
-------	--

回调函数描述	
函数名称	<pre>typedef void(*SGP_FIREALARMCALLBACK)(SGP_FIRE_ALARM notify, void *pUser);</pre>

功能描述	火灾报警回调函数	
参数说明	notify	
	pUser	
返回值	无	

版权所有©武汉高德红外股份有限

公司

注册自动调焦回调函数

SGP_RegisterAutoFocusCallback

选项：	说明
描述：	注册自动调焦回调函数
详细描述：	
函数：	<pre>void SGP_RegisterAutoFocusCallback (SGP_HANDLE handle, SGP_AUTOFOCUSCALLBACK callback, void *pUser);</pre>
参数：	<pre>handle [in] 传入设备对象 callback [in] 回调函数地址</pre>

	pUser [in] 回调函数入参
返回值:	无
备注:	
使用示例:	<pre>/** 示例中部分类、变量、函数的解释： 1, handle 设备对象。 **/ static void GetFocusResult (int result , void *ptr) { printf("Focus result is %d\n", result); //TODO..... } void MainWindow::Init() { SGP_RegisterAutoFocusCallback (handle, GetFocusResult , this); //TODO..... }</pre>

回调函数描述	
函数名称	<pre>typedef void(*SGP_AUTOFOCUSCALLBACK)(int result, void *pUser);</pre>

功能描述	自动调焦回调函数	
参数说明	result	0:调焦完成，结果不清晰 1:调焦完成，结果清晰
	pUser	
返回值	无	

版权所有©武汉高德红外股份有限

公司

结构体定义描述
<pre>struct SGP_ACCESS_VIOLATION_INFO { int audio_flag;//是否音频联动 0:否; 1:是 int audio_index;//音频文件索引0-2 int audio_mode;//音频模式 0:持续时间; 1:播放次数 int audio_value;//音频模式值 0-100 (次/秒) int allow_count;//允许登录次数3-10次 int flag;//是否开启 0:不开启; 1:开启 int sendmail;//是否发送邮件 0:否; 1:是 int light_flag;//是否闪光灯 0:否; 1:是 int light_hold;//闪光灯持续时间10-300s int output_flag;//是否外部输出 0:不输出 1:输出</pre>

```
int output_hold;//外部输出持续时间10-300s  
};
```

结构体定义描述

```
struct SGP_ACCESSVIOLATIONNOTIFY  
{  
    char user[STRING\_LENGTH];//异常登录用户  
    char ip[STRING\_LENGTH]; //异常登录IP  
    char time[STRING\_LENGTH];//异常登录时间  
};
```

结构体定义描述

```
struct SGP_ALARM_INPUT_INFO  
{  
    int flag;//是否开启 0 不开启 1 开启  
    int alarm_shake;//报警抖动0-100s  
    int type;//输入类型：0 常开型 1 常闭型  
    int record_delay;//录制延时 10-300  
    int record_flag;//是否录制 0:不录制； 1:录制  
    int record_stream;//录制类型 0:不录制； 1:只录制可见光； 2:只录制红外； 3:录制红外和可见光  
    int capture_flag;//是否截图 0:否； 1:是
```

```

    int capture_stream;//截图类型 0:不截图; 1:只截图可见光; 2:只截图红外; 3:截图红外和可见光

    int sendmail;//是否发送邮件 0:不发送; 1:发送

    int light_flag;//是否开启闪光灯 0:否; 1:是

    int light_hold;//闪光灯持续时间,10-300s

    int output_flag;//是否外部输出 0:不输出 1:输出

    int output_hold;//外部输出持续时间10-300s

    int audio_flag;//是否音乐提醒 1:是; 0:否

    int audio_index;//音乐文件索引,0-2

    int audio_mode;//音乐播放模式 1:播放次数; 2:持续时间

    int audio_value;//音乐播放值,随模式定义:(持续时间:秒数)(播放次数:播放次数0-100)

    int effect_day_num;//时间数组数量

    SGP\_EFFECT\_DAY effect_day[7];//时间数组
};

```

结构体定义描述

```

struct SGP_ALARMINPUTNOTIFY
{
    char time[STRING\_LENGTH];//报警时间,格式为2020-05-21 12:22:33

    char vl_image_url[STRING\_LENGTH];//报警记录可见光截图,http jpeg路径

    char ir_image_url[STRING\_LENGTH];//报警记录红外截图,http jpeg路径

    char vl_image_content[STRING\_LENGTH];//可见光JPEG图片得BASE64格式

