SP210SDK 开发说明

SDK v1.5.4

目录

SP210 C ABI 开发函数说明 2

SDK 接入与 **Qt** 再分发 5

设备、管理、存储与校准 6

协议、传输与光谱计算 39

取消与连续采集 50

固件包、更新与核验 53

SP210 C ABI 开发函数说明

本参考覆盖 SP210SDK/sp210_c_api.h 的全部公开函数。函数签名与枚举数值以和 SDK 一起发布的公共头文件为准，头文件及动态库应来自同一个发布包。SDK 内部使用 Qt，调用方通过纯 C 公共头文件及 SDK 动态库接入。

分类	内容
SDK 接入与 Qt 再分发	Windows 和 Linux 部署、LGPLv3 义务及运行库替换
设备与管理	版本、错误、发现设备、产品信息、参数、IO、存储及校准
协议与传输	光谱计算、组帧与解析、串口事务与会话
连续采集	取消令牌及连续采集回调
固件升级	固件包、查询、升级、核验及辅助函数

调用约定与错误

- Windows 调用约定为 cdecl；公开枚举为 32 位，禁止使用 short-enum 编译选项。size_t 与进程指针同宽，C# 对应 nuint；int32_t 对应 int，int64_t 对应 long，uint8_t 对应 byte，不要将 C 布尔字节直接声明为默认 .NET bool。
- 字符串输入和输出为 UTF-8。输出字节数包含结尾 NUL。SDK 分配的静态字符串指针只读，不由调用方释放；其他输出使用调用方缓冲区。
- 所有接口同步执行，串口可能阻塞；可由客户自己的工作线程调用。同一端口互斥，不同端口可并行。扫描结果与基于索引的两次读取应由调用方协调，避免扫描改变列表。
- 普通可变长度输出先传 NULL 和容量 0，再根据输出长度分配并重试；非空结果的查询一般返回 SP210_RESULT_BUFFER_TOO_SMALL。字符串容量以字节计，数值数组容量以元素计，二进制容量以字节计。设备读取类接口两次调用可能各访问一次设备；具体例外见各函数说明。
- 单帧采集的查询也会采集一帧；如需保留同一帧，预分配 32767 个 int32_t，只调用一次。波长数组从物理像素 0 开始，样本 i 对应下标 starting_pixel + i；起始像素读取失败或索引越界时不得猜测波长。
- 调用失败后应立即在同一线程调用 SP210_GetLastError；其他成功的结果码接口会清除该线程的错误。不要让 C++ 或托管异常穿过 ABI 边界。

SP210_Result	数值	含义
SP210_RESULT_OK	0	调用成功
SP210_RESULT_INVALID_ARGUMENT	1	参数或调用上下文无效
SP210_RESULT_INVALID_HANDLE	2	句柄无效或已销毁
SP210_RESULT_BUFFER_TOO_SMALL	3	输出缓冲区不足；参考所需长度
SP210_RESULT_DEVICE_NOT_FOUND	4	未找到支持的设备
SP210_RESULT_IO_ERROR	5	设备通信、读取或写入失败
SP210_RESULT_OUT_OF_MEMORY	6	内存不足
SP210_RESULT_INTERNAL_ERROR	7	内部错误
SP210_RESULT_BUSY	8	资源或取消令牌忙
SP210_RESULT_CANCELLED	9	操作已取消
SP210_RESULT_PROTOCOL_ERROR	10	响应帧或协议校验失败

句柄与资源

句柄	创建	销毁
SP210_ManagerHandle	SP210_ManagerCreate	SP210_ManagerDestroy
SP210_DeviceHandle	管理器 Open/GetCachedDevice	SP210_DeviceDestroy
SP210_CancelHandle	SP210_CancelCreate	SP210_CancelDestroy
SP210_FirmwarePackageHandle	FirmwareParsePackage/ LoadPackage	SP210_FirmwarePackageDestroy
SP210_FirmwareResultHandle	FirmwareQuery/Update	SP210_FirmwareResultDestroy
SP210_TransportSessionHandle	会话回调参数	SDK 在回调结束时失效，调用方不能销毁或保存

句柄不透明，不得解引用或用 free 释放。管理器销毁后，已打开的设备句柄仍可使用；销毁句柄不会中断已取得对象引用的调用，但销毁后不得继续使用该句柄。取消令牌只能在上一次操作返回后复用，复用前须 Reset。

结构体和枚举

带 struct_size 的输入或输出结构体，调用前必须置为本版本结构体的 sizeof；保留字段置零。现有结构体布局不随扩展改变；具体字段顺序和平台对齐以公共头文件及对应 C ABI 为准。

结构体	字段与用途
SP210_ProductInfo	struct_size；数值 product_id；sensor_name[64] UTF-8；ADC 位数 sensor_ad_resolution；物理像素数 sensor_pixel_count；最小/最大积分微秒 minimum_integration_time_us、maximum_integration_time_us；spectral_data_length、optical_black_count；字节标志 adjustable_saturation_voltage、adjustable_gain、has_tec；reserved[5]。
SP210_InputIOState	raw_value 是 16 位原始值；io15_high、io16_high、io17_high 是字节状态；reserved[3]。
SP210_OutputIOState	raw_value 是 16 位原始值；io2_high、io3_high、io4_high 是字节状态；reserved[3]。
SP210_TransportTransaction	struct_size；完整请求帧指针 request 和字节数 request_length；等待 delay_ms；最大响应字节数 response_maximum_length；尝试次数 max_retries；首字节超时 first_byte_timeout_ms。请求和响应上限不超过 65545 字节，响应上限至少为 1，尝试次数至少为 1，等待和超时不能为负；输出缓冲区须覆盖响应上限。
SP210_FirmwarePackageInfo	struct_size、原始包长度 raw_length、载荷长度 payload_length、header_crc、payload_crc、Unix 时间戳秒数 timestamp_unix_seconds。
SP210_FirmwareResultInfo	struct_size、固件状态 status、字节标志 transfer_completed、device_accepted 和 reserved[2]。接口调用成功不等于固件操作成功，必须检查状态。
SP210_FirmwareTransport	struct_size、同步 read / write 函数指针与借用的 user_data。
SP210_FirmwareOptions	struct_size；query_timeout_ms、query_quiet_ms、handshake_timeout_ms、packet_timeout_ms、processing_timeout_ms；max_retries 和 data_block_size。超时均须大于 0，重试次数为 0–100，数据块大小为 128 或 1024；先用默认选项初始化再修改。

SP210_UserStorageBlock 的 PART1–PART8 数值为 0x52–0x59，每区严格 2048 字节。SP210_FunctionCode、SP210_FunctionItem、SP210_ProtocolError 定义协议常量，完整值见公共头文件；名称中的 RO、RW、RW_AUTH、WO 分别表示只读、读写、授权写和只写。旧强度校准名称仅是存储区兼容别名，不表示 double 数组。SP210_DEFAULT_DEVICE_ADDRESS 为 0x01。

固件枚举：SP210_FirmwareOperation 的 UNKNOWN/UPGRADE/DOWNGRADE/REFLASH 为 0–3；SP210_FirmwareVerification 的 INCOMPLETE/MISMATCH/TRANSFER_UNCONFIRMED/ACCEPTANCE_UNCONFIRMED/VERIFIED 为 0–4；SP210_FirmwareStatus 从 SUCCEEDED=0 到 AWAITING_MANUAL_VERIFICATION=10，含 BUSY、PORT_UNAVAILABLE、QUERY_TIMEOUT、UNRECOGNIZED_REPLY、HANDSHAKE_TIMEOUT、TRANSFER_FAILED、DEVICE_CANCELLED、DEVICE_REJECTED、CANCELLED；SP210_FirmwareStage 从 QUERYING=0 到 AWAITING_MANUAL_VERIFICATION=5。包文本字段 FILE_NAME/NAME/VERSION 为 0–2；结果文本字段 MESSAGE/DEVICE_VERSION/DEVICE_BOARD/INTERRUPTION_REASON 为 0–3。完整名称见公共头文件。

回调约定

所有回调使用 SP210_CALL，在调用期间同步执行；user_data 由客户持有。连续采集 SP210_SpectrumCallback 收到的样本仅在回调期间有效，返回 0 停止；参数回调 SP210_AcquisitionParametersCallback 可更新下一帧参数，并返回 SP210_Result。SP210_MasterModeAction 在已解锁的独占会话内执行，返回结果会传回调用方。SP210_TransportSessionAction 只可在其线程及回调期间使用会话句柄。

固件进度回调包含阶段、已传输/总字节、是否可取消、UTF-8 消息；诊断回调接收 UTF-8 消息。自定义固件读取回调不得超过给定容量，无数据时返回 OK 并将收到字节数置 0；写入回调接收本次要发送的字节。回调及消息指针只在调用期间有效。.NET 须保持 delegate 和其上下文存活至同步操作返回，捕获回调内异常并转换为合适的返回值，不能向原生代码传播异常。

