MMS

服务器端开发包接口 API 使用说明书

版本：V2.0

深圳天勺电力软件有限公司

深圳　中国

1. 开发包简介

该开发包根据已经正式发行的 IEC 61850 最新版本开发而成，采用 C 语音编写，可跨平台使用。

为了最大限度地简化使用，完全封装了 IEC 61850 中的诸多底层细节和各种复杂模型的实现逻辑。用户利用它作开发的时候无需了解底层细节，只需要调用功能接口函数完成相应的功能研发，具有方便、快捷的优点。

本接口说明文档为接口全集，客户可以根据具体业务选择相应的接口进行调用。本开发包既可以用在开发基于 IEC61850 标准的所有设备中（包括各个应用领域），也可以集成在网关机和通信管理机中。现场运行稳定可靠，调用简单高效。

2. 接口描述

2.1 获取节点数据路径函数

函数声明	**char* get_path_by_node(TREE_NODE* treeNode)**
功能	可通过该接口获取当前指定节点的路径
参数	TREE_NODE* treeNode：节点对象指针
返回值	数据路径字符串(不带 fc)
示例	char* path = get_path_by_node(treeNode);

2.2 日志完整性周期服务函数

函数声明	**void intgPdLog(SCL_INFO *ied)**
功能	使用线程将该接口执行，可启动日志服务的完整性周期服务
参数	SCL_INFO* ied：模型信息对象指针
返回值	无
示例	intgPdLog(ied);

2.3 设置模型文件路径函数

函数声明	**int SZTS_set_doc_path(char *theDocPath)**
功能	设置模型文件路径，用于解析指定的模型文件
参数	char * theDocPath：模型文件路径
返回值	成功：1，失败：0
示例	int result = SZTS_set_doc_path(theDocPath);

2.4 获取设备节点函数

函数声明	**SCL_INFO* SZTS_get_all_ied_node()**
功能	解析模型文件中的 IED 节点，做成链表，并返回链表的头节点
参数	无
返回值	模型文件中 IED 节点链表头节点
示例	SCL_INFO* sclInfo = SZTS_get_all_ied_node();

2.5 拷贝树函数

函数声明	**TREE_NODE* cp_tree_node_link(TREE_NODE* srcHead)**
功能	拷贝一个树对象的全部信息
参数	TREE_NODE* srcHead：需要拷贝树对象的头节点
返回值	拷贝出来的树对象的头节点
示例	TREE_NODE* treeNode = cp_tree_node_link(srcHead);

2.6 通过路径获取节点函数

函数声明	**TREE_NODE* get_tree_node_by_path(char *path)**
功能	可获取指定路径的节点对象
参数	char* path：需要获取的节点的路径(不带 FC)
返回值	路径对应的节点对象
示例	TREE_NODE* treeNode =get_tree_node_by_path(path);

2.7 通过路径获取节点函数

函数声明	**TREE_NODE* get_tree_node_by_path_new(char *path)**
功能	可获取指定路径的节点对象

参数	char*　path：需要获取的节点的路径(带 FC)
返回值	路径对应的节点对象
示例	TREE_NODE* treeNode =get_tree_node_by_path_new(path);

2.8 获取当前时间函数

函数声明	**char*　get_time()**
功能	获取字符串时间格式的当前系统时间
参数	无
返回值	字符串时间格式的当前系统时间
示例	char* nowTime = get_time();

2.9 获取报告控制块信息函数

函数声明	**RPT_CTRL*　get_rpt_ctrl_by_ied_name(char*　iedName)**
功能	解析模型文件中指定的 IED 节点下的全部报告控制块，并生成链表，返回链表的头节点
参数	char*　iedName：设备 IED 节点名称
返回值	指定 IED 节点下的报告控制块链表头节点
示例	RPT_CTRL* rptCtrl = get_rpt_ctrl_by_ied_name(iedName);