```

```

    char ir_image_content[STRING\_LENGTH]; // 红外JPEG图
片得BASE64格式
    char vl_video_url[STRING\_LENGTH]; // 可见光录像地址
    char ir_video_url[STRING\_LENGTH]; // 红外录像地址
};

```

结构体定义描述

```

struct SGP_FIRE_ALARM
{
    float high_temp; // 高温温度, 高温报警时有效
    float low_temp; // 低温温度, 低温报警时有效
    float avg_temp; // 平均温度, 平均温报警时有效
    char time[STRING\_LENGTH]; // 报警时间, 格式为2020-05-
21 12:22:33
    char capture_time[STRING\_LENGTH]; // 报警抓图时间, 格
式为2020-05-21 12:22:33
    char vl_image_url[STRING\_LENGTH]; // 报警记录可见光截
图, http jpeg路径
    char ir_image_url[STRING\_LENGTH]; // 报警记录红外截图
, http jpeg路径
    char vl_video_url[STRING\_LENGTH]; // 可见光录像地址
    char ir_video_url[STRING\_LENGTH]; // 红外录像地址
};

```

结构体定义描述

```
struct SGP_ANALYTIC_TEMP
{
    int rule_id;    //规则id
    char rule_name[STRING\_LENGTH]; //规则名称 32字符
    以内
    int type;//对象类型 1点; 2线; 3矩形; 4多边形; 5圆形
    float max_temp;//最高温度值
    float min_temp;//最低温度值
    float avg_temp;//平均温度值
};
```

结构体定义描述

```
struct SGP_ANALYTIC_TEMPS
{
    int analytic_num;
    SGP\_ANALYTIC\_TEMP analytic[ANALYTIC\_MAX\_NUM];
    float global_max_temp;//全局最高温度值
    float global_min_temp;//全局最低温度值
    float global_avg_temp;//全局平均温度值
};
```

结构体定义描述

```
struct SGP_COLD_HOT_TRACE_INFO
```

```

{
    int light_hold;//闪光灯持续时间 10-300s
    int light_flag;//是否开启闪光灯 0:否; 1:是
    int alarm_shake;//报警抖动,单位s,0-100
    int capture_flag;//是否截图 0:否; 1:是
    int capture_stream;//截图类型 1:只截图可见光; 2:只
截图红外; 3:截图红外和可见光 (web2.0 截图类型 0:不截图)
    char high_color[STRING\_LENGTH];//高温颜色:0xRGB
    int high_flag;//高温是否检测 0:不检测; 1:检测
    float high_temp;//高温温度,-40~2000
    char low_color[STRING\_LENGTH];//低温颜色:0xRGB
    int low_flag;//低温是否检测 0:不检测; 1:检测
    float low_temp;//低温温度,-40~2000
    int record_delay;//录像时间, 10~300s
    int record_flag;//是否录制 0:不录制; 1:录制
    int record_stream;//录制类型 1:只录制可见光; 2:只录
制红外; 3:录制红外和可见光
(web2.0 录制类型 0:不录制)
    int sendmail;//是否发送邮件 0:不发送; 1:发送
    int trace_flag;//是否开启 0:不开启; 1:开启
    int effect_day_num;//时间数组数量
    int output_flag;//是否外部输出 0:不输出 1:输出
    int output_hold;//外部输出持续时间 10-300s
    int audio_flag;//是否音乐提醒 1:是; 0:否
    int audio_index;//音乐文件索引, 0-2
    int audio_mode;//音乐播放模式 1:播放次数; 2:持续时间
    int audio_value;//音乐播放值,随模式定义:(持续时间:秒
数)(播放次数:播放次数0-100)
    SGP\_EFFECT\_DAY effect_day[7];//时间数组
    int high_condition;//全局最高温对应的控制条件, 1:大
于, 0:小于

```

```
    int low_condition;//全局最低温对应的控制条件, 1:大于  
    , 0:小于  
};
```

结构体定义描述

```
struct SGP_CONFIG  
{  
    int type;//报警类型 1:高温报警; 2:低温报警; 3:平均温  
报警; 4:高低温报警  
    int condition;//条件 1:高于; 2:低于 3:匹配;  
    float high_temp;//配置高温  
    float low_temp;//配置低温  
    float avg_temp;//配置平均温  
    int objtype;//类型 0:冷热点; 1:点; 2:线; 3:矩形;  
4:多边形;5:圆形  
    SGP\_POINT points[7];  
};
```