采集、固件及 transport 回调可能持有端口锁，其中设备操作会被拒绝；transport 回调可使用本会话的 exchange，取消操作可从另一线程调用。master 回调只允许操作已解锁的当前设备。不要在回调中等待另一个线程完成同端口操作。

客户端最小流程

1. 创建管理器、扫描、按索引取 UTF-8 端口并打开设备；每个成功获得的句柄对应销毁函数。
2. 设置 SP210_ProductInfo.struct_size 后读取设备信息，按产品允许范围选择积分时间。
3. 单帧采集一次获取强度；只有读取到有效起始像素和波长时才配对波长。
4. 当操作失败时立即读取当前线程错误；固件更新另需读取固件状态并在重启后核验版本。

示例代码见 C++ 控制台 和 .NET 10 控制台。

文档与头文件一致性检查

仓库维护者可在仓库根目录运行 python SP210DevDemo/ApiReference/check_coverage.py。检查脚本核对每个导出函数是否都有且只有一份 API 文档和 P/Invoke 声明，并核对文档中的 C 签名和参数条目。也可显式指定其他公共头文件路径。

SDK 接入与 Qt 再分发

返回 API 参考目录

安装与调用

从 GitHub Release 下载同版本的 SDK ZIP，按应用进程架构选取平台目录。SDK 提供纯 C ABI，动态加载 Qt Core 和 Qt Serial Port。此发布流程使用 **Qt 6.11.1** 社区版。

平台	构建与部署
Windows x64	C/C++ 引用 `include/sp210_c_api.h` 并链接 `lib/SP210SDK.lib`。C# 参考 `Native.cs` 中的 Cdecl P/Invoke。将 `Windows-x64/bin` 中的 SDK 和 Qt DLL 放在应用加载原生依赖的位置，通常是程序输出目录。按需安装 Visual C++ x64 Redistributable。
Linux x64 / ARM64	引用 `include/sp210_c_api.h` 并链接 `lib/libSP210SDK.so`。安装 ABI 兼容的 Qt 6 Core、Serial Port 和系统依赖，确保动态加载器能找到 SDK 和 Qt。

Windows 的 C++ 控制台 和 .NET 10 控制台 从 `dll/` 复制依赖至输出目录。构建后运行示例时传入 `--self-test`，可离线验证 SDK 加载和基本调用。

替换 Qt 运行库

准备与系统、架构及原运行库 ABI 兼容的 Qt Core、Qt Serial Port 和配套依赖，保留原文件用于恢复。SDK 和 Windows 示例输出均附有 `licenses/REPLACE-QT.txt`。

- 在 **Windows** 上，退出应用，在实际加载目录中换成兼容的 release 版 `Qt6Core.dll` 和 `Qt6SerialPort.dll`，然后重启。重建示例前，也要更新项目 `dll/` 中的副本。
- 在 **Linux** 上，将兼容 Qt 安装至可写目录，按实际安装位置设置共享库搜索路径。

```
qt_lib=/path/to/modified-qt/lib
  sdk_lib=/path/to/sdk/lib
  export LD_LIBRARY_PATH="$qt_lib:$sdk_lib${LD_LIBRARY_PATH:+:$LD_LIBRARY_PATH}"
  ldd "$sdk_lib/libSP210SDK.so"
  /path/to/your-application
```

在 `ldd` 输出中检查 `libQt6Core.so.6` 和 `libQt6SerialPort.so.6` 是否指向选定目录，并在同一环境启动程序。恢复原搜索路径即可切回原运行库。

使用 Qt 社区版交付自己的产品

交付应用、安装包或 SDK 副本前，核对随产品分发的 Qt 库、插件和第三方组件。

1. 使用提供 LGPLv3 选项的 Qt 模块，并保持动态库可由客户替换为 ABI 兼容的修改版本。交付条款应允许用户修改 Qt 部分，以及为调试这些修改而进行逆向工程。
2. 随每份分发提供显著的 Qt 使用、版权及许可声明，以及 LGPLv3 和 GPLv3 两份许可全文。保留相关第三方许可和通知。说明运行库的位置、替换方法及如何运行替换后的程序。
3. 如果把 Qt 二进制一并交付，分发方还需要按适用条款提供实际交付版本的完整对应源码，包含适用的修改和构建所需脚本。对网络下载方式，按 GPLv3 第 6(d) 条提供等效的免费源码下载途径，并维持其可用。随包的许可正文和替换说明只覆盖这些义务中的一部分。

Qt 社区版许可说明 · LGPLv3 第 4 条 · GPLv3 第 6 条

设备、管理、存储与校准

返回 API 参考目录

SP210_GetApiVersion

返回 Core/SDK 共用的版本，按 `(major << 16) | (minor << 8) | patch` 编码，每个分量占 8 位。

```
SP210_API uint32_t SP210_CALL SP210_GetApiVersion(void);
```

参数：无。

返回：按上述规则编码的版本号。

SP210_ResultString

将结果码转换为只读静态 UTF-8 文本指针；调用方不得释放或修改该指针。

```
SP210_API const char* SP210_CALL SP210_ResultString(SP210_Result result);
```

参数	说明
`result`	要转换的 `SP210_Result` 结果码。

返回：SDK 持有的只读 NUL 结尾 UTF-8 字符串指针。

SP210_GetLastError

取得当前线程最近一次错误的 UTF-8 文本，本函数不会覆盖该错误；其他成功的结果码接口会清除它。

```
SP210_API SP210_Result SP210_CALL SP210_GetLastError(char* buffer, size_t buffer_size, size_t* required_size);
```

参数	说明
`buffer`	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
`buffer_size`	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
`required_size`	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：`SP210_RESULT_OK` 表示已复制；缓冲区不足返回 `SP210_RESULT_BUFFER_TOO_SMALL`。本函数不会改变原错误文本。

SP210_ManagerCreate

创建一个设备管理器，成功后用 `SP210_ManagerDestroy` 释放。

```
SP210_API SP210_Result SP210_CALL SP210_ManagerCreate(SP210_ManagerHandle* manager);
```

参数	说明
`manager`	输出新建管理器句柄，成功后须销毁。

返回：`SP210_RESULT_OK` 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_ManagerDestroy

释放管理器句柄；已打开的设备句柄仍可使用，须分别销毁。

```
SP210_API SP210_Result SP210_CALL SP210_ManagerDestroy(SP210_ManagerHandle manager);
```

参数	说明
manager	有效管理器句柄。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_ManagerScan

同步扫描可用串口并更新管理器设备列表，返回发现数量；此操作访问串口，可能等待设备。

```
SP210_API SP210_Result SP210_CALL SP210_ManagerScan(SP210_ManagerHandle manager, size_t*
device_count);
```

参数	说明
manager	有效管理器句柄。
device_count	输出扫描发现的设备数量，类型为 size_t。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_ManagerGetPortName

按最近一次扫描结果的索引读取端口名称；两阶段查询期间应避免并发扫描修改列表。

```
SP210_API SP210_Result SP210_CALL SP210_ManagerGetPortName(SP210_ManagerHandle manager, size_t
index, char* buffer, size_t buffer_size, size_t* required_size);
```

参数	说明
manager	有效管理器句柄。
index	最近一次扫描结果中的零基设备索引。
buffer	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
buffer_size	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
required_size	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ManagerOpenDevice

探测指定端口并打开支持的设备，返回独立设备句柄；成功后须调用 SP210_DeviceDestroy。

```
SP210_API SP210_Result SP210_CALL SP210_ManagerOpenDevice(SP210_ManagerHandle manager, const char* port_name, SP210_DeviceHandle* device);
```

参数	说明
manager	有效管理器句柄。
port_name	以 NUL 结尾的 UTF-8 串口名称，例如 Windows 的 COM3。
device	输出设备句柄，成功后须销毁。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_ManagerOpenDeviceAtAddress

按指定设备地址探测端口并打开设备；成功后须销毁设备句柄。

```
SP210_API SP210_Result SP210_CALL SP210_ManagerOpenDeviceAtAddress(SP210_ManagerHandle manager, const char* port_name, uint8_t address, SP210_DeviceHandle* device);
```

参数	说明
manager	有效管理器句柄。
port_name	以 NUL 结尾的 UTF-8 串口名称，例如 Windows 的 COM3。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
device	输出设备句柄，成功后须销毁。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_ManagerGetCachedDevice

从管理器已有缓存获取设备句柄，不访问串口；未缓存时返回设备未找到。

```
SP210_API SP210_Result SP210_CALL SP210_ManagerGetCachedDevice(SP210_ManagerHandle manager, const char* port_name, SP210_DeviceHandle* device);
```

参数	说明
manager	有效管理器句柄。
port_name	以 NUL 结尾的 UTF-8 串口名称，例如 Windows 的 COM3。
device	输出设备句柄，成功后须销毁。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_ProbeDeviceAddress

探测指定端口上的设备地址，涉及串口通信。

```
SP210_API SP210_Result SP210_CALL SP210_ProbeDeviceAddress(const char* port_name, uint8_t* address);
```