2.10 根据报告控制块最大使能数拷贝报告控制块信息节点函数

函数声明	**RPT_CTRL*　copy_rpt_ctrl_link(RPT_CTRL*　head)**
功能	按照模型文件中的最大使能报告控制块的参数配置，处理并实例化出对应最大使能个数的报告控制块
参数	RPT_CTRL*　head：报告控制块链表的头节点
返回值	经过拷贝处理后的报告控制块链表的头节点
示例	RPT_CTRL* cp = copy_rpt_ctrl_link(head);

2.11 获取模型节点信息函数

函数声明	**TREE_NODE*　get_tree_node_link_by_ied_name(char*　iedName)**
功能	解析模型文件中指定的 IED 节点的全部数据对象节点，并生成树对象，返回树对象的头节点
参数	char*　iedName：设备 IED 节点名称
返回值	指定 IED 节点下的模型节点树结构的头节点

示例	TREE_NODE* treeNode= get_tree_node_link_by_ied_name(iedName);

2.12 获取定值控制块信息函数

函数声明	**SETTING_CTRL*　get_setting_ctrl_by_ied_name(char *iedName)**
功能	解析模型文件中指定的 IED 节点下的全部定值控制块，并生成链表，返回链表的头节点
参数	char*　iedName：设备 IED 节点名称
返回值	指定 IED 节点下的定值控制块链表头节点
示例	SETTING_CTRL*settingCtrl= get_setting_ctrl_by_ied_name(iedName);

2.13 获取日志控制块信息函数

函数声明	**LOG_CTRL*　get_log_ctrl_by_ied_name(char*　iedName)**
功能	解析模型文件中指定的 IED 节点下的全部日志控制块，并生成链表，返回链表的头节点
参数	char*　iedName：设备 IED 节点名称
返回值	指定 IED 节点下的日志控制块链表头节点
示例	LOG_CTRL*logCtrl= get_log_ctrl_by_ied_name(iedName);

2.14 获取数据集信息函数

函数声明	**DATASET*　get_dataSet_by_ied_name(char*　iedName)**
功能	解析模型文件中指定的 IED 节点下的全部数据集，并生成链表，返回链表的头节点
参数	char*　iedName：设备 IED 节点名称
返回值	指定 IED 节点下的数据集链表头节点
示例	DATASET* dataset= get_dataSet_by_ied_name(iedName);

2.15 设置创建数据集保存路径函数

函数声明	**int　SZTS_set_create_dataSet_save_path(char*　path)**
功能	自主创建的数据集保存的 dataSet.xml 文件的所在路径，用于解析自主创建的数据集信息
参数	char*　path：自主创建的数据集保存的 dataSet.xml 文件的所在路径
返回值	成功：1，失败：0
示例	int result=SZTS_set_create_dataSet_save_path(path);

2.16 解析创建的持久数据集函数

函数声明	**void　get_dataSet_self(DATASET*　dataSetHead, SCL_INFO*　ied)**
功能	解析创建并保存在 dataSet.xml 中的自主创建的持久型数据集
参数	DATASET*　dataSetHead：数据集链表头节点 SCL_INFO*　ied：指定得 IED 设备对象节点
返回值	无
示例	get_dataSet_self(dataSetHead, ied);

2.17 获取 GOOSE 函数

函数声明	**GOOSE_INFO*　get_goose_info_link_by_ied_name(char*　iedName)**
功能	解析模型文件中指定的 IED 节点下的全部 GOOSE 控制块，并生成链表，返回链表的头节点
参数	char*　iedName：设备 IED 节点名称
返回值	指定 IED 节点下的 GOOSE 控制块链表头节点
示例	GOOSE_INFO*goossInfo= get_goose_info_link_by_ied_name(iedName);

2.18 获取 SV 函数

函数声明	**SMV_INFO*　get_smv_info_link_by_ied_name(char*　iedName)**
功能	解析模型文件中指定的 IED 节点下的全部 MSV 控制块，并生成链表，返回链表的头节点
参数	char*　iedName：设备 IED 节点名称
返回值	指定 IED 节点下的 MSV 控制块链表头节点
示例	SMV_INFO*smvInfo= get_smv_info_link_by_ied_name(iedName);