结构体定义描述

```
struct SGP_EFFECT_DAY  
{  
    int day;//1-7,星期几  
    int period_num;//时间段数量  
    SGP\_PERIOD period[6];//时间段
```



```
};
```

结构体定义描述

```
struct SGP_EMAIL_INFO
{
    int alarm;//是否使用报警邮件 0:否;1:是
    int alarm_value;//报警邮件间隔 1-3600秒
    int enclosure;//是否带附件 0:否; 1:是
    int encry_type;//加密方式 0:none; 1:tls; 2:ssl
    char from[STRING\_LENGTH];//发件人
    int health;//是否使用健康邮件 0:否; 1:是
    int health_value;//健康邮件间隔 1-3600秒
    int is_anon;//是否匿名 0:否; 1:是
    char password[STRING\_LENGTH];//登录密码,密文传输
    int smtp_port;//smtp服务端口,默认25
    char smtp_server[STRING\_LENGTH];//smtp服务器,默认
空xxx.xxx.xxx.xxx
    char subject[STRING\_LENGTH];//主题
    char username[STRING\_LENGTH];//登录服务器用户名
    int mailto_num;//收件人数量
    char mailto[5][STRING\_LENGTH];//收件人列表
};
```

结构体定义描述

```

struct SGP_FILL_LIGHT_INFO
{
    int brightness; /*亮度 0-100,0 - 20一档;21 - 40二
档;41 - 60三档;61 - 80四档;81 - 100五档*/
    int light; //灯开启状态 0:关闭; 1:开启
    int mode; //灯模式 0:手动; 1:自动
};

```

结构体定义描述

```

enum SGP_FOCUS_TYPE
{
    SGP_FOCUS_STOP = 0,          // 电机停止
    SGP_FOCUS_FAR = 1,           // 远焦
    SGP_FOCUS_NEAR = 2,          // 近焦
    SGP_FOCUS_FAR_FINE = 3,      // 远焦微调
    SGP_FOCUS_NEAR_FINE = 4,     // 近焦微调
    SGP_FOCUS_AUTO = 5,          // 自动聚焦
    SGP_FOCUS_PLACE = 6,        // 设置位置
};

```

结构体定义描述

```

struct SGP_GENERAL_INFO
{

```

```

        char datetime[STRING\_LENGTH]; //系统时间，格式为
2020-05-21 12:55:12
        char ir_rtsp_url[STRING\_LENGTH]; //红外主码流
rtsp地址
        char ir_sub_rtsp_url[STRING\_LENGTH]; //红外辅码
流rtsp地址
        int ir_model_w; //红外模组宽
        int ir_model_h; //红外模组高
        int ir_output_w; //红外通道输出宽
        int ir_output_h; //红外通道输出高
        int range_num; //测温范围数量
SGP\_RANGE range[RANGE\_MAX\_NUM]; //测温范围
        char vl_rtsp_url[STRING\_LENGTH]; //可见光主码流
rtsp地址(双光产品支持)
        char vl_sub_rtsp_url[STRING\_LENGTH]; //可见光辅
码流rtsp地址(双光产品支持)
};

```

结构体定义描述

```

struct SGP_IAMGE_EFFECT_PARAM_IR_CONFIG
{
    int auto_shutter; //快门自动补偿时间1-20(单位分钟)
    int brightness; //亮度，取值范围0-100
    int contrast; //对比度，取值范围0-100
    int reverse; //是否反转，0:不反转 1 反转
    int time_flag; //降噪时域滤波开关:0关闭;1开启
    int time_value; //降噪时域滤波值 0-100

```

```

int space_flag;//降噪空域滤波开关:0关闭;1开启
int space_value;//降噪空域滤波值 0-100
int iee_flag;//细节增强开关:0关闭;1开启
int iee_value;//细节增强值0-100
int saturation;//饱和度，取值范围0-100(红外设备不支持)
int sharpness;//锐度，取值范围0-100
    int rotate;//旋转参数(顺时针，0:0°，1:90°，2:180°，3:270°)
};