参数	说明
port_name	以 NUL 结尾的 UTF-8 串口名称，例如 Windows 的 COM3。
address	输出探测到的设备地址。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_CheckDevice

探测端口是否连接支持的设备，通过字节标志返回结果。

```
SP210_API SP210_Result SP210_CALL SP210_CheckDevice(const char* port_name, uint8_t* is_supported);
```

参数	说明
port_name	以 NUL 结尾的 UTF-8 串口名称，例如 Windows 的 COM3。
is_supported	输出设备支持标志：1 是，0 否。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_ReadProductId

从指定端口及地址读取原始产品 ID 字节；两阶段调用可能重复访问设备。

```
SP210_API SP210_Result SP210_CALL SP210_ReadProductId(const char* port_name, uint8_t address, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
port_name	以 NUL 结尾的 UTF-8 串口名称，例如 Windows 的 COM3。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceReadModbusText

读取 Modbus 销售信息字段的 10 个寄存器，按 ASCII 解码并去除末尾的 0x00 和 0xFF 填充。每次调用（包括查询缓冲区大小）都会访问设备；可使用 21 字节缓冲区一次读取。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceReadModbusText(SP210_DeviceHandle device, int32_t item, char* buffer, size_t buffer_size, size_t* required_size);
```

参数	说明
device	有效设备句柄。
item	公共头文件中的 SP210_MODBUS_* 字段常量，值为 0、10、20、30、40 或 50。
buffer	调用方提供的输出缓冲区，可传 NULL 查询大小。
buffer_size	输出缓冲区容量，单位字节，包含末尾 NUL 的空间。
required_size	输出所需缓冲区大小，单位字节，包括末尾 NUL；不得为 NULL。

返回：成功返回 SP210_RESULT_OK；缓冲区不足返回 SP210_RESULT_BUFFER_TOO_SMALL，其他失败返回对应结果码。

SP210_DeviceWriteModbusText

写入 Modbus 销售信息字段的完整 10 个寄存器；剩余字节填充 0x00。每次调用发送一次写入。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceWriteModbusText(SP210_DeviceHandle device, int32_t item, const char* value);
```

参数	说明
device	有效设备句柄。
item	公共头文件中的 SP210_MODBUS_* 字段常量，值为 0、10、20、30、40 或 50。
value	NUL 结尾的 ASCII 字符串，最多 20 个非 NUL 字节，不得为 NULL。

返回：成功返回 SP210_RESULT_OK；失败返回对应结果码。

SP210_DeviceDestroy

释放设备句柄，不得在销毁后继续使用该句柄。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceDestroy(SP210_DeviceHandle device);
```

参数	说明
device	有效设备句柄。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetPortName

读取设备关联的 UTF-8 端口名称。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetPortName(SP210_DeviceHandle device, char*
buffer, size_t buffer_size, size_t* required_size);
```

参数	说明
device	有效设备句柄。
buffer	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
buffer_size	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
required_size	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceGetAddress

读取设备对象的通信地址。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetAddress(SP210_DeviceHandle device, uint8_t*
address);
```

参数	说明
device	有效设备句柄。
address	输出设备句柄保存的通信地址。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetProductInfo

取得设备缓存的产品信息；调用前初始化结构体 struct_size。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetProductInfo(SP210_DeviceHandle device,
SP210_ProductInfo* info);
```

参数	说明
device	有效设备句柄。
info	调用方输出结构体指针；调用前将 struct_size 设为结构体大小。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetProductId

取得设备缓存的原始产品 ID 字节。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetProductId(SP210_DeviceHandle device, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
device	有效设备句柄。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ProductInfoFromId

离线解析 1–4 字节产品 ID 为产品信息；调用前初始化 info.struct_size。

```
SP210_API SP210_Result SP210_CALL SP210_ProductInfoFromId(const uint8_t* product_id, size_t length, SP210_ProductInfo* info);
```

参数	说明
product_id	输入原始产品 ID 字节；长度参数以字节计。
length	输入产品 ID 字节数，范围 1–4。
info	调用方输出结构体指针；调用前将 struct_size 设为结构体大小。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetSerialNumber

从设备读取序列号并更新缓存。两阶段查询可能执行两次设备读取。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetSerialNumber(SP210_DeviceHandle device, char* buffer, size_t buffer_size, size_t* required_size);
```

参数	说明
device	有效设备句柄。
buffer	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
buffer_size	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
required_size	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceGetIntegrationTime

读取积分时间，单位微秒。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetIntegrationTime(SP210_DeviceHandle device,
int64_t* integration_time_us);
```

参数	说明
device	有效设备句柄。
integration_time_us	输出积分时间，单位微秒。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetIntegrationTime

设置积分时间，单位微秒；须处于设备产品信息定义的允许范围。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetIntegrationTime(SP210_DeviceHandle device,
int64_t integration_time_us);
```

参数	说明
device	有效设备句柄。
integration_time_us	积分时间，单位微秒；采集时须处于产品支持范围。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetSpectrumStartingPixel

读取输出窗口的物理起始像素，用于将样本索引映射到完整波长轴。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetSpectrumStartingPixel(SP210_DeviceHandle device,
int32_t* starting_pixel);
```

参数	说明
device	有效设备句柄。
starting_pixel	输出设备光谱窗口的物理起始像素，零基。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetWavelengths

取得从物理像素 0 开始的波长数组；样本 i 对应 starting_pixel + i，失败或越界时不要猜测波长。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetWavelengths(SP210_DeviceHandle device, double* values, size_t capacity, size_t* value_count);
```

参数	说明
device	有效设备句柄。
values	调用方提供的数值输出数组，可传 NULL 查询元素数量。
capacity	输出 values 数组的容量，单位为元素个数。
value_count	输出数组实际或所需的元素数量，单位为元素。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceAcquireSpectrum

按指定积分时间、平均次数和平滑等级采集一帧。每次调用包括空缓冲查询都会采集新帧；需要保留同一帧时预分配 32767 个 int32_t 并只调用一次。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceAcquireSpectrum(SP210_DeviceHandle device, int64_t integration_time_us, int32_t average_times, int32_t smoothing_level, int32_t* values, size_t capacity, size_t* value_count);
```

参数	说明
device	有效设备句柄。
integration_time_us	积分时间，单位微秒；采集时须处于产品支持范围。
average_times	平均次数；设备采集要求 1–128。
smoothing_level	平滑等级，0–10。
values	调用方提供的数值输出数组，可传 NULL 查询元素数量。
capacity	输出 values 数组的容量，单位为元素个数。
value_count	输出数组实际或所需的元素数量，单位为元素。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceAcquireSpectrumForProduct

按显式指定的产品 ID 采集一帧；其他参数、两阶段查询和单帧保留规则与 SP210_DeviceAcquireSpectrum 相同。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceAcquireSpectrumForProduct(SP210_DeviceHandle device, const uint8_t* product_id, size_t product_id_length, int64_t integration_time_us, int32_t average_times, int32_t smoothing_level, int32_t* values, size_t capacity, size_t* value_count);
```

参数	说明
device	有效设备句柄。
product_id	输入原始产品 ID 字节；长度参数以字节计。
product_id_length	输入产品 ID 字节数，范围 1–4。
integration_time_us	积分时间，单位微秒；采集时须处于产品支持范围。
average_times	平均次数；设备采集要求 1–128。
smoothing_level	平滑等级，0–10。
values	调用方提供的数值输出数组，可传 NULL 查询元素数量。
capacity	输出 values 数组的容量，单位为元素个数。
value_count	输出数组实际或所需的元素数量，单位为元素。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceGetADResolution

读取ADC 分辨率，单位位。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetADResolution(SP210_DeviceHandle device, int32_t* value);
```

参数	说明
device	有效设备句柄。
value	输出ADC 分辨率，单位位。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetFilterState

读取滤波器状态，0 或 1。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetFilterState(SP210_DeviceHandle device, int32_t* value);
```

参数	说明
device	有效设备句柄。
value	输出滤波器状态，0 或 1。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetCuterState

读取去底器状态，0 或 1。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetCuterState(SP210_DeviceHandle device, int32_t* value);
```

参数	说明
device	有效设备句柄。
value	输出去底器状态，0 或 1。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetXenonLampTriggerDelay

读取氙灯触发延迟，单位毫秒。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetXenonLampTriggerDelay(SP210_DeviceHandle device, int32_t* value);
```

参数	说明
device	有效设备句柄。
value	输出氙灯触发延迟，单位毫秒。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetTEC0Status

读取TEC0 状态码。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetTEC0Status(SP210_DeviceHandle device, int32_t* value);
```

参数	说明
device	有效设备句柄。
value	输出TEC0 状态码。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetCCDOutputPixels

读取CCD 输出像素数。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetCCDOutputPixels(SP210_DeviceHandle device,
int32_t* value);
```

参数	说明
device	有效设备句柄。
value	输出CCD 输出像素数。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetSpectrumPixelLength

读取光谱输出窗口像素长度。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetSpectrumPixelLength(SP210_DeviceHandle device,
int32_t* value);
```

参数	说明
device	有效设备句柄。
value	输出光谱输出窗口像素长度。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetAverageTimes

读取平均次数，1–128。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetAverageTimes(SP210_DeviceHandle device, int32_t*
value);
```

参数	说明
device	有效设备句柄。
value	输出平均次数，1–128。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetSmoothingCoefficient

读取平滑等级，0–10。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetSmoothingCoefficient(SP210_DeviceHandle device,
int32_t* value);
```

参数	说明
device	有效设备句柄。
value	输出平滑等级，0–10。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetRS232BaudRate

读取RS232 波特率，支持 2400、4800、9600、19200、38400、57600、115200。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetRS232BaudRate(SP210_DeviceHandle device,
int32_t* value);
```

参数	说明
device	有效设备句柄。
value	输出RS232 波特率，支持 2400、4800、9600、19200、38400、57600、115200。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetSIO3Frequency

读取SIO3 频率，0–1000 Hz。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetSIO3Frequency(SP210_DeviceHandle device,
int32_t* value);
```

参数	说明
device	有效设备句柄。
value	输出SIO3 频率，0–1000 Hz。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetSIO3DutyCycle

读取SIO3 占空比，0–100%。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetSIO3DutyCycle(SP210_DeviceHandle device,
int32_t* value);
```

参数	说明
device	有效设备句柄。
value	输出SIO3 占空比，0–100%。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetSIO3Pulse

读取SIO3 脉冲数，0–999999。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetSIO3Pulse(SP210_DeviceHandle device, int32_t*
value);
```

参数	说明
device	有效设备句柄。
value	输出SIO3 脉冲数，0–999999。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetStartingWavelength

读取设备配置的起始波长，单位为整数纳米。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetStartingWavelength(SP210_DeviceHandle device,
int32_t* value);
```

参数	说明
device	有效设备句柄。
value	接收波长的非空指针，单位纳米。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetEndWavelength

读取设备配置的结束波长，单位为整数纳米。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetEndWavelength(SP210_DeviceHandle device,
int32_t* value);
```

参数	说明
device	有效设备句柄。
value	接收波长的非空指针，单位纳米。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetStartingWavelength

设置设备配置的起始波长，单位为整数纳米。必须在当前设备的 SP210_DeviceWithMasterMode 回调中调用。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetStartingWavelength(SP210_DeviceHandle device,
int32_t value);
```

参数	说明
device	有效设备句柄。
value	波长，单位纳米；有效范围为 0–65535。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetEndWavelength

设置设备配置的结束波长，单位为整数纳米。必须在当前设备的 SP210_DeviceWithMasterMode 回调中调用。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetEndWavelength(SP210_DeviceHandle device, int32_t
value);
```

参数	说明
device	有效设备句柄。
value	波长，单位纳米；有效范围为 0–65535。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetSensorGain

读取传感器增益档位，0–3。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetSensorGain(SP210_DeviceHandle device, int32_t* value);
```

参数	说明
device	有效设备句柄。
value	输出传感器增益档位，0–3。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetTECChannelCount

读取TEC 通道数，0–4。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetTECChannelCount(SP210_DeviceHandle device, int32_t* value);
```

参数	说明
device	有效设备句柄。
value	输出TEC 通道数，0–4。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetAverageTimes

设置平均次数，1–128。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetAverageTimes(SP210_DeviceHandle device, int32_t value);
```

参数	说明
device	有效设备句柄。
value	输入平均次数，1–128。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetSmoothingCoefficient

设置平滑等级，0–10。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetSmoothingCoefficient(SP210_DeviceHandle device, int32_t value);
```

参数	说明
device	有效设备句柄。
value	输入平滑等级，0–10。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetRS232BaudRate

设置RS232 波特率，支持 2400、4800、9600、19200、38400、57600、115200。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetRS232BaudRate(SP210_DeviceHandle device, int32_t value);
```

参数	说明
device	有效设备句柄。
value	输入RS232 波特率，支持 2400、4800、9600、19200、38400、57600、115200。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetSIO3Frequency

设置SIO3 频率，0–1000 Hz。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetSIO3Frequency(SP210_DeviceHandle device, int32_t value);
```

参数	说明
device	有效设备句柄。
value	输入SIO3 频率，0–1000 Hz。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetSIO3DutyCycle

设置SIO3 占空比，0–100%。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetSIO3DutyCycle(SP210_DeviceHandle device, int32_t value);
```

参数	说明
device	有效设备句柄。
value	输入SIO3 占空比，0–100%。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetSIO3Pulse

设置SIO3 脉冲数，0–999999。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetSIO3Pulse(SP210_DeviceHandle device, int32_t value);
```

参数	说明
device	有效设备句柄。
value	输入SIO3 脉冲数，0–999999。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetSensorGain

设置传感器增益档位，0–3。写入须在当前设备 master-mode 回调内。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetSensorGain(SP210_DeviceHandle device, int32_t value);
```

参数	说明
device	有效设备句柄。
value	输入传感器增益档位，0–3。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetTECChannelCount

设置TEC 通道数，0–4。写入须在当前设备 master-mode 回调内。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetTECChannelCount(SP210_DeviceHandle device,
int32_t value);
```

参数	说明
device	有效设备句柄。
value	输入TEC 通道数，0–4。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetTemperature

读取设备温度，单位摄氏度。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetTemperature(SP210_DeviceHandle device, double*
value);
```

参数	说明
device	有效设备句柄。
value	输出设备温度，单位摄氏度。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetHumidity

读取设备相对湿度，单位百分比。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetHumidity(SP210_DeviceHandle device, double*
value);
```

参数	说明
device	有效设备句柄。
value	输出设备相对湿度，单位百分比。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetSupplyVoltage

读取供电电压，单位伏特。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetSupplyVoltage(SP210_DeviceHandle device, double* value);
```

参数	说明
device	有效设备句柄。
value	输出供电电压，单位伏特。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetADReferenceVoltage

读取ADC 参考电压，单位伏特。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetADReferenceVoltage(SP210_DeviceHandle device, double* value);
```

参数	说明
device	有效设备句柄。
value	输出ADC 参考电压，单位伏特。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetTEC0SetTemperature

读取TEC0 设定温度，单位摄氏度。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetTEC0SetTemperature(SP210_DeviceHandle device, double* value);
```

参数	说明
device	有效设备句柄。
value	输出TEC0 设定温度，单位摄氏度。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetTEC0ActualTemperature

读取TEC0 实际温度，单位摄氏度。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetTEC0ActualTemperature(SP210_DeviceHandle device, double* value);
```

参数	说明
device	有效设备句柄。
value	输出TEC0 实际温度，单位摄氏度。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetTEC0HeatSinkTemperature

读取TEC0 散热器温度，单位摄氏度。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetTEC0HeatSinkTemperature(SP210_DeviceHandle device, double* value);
```

参数	说明
device	有效设备句柄。
value	输出TEC0 散热器温度，单位摄氏度。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetTEC0Voltage

读取TEC0 电压，单位伏特。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetTEC0Voltage(SP210_DeviceHandle device, double* value);
```

参数	说明
device	有效设备句柄。
value	输出TEC0 电压，单位伏特。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetTEC0Current

读取TEC0 电流。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetTEC0Current(SP210_DeviceHandle device, double* value);
```

参数	说明
device	有效设备句柄。
value	输出TEC0 电流。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetTEC0PowerVoltage

读取TEC0 电源电压，单位伏特。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetTEC0PowerVoltage(SP210_DeviceHandle device, double* value);
```

参数	说明
device	有效设备句柄。
value	输出TEC0 电源电压，单位伏特。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetCMOSSaturationVoltage

读取CMOS 饱和电压，单位伏特。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetCMOSSaturationVoltage(SP210_DeviceHandle device, double* value);
```

参数	说明
device	有效设备句柄。
value	输出CMOS 饱和电压，单位伏特。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetHardwareTriggerAutoMode

读取硬件触发自动模式，0 或 1。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetHardwareTriggerAutoMode(SP210_DeviceHandle device, uint8_t* value);
```

参数	说明
device	有效设备句柄。
value	输出硬件触发自动模式，0 或 1。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetTEC0SwitchStatus

读取TEC0 开关状态，0 或 1。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetTEC0SwitchStatus(SP210_DeviceHandle device, uint8_t* value);
```

参数	说明
device	有效设备句柄。
value	输出TEC0 开关状态，0 或 1。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetSpectralDataSequentialModeReverse

读取光谱数据顺序反转状态，0 或 1。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetSpectralDataSequentialModeReverse(SP210_DeviceHandle device, uint8_t* value);
```

参数	说明
device	有效设备句柄。
value	输出光谱数据顺序反转状态，0 或 1。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetHardwareTriggerAutoMode

设置硬件触发自动模式，0 或 1。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetHardwareTriggerAutoMode(SP210_DeviceHandle device, uint8_t value);
```

