MCS设计及二次开发手册

声明

由于产品版本升级或其他原因，本手册内容会 不定期进行更新。

本手册仅作为使用指导，本手册中的所有陈述、信息和建议不构成任何明示或暗示的担保。如有任何疑问或争议，请以我司最终解释为准。

本公司对使用本手册或使用本公司产品导致的任何特殊、附带、偶然或间接的损害不承担责任，包括但不限于商业利润损失、数据或文档丢失产生的损失，因遭受网络攻击、黑客攻击、病毒感染等造成的产品工作异常、信息泄露。

修改记录

对应版本	修订内容	修订人	日期
1.0.0	初次提交	路凯	2022-09-05

1. 内容简介

1.1 前言

本手册可能包含技术上不准确的地方或文字错误。

本手册的内容将做定期的更新，恕不另行通知；更新的内容将会在本手册的新版本中加入。

我们随时会改进或更新本手册中描述的产品或程序。

1.2 概述

MCS（消息中心服务）是我们为JES系统架构设计的提供基本消息管理的服务进程。本应用为守护进程，启动之后即可提供相关服务。

服务接口依赖JES通信系统，提供libjes_mcs.so 和 jes_mcs.h作为二次开发的sdk。

提供服务如下：报警消息发送，TLV消息发送，报警联动配置，布防时间配置，本地报警控制。

1.3 阅读对象

本文档的阅读对象为在JES-IPC上进行其他业务应用开发的人员，如存储应用、协议应用、算法应用等。

1.4 名词解释

名词	解释
MCS	Message Center service 消息中心服务的缩写
MSG	Message 缩写
TLV	一种二进制序列化数据，有规定的协议，具体协议见wiki 7. TLV格式定义 - 【云视通】南向设备协议 - 中维世纪 (jovision.com)

1.5 系统要求

目前主要运行在各个IPC芯片上，目前仅支持 linux 操作系统。

2. 编程引导

2.1 编程引导

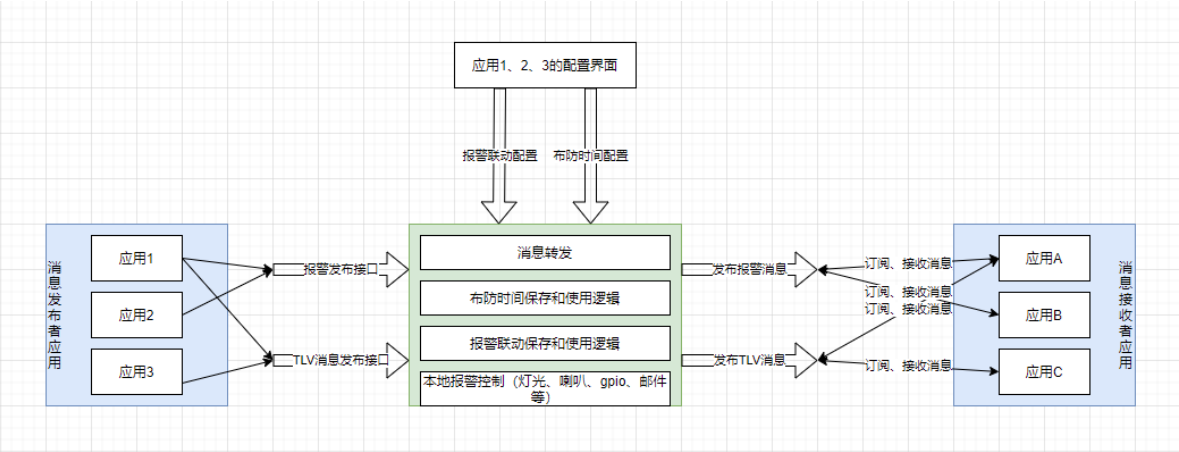
- 运行mcs服务
- 使用jes_mcs.h 和 jes_mcs.so 开发应用

