

# 接口函数说明

## 1. 版本说明

| 版本   | 描述                                                                                                                                                                                                                                                                                                     | 日期         | 作者  |
|------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|-----|
| V1.0 | 第一个版本                                                                                                                                                                                                                                                                                                  | 2023.05.02 | joy |
| V1.1 | int32 ts_scanner_set_light_state(DB_LED_STATE_E<br>eLedState, uint8 isSaveParam); 修改成<br>int32 tx_scanner_set_light_state(DB_LED_STATE_E<br>eLedState, uint8 isSaveParam);<br>int32 ts_scanner_get_light_state(void);修改成<br>int32 tx_scanner_get_light_state(void);                                    | 2023.08.03 | zzz |
| V1.2 | 1. 添加设置和获取定位灯状态<br>2. 添加设置和获取单次扫描的时间<br>3. 添加设置和获取使能前缀<br>4. 添加设置和获取设置前缀内容,最大长度为 10<br>5. 添加设置和获取使能后缀<br>6. 添加设置和获取设置后缀内容,最大长度为 10<br>7. 添加设置和获取提示音音量范围(0-100)<br>8. 添加设置和获取心跳类型<br>9. 添加设置和获取心跳间隔时间<br>10. 添加设置和获取心跳等待应答时间<br>11. 添加设置设备复位功能<br>12. 添加设置扫码开关是否能扫码<br>13. 添加设置设置码使能功能,指令禁止只能用指令开<br>启. | 2023.08.14 | zzz |

## 2. 概述

在 Windows 上使用的 USB HID POS 的 SDK 包.

特别说明涉及到永久参数的函数接口,请勿频繁使用.如果是出厂或者偶尔设置推荐使用永久参数,如果是每次读码都进行参数修改,请使用临时参数.频繁进行永久参数操作会减小扫码设备的使用寿命.

## 3. 接口说明

```
typedef unsigned char uint8;
```

```

typedef signed char int8;
typedef unsigned short uint16;
typedef signed short int16;
typedef unsigned int uint32;
typedef signed int int32;
typedef char CHAR;
typedef uint16 wchar;
typedef unsigned int BOOL;
typedef long long int64;
typedef unsigned long long uint64;

#ifndef SUCC
#define SUCC0
#endif

#ifndef FAIL
#define FAIL -1
#endif

/**产品信息相关宏定义**/
#define DB_PRODUCT_MODEL_MAX_LEN 20 /**产品名称最大长度**/
#define DB_PRODUCT_SN_MAX_LEN 30 /**产品序列号(ID)最大长度**/
#define DB_PRODUCT_DATE_MAX_LEN 15 /**日期最大长度(如:20150809)**/
#define DB_PRODUCT_HVER_MAX_LEN 15 /**硬件版本最大长度**/
#define DB_PRODUCT_SVER_MAX_LEN 15 /**软件版本由代码控制,最大长度**/
#define DB_PRODUCT_FLAG (0xCF)

/**解码数据回调函数**/
typedef int32 (*tx_scanner_decode_data_cb_fun)(uint8 ucCodeType, uint8 *pBuf, uint32 uiBufLen);
/**通信接口(USB)连接状态 0:断开,1:连接**/
typedef void (*tx_scanner_comm_state_cb_fun)(uint8 ucState);

/**产品信息**/
typedef struct DB_PRODUCT_INFO_
{
    uint8 ucModel[DB_PRODUCT_MODEL_MAX_LEN]; /**产品名称**/
    uint8 ucSn[DB_PRODUCT_SN_MAX_LEN]; /**产品序列号(SN)**/
    uint8 ucDate[DB_PRODUCT_DATE_MAX_LEN]; /**日期**/
    uint8 ucHVer[DB_PRODUCT_HVER_MAX_LEN]; /**硬件版本**/
    uint8 ucSVer[DB_PRODUCT_SVER_MAX_LEN]; /**软件版本,由代码控制**/
    uint8 ucFlag; /**产品信息标志位 DB_PRODUCT_FLAG**/
}DB_PRODUCT_INFO_T;

```

```

/**解码触发方式**/
typedef enum DB_TRIGGER_MODE_
{
    DB_TRIGGER_M_LEVEL = 0x00,  /**00[被动](电平)按键保持**/    //默认
    DB_TRIGGER_M_PULSE,         /**01[被动](脉冲)单次按键触发模式**/
    DB_TRIGGER_M_CONTINUOUS,    /**02[主动]连续模式**/
    DB_TRIGGER_M_AUTOSENS,     /**03[主动]自动感应模式**/

```

```

}DB_TRIGGER_MODE_E;