参数	说明
device	有效设备句柄。
value	输入硬件触发自动模式，0 或 1。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetTEC0SwitchStatus

设置TEC0 开关状态，0 或 1。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetTEC0SwitchStatus(SP210_DeviceHandle device, uint8_t value);
```

参数	说明
device	有效设备句柄。
value	输入TEC0 开关状态，0 或 1。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetSpectralDataSequentialModeReverse

设置光谱数据顺序反转状态，0 或 1。写入须在当前设备 master-mode 回调内。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetSpectralDataSequentialModeReverse(SP210_DeviceHandle device, uint8_t value);
```

参数	说明
device	有效设备句柄。
value	输入光谱数据顺序反转状态，0 或 1。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetFirmwareVersion

读取主板固件版本。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetFirmwareVersion(SP210_DeviceHandle device, char* buffer, size_t capacity, size_t* required_size);
```

参数	说明
device	有效设备句柄。
buffer	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
capacity	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
required_size	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceGetHardwareVersion

读取主板硬件版本。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetHardwareVersion(SP210_DeviceHandle device, char* buffer, size_t capacity, size_t* required_size);
```

参数	说明
device	有效设备句柄。
buffer	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
capacity	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
required_size	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceGetCurrentCommunicationInterface

读取当前通信接口。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetCurrentCommunicationInterface(SP210_DeviceHandle device, char* buffer, size_t capacity, size_t* required_size);
```

参数	说明
device	有效设备句柄。
buffer	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
capacity	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
required_size	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceGetTEC0ChannelReopened

读取TEC0 通道重新打开状态文本。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetTEC0ChannelReopened(SP210_DeviceHandle device, char* buffer, size_t capacity, size_t* required_size);
```

参数	说明
device	有效设备句柄。
buffer	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
capacity	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
required_size	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceGetTEC0FirmwareVersion

读取TEC0 固件版本。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetTEC0FirmwareVersion(SP210_DeviceHandle device, char* buffer, size_t capacity, size_t* required_size);
```

参数	说明
device	有效设备句柄。
buffer	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
capacity	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
required_size	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceGetTEC0HardwareVersion

读取TEC0 硬件版本。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetTEC0HardwareVersion(SP210_DeviceHandle device, char* buffer, size_t capacity, size_t* required_size);
```

参数	说明
device	有效设备句柄。
buffer	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
capacity	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
required_size	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceGetCachedSerialNumber

读取已缓存序列号，不触发设备读取，缓存为空时返回空字符串。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetCachedSerialNumber(SP210_DeviceHandle device, char* buffer, size_t capacity, size_t* required_size);
```

参数	说明
device	有效设备句柄。
buffer	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
capacity	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
required_size	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceGetProductName

读取设备缓存的产品名称，不触发设备读取。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetProductName(SP210_DeviceHandle device, char* buffer, size_t capacity, size_t* required_size);
```

参数	说明
device	有效设备句柄。
buffer	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
capacity	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
required_size	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceGetFixedDelayTriggerTime

读取固定延时触发时间，单位微秒。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetFixedDelayTriggerTime(SP210_DeviceHandle device, int64_t* microseconds);
```

参数	说明
device	有效设备句柄。
microseconds	输出固定触发延时，单位微秒。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetCMOSSaturationVoltage

设置CMOS 饱和电压，单位伏特。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetCMOSSaturationVoltage(SP210_DeviceHandle device, double value);
```

参数	说明
device	有效设备句柄。
value	输入CMOS 饱和电压，单位伏特。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetSerialNumber

设置序列号，要求 1–40 个 ASCII 字母或数字；必须在当前设备的 master-mode 回调中调用。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetSerialNumber(SP210_DeviceHandle device, const char* serial_number);
```

参数	说明
device	有效设备句柄。
serial_number	要写入的 NUL 结尾 UTF-8 序列号；要求 1–40 个 ASCII 字母或数字。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceSetSpectrumOutputWindow

设置光谱物理起始像素和长度；按设备需要提供密码并执行受保护写入。写入或验证失败时参数可能已部分改变，须重新读取。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetSpectrumOutputWindow(SP210_DeviceHandle device, int32_t starting_pixel, int32_t pixel_length, const char* password);
```

参数	说明
device	有效设备句柄。
starting_pixel	物理起始像素，零基。
pixel_length	光谱输出窗口的像素数量。
password	可选 UTF-8 解锁密码；按设备窗口变更需要提供。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetUserInputIO

读取输入 IO 的原始 16 位值及 IO15/16/17 电平。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetUserInputIO(SP210_DeviceHandle device,
SP210_InputIOState* state);
```

参数	说明
device	有效设备句柄。
state	输出 IO 状态结构体指针。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetUserOutputIO

读取输出 IO 的原始 16 位值及 IO2/3/4 电平。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetUserOutputIO(SP210_DeviceHandle device,
SP210_OutputIOState* state);
```

参数	说明
device	有效设备句柄。
state	输出 IO 状态结构体指针。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetUserInputIOBytes

读取输入 IO 的原始协议字节。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetUserInputIOBytes(SP210_DeviceHandle device,
uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
device	有效设备句柄。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceGetUserOutputIOBytes

读取输出 IO 的原始协议字节。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetUserOutputIOBytes(SP210_DeviceHandle device, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
device	有效设备句柄。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceSetUserOutputIO

根据输出 IO 状态结构体写入用户输出 IO；io2_high、io3_high、io4_high 决定 raw_value 的低 3 位，其余位保留。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetUserOutputIO(SP210_DeviceHandle device, const SP210_OutputIOState* state);
```

参数	说明
device	有效设备句柄。
state	输入输出 IO 状态结构体指针。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceWithMasterMode

在独占串口会话中使用密码解锁，并同步执行回调。回调只能对当前设备进行允许的操作；从解锁请求前计时，超 10 秒不再发送受保护写入，写入不自动重试。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceWithMasterMode(SP210_DeviceHandle device, const char* password, SP210_MasterModeAction action, void* user_data);
```

参数	说明
device	有效设备句柄。
password	NUL 结尾的 UTF-8 解锁密码。
action	同步执行受保护设备操作的回调，不可为 NULL。
user_data	原样传入回调的调用方上下文，可为 NULL；在同步调用返回前须保持有效。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceReadUserStorage

读取 Part1–Part8 中的一个 2048 字节块，内容为不解释的原始数据。空缓冲区查询只返回大小，不访问串口；串口忙立即失败。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceReadUserStorage(SP210_DeviceHandle device,
SP210_UserStorageBlock block, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
device	有效设备句柄。
block	存储区枚举 SP210_USER_STORAGE_PART1–PART8（0x52–0x59）。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceUnlockAndWriteUserStorage

在同一独占会话内每次重新解锁并写入一个完整 2048 字节块。10 秒窗口从解锁请求前计时，不自动重试，串口忙立即失败。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceUnlockAndWriteUserStorage(SP210_DeviceHandle
device, SP210_UserStorageBlock block, const uint8_t* bytes, size_t length, const char* password);
```

参数	说明
device	有效设备句柄。
block	存储区枚举 SP210_USER_STORAGE_PART1–PART8（0x52–0x59）。
bytes	输入字节数组，长度由 length 指定。
length	输入块长度，必须严格为 2048 字节。
password	NUL 结尾的 UTF-8 解锁密码。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceGetWavelengthCalibration

读取波长校准区的系数数组；这是独立于 Part1–Part8 的 0x51 功能项。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetWavelengthCalibration(SP210_DeviceHandle device, double* values, size_t capacity, size_t* value_count);
```

参数	说明
device	有效设备句柄。
values	调用方提供的数值输出数组，可传 NULL 查询元素数量。
capacity	输出 values 数组的容量，单位为元素个数。
value_count	输出数组实际或所需的元素数量，单位为元素。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceSetWavelengthCalibration

低级校准写入口，接受 1–8 个有限系数；只能在当前设备 master-mode 回调中调用。验证失败时设备可能已经改变。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceSetWavelengthCalibration(SP210_DeviceHandle device, const double* values, size_t count);
```

参数	说明
device	有效设备句柄。
values	输入数值数组，元素数量由 count 指定。
count	输入数组元素数量；校准写入要求 1–8 个有限系数。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceUnlockAndWriteWavelengthCalibration

先成功读取设备序列号，然后使用密码在同一会话解锁、写入并验证 1–8 个有限系数。actual 至少容纳 8 个 double，缓冲不足时不写入；验证失败时设备可能已改变。