2.19 设置模型节点的实例化初始值函数

函数声明	**void　set_instance_val_to_node(char*　iedName)**
功能	对指定的 IED 节点下的树对象中的数据节点，赋予模型文件中实例化部分对应的实例化值
参数	char*　iedName：设备 IED 节点名称
返回值	无
示例	set_instance_val_to_node(iedName);

2.20 获取短地址映射信息函数

函数声明	**S_ADDR_MAP*　get_s_addr_map(char*　iedName)**
功能	解析模型文件中短地址，并与对应的点的索引路径形成映射关系，生成链表，并返回链表的头节点
参数	char*　iedName：设备 IED 节点名称
返回值	短地址与节点路劲映射的链表的头节点(节点路径不带 FC)
示例	S_ADDR_MAP*sAddr= get_s_addr_map(iedName);

2.21 缓存报告服务函数

函数声明	**void　buffDataReport(SCL_INFO*　ied)**
功能	开启缓存报告的缓存事件以及缓存报告的发送服务，该方法内部通过 while 循环进行间歇遍历，无退出动作，建议使用独立线程启动该服务
参数	SCL_INFO*　ied：设备 IED 节点对象
返回值	无
示例	buffDataReport(ied);

2.22 非缓存报告服务函数

函数声明	**void　unBuffDataReport_bufTm(SCL_INFO*　ied)**
功能	开启非缓存报告的缓存事件以及其缓存事件报告的发送服务，该方法内部通过 while 循环进行间歇遍历，无退出动作，建议使用独立线程启动该服务
参数	SCL_INFO*　ied：设备 IED 节点对象
返回值	无
示例	unBuffDataReport_bufTm (ied);

2.23 完整性周期报告服务函数

函数声明	**void　unBuffDataReport_bufTm(SCL_INFO*　ied)**
功能	开启非缓存报告的缓存事件以及其缓存事件报告的发送服务，该方法内部通过 while 循环进行间歇遍历，无退出动作，建议使用独立线程启动该服务
参数	SCL_INFO*　ied：设备 IED 节点对象
返回值	无
示例	unBuffDataReport_bufTm (ied);

2.24 获取配置文件信息函数

函数声明	**CFG_FILE*　SZTS_readCfgFile(char*　fileName)**
功能	解析配置文件的信息，并生成存放配置文件信息的内存对象
参数	char*　fileName：.cfg 后缀的配置文件的路径
返回值	配置文件的信息对象
示例	CFG_FILE*cfgInfo= SZTS_readCfgFile(fileName);

2.25 解析获取模型文件函数

函数声明	**SCL_INFO*　SZTS_sclParse(CFG_FILE*　cfgFile)**
功能	解析配置文件中配置的模型文件路径、设备名，访问点对应的模型文件的全部数据节点信息，生成链表对象，并返回链表头节点
参数	CFG_FILE*　cfgFile：.cfg 配置文件信息对象
返回值	设备信息对象链表头节点
示例	SCL_INFO*sclInfo= SZTS_sclParse(cfgFile);

2.26 启动服务器函数

函数声明	**int　SZTS_startServer()**
功能	开启服务器监听和接收消息服务，服务器对象默认为模型文件中的第一个 IED，该方法内部通过 while 循环进行间歇遍历，无退出动作，建议使用独立线程启动该服务
参数	无
返回值	有返回值即为服务运行结束，无返回值即为服务运行中
示例	int result = SZTS_startServer();

2.27 启动指定的服务器函数

函数声明	**int　SZTS_start_server_by_ied(SCL_INFO*　ied)**
功能	开启服务器监听和接收消息服务，服务器对象为指定的一个 IED，该方法内部通过 while 循环进行间歇遍历，无退出动作，建议使用独立线程启动该服务
参数	SCL_INFO*　ied：设备 IED 节点对象
返回值	有返回值即为服务运行结束，无返回值即为服务运行中
示例	int result = SZTS_start_server_by_ied(ied);