2.1.1 目录结构

```
├── build   独立编译目录，单独调试时使用，放到buildroot中统一编译时无用
├── doc     文档目录
├── lib     依赖库以及头文件
├── release 独立安装目录，单独调试时使用，放到buildroot中统一编译时无用
|   ├── hi3516dv300     hi3516dv300平台的安装文件
|   |   ├── bin         存放mcs服务程序
|   |   └── sdk         存放jes_mcs.so/jes_mcs.a
|   ├── include
|       └── jes         存放jes_mcs.h
```

```
└── src          源码目录
    ├── app      mcs应用程序源码，依赖message_center和jbus实现服务应用
    ├── lib      message_center 业务代码
    ├── sdk      jes_mcs二次开发sdk源码
    └── test     测试demo的源码
```

2.1.2 MCS模块图

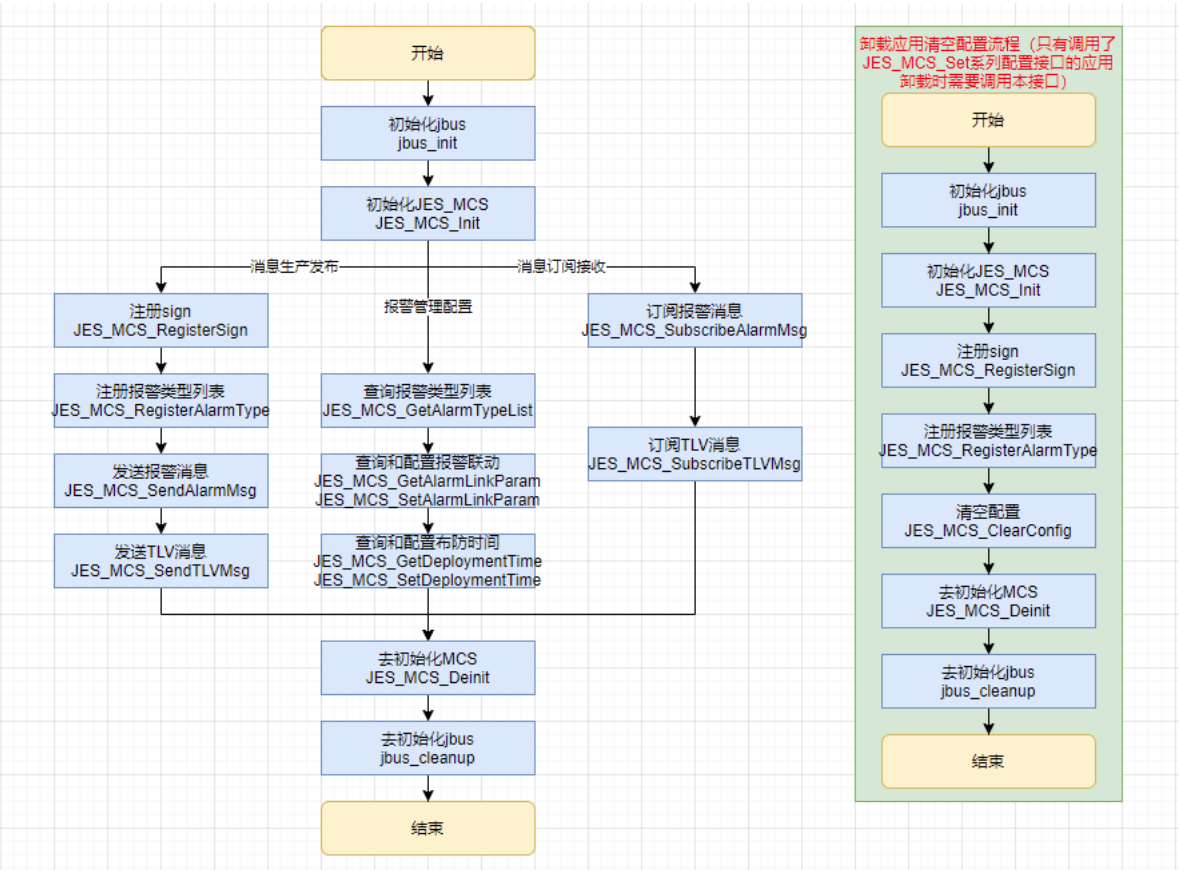


2.1.3 SDK开发指南

- 环境准备：烧有文件系统的硬件，内部有jbus、jbufffer运行环境，且相关服务dbus、mcs都在正常运行。
- jes_mcs.h头文件和库文件
- jes_mcs_test.cpp demo源码
- 参考demo源码和开发手册，进行新应用的开发。

2.2 JES_MCS接口调用流程

2.2.1 流程图



2.2.2 示例代码

参见demo，jes_mcs_test.cpp

3. 接口定义

3.1 初始化

3.1.1 MCS 初始化

接口声明：

```
jcs_hdl_t JES_MCS_Init(jbus_hdl_t jbus);
```

接口描述： MCS 初始化

接口参数：

参数名称	参数类型	参数说明
jbus	in	jbus_init 的返回值

返回值： 失败返回 NULL， 成功返回 句柄

专用错误码：无

Remarks：

MCS 为 Message Center Service 的缩写
jbus_init 位于 jbus.h 中，每个进程只可以调用一次 jbus_init

See Also：无

3.1.2 去初始化

接口声明：

```
void JES_MCS_Deinit(jcs_hdl_t hdl);
```

接口描述： 去初始化

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值

返回值： 无

专用错误码：无

Remarks：无

See Also：无

3.1.3 注册应用标识

接口声明：

```
int JES_MCS_RegisterSign(jcs_hdl_t hdl, const char *sign);
```

接口描述： 注册应用标识

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
sign	in	应用唯一标识