```

结构体定义描述

```

struct SGP_IAMGE_EFFECT_PARAM_VL_CONFIG
{
    int blc;//背光补偿:0关闭; 1上; 2下; 3左; 4右; 5中; 6自动
    int brightness;//亮度，取值范围0-100
    int contrast;//对比度，取值范围0-100
    int exp;//曝光补偿：0-100
    int hlc;//强光抑制:0关闭;1开启
    int reverse;//是否反转，0:不反转 1 反转
    int saturation;//饱和度，取值范围0-100
    int sharpness;//锐度，取值范围0-100
    int wdr;//宽动态 0:关闭; 1:20%; 2:40%; 3:60%; 4:80%; 5:100%
};

```

结构体定义描述

```
struct SGP_IMAGE_FUSION
{
    int percent;//融合比例值0-100
    int ir_left;//红外图像左边裁剪像素值0~50
    int ir_right;//红外图像右边裁剪像素值0~50
    int ir_top;//红外图像上边裁剪像素值0~50
    int ir_botton;//红外图像下边裁剪像素值0~50
    int vl_left;//可见光图像左边裁剪像素值0~1000
    int vl_right;//可见光图像右边裁剪像素值0~1000
    int vl_top;//可见光图像上边裁剪像素值0~1000
    int vl_botton;//可见光图像下边裁剪像素值0~1000
    SGP\_IMAGE\_FUSION\_MATCH\_POINTS ir_match_points;//
    红外校准点
    SGP\_IMAGE\_FUSION\_MATCH\_POINTS vl_match_points;//
    可见光校准点
};
```

结构体定义描述

```
struct SGP_IMAGE_FUSION_MATCH_POINTS
{
    SGP\_POINT point1;//第1个校准点
    SGP\_POINT point2;//第2个校准点
    SGP\_POINT point3;//第3个校准点
    SGP\_POINT point4;//第4个校准点
};
```

```
};  
    SGP\_POINT point5;//第5个校准点
```

结构体定义描述

```
enum SGP_IMAGE_TYPE  
{  
    SGP_VL_IMAGE = 1, //可见光图片  
    SGP_IR_IMAGE = 2, //红外图片  
};
```

结构体定义描述

```
enum SGP_IR_IMAGE_EFFECT_ENUM  
{  
    SGP_IR_AUTO_SHUTTER = 1, //快门自动补偿时间1-20(单位分钟)  
    SGP_IR_BRIGHTNESS = 2, //亮度, 取值范围0-100  
    SGP_IR_CONTRAST = 3, //对比度, 取值范围0-100  
    SGP_IR_REVERSE = 4, //是否反转, 0:不反转 1 反转  
    SGP_IR_TIME_FLAG = 5, //降噪时域滤波开关:0关闭;1开启  
    SGP_IR_TIME_VALUE = 6, //降噪时域滤波值 0-100  
    SGP_IR_SPACE_FLAG = 7, //降噪空域滤波开关:0关闭;1开启  
    SGP_IR_SPACE_VALUE = 8, //降噪空域滤波值 0-100
```

<pre>SGP_IR_IEE_FLAG = 9, //细节增强开关:0关闭;1开启 SGP_IR_IEE_VALUE = 10, //细节增强值0-100 SGP_IR_SATURATION = 11, //饱和度, 取值范围0-100 SGP_IR_SHARPNESS = 12, //锐度, 取值范围0-100 SGP_IR_ROTATE = 13, //旋转 };</pre>

结构体定义描述

```
struct SGP_MEMORYFULLNOTIFY
{
    int total; //总存储, 单位M
    int free; //可用大小, 单位M
    int limit; //报警阈值, 可用小于报警阈值时报警, 单位M
};
```

结构体定义描述

```
struct SGP_NET_EXCEPTION_INFO
{
    int audio_flag; //是否音频联动 0:否; 1:是
    int audio_index; //音频文件索引0-2
    int audio_mode; //音频模式 0:持续时间; 1:播放次数
    int audio_value; //音频模式值 0-100 (次/秒)
    int flag; //是否开启 0:不开启; 1:开启
```

```
int light_flag;//是否闪光灯 0:否; 1:是
int light_hold;//闪光灯持续时间10-300s
int output_flag;//是否外部输出 0:不输出 1:输出
int output_hold;//外部输出持续时间1-300s
};
```