2.28 停止服务器

函数声明	**int SZTS_stopServer()**
功能	停止服务器监听和接收消息服务，停止的服务器对象默认为模型文件中的第一个 IED
参数	无
返回值	有返回值即为服务运行结束，无返回值即为服务运行中
示例	int result = SZTS_stopServer();

2.29 停止指定的服务器函数

函数声明	**int SZTS_stop_server_by_ied(SCL_INFO* ied)**
功能	停止服务器监听和接收消息服务，停止的服务器对象为指定的一个 IED
参数	SCL_INFO* ied：设备 IED 节点对象
返回值	有返回值即为服务运行结束，无返回值即为服务运行中
示例	int result = SZTS_stop_server_by_ied(ied);

2.30 设置对应路径模型节点值函数

函数声明	**int setNodeValueByPath(char* path, char* value)**
功能	通过节点路径更新节点值，方法内部校验并处理相关的非缓存报告、缓存报告、日志业务
参数	char* path：需要修改的节点路径(带 FC) char* value：目标值
返回值	1：改值成功，0：改值失败
示例	int result = setNodeValueByPath(path, value);

2.31 设置对应路径模型节点值-只设置值函数

函数声明	**int setNodeValueByPathOnly(char* path, char* value)**
功能	通过节点路径更新节点值，只更新值，不产生报告事件、日志事件
参数	char* path：需要修改的节点路径(带 FC) char* value：目标值
返回值	1：改值成功，0：改值失败
示例	int result = setNodeValueByPathOnly (path, value);

2.32 获取设备 mac 地址函数

函数声明	**void get_mac_address(char* mac_address)**
功能	获取当前设备的 mac 地址，存放到指定的内存空间
参数	char* mac_address：已分配可容纳 mac 地址空间的字符串指针
返回值	设备 mac 地址
示例	get_mac_address(char* mac_address);

2.33 缓存报告控制块的预留时间 超时校验函数

函数声明	**void handle_buffer_rptCtrl_client_resv_check()**
功能	开启缓存报告控制块预留时间的超时检测机制服务，对每个 TCP 连接在关联过缓存报告控制块的情况中，在失去关联时，进行设定的缓存报告控制块预留时间进行超时校验，在超时时间到达前，除最近一次关联过该缓存报告控制块的 TCP 连接可设置并使用该缓存报告控制块，其他的 TCP 连接不可设置和使用；倘若在超时间内最近一次关联该缓存报告控制块的 TCP 连接未继续关联该缓存报告控制块，则取消预留限制，任何 TCP 连接均可对此缓存报告控制块进行设置和使用；该方法内部通过 while 循环进行间歇遍历，无退出动作，建议使用独立线程启动该服务
参数	无
返回值	无
示例	handle_buffer_rptCtrl_client_resv_check();