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：无

See Also：无

3.1.4 注册报警类型

接口声明：

```
int JES_MCS_RegisterAlarmType(jcs_hdl_t hdl, const char *alarmType, const char
*alarmTypeName);
```

接口描述： 注册报警类型

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
alarmType	in	报警类型，对应 AlarmMsg_t 中的 type
alarmTypeName	in	报警类型名称用于界面显示，须是用户可阅读的描述

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：

报警源需要把自己所有的报警类型以及报警类型对应的当前语言的名称注册进来，用于配置报警类型和报警联动的关系
此接口为报警源应用调用，发送报警前注册一次即可，允许重复注册如修改语言后重新注册可以替换alarmTypeName
应先调用JES_MCS_RegisterSign接口，应用支持多个报警类型时需要多次调用接口全部注册进去

See Also：无

3.1.5 获取注册的报警类型名称列表

接口声明：

```
int JES_MCS_GetAlarmTypeList(jcs_hdl_t hdl, const char *sign,
AlarmTypeInfoList_t *typeList);
```

接口描述： 获取注册的报警类型名称列表

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
sign	in	报警源应用的 sign，为 NULL 时表示查询所有应用注册的报警类型列表
typeList	out	输出报警类型列表

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：

用于界面展示及参数配置

See Also：无

3.1.6 发送报警消息

接口声明：

```
int JES_MCS_SendAlarmMsg(jcs_hdl_t hdl, AlarmMsg_t *msg);
```

接口描述： 发送报警消息

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
msg	in	报警消息

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：无

See Also：无

3.1.7 发送TLV消息

接口声明：

```
int JES_MCS_SendTLVMsg(jcs_hdl_t hdl, int channelid, uint8_t *data, int len);
```

接口描述： 发送TLV消息

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
channelid	in	通道号NVR上使用，IPC固定传0
data	in	tlv格式消息
len	in	数据长度