结构体定义描述

```
struct SGP_NET_INFO
{
    int card;//网卡类型:0有线网卡
    char dns1[STRING\_LENGTH];//dns服务器
xxx.xxx.xxx.xxx
    char dns2[STRING\_LENGTH];//dns服务器
xxx.xxx.xxx.xxx
    char gateway[STRING\_LENGTH];//网关xxx.xxx.xxx.xxx
    char host_name[STRING\_LENGTH];//主机名
    int ip_version;//版本 0:ipv4; 1:ipv6
    char ipaddr[STRING\_LENGTH];//网络ip地址
xxx.xxx.xxx.xxx
    char mac[STRING\_LENGTH];//Mac地址
    int mode;//模式 0:静态; 1:动态
    char netmask[STRING\_LENGTH];//子网掩码
xxx.xxx.xxx.xxx
};
```


结构体定义描述

```
struct SGP_NETWORKERRORNOTIFY
{
    int type;//类型 1:ip冲突; 2:ping不通网关, ip为网关
    char ip[STRING\_LENGTH];//ip地址
};
```

结构体定义描述

```
struct SGP_PERIOD
{
    char start[STRING\_LENGTH];//开始时间, 格式
    HH:mm:ss
    char end[STRING\_LENGTH];//结束时间, 格式 HH:mm:ss
};
```

结构体定义描述

```
struct SGP_POINT
{
    int x;//x坐标 范围 参照红外图像
    int y;//y坐标 范围 参照红外图像
};
```

结构体定义描述

```
struct SGP_PORT_INFO
{
    int http_port;//http服务器端口，默认端口80保留设置
    int max_connectios;//最大web连接数， 1-20
    int onvif_check;//Onvif登录校验 0:不校验； 1:校验
    int rtsp_port;//红外rtsp端口，1024-65535，端口用于
rtsp流服务，默认端口554保留设置
    int tcp_port;//web消息交互端口，不可设置
};
```

结构体定义描述

```
struct SGP_RANGE
{
    int id;//测温档位类型（低温:0,高温:1,其他:2）
    int min;//测温范围最低温
    int max;//测温范围最高温
};
```

结构体定义描述

```
struct SGP_RECORD_INFO
{
```

```

    int record_interval;//延时录制时间1-3600秒
    int record_max_size;//录制文件最大大小，单位M，1-1000
    int record_time;//录制时长，单位秒，1-3600分钟
};

```

结构体定义描述

```

struct SGP_RECT
{
    int x;//x坐标， 1-图像宽
    int y;//y坐标， 1-图像高
    int w;//区域宽，与坐标共同作用，取值范围1-图像宽
    int h;//区域高，与坐标共同作用，取值范围1-图像高
};

```

结构体定义描述

```

struct SGP_RULE
{
    int id;//分析对象id，内部分配，无需设置。
    int alarm_condition;//报警条件:1高于;2低于;3匹配;4高于和低于(web2.0支持1和2)
    int alarm_flag;//是否报警:0不需要;1需要
    int alarm_time;//去抖动时间,0-10秒

```

```

    int alarm_interal;//报警间隔时间，单位：秒。允许设置
的数据为：30， 60， 300， 600，900,1800，3600
    int alarm_type;//报警类型:1高温报警;2低温报警;3平均温
报警;4最高温+最低温报警警(web2.0 只支持1、2、3)
    float avg_temp;//平均温(基于设备的测温范围)
    int flag;//是否启用配置:0不启用;1启用
    float high_temp;//报警高温阈值(基于设备的测温范围)
    float low_temp;//报警低温阈值(基于设备的测温范围)
    int points_num;
    SGP\_POINT points[7];//矩形，圆是四个点，顺时针顺序
    char rule_name[STRING\_LENGTH];//规则名称，支持50字
符
    int show_location;//名称显示位置:1上方;2下方;3左方
;4右方;5中间
    float temp_mod;//温度误差
    int type;//对象类型:1点;2线;3矩形;4多边形;5圆
    float atmo_trans;//大气透过率0.01-1
    float dist;//距离，单位米，0.1-20.0
    float emiss;//发射率 0.1-1.0
    int emiss_mode;//发射率类型:1标准;2自定义
    int humi;//湿度,范围1-100
    float opti_trans;//光学透过率0.01-1
    float ref_temp;//反射温度-20~550,单位摄氏度(web2.0
-40到2000)
    int show_type;//显示内容，范围1~8,1~5(最高温，最低温
，平均温度，仅名称，不显示),6~8属于预留部分
};