```

```

/**LED 灯的状态(定位灯和照明灯)**/

```

```

typedef enum DB_LED_STATE_
{
    DB_LED_S_NORMAL = 0x00,    /**正常读码的时候亮**/
    DB_LED_S_ALWAYS_OFF,      /**一直关闭**/
    DB_LED_S_ALWAYS_ON,       /**一直开启**/

```

```

    DB_LED_S_MAX,

```

```

}DB_LED_STATE_E;

```

```

/**心跳类型**/

```

```

typedef enum DB_HEARTBEAT_TYPE_
{
    DB_HEARTBEAT_T_DISABLE = 0x00, /**禁止，默认**/
    DB_HEARTBEAT_T_NOT_ACK,        /**心跳不需要 ACK**/
    DB_HEARTBEAT_T_NEED_ACK,       /**心跳需要 ACK**/

```

```

    DB_HEARTBEAT_T_MAX,

```

```

}DB_HEARTBEAT_TYPE_E;

```

```

/*****

```

```

*功能描述: 初始化

```

```

*参数说明:

```

```

*   输入: 无

```

```

*   输出: 无

```

```

*   返回值: int32,成功:SUCC,失败:FAIL

```

```

*****/

```

```

int32 tx_scanner_init(void);

```

```

/*****

```

```

*功能描述: 去初始化

```

\*参数说明:

\* 输入: 无

\* 输出: 无

\* 返回值: int32,成功:Succ,失败:Fail

\*\*\*\*\*/

int32 tx\_scanner\_deinit(void);

/\*\*\*\*\*

\*功能描述: 获取 SDK 包版本

\*参数说明:

\* 输入: ucVerLen,缓存区 pVer 空间长度

\* 输出: pVer,版本信息

\* 返回值: int32,成功:版本信息的长度,失败:Fail

\*\*\*\*\*/

int32 tx\_scanner\_get\_version(uint8 \*pVer, uint8 ucVerLen);

/\*\*\*\*\*

\*功能描述: 注册解码数据回调函数

\*参数说明:

\* 输入: fDataFun,解码数据函数

\* 输出: 无

\* 返回值: int32,成功:Succ,失败:Fail

\*\*\*\*\*/

int32 tx\_scanner\_decode\_data\_fun\_register(tx\_scanner\_decode\_data\_cb\_fun fDataFun);

/\*\*\*\*\*

\*功能描述: 注册设备连接状态回调函数

\*参数说明:

\* 输入: fStateFun,设备连接状态函数

\* 输出: 无

\* 返回值: int32,成功:Succ,失败:Fail

\*\*\*\*\*/

int32 tx\_scanner\_comm\_state\_fun\_register(tx\_scanner\_comm\_state\_cb\_fun fStateFun);

/\*\*\*\*\*

\*功能描述: 获取设备版本信息

\*参数说明:

\* 输入: usLen,缓存区 pucVerInfo 空间长度

\* 输出: pucVerInfo,设备版本信息

\* 返回值: int32,成功:设备版本信息长度,失败:Fail

\*\*\*\*\*/

int32 tx\_scanner\_get\_version\_info(uint8 \*pucVerInfo, uint16 usLen);

/\*\*\*\*\*

\*功能描述: 获取设备产品信息

\*参数说明:

\* 输入: usLen,缓存区 pucProInfo 空间长度

\* 输出: pucProInfo,设备产品信息

\* 返回值: int32,成功:设备产品信息长度,失败:FAIL

\*\*\*\*\*/

int32 tx\_scanner\_get\_all\_product\_info(uint8 \*pucProInfo, uint16 usLen);

/\*\*\*\*\*

\*功能描述: 开始解码

\*参数说明:

\* 输入: 无

\* 输出: 无

\* 返回值: int32,成功:Succ,失败:FAIL

\*\*\*\*\*/

int32 tx\_scanner\_decode\_start(void);

/\*\*\*\*\*

\*功能描述: 结束解码

\*参数说明:

\* 输入: 无

\* 输出: 无

\* 返回值: int32,成功:Succ,失败:FAIL

\*\*\*\*\*/

int32 tx\_scanner\_decode\_stop(void);

/\*\*\*\*\*

\*功能描述: 开始不超时解码

\*参数说明:

\* 输入: 无

\* 输出: 无

\* 返回值: int32,成功:Succ,失败:FAIL

\*\*\*\*\*/

int32 tx\_scanner\_no\_time\_decode\_start(void);

/\*\*\*\*\*

\*功能描述: 设置触发方式

\*参数说明:

\* 输入: eMode,触发方式

\* isSaveParam,0:临时生效,1:永久生效

\* 输出: 无

\* 返回值: int32,成功:Succ,失败:FAIL

\*\*\*\*\*/

int32 tx\_scanner\_set\_trigger\_mode(DB\_TRIGGER\_MODE\_E eMode, uint8 isSaveParam);