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：

主要用于智能分析结果数据、外设统计数据等的实时上报
tlv格式定义详见 http://wiki.jovision.com:8090/pages/viewpage.action?pageId=7439100
TLV格式封装和解析，请使用jvbase中的utl_tlv.h(c接口)或JCTLV.h(c++接口)
针对c++还提供了TLVParser.hpp工具类提供了TLV协议基础数据的封装和解析

See Also：无

3.1.8 订阅报警消息

接口声明：

```
int JES_MCS_SubscribeAlarmMsg(jcs_hdl_t hdl, JESMCSAlarmMsgCB cb, void *arg);
```

接口描述： 订阅报警消息

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
cb	in	报警事件回调
arg	in	用户参数

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：无

See Also：无

3.1.9 订阅TLV消息

接口声明：

```
int JES_MCS_SubscribeTLVMsg(jcs_hdl_t hdl, JESMCSTLVMsgCB cb, void *arg);
```

接口描述： 订阅TLV消息

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
cb	in	报警事件回调
arg	in	用户参数

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：

TLV消息主要指智能分析结果数据、外设统计数据等

See Also：

JES_MCS_SendAlarmMsg

3.1.10 查询报警联动

接口声明：

```
int JES_MCS_GetAlarmLinkParam(jcs_hdl_t hdl, const char *alarmType,
AlarmLinkParam_t *param, JBOOL bDefault);
```

接口描述： 查询报警联动

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
alarmType	in	报警类型，从 JES_MCS_GetAlarmTypeList 查询
param	out	该报警类型对应的联动参数
bDefault	in	是否获取默认参数

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：无

See Also：无

3.1.11 配置报警联动

接口声明：

```
int JES_MCS_SetAlarmLinkParam(jcs_hdl_t hdl, const char *alarmType, const
AlarmLinkParam_t *param);
```

接口描述： 配置报警联动

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
alarmType	in	报警类型，从 JES_MCS_GetAlarmTypeList 查询
param	in	该报警类型对应的联动参数

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：无

See Also：无

3.1.12 查询布防时间

接口声明：

```
int JES_MCS_GetDeploymentTime(jcs_hdl_t hdl, const char *alarmType,
DeploymentParam_t *param, JBOOL bDefault);
```

接口描述： 查询布防时间

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
alarmType	in	报警类型，从 JES_MCS_GetAlarmTypeList 查询
param	out	该报警类型对应的联动参数
bDefault	in	是否获取默认参数

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：无

See Also：无

3.1.13 配置布防时间

接口声明：

```
int JES_MCS_SetDeploymentTime(jcs_hdl_t hdl, const char *alarmType, const
DeploymentParam_t *param);
```

接口描述： 配置布防时间

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
alarmType	in	报警类型，从 JES_MCS_GetAlarmTypeList 查询
param	in	该报警类型对应的联动参数

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：无

See Also：无

3.1.14 查询邮件参数

接口声明：

```
int JES_MCS_GetEmailParam(jcs_hdl_t hdl, EmailParam_t *param, JBOOL bDefault);
```

接口描述： 查询邮件参数

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
param	out	邮件参数
bDefault	in	是否获取默认参数

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：无

See Also：无

3.1.15 配置邮件参数

接口声明：

```
int JES_MCS_SetEmailParam(jcs_hdl_t hdl, const EmailParam_t *param);
```

接口描述： 配置邮件参数

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
param	in	邮件参数

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：无

See Also：无

3.1.16 发送测试邮件

接口声明：

```
int JES_MCS_TestEmailParam(jcs_hdl_t hdl);
```

接口描述： 发送测试邮件

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：

测试 JES_MCS_SetEmailParam 配置的参数是否正确，本接口会阻塞等待发送结果

See Also：无

3.1.17 查询HTTP上报服务器参数

接口声明：

```
int JES_MCS_GetHttpServerParam(jcs_hdl_t hdl, ReportServerParam_t *param, JBOOL
bDefault);
```

接口描述： 查询HTTP上报服务器参数

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
param	out	HTTP上报服务器参数
bDefault	in	是否获取默认参数