```
SP210_API SP210_Result SP210_CALL
SP210_DeviceUnlockAndWriteWavelengthCalibration(SP210_DeviceHandle device, const double* values,
size_t count, const char* password, double* actual, size_t capacity, size_t* value_count);
```

参数	说明
device	有效设备句柄。
values	输入数值数组，元素数量由 count 指定。
count	输入数组元素数量；校准写入要求 1–8 个有限系数。
password	NUL 结尾的 UTF-8 解锁密码。
actual	调用方提供的写后读回系数数组；容量至少为 8 个 double。
capacity	actual 容量，单位 double 元素，至少为 8 才会执行写入。
value_count	输出数组实际或所需的元素数量，单位为元素。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_DeviceGetHardwareTriggeredSpectrum

读取硬件触发光谱并可平滑处理；每次调用均读取设备，空缓冲查询也会读取，若需要同一帧请预分配足够空间。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceGetHardwareTriggeredSpectrum(SP210_DeviceHandle
device, int32_t smoothing_level, int32_t* values, size_t capacity, size_t* value_count);
```

参数	说明
device	有效设备句柄。
smoothing_level	平滑等级，0–10。
values	调用方提供的数值输出数组，可传 NULL 查询元素数量。
capacity	输出 values 数组的容量，单位为元素个数。
value_count	输出数组实际或所需的元素数量，单位为元素。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

协议、传输与光谱计算

返回 API 参考目录

SP210_CalculateWavelengths

离线根据有限多项式系数、像素数量和起始物理像素计算波长数组，不访问设备。

```
SP210_API SP210_Result SP210_CALL SP210_CalculateWavelengths(const double* coefficients, size_t coefficient_count, int32_t pixel_count, int32_t starting_pixel, double* values, size_t capacity, size_t* value_count);
```

参数	说明
coefficients	输入有限 double 多项式系数数组。
coefficient_count	输入有限系数的数量，至少为 1。
pixel_count	需要计算的像素数量，必须大于 0。
starting_pixel	物理起始像素，零基。
values	调用方提供的数值输出数组，可传 NULL 查询元素数量。
capacity	输出 values 数组的容量，单位为元素个数。
value_count	输出数组实际或所需的元素数量，单位为元素。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_SmoothSpectrum

离线按指定等级处理输入光谱，输出 int32 样本数组，不访问设备。

```
SP210_API SP210_Result SP210_CALL SP210_SmoothSpectrum(const int32_t* input, size_t count, int32_t level, int32_t* values, size_t capacity, size_t* value_count);
```

参数	说明
input	输入 int32 光谱数组，数量由 count 指定。
count	输入光谱样本数量，单位为 int32 元素。
level	平滑等级，0–10。
values	调用方提供的数值输出数组，可传 NULL 查询元素数量。
capacity	输出 values 数组的容量，单位为元素个数。
value_count	输出数组实际或所需的元素数量，单位为元素。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ProtocolReadCommand

离线生成指定功能项的读命令及 CRC。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolReadCommand(uint16_t item, uint8_t address,
uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
item	协议功能项编号，参考 SP210_FunctionItem。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ProtocolWriteBytes

离线生成指定功能项和宽度的二进制写命令及 CRC；width 为 0–65535，输入长度至少为 width，取前 width 字节。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolWriteBytes(uint16_t item, int32_t width, const
uint8_t* bytes, size_t length, uint8_t address, uint8_t* buffer, size_t capacity, size_t*
required_size);
```

参数	说明
item	协议功能项编号，参考 SP210_FunctionItem。
width	协议写入的数据宽度，单位字节。
bytes	输入字节数组，长度由 length 指定。
length	输入字节数组的长度，单位字节。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ProtocolWriteString

离线生成指定功能项和宽度的 UTF-8 字符串写命令及 CRC；width 为 0–65535，文本编码后不得超过宽度，不足的部分补零。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolWriteString(uint16_t item, int32_t width, const char* text, uint8_t address, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
item	协议功能项编号，参考 SP210_FunctionItem。
width	协议写入的数据宽度，单位字节。
text	NUL 结尾的 UTF-8 输入文本。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ProtocolWriteInt32

离线生成 int32 写命令及 CRC；width 为 1–4 字节，数值必须能按该宽度表示。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolWriteInt32(uint16_t item, int32_t width, int32_t value, uint8_t address, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
item	协议功能项编号，参考 SP210_FunctionItem。
width	协议写入的数据宽度，单位字节。
value	要编码到写命令中的输入数值。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ProtocolWriteInt64

离线生成 int64 写命令及 CRC；width 为 1–8 字节，数值必须能按该宽度表示。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolWriteInt64(uint16_t item, int32_t width, int64_t value, uint8_t address, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
item	协议功能项编号，参考 SP210_FunctionItem。
width	协议写入的数据宽度，单位字节。
value	要编码到写命令中的输入数值。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ProtocolWriteDoubles

离线将 double 数组编码为写命令及 CRC；width 为不超过 65535 的 8 字节倍数，元素数不能超过 width/8，不足的部分补零。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolWriteDoubles(uint16_t item, int32_t width, const double* values, size_t count, uint8_t address, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
item	协议功能项编号，参考 SP210_FunctionItem。
width	协议写入的数据宽度，单位字节。
values	输入数值数组，元素数量由 count 指定。
count	输入数值数组的元素数量。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ProtocolUnlockCommand

离线生成主控模式解锁命令；只生成字节，不打开设备或执行解锁。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolUnlockCommand(const char* password, uint8_t address, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
password	NUL 结尾的 UTF-8 解锁密码。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ProtocolAcquireCommand

离线生成光谱采集命令；积分微秒编码范围为 0 到 UINT32_MAX，平均次数由 Core 规范化到 1–128。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolAcquireCommand(int64_t integration_time_us, int32_t average_times, uint8_t address, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
integration_time_us	积分时间，单位微秒；采集时须处于产品支持范围。
average_times	平均次数；设备采集要求 1–128。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ProtocolValidateResponse

离线检查响应与请求的帧关联及校验，通过 valid 返回真伪。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolValidateResponse(const uint8_t* response, size_t response_length, const uint8_t* request, size_t request_length, uint8_t* valid);
```

参数	说明
response	输入完整响应帧字节，长度由相应长度参数指定。
response_length	响应帧的字节数。
request	输入完整请求帧，长度由 request_length 指定。
request_length	请求帧的字节数。
valid	输出验证结果字节：1 表示有效，0 表示无效。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_ProtocolParseReadInt32

离线验证并解析读取响应中的 int32 值，协议无效时返回协议错误。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolParseReadInt32(const uint8_t* response, size_t length, uint8_t address, int32_t* value);
```

参数	说明
response	输入完整响应帧字节，长度由相应长度参数指定。
length	输入字节数组的长度，单位字节。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
value	输出解析得到的数值。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_ProtocolParseReadInt64

离线验证并解析读取响应中的 int64 值。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolParseReadInt64(const uint8_t* response, size_t length, uint8_t address, int64_t* value);
```

参数	说明
response	输入完整响应帧字节，长度由相应长度参数指定。
length	输入字节数组的长度，单位字节。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
value	输出解析得到的数值。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_ProtocolParseReadDouble

离线验证并解析读取响应中的 double 值。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolParseReadDouble(const uint8_t* response, size_t length, uint8_t address, double* value);
```

参数	说明
response	输入完整响应帧字节，长度由相应长度参数指定。
length	输入字节数组的长度，单位字节。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
value	输出解析得到的数值。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_ProtocolParseReadString

离线验证并解析读取响应为 UTF-8 文本。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolParseReadString(const uint8_t* response, size_t length, uint8_t address, char* buffer, size_t capacity, size_t* required_size);
```

参数	说明
response	输入完整响应帧字节，长度由相应长度参数指定。
length	输入字节数组的长度，单位字节。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
buffer	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
capacity	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
required_size	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ProtocolParseReadBytes

离线验证并解析读取响应的原始数据字节。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolParseReadBytes(const uint8_t* response, size_t length, uint8_t address, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
response	输入完整响应帧字节，长度由相应长度参数指定。
length	输入字节数组的长度，单位字节。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ProtocolParseError

离线解析设备错误帧，输出功能码、是否错误和错误文本；is_error 为 0 本身不代表其他响应校验已成功。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolParseError(const uint8_t* response, size_t length, uint8_t address, uint8_t* function_code, uint8_t* is_error, char* buffer, size_t capacity, size_t* required_size);
```

参数	说明
response	输入完整响应帧字节，长度由相应长度参数指定。
length	输入字节数组的长度，单位字节。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
function_code	输出错误帧中的功能码。
is_error	输出是否解析出错误帧：1 是，0 否。
buffer	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
capacity	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
required_size	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ProtocolUserStorageWriteCommand

离线生成 Part1–Part8 之一的完整 2048 字节块写帧，不执行解锁或发送。真实写入请使用专用安全接口。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolUserStorageWriteCommand(SP210_UserStorageBlock block, const uint8_t* data, size_t length, uint8_t address, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
block	存储区枚举 SP210_USER_STORAGE_PART1–PART8（0x52–0x59）。
data	输入存储块字节数组，必须为 2048 字节。
length	输入块长度，必须严格为 2048 字节。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ProtocolParseUserStorageResponse

离线校验并解析指定存储区的响应，输出完整 2048 字节。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolParseUserStorageResponse(SP210_UserStorageBlock block, const uint8_t* response, size_t length, uint8_t address, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
block	存储区枚举 SP210_USER_STORAGE_PART1–PART8（0x52–0x59）。
response	输入完整响应帧字节，长度由相应长度参数指定。
length	输入字节数组的长度，单位字节。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_ProtocolValidateUserStorageWriteAck

离线校验指定地址的存储区写入确认帧。