/\*\*\*\*\*\*

\*功能描述: 获取触发方式

\*参数说明:

\* 输入: 无

\* 输出: 无

\* 返回值: int32,成功:触发方式,失败:FAIL

\*\*\*\*\*/

int32 tx\_scanner\_get\_trigger\_mode(void);

/\*\*\*\*\*\*

\*功能描述: 设置补光灯状态

\*参数说明:

\* 输入: eLedState,补光灯状态

\* isSaveParam,0:临时生效,1:永久生效

\* 输出: 无

\* 返回值: int32,成功:Succ,失败:FAIL

\*\*\*\*\*/

int32 tx\_scanner\_set\_light\_state(DB\_LED\_STATE\_E eLedState, uint8 isSaveParam);

/\*\*\*\*\*\*

\*功能描述: 获取补光灯状态

\*参数说明:

\* 输入: 无

\* 输出: 无

\* 返回值: int32,成功:补光灯状态,失败:FAIL

\*\*\*\*\*/

int32 tx\_scanner\_get\_light\_state(void);

/\*\*\*\*\*\*

\*功能描述: 设置定位灯状态

\*参数说明:

\* 输入: eLedState,定位灯状态

\* isSaveParam,0:临时生效,1:永久生效

\* 输出: 无

\* 返回值: int32,成功:Succ,失败:FAIL

\*\*\*\*\*/

int32 tx\_scanner\_set\_focus\_state(DB\_LED\_STATE\_E eLedState, uint8 isSaveParam);

/\*\*\*\*\*\*

\*功能描述: 获取定位灯状态

\*参数说明:

\* 输入: 无

\* 输出: 无

```

*   返回值: int32,成功:定位灯状态,失败:FAIL
*****/
int32 tx_scanner_get_focus_state(void);

/*****
*功能描述: 设置使能前缀
*参数说明:
*   输入: ucEnable,0:禁止,1:使能
*         isSaveParam,0:临时生效,1:永久生效
*   输出: 无
*   返回值: int32,成功:SUCC,失败:FAIL
*****/
int32 tx_scanner_set_en_prefix(uint8 ucEnable, uint8 isSaveParam);

/*****
*功能描述: 获取使能前缀
*参数说明:
*   输入: 无
*   输出: 无
*   返回值: int32,成功:0:禁止,1:使能,失败:FAIL
*****/
int32 tx_scanner_get_en_prefix(void);

/*****
*功能描述: 设置前缀,前缀最大长度:10 字节
*参数说明:
*   输入: pData,前缀数据
*         ucDataLen,前缀数据长度
*         isSaveParam,0:临时生效,1:永久生效
*   输出: 无
*   返回值: int32,成功:SUCC,失败:FAIL
*****/
int32 tx_scanner_set_prefix(uint8* pData, uint8 ucDataLen, uint8 isSaveParam);

/*****
*功能描述: 获取前缀,前缀最大长度:10 字节
*参数说明:
*   输入: ucDataLen,前缀缓存区的长度
*   输出: pData,前缀缓存区
*   返回值: int32,成功:获取的前缀长度,失败:FAIL
*****/
int32 tx_scanner_get_prefix(uint8* pData, uint8 ucDataLen);

/*****
*功能描述: 设置使能后缀

```

\*参数说明:

- \* 输入: ucEnable,0:禁止,1:使能
- \* isSaveParam,0:临时生效,1:永久生效
- \* 输出: 无
- \* 返回值: int32,成功:Succ,失败:Fail

\*\*\*\*\*/

int32 tx\_scanner\_set\_en\_suffix(uint8 ucEnable, uint8 isSaveParam);

/\*\*\*\*\*

\*功能描述: 获取使能后缀

\*参数说明:

- \* 输入: 无
- \* 输出: 无
- \* 返回值: int32,成功:0:禁止,1:使能,失败:Fail

\*\*\*\*\*/

int32 tx\_scanner\_get\_en\_suffix(void);

/\*\*\*\*\*

\*功能描述: 设置后缀,后缀最大长度:10 字节

\*参数说明:

- \* 输入: pData,后缀数据
- \* ucDataLen,后缀数据长度
- \* isSaveParam,0:临时生效,1:永久生效
- \* 输出: 无
- \* 返回值: int32,成功:Succ,失败:Fail

\*\*\*\*\*/

int32 tx\_scanner\_set\_suffix(uint8\* pData, uint8 ucDataLen, uint8 isSaveParam);

/\*\*\*\*\*

\*功能描述: 获取后缀,后缀最大长度:10 字节

\*参数说明:

- \* 输入: ucDataLen,后缀缓存区的长度
- \* 输出: pData,后缀缓存区
- \* 返回值: int32,成功:获取的后缀长度,失败:Fail