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：

可用于报警上报和TLV上报，使用相同的服务器地址，不同的服务接口
报警上报接口 /alarm_report，TLV上报接口 /result_report

See Also：无

3.1.18 配置HTTP上报服务器参数

接口声明：

```
int JES_MCS_SetHttpServerParam(jcs_hdl_t hdl, const ReportServerParam_t *param);
```

接口描述： 配置HTTP上报服务器参数

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
param	in	HTTP上报服务器参数

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：无

See Also：无

3.1.19 查询FTP上报服务器参数

接口声明：

```
int JES_MCS_GetFtpServerParam(jcs_hdl_t hdl, ReportServerParam_t *param, JBOOL
bDefault);
```

接口描述： 查询FTP上报服务器参数

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
param	out	FTP上报服务器参数
bDefault	in	是否获取默认参数

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：无

See Also：无

3.1.20 配置FTP上报服务器参数

接口声明：

```
int JES_MCS_SetFtpServerParam(jcs_hdl_t hdl, const ReportServerParam_t *param);
```

接口描述： 配置FTP上报服务器参数

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
param	in	FTP上报服务器参数

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：无

See Also：无

3.1.21 查询TLV上报参数

接口声明：

```
int JES_MCS_GetTLVReportParam(jcs_hdl_t hdl, const char *sign,
ResultReportParam_t *param, JBOOL bDefault);
```

接口描述： 查询TLV上报参数

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
param	out	该报警类型对应的联动参数
bDefault	in	是否获取默认参数

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：

消息中心为AI应用提供通用的智能数据通过http/https协议上报第三方平台的服务
通过本接口配置是否需要发送，服务器地址配置依赖 JES_MCS_SetHttpServerParam
当配置了服务器地址，且本接口的param.httpEnable为true时，JES_MCS_SendTLVMsg 将会把TLV消息也发送到配置的http服务器上

See Also：

JES_MCS_SetHttpServerParam

3.1.22 配置TLV上报参数

接口声明：

```
int JES_MCS_SetTLVReportParam(jcs_hdl_t hdl, const char *sign, const
ResultReportParam_t *param);
```

接口描述： 配置TLV上报参数

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值
alarmType	in	报警类型，从 JES_MCS_GetAlarmTypeList 查询
param	in	该报警类型对应的联动参数

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：无

See Also：无

3.1.23 清空配置

接口声明：

```
int JES_MCS_ClearConfig(jcs_hdl_t hdl);
```

接口描述： 清空配置

接口参数：

参数名称	参数类型	参数说明
hdl	in	JES_MCS_Init 返回值

返回值： <0 失败，==0 成功

专用错误码：无

Remarks：

如果有使用JES_MCS_SetXXX这些接口，则应用被卸载时须调用本接口清空配置
本接口需要在调用 JES_MCS_RegisterSign JES_MCS_RegisterAlarmType 之后调用

See Also：无

4. 数据定义

4.1 结构体定义

4.1.1 报警消息

```
typedef struct{
    AlarmChnType_e chntype;
    int channelid;
    char type[64];
    AlarmStatus_e status;
    JTime_t time;
    char msg[128];
    char snapfile[128];
    char addition[1024];
    char uuid[128];
} AlarmMsg_t;
```

参数	说明
chntype	报警通道类型
channelid	通道号，从0开始
type[64]	报警类型：详见wiki报警类型管理
status	报警状态
time	报警时间
msg[128]	显示信息：用于界面展示，请保证是翻译后的可读内容
snapfile[128]	抓拍的图片文件的路径，为空时mcs服务会自动抓拍
addition[1024]	附加信息：base64编码的json数据，可用于如人脸报警携带人脸属性信息，json格式定义详见开发文档
uuid[128]	发送者无需填写内部生成，接收者使用

4.1.2 报警类型信息

```
typedef struct {
    char type[64];
    char typeName[64];
} AlarmTypeInfo_t;
```