2.34 客户端连接信息回调函数

函数原型定义

```
typedef int (*connectInfoFun)(char *ip, int port, int status);
```

函数全局变量

```
connectInfoFun connectInfo_fun;
```

注册回调函数方法

```
extern int regist_connectInfoFun(connectInfoFun f);
```

输入

char *ip：连接过来的客户端 ip

int port：连接过来的客户端端口

int status：连接过来的客户端状态

返回

预留返回值，未使用

用途

可通过该回调获得客户端的连接信息

2.35 接收客户端报文信息回调函数

函数原型定义

```
typedef int (*recvDataFun)(char *data, int length);
```

函数全局变量

```
recvDataFun recvData_fun;
```

注册回调函数方法

```
extern int regist_recvDataFun(recvDataFun f);
```

输入

char *data：接收的报文数据

int length：接收的报文长度

返回

预留返回值，未使用

用途

可通过该回调获得客户端的请求报文信息

2.36 判断否允许写入文件回调函数

函数原型定义

```
typedef int (*fileDataIsOkFun)(char *path);
```

函数全局变量

```
fileDataIsOkFun fileDataIsOk_fun;
```

注册回调函数方法

```
extern int regist_fileDataIsOkFun(fileDataIsOkFun f);
```

输入

char *path：文件路径

返回

1：允许写文件操作，0：不允许写文件操作

用途

可通过该回调可实现对写文件动作的允许与否

2.37 遥控选择回调函数

函数原型定义

```
typedef int (*selectFun)(char *reference);
```

函数全局变量

```
selectFun select_fun;
```

注册回调函数方法

```
extern int regist_selectFun(selectFun f);
```

输入

char *reference：遥控点路径

返回

1：允许选择，0：不允许选择

用途

可通过该回调实现对遥控选择的判断介入

2.38 遥控带值选择回调函数

函数原型定义

```
typedef int (*selectWithValueFun)(char *reference, Data *ctlVal, int *addCase);
```

函数全局变量

```
selectWithValueFun selectWithValue_fun;
```

注册回调函数方法

```
extern int regist_selectWithValueFun(selectWithValueFun f);
```

输入

char *reference：遥控点路径

Data *ctlVal：控制值

输出

int *addCase：不允许选择原因

返回

1：允许带值选择，0：不允许带值选择

用途

可通过该回调实现对遥控带值选择的判断介入

2.39 遥控取消回调函数

函数原型定义

```
typedef int (*cancelFun)(char *reference, Data *ctlVal, int *addCase);
```

函数全局变量

```
cancelFun cancel_fun;
```

注册回调函数方法

```
extern int regist_cancelFun(cancelFun f);
```

输入

char *reference：遥控点路径

Data *ctlVal：控制值

输出

int *addCase：不允许取消原因

返回

1：允许取消，0：不允许取消

用途

可通过该回调实现对遥控取消的判断介入

2.40 遥控执行回调函数

函数原型定义

```
typedef int (*operateFun)(char *reference, Data *ctlVal, int *addCase);
```

函数全局变量

operateFun operate_fun;

注册回调函数方法

extern int regist_operateFun(operateFun f);

输入

char *reference：遥控点路径

Data *ctlVal：控制值

输出

int *addCase：不允许遥控执行的原因

返回

1：允许遥控执行，0：不允许遥控执行

用途

可通过该回调实现对遥控执行的判断介入

2.41 定值切区回调函数

函数原型定义

typedef int (*selectActiveSgFun)(char *reference, int sgNumber);

函数全局变量

selectActiveSgFun selectActiveSg_fun;

注册回调函数方法

extern int regist_selectActiveSgFun(selectActiveSgFun f);

输入

char *reference：定值控制块路径

int sgNumber：切区的区号

返回

预留返回值，未使用

用途

可通过该回调获取定值切区的相关信息

2.42 改值操作回调函数

函数原型定义

```
typedef int (*setDataValuesFun)(char *reference, char *fc);
```

函数全局变量

```
setDataValuesFun setDataValues_fun;
```

注册回调函数方法

```
extern int regist_setDataValuesFun(setDataValuesFun f);
```

输入

char *reference：改值对象的路径

char *fc：改值对象的 FC

返回

1：允许改值，0：不允许改值

用途

可通过该回调实现改值操作的判断介入

2.43 改值操作回调-sv 取代 函数

函数原型定义

```
typedef int (*setSVDataValuesFun)(char *reference, char *value);
```

函数全局变量

```
setSVDataValuesFun setSVDataValues_fun;
```

注册回调函数方法

```
extern int regist_setSVDataValuesFun(setSVDataValuesFun f);
```

输入

char *reference：改值对象的路径

char *value：目的值

返回

1：允许改值，0 不允许改值

用途：

可通过该回调实现取代的业务逻辑，在改值操作时，优先执行此回调中实现的取代业务