```

结构体定义描述

```
struct SGP_RULE_ARRAY
{
    int rule_num;
    SGP\_RULE rule[ANALYTIC\_MAX\_NUM]; //规则列表
};
```

结构体定义描述

```
struct SGP_SHIELD_AREA_INFO
{
    int rect_num;
    SGP_RECT rect[SHIELD\_AREA\_MAX\_NUM]; //区域数组,左上角坐标0, 0标准,最多支持两个
};
```

结构体定义描述

```
enum SGP_SHUTTER_ENUM
{
    SGP_SHUTTER = 1, //快门操作
    SGP_SHUTTER_OPEN = 2, //快门常开
    SGP_SHUTTER_CLOSE = 3, //快门常闭
    SGP_SHUTTER_AUTO = 4, //自动快门
};
```

```
};
```

结构体定义描述

```
struct SGP_TEMP_ALARM_INFO
{
    int audio_flag;//是否音乐提醒 1:是; 0:否
    int audio_index;//音乐文件索引, 0-2
    int audio_mode;//音乐播放模式 1:播放次数; 2:持续时间
    int audio_value;//音乐播放值,随模式定义:(持续时间:秒数)(播放次数:播放次数0-100)
    int alarm_flag;//是否开启报警 1:开启; 0:不开启(新web上该字段弃用)
    int light_hold;//闪光灯持续时间, 10-300s
    int light_flag;//是否开启闪光灯 0:否; 1:是
    int alarm_shake;//报警抖动0-100s(新web上仅在全局温度-报警参数设置-去抖动, 这个功能上使用)
    int capture_flag;//是否截图 0:否; 1:是( web2.0上该字段弃用)
    int capture_stream;//截图类型 0:不截图; 1:只截图可见光; 2:只截图红外; 3:截图红外和可见光
    int record_delay;//录制时间 10-300s
    int record_flag;//是否录制 0:不录制; 1:录制( web2.0上该字段弃用)
    int record_stream;//录制类型 0:不录制; 1:只录制可见光; 2:只录制红外; 3:录制红外和可见光
    int sendmail;//是否发送邮件 0:不发送; 1:发送
    int effect_day_num;//时间数组数量
    int output_flag;//是否外部输出 0:不输出 1:输出
```

```

    int output_hold;//外部输出持续时间10-300s
    SGP\_EFFECT\_DAY effect_day[7];//时间数组
};

```

结构体定义描述

```

struct SGP_TEMPALARMNOTIFY
{
    char vl_image_url[STRING\_LENGTH];//报警记录可见光截图
    char vl_video_url[STRING\_LENGTH];//可见光视频地址
    char ir_image_url[STRING\_LENGTH];//报警记录红外截图
    char ir_video_url[STRING\_LENGTH];//红外视频地址
    float high_temp;//高温温度,高温报警时有效
    float low_temp;//低温温度,低温报警时有效
    float avg_temp;//平均温度,平均温报警时有效
    int temp_flag;//报警类型,0代表平均温,1代表高温报警,2代表低温报警,3代表高低温报警(web2.0 不支持)
    int type;//1:温度报警; 2:热点报警; 3:冷点报警(web2.0 不支持)
    char name[STRING\_LENGTH];//名称
    char time[STRING\_LENGTH];//报警时间,格式为2020-05-21 12:22:33
    SGP\_CONFIG config;//配置
};

```

结构体定义描述

```
struct SGP_OBJTEMPALARMNOTIFY
{
    char vl_image_url[200]; //报警记录可见光截图地址
    char vl_video_url[200]; //可见光视频地址
    char ir_image_url[200]; //报警记录红外截图地址
    char ir_video_url[200]; //红外视频地址

    char obj_name1[STRING\_LENGTH]; //分析对象1的名称
    char obj_name2[STRING\_LENGTH]; //分析对象2的名称
    float fTemp1; //分析对象1的温度
    float fTemp2; //分析对象2的温度
    float fTempDiff; //分析对象1、2的温差值
    float fTempThreshold; //分析对象1、2的温差阈值
    int iTempFlag; //判断条件： 0:温差大于阈值； 1:温差小于阈值

    float iTempType1; //分析对象1的温度类型，0：最高温，1:最低温，2:平均温
    float iTempType2; //分析对象2的温度类型，0：最高温，1:最低温，2:平均温

    char name[STRING\_LENGTH]; //名称
    char time[STRING\_LENGTH]; //报警时间，格式为2020-05-21 12:22:33
};
```