```
SP210_API SP210_Result SP210_CALL SP210_ProtocolValidateUserStorageWriteAck(const uint8_t*
response, size_t length, uint8_t address);
```

参数	说明
response	输入完整响应帧字节，长度由相应长度参数指定。
length	输入字节数组的长度，单位字节。
address	设备地址（默认 0x01）；使用与实际设备一致的地址。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_TransportTransact

执行单次完整 SP210 帧事务；空缓冲查询返回响应上限且不访问串口。请求必须是有效 SP210 帧，不能直接发送 Part1–Part8 存储帧。

```
SP210_API SP210_Result SP210_CALL SP210_TransportTransact(const char* port_name, const
SP210_TransportTransaction* transaction, uint8_t* response, size_t capacity, size_t*
required_size);
```

参数	说明
port_name	以 NUL 结尾的 UTF-8 串口名称，例如 Windows 的 COM3。
transaction	完整事务参数指针，先设置 struct_size；请求帧及指针在本次调用期间保持有效。
response	调用方提供的响应字节缓冲区；NULL 查询上限时不发送事务。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_TransportWithSession

独占指定端口，在同步回调中复用串口会话；会话句柄仅在回调期间及回调线程有效。

```
SP210_API SP210_Result SP210_CALL SP210_TransportWithSession(const char* port_name,
SP210_TransportSessionAction action, void* user_data);
```

参数	说明
port_name	以 NUL 结尾的 UTF-8 串口名称，例如 Windows 的 COM3。
action	同步会话回调，不可为 NULL；返回结果传回调用方。
user_data	原样传入回调的调用方上下文，可为 NULL；在同步调用返回前须保持有效。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_TransportSessionExchange

在会话回调中进行一次事务；缓冲查询不访问串口，限制与 SP210_TransportTransact 相同。

```
SP210_API SP210_Result SP210_CALL SP210_TransportSessionExchange(SP210_TransportSessionHandle session, const SP210_TransportTransaction* transaction, uint8_t* response, size_t capacity, size_t* required_size);
```

参数	说明
session	当前会话的借用句柄，只在创建它的回调线程及回调期间有效。
transaction	完整事务参数指针，先设置 struct_size；请求帧及指针在本次调用期间保持有效。
response	调用方提供的响应字节缓冲区；NULL 查询上限时不发送事务。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

取消与连续采集

返回 API 参考目录

SP210_CancelCreate

创建可用于连续采集或固件操作的取消令牌，成功后使用 SP210_CancelDestroy。

```
SP210_API SP210_Result SP210_CALL SP210_CancelCreate(SP210_CancelHandle* token);
```

参数	说明
token	输出新建取消令牌，成功后须销毁。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_CancelRequest

请求取消，可由其他线程调用；实际返回时间取决于操作阶段。

```
SP210_API SP210_Result SP210_CALL SP210_CancelRequest(SP210_CancelHandle token);
```

参数	说明
token	有效取消令牌；操作完成后 Reset 才能复用。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_CancelReset

上一次使用令牌的操作返回后重置，以便再次使用；正在使用时返回 BUSY。

```
SP210_API SP210_Result SP210_CALL SP210_CancelReset(SP210_CancelHandle token);
```

参数	说明
token	有效取消令牌；操作完成后 Reset 才能复用。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_CancelDestroy

销毁令牌并发出停止请求；销毁后不得再使用该句柄。

```
SP210_API SP210_Result SP210_CALL SP210_CancelDestroy(SP210_CancelHandle token);
```

参数	说明
token	有效取消令牌；操作完成后 Reset 才能复用。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceContinuouslyAcquireSpectrum

在调用线程中持续采集并复用同一串口会话，每帧同步回调；样本仅在回调期间有效，回调返回 0 停止。外部线程可请求取消，令牌复用前须 Reset。

```
SP210_API SP210_Result SP210_CALL SP210_DeviceContinuouslyAcquireSpectrum(SP210_DeviceHandle device, int64_t integration_time_us, int32_t average_times, int32_t smoothing_level, SP210_CancelHandle token, SP210_SpectrumCallback callback, SP210_AcquisitionParametersCallback parameters, void* user_data);
```

参数	说明
device	有效设备句柄。
integration_time_us	积分时间，单位微秒；采集时须处于产品支持范围。
average_times	平均次数；设备采集要求 1–128。
smoothing_level	平滑等级，0–10。
token	有效取消令牌；操作完成后 Reset 才能复用。
callback	每帧同步调用的光谱回调，不可为 NULL；返回 0 停止，样本指针只在回调内有效。
parameters	可选的逐帧参数回调；可为 NULL，非空时在回调中提供有效的积分、平均和平滑参数。
user_data	原样传入回调的调用方上下文，可为 NULL；在同步调用返回前须保持有效。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_DeviceContinuouslyAcquireSpectrumForProduct

以显式产品 ID 连续采集，生命周期、回调和取消规则与 SP210_DeviceContinuouslyAcquireSpectrum 相同。

```
SP210_API SP210_Result SP210_CALL
SP210_DeviceContinuouslyAcquireSpectrumForProduct(SP210_DeviceHandle device, const uint8_t*
product_id, size_t product_id_length, int64_t integration_time_us, int32_t average_times, int32_t
smoothing_level, SP210_CancelHandle token, SP210_SpectrumCallback callback,
SP210_AcquisitionParametersCallback parameters, void* user_data);
```

参数	说明
device	有效设备句柄。
product_id	输入原始产品 ID 字节；长度参数以字节计。
product_id_length	输入产品 ID 字节数，范围 1–4。
integration_time_us	积分时间，单位微秒；采集时须处于产品支持范围。
average_times	平均次数；设备采集要求 1–128。
smoothing_level	平滑等级，0–10。
token	有效取消令牌；操作完成后 Reset 才能复用。
callback	每帧同步调用的光谱回调，不可为 NULL；返回 0 停止，样本指针只在回调内有效。
parameters	可选的逐帧参数回调；可为 NULL，非空时在回调中提供有效的积分、平均和平滑参数。
user_data	原样传入回调的调用方上下文，可为 NULL；在同步调用返回前须保持有效。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

固件包、更新与核验

返回 API 参考目录

SP210_FirmwareParsePackage

从内存解析并校验固件包，成功时返回包句柄；须用 SP210_FirmwarePackageDestroy 释放。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareParsePackage(const uint8_t* bytes, size_t length,
const char* file_name, SP210_FirmwarePackageHandle* package);
```

参数	说明
bytes	输入字节数组，长度由 length 指定。
length	输入字节数组的长度，单位字节。
file_name	与内存固件包关联的 UTF-8 文件名。
package	输出固件包句柄，成功后须销毁。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_FirmwareLoadPackage

从 UTF-8 路径加载并校验固件包，成功后须销毁包句柄。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareLoadPackage(const char* path,
SP210_FirmwarePackageHandle* package);
```

参数	说明
path	固件包路径，NUL 结尾的 UTF-8 字符串。
package	输出固件包句柄，成功后须销毁。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_FirmwarePackageDestroy

销毁固件包句柄。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwarePackageDestroy(SP210_FirmwarePackageHandle
package);
```

参数	说明
package	有效固件包句柄。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_FirmwarePackageGetInfo

读取包长度、CRC 和时间戳；调用前设置 `info.struct_size`。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwarePackageGetInfo(SP210_FirmwarePackageHandle package, SP210_FirmwarePackageInfo* info);
```

参数	说明
`package`	有效固件包句柄。
`info`	调用方输出结构体指针；调用前将 `struct_size` 设为结构体大小。

返回：`SP210_RESULT_OK` 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_FirmwarePackageGetText

按字段读取包文件名、名称或版本文本。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwarePackageGetText(SP210_FirmwarePackageHandle package, SP210_FirmwarePackageText field, char* buffer, size_t capacity, size_t* required_size);
```

参数	说明
`package`	有效固件包句柄。
`field`	选择包文件名、名称或版本的 `SP210_FirmwarePackageText`。
`buffer`	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
`capacity`	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
`required_size`	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：`SP210_RESULT_OK` 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 `SP210_RESULT_BUFFER_TOO_SMALL`。

SP210_FirmwarePackageGetBytes

读取固件包原始字节。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwarePackageGetBytes(SP210_FirmwarePackageHandle package, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
`package`	有效固件包句柄。
`buffer`	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
`capacity`	输出二进制缓冲区容量，单位字节。
`required_size`	输出实际或所需的二进制缓冲区大小，单位字节。

返回：`SP210_RESULT_OK` 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 `SP210_RESULT_BUFFER_TOO_SMALL`。