\*\*\*\*\*/

int32 tx\_scanner\_get\_suffix(uint8\* pData, uint8 ucDataLen);

/\*\*\*\*\*

\*功能描述: 设置提示音音量

\*参数说明:

- \* 输入: ucVol,范围:0-100,0 是静音
- \* isSaveParam,0:临时生效,1:永久生效
- \* 输出: 无
- \* 返回值: int32,成功:Succ,失败:Fail

```

*****/
int32 tx_scanner_set_beep_volume(uint8 ucVol, uint8 isSaveParam);

/*****
*功能描述: 获取提示音音量
*参数说明:
*    输入: 无
*    输出: 无
*    返回值: int32,成功:音量,失败:FAIL
*****/
int32 tx_scanner_get_beep_volume(void);

/*****
*功能描述: 设置单次扫描的时间
*参数说明:
*    输入: usTimeMs,单次的时间,单位:ms,范围:0-65535ms,0 表示没有时间限制
*           isSaveParam,0:临时生效,1:永久生效
*    输出: 无
*    返回值: int32,成功:SUCC,失败:FAIL
*****/
int32 tx_scanner_set_once_scan_time(uint16 usTimeMs, uint8 isSaveParam);

/*****
*功能描述: 获取单次扫描的时间
*参数说明:
*    输入: 无
*    输出: 无
*    返回值: int32,成功:单次扫描的时间,失败:FAIL
*****/
int32 tx_scanner_get_once_scan_time(void);

/*****
*功能描述: 设置心跳类型
*参数说明:
*    输入: eType,心跳类型
*           isSaveParam,0:临时生效,1:永久生效
*    输出: 无
*    返回值: int32,成功:SUCC,失败:FAIL
*****/
int32 tx_scanner_set_heartbeat_type(DB_HEARTBEAT_TYPE_E eType, uint8 isSaveParam);

/*****
*功能描述: 获取心跳类型
*参数说明:

```

```

*    输入: 无
*    输出: 无
*    返回值: int32,成功:心跳类型,失败:FAIL
*****/
int32 tx_scanner_get_heartbeat_type(void);

/*****
*功能描述: 设置心跳间隔时间
*参数说明:
*    输入: uiTimeMs,心跳间隔时间,单位:ms,范围 0-2147483648
*          isSaveParam,0:临时生效,1:永久生效
*    输出: 无
*    返回值: int32,成功:Succ,失败:FAIL
*****/
int32 tx_scanner_set_heartbeat_gap_time(uint32 uiTimeMs, uint8 isSaveParam);

/*****
*功能描述: 获取心跳间隔时间
*参数说明:
*    输入: 无
*    输出: 无
*    返回值: int32,成功:心跳间隔时间,失败:FAIL
*****/
int32 tx_scanner_get_heartbeat_gap_time(void);

/*****
*功能描述: 设置心跳等待应答时间
*参数说明:
*    输入: uiTimeMs,心跳等待应答时间,单位:ms,范围 0-2147483648
*          isSaveParam,0:临时生效,1:永久生效
*    输出: 无
*    返回值: int32,成功:Succ,失败:FAIL
*****/
int32 tx_scanner_set_heartbeat_wait_ack_time(uint32 uiTimeMs, uint8 isSaveParam);

/*****
*功能描述: 获取心跳等待应答时间
*参数说明:
*    输入: 无
*    输出: 无
*    返回值: int32,成功:心跳等待应答时间,失败:FAIL
*****/
int32 tx_scanner_get_heartbeat_wait_ack_time(void);

```

/\*\*\*\*\*\*

\*功能描述: 设备复位

\*参数说明:

\* 输入: 无

\* 输出: 无

\* 返回值: int32,成功:SUCCE,失败:FAIL

\*\*\*\*\*/

int32 tx\_scanner\_reset(void);

/\*\*\*\*\*\*

\*功能描述: 扫码开关,临时有效

\*参数说明:

\* 输入: ucEnable,0:禁止扫码,1:使能扫码

\* 输出: 无

\* 返回值: int32,成功:SUCCE,失败:FAIL

\*\*\*\*\*/

int32 tx\_scanner\_scan\_sw(uint8 ucEnable);

/\*\*\*\*\*\*

\*功能描述: 设置码开关,使用指令禁止了设置码,只能使用指令使能设置码

\*参数说明:

\* 输入: ucEnable,0:禁止设置码,1:使能设置码

\* isSaveParam,0:临时生效,1:永久生效

\* 输出: 无

\* 返回值: int32,成功:SUCCE,失败:FAIL

\*\*\*\*\*/

int32 tx\_scanner\_cmd\_setcode\_sw(uint8 ucEnable, uint8 isSaveParam);