结构体定义描述
<pre>struct SGP_THERMOMETRY_PARAM { int color_bar;//色带1-26 int color_show;//色带显示0~1 int flag;//测温开关0~1 float mod_temp;//温度修正 int show_mode;//温度显示方式：1 最高温 2 最低温 3 平均温 4 最高温 + 最低温 5 最高温 + 平均温 6 平均温 + 最低温 7 最高温 + 最低温 + 平均温 8不显示 int gear;//测温范围 int show_string; //是否使用字符叠加 其他机型 1:关闭; 2, 4, 5:右下; 3:右上 //IPM630 1:关闭; 5:右下 //IPT640M 1:关闭; 2:左上; 3:右 上; 4:左下; 5:右下 char show_desc[STRING_LENGTH];//显示字符串 float atmo_trans;//大气透过率0.01-1</pre>

<pre> float dist;//距离，单位米，0.1-20.0 float emiss;//发射率 0.1-1.0 int emiss_mode;//发射率类型:1标准;2自定义 int humi;//湿度,范围1-100 float opti_trans;//光学透过率0.01-1 float ref_temp;//反射温度-20~550,单位摄氏度(web2.0 的反射温度范围: -40-2000) int isot_flag;//等温线开关0:关闭;1开启(工业机芯支持) float isot_high;//高温阈值0~400 char isot_high_color[STRING_LENGTH];//高温颜色,十 六进制值,如红色:0xff0000 int isot_low;//低温阈值-50~-100 char isot_low_color[STRING_LENGTH];//低温颜色,十六 进制值,如红色:0xff0000 int isot_type;//范围类型:1 关闭等温线效果 2 开启高等 温线 3 开启低等温线 4 开启区间内等温线 5 开启区间外等温线 float ambient;//环境温度 }; </pre>

结构体定义描述
<pre> struct SGP_VERSION_INFO { </pre>

```

char model[STRING\_LENGTH];          //设备型号
char version[STRING\_LENGTH];        //系统版本
char serial[STRING\_LENGTH];         //序列号
char fpga_version[STRING\_LENGTH];    //FPGA版本
char measure_version[STRING\_LENGTH]; //测温版本
char sdk_version[STRING\_LENGTH];     //sdk版本
};

```

结构体定义描述

```

struct SGP_VIDEO_PARAM
{
    int bit_size;//主码流固定码流值，
    可变码流时也需设置（宽*高*1.5*fps*8/压缩率）其中压缩率范围
    （18-500）
    如分辨率1280x720    取值范围：540Kb/s - 15000Kb/s
    int codec;//主码流编码 0:h264; 1:h265; 2:mjpeg
    int fps;//主码流帧率1-25
    int gop_size;//主码流帧间隔1-50
    int level;//编码质量等级，等级效果随实际变化，如使用
    ffmpeg，需服务端自映射（
    默认medium， 可以上下延续几个等级）1:最好 2:更好 3:好 4:差
    5:更差 6:最差
    int rate_control;//主码流控制 0:可变码流;1:固定码流
    char ratio[STRING\_LENGTH];/*主码流分辨率
                                1920x1080
                                1280x960
                                1280x720
                                可见光辅码流分辨率

```

```

704x576
640x480
红外主码流分辨率
512x384(640*512)
红外辅码流分辨率
(384*288)256x192*/
int svc;//帧率可分层编码，H264有效，其他格式也需传入
0:分层编码；1:不分层编码
};

```

结构体定义描述

```

enum SGP_VIDEO_PARAM_ENUM
{
    SGP_VL = 1,          //可见光主码流
    SGP_VL_SUB = 2,      //可见光辅码流
    SGP_IR = 3,          //红外主码流
    SGP_IR_SUB = 4,      //红外辅码流
};

```

结构体定义描述

```

enum SGP_VIDEO_TYPE
{
    SGP_VL_VIDEO = 1, //可见光录像
    SGP_IR_VIDEO = 2, //红外录像
};