SP210_FirmwareCompareVersions

比较当前与目标版本，输出升级、降级、重刷或未知操作。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareCompareVersions(const char* current, const char* target, SP210_FirmwareOperation* operation);
```

参数	说明
current	当前版本的 NUL 结尾 UTF-8 字符串。
target	目标版本的 NUL 结尾 UTF-8 字符串。
operation	输出版本比较得到的 SP210_FirmwareOperation。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_FirmwareQueryPort

通过串口同步查询设备固件状态，成功返回结果句柄；必须读取结果状态并销毁句柄。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareQueryPort(const char* port, SP210_CancelHandle token, SP210_FirmwareDiagnosticCallback diagnostic, void* user_data, SP210_FirmwareResultHandle* result);
```

参数	说明
port	以 NUL 结尾的 UTF-8 串口名称。
token	有效取消令牌；操作完成后 Reset 才能复用。
diagnostic	可选的固件诊断回调；消息指针仅在回调期间有效。
user_data	原样传入回调的调用方上下文，可为 NULL；在同步调用返回前须保持有效。
result	输出固件结果句柄，成功后须销毁。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_FirmwareUpdatePort

通过串口同步发送固件包并返回结果句柄。返回 OK 只说明生成了结果，必须检查状态；传输完成后等待重启并核验版本。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareUpdatePort(const char* port,
SP210_FirmwarePackageHandle package, SP210_CancelHandle token, SP210_FirmwareProgressCallback
progress, SP210_FirmwareDiagnosticCallback diagnostic, void* user_data,
SP210_FirmwareResultHandle* result);
```

参数	说明
port	以 NUL 结尾的 UTF-8 串口名称。
package	有效固件包句柄。
token	有效取消令牌；操作完成后 Reset 才能复用。
progress	可选的固件进度回调；消息指针仅在回调期间有效。
diagnostic	可选的固件诊断回调；消息指针仅在回调期间有效。
user_data	原样传入回调的调用方上下文，可为 NULL；在同步调用返回前须保持有效。
result	输出固件结果句柄，成功后须销毁。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_FirmwareGetDefaultOptions

取得默认固件选项，调用前设置 options.struct_size，按需修改后用于自定义传输。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareGetDefaultOptions(SP210_FirmwareOptions*
options);
```

参数	说明
options	输出选项结构体；调用前设置 struct_size。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_FirmwareQueryTransport

通过调用方提供的同步传输回调查询固件；options 可为 NULL 使用默认值，成功后读取并销毁结果句柄。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareQueryTransport(const SP210_FirmwareTransport*
transport, const SP210_FirmwareOptions* options, SP210_CancelHandle token,
SP210_FirmwareDiagnosticCallback diagnostic, void* user_data, SP210_FirmwareResultHandle*
result);
```

参数	说明
transport	调用方实现的同步读写回调结构体；先设置 struct_size 并在调用返回前保持有效。
options	可选选项结构体；NULL 使用默认值，非空时须设置 struct_size。
token	有效取消令牌；操作完成后 Reset 才能复用。
diagnostic	可选的固件诊断回调；消息指针仅在回调期间有效。
user_data	原样传入回调的调用方上下文，可为 NULL；在同步调用返回前须保持有效。
result	输出固件结果句柄，成功后须销毁。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_FirmwareUpdateTransport

通过自定义传输回调更新固件；options 可为 NULL。返回 OK 只表示结果对象已生成，须检查状态并在重启后核验。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareUpdateTransport(const SP210_FirmwareTransport*
transport, const SP210_FirmwareOptions* options, SP210_FirmwarePackageHandle package,
SP210_CancelHandle token, SP210_FirmwareProgressCallback progress,
SP210_FirmwareDiagnosticCallback diagnostic, void* user_data, SP210_FirmwareResultHandle*
result);
```

参数	说明
transport	调用方实现的同步读写回调结构体；先设置 struct_size 并在调用返回前保持有效。
options	可选选项结构体；NULL 使用默认值，非空时须设置 struct_size。
package	有效固件包句柄。
token	有效取消令牌；操作完成后 Reset 才能复用。
progress	可选的固件进度回调；消息指针仅在回调期间有效。
diagnostic	可选的固件诊断回调；消息指针仅在回调期间有效。
user_data	原样传入回调的调用方上下文，可为 NULL；在同步调用返回前须保持有效。
result	输出固件结果句柄，成功后须销毁。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_FirmwareResultDestroy

销毁固件查询或更新结果句柄。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareResultDestroy(SP210_FirmwareResultHandle result);
```

参数	说明
result	有效固件结果句柄。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_FirmwareResultGetInfo

取得固件状态、传输完成与设备接受标志；调用前设置 info.struct_size。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareResultGetInfo(SP210_FirmwareResultHandle result,
SP210_FirmwareResultInfo* info);
```

参数	说明
result	有效固件结果句柄。
info	调用方输出结构体指针；调用前将 struct_size 设为结构体大小。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_FirmwareResultGetText

按字段取得结果消息、设备版本、板卡或中断原因。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareResultGetText(SP210_FirmwareResultHandle result,
SP210_FirmwareResultText field, char* buffer, size_t capacity, size_t* required_size);
```

参数	说明
result	有效固件结果句柄。
field	选择消息、版本、板卡或中断原因的 SP210_FirmwareResultText。
buffer	调用方提供的 UTF-8 输出缓冲区，可传 NULL 查询大小。
capacity	输出文本缓冲区容量，单位字节，须包含末尾 NUL 的空间。
required_size	输出所需缓冲区大小，单位字节，包括结尾 NUL。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_FirmwareResultGetRawReply

取得固件结果中的原始设备响应字节。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareResultGetRawReply(SP210_FirmwareResultHandle result, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
result	有效固件结果句柄。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_FirmwareEvaluateVerification

结合更新结果、目标版本和重启后的查询结果，计算固件验证状态；传输完成本身不等于已验证。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareEvaluateVerification(SP210_FirmwareResultHandle update_result, const char* target_version, SP210_FirmwareResultHandle query_result, SP210_FirmwareVerification* verification);
```

参数	说明
update_result	固件更新结果句柄。
target_version	目标固件版本的 NUL 结尾 UTF-8 字符串。
query_result	设备重启后查询得到的固件结果句柄。
verification	输出 SP210_FirmwareVerification 核验状态。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_FirmwareParseDeviceInfo

离线从设备原始回复中解析版本和板卡文本；一次查询得到两个所需长度，实际调用时两个输出缓冲区都必须足够。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareParseDeviceInfo(const uint8_t* reply, size_t length, char* version, size_t version_capacity, size_t* version_required, char* board, size_t board_capacity, size_t* board_required);
```

参数	说明
reply	输入设备原始回复字节，长度由 length 指定。
length	输入字节数组的长度，单位字节。
version	调用方提供的 UTF-8 版本输出缓冲区，可传 NULL 查询大小。
version_capacity	版本缓冲区容量，单位字节，包含 NUL 空间。
version_required	输出版本文本所需字节数，包含 NUL。
board	调用方提供的 UTF-8 板卡输出缓冲区，可传 NULL 查询大小。
board_capacity	板卡缓冲区容量，单位字节，包含 NUL 空间。
board_required	输出板卡文本所需字节数，包含 NUL。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。

SP210_FirmwareCRC32

离线计算固件格式使用的 CRC-32。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareCRC32(const uint8_t* bytes, size_t length, uint32_t* crc);
```

参数	说明
bytes	输入字节数组，长度由 length 指定。
length	输入字节数组的长度，单位字节。
crc	输出对应宽度的 CRC 值。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_FirmwareCRC16Xmodem

离线计算 CRC-16/XMODEM。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareCRC16Xmodem(const uint8_t* bytes, size_t length, uint16_t* crc);
```

参数	说明
bytes	输入字节数组，长度由 length 指定。
length	输入字节数组的长度，单位字节。
crc	输出对应宽度的 CRC 值。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误。

SP210_FirmwareMakePacket

离线为给定块号和载荷生成固件传输数据包；载荷须为 128 或 1024 字节。当前实现对其他长度返回空输出而非参数错误，调用方须检查输出长度。

```
SP210_API SP210_Result SP210_CALL SP210_FirmwareMakePacket(uint8_t block, const uint8_t* payload, size_t length, uint8_t* buffer, size_t capacity, size_t* required_size);
```

参数	说明
block	固件数据包块号，uint8。
payload	输入固件数据包载荷字节数组，长度由 length 指定。
length	输入字节数组的长度，单位字节。
buffer	调用方提供的二进制输出缓冲区，可传 NULL 查询大小。
capacity	输出二进制缓冲区容量，单位字节。
required_size	输出实际或所需的二进制缓冲区大小，单位字节。

返回：SP210_RESULT_OK 表示接口调用成功；失败时返回对应结果码并可立即读取当前线程错误；输出缓冲区不足时返回 SP210_RESULT_BUFFER_TOO_SMALL。