```

--

结构体定义描述

```
enum SGP_VL_IMAGE_EFFECT_ENUM
{
    SGP_VL_BLC = 1, //背光补偿:0关闭; 1上; 2下; 3左; 4
    右; 5中; 6自动
    SGP_VL_BRIGHTNESS = 2, //亮度, 取值范围0-100
    SGP_VL_CONTRAST = 3, //对比度, 取值范围0-100
    SGP_VL_EXP = 4, //曝光补偿: 0-100
    SGP_VL_HLC = 5, //强光抑制:0关闭;1开启
    SGP_VL_REVERSE = 6, //是否反转, 0:不反转 1 反转
    SGP_VL_SATURATION = 7, //饱和度, 取值范围0-100
    SGP_VL_SHARPNESS = 8, //锐度, 取值范围0-100
    SGP_VL_WDR = 9, //宽动态 0:关闭; 1:20%; 2:40%;
    3:60%; 4:80%; 5:100%
};
```

结构体定义描述

```
struct SGP_MEASURE_TEMP_INFO
{
    float jwtemp; //焦温温度
    float realshuttertemp; //实时快门温度
    float lastshuttertemp; //上次快门温度
    float realmirrortemp; //实时镜筒温度
};
```

```

int      jwgears;//焦温档位
int devgain;//探测器参数Gain
int devint;//探测器参数Int
int devres;//探测器参数Res
int devgsk;//探测器参数gsk
float centertemp;//全图中心温度
float centermaxtemp;//全图最高温度
float centermintemp;//全图最低温度
int      centerx16;//中心点X16
int centery16;//中心点Y16
int avgshutter;//快门本底均值
int rasel;//探测器参数RASEL
int hssd;//探测器参数HSSD
int gsktestnum;//探测器参数gsk_test_num
int gskval;//探测器参数gsk_val
bool tempStabilityState;//焦温波动判断设备稳定性
float sharpnessleftup;//图像内左上区域的清晰度
float sharpnessrightup;//图像内右上区域的清晰度
float sharpnesscenter;//图像内中间区域的清晰度
float sharpnessleftdown;//图像内左下区域的清晰度
float sharpnessrightdown;//图像内右下区域的清晰度
};

```

结构体定义描述

```

enum SGP_IR_IMAGE_INFO
{
    int denoise_flag_2d;//2D降噪开关
    int denoise_value_2d;//2D降噪值
}

```

```
int denoise_flag_3d;//3D降噪开关
int denoise_value_3d;//3D降噪值
};
```

结构体定义描述

```
enum SGP_IR_IMAGE_INFO_ENUM
{
    SGP_2D_FLAG = 1, //2D降噪开关
    SGP_2D_VALUE = 2, //2D降噪值
    SGP_3D_FLAG = 3, //3D降噪开关
    SGP_3D_VALUE = 4, //3D降噪值
    SGP_SAVE_IR_IMAGE_INFO = 5, //保存参数信息
};
```

类型定义描述

```
typedef unsigned long long SGP_HANDLE;
```

宏定义

定义	数值	描述
STRING_LENGTH	50	一般长度
RANGE_MAX_NUM	3	测温范围值
ANALYTIC_MAX_NUM	21	分析对象个数
SHIELD_AREA_MAX_NUM	2	屏蔽区域个数

版权所有©武汉高德红外股份有限

公司

错误码

错误码	定义	描述
0	SGP_OK	正常
1	SGP_ERR	错误
10001	SGP_ERR_10001	消息内容为空
10002	SGP_ERR_10002	消息内容无效
10003	SGP_ERR_10003	消息字段为空
10004	SGP_ERR_10004	无效用户名
10005	SGP_ERR_10005	未鉴权
10006	SGP_ERR_10006	密码修改失败
10007	SGP_ERR_10007	用户无权限操作

10008	SGP_ERR_10008	用户操作失败
10009	SGP_ERR_10009	密码错误
10010	SGP_ERR_10010	用户锁定
10011	SGP_ERR_10011	用户或密码错误
10012	SGP_ERR_10012	同版本升级
10013	SGP_ERR_10013	低版本升级
10014	SGP_ERR_10014	非法IP
10015	SGP_ERR_10015	IP冲突
10016	SGP_ERR_10016	非法子网掩码
10017	SGP_ERR_10017	非法网关
10018	SGP_ERR_10018	超过最大在线人数
10019	SGP_ERR_10019	DHCP错误
10020	SGP_ERR_10020	密码重复
10021	SGP_ERR_10021	请求温度矩阵时，正在打快门
10022	SGP_ERR_10022	请求温度矩阵时，正在切测温范围