参数	说明
type[64]	报警类型对应AlarmMsg_t中的type
typeName[64]	报警类型名称，用于界面展示

4.1.3 报警类型列表

```
typedef struct {
    int count;
    AlarmTypeInfo_t typeList[32];
} AlarmTypeInfoList_t;
```

参数	说明
count	int count;
typeList[32]	AlarmTypeInfo_t typeList[32];

4.1.4 报警灯光联动参数

```
typedef struct {
    JBOOL enable;
    int strength;
} AlarmLinkLight_t;
```

参数	说明
enable	是否联动闪烁
strength	闪烁强度: 白光灯时：1低频闪烁，2中频闪烁，3高频闪烁（或常量），RGB灯时0-100频率增加

4.1.5 报警声音联动参数

```
typedef struct {
    JBOOL enable;
    char filename[128];
} AlarmLinkSound_t;
```

参数	说明
enable	报警声音使能
filename[128]	报警音文件路径

4.1.6 报警输出联动参数

```
typedef struct {
    int count;
    int id[4];
} AlarmLinkOut_t;
```

参数	说明
count	联动id的数量
id[4]	报警输出id，最多可同时联动4个

4.1.7 报警联动参数

```
typedef struct {
    int delay;
    JBOOL recordEnable;
    JBOOL snapEnable;
    JBOOL emailEnable;
    JBOOL httpEnable;
    JBOOL ftpEnable;
    AlarmLinkLight_t whiteLight;
    AlarmLinkLight_t rgbLight;
    AlarmLinkSound_t alarmSound;
    AlarmLinkOut_t alarmOut;
}AlarmLinkParam_t;
```

参数	说明
delay	报警延时，用于脉冲式报警本地报警联动的结束
recordEnable	报警录像开启
snapEnable	抓拍联动
emailEnable	是否发送邮件
httpEnable	是否发送到http，http地址通过SetHttpServerParam接口配置，路径固定为/alarm_report
ftpEnable	是否上传图片到ftp，ftp地址通过SetFtpServerParam接口配置，路径固定为/设备ID/设备名称*报警类型*时间.jpg
whiteLight	白光灯联动配置
rgbLight	红蓝灯联动配置
alarmSound	报警声音配置
alarmOut	可用于联动的报警输出

4.1.8 布防时间参数

```
typedef struct
{
    int wday;
    JTime_t startTime;
    JTime_t endTime;
}DeploymentTime_t;
```

参数	说明
wday	星期几，0-6，0表示星期天
startTime	开始时间
endTime	结束时间

4.1.9 布防参数

```
typedef struct
{
    JBOOL allTime;
    int count;
    DeploymentTime_t times[28];
}DeploymentParam_t;
```

参数	说明
allTime	全时段布防
count	时间数量
times[28]	布防时间

4.1.10 布防参数

```
typedef struct
{
    JBOOL httpEnable;
}ResultReportParam_t;
```

参数	说明
httpEnable	结果数据(TLV格式)是否上报至自定义http服务，http地址通过SetHttpServerParam接口配置，路径固定为/result_report

4.1.11 邮件参数

```
typedef struct
{
    char sender[64];
    char server[64];
    char username[64];
    char passwd[64];
    char encrytmode[8];
    int port;
    char receiver0[64];
    char receiver1[64];
    char receiver2[64];
    char receiver3[64];
}EmailParam_t;
```

参数	说明
sender[64]	发件人
server[64]	服务器
username[64]	用户名
passwd[64]	密码
encrytmode[8]	加密方式:none,ssl,tls
port	端口号
receiver0[64]	收件人1
receiver1[64]	收件人2
receiver2[64]	收件人3
receiver3[64]	收件人4

4.1.12 上报服务参数

```
typedef struct
{
    JBOOL enable;
    char addr[64];
    int port;
    char username[64];
    char passwd[64];
}ReportServerParam_t;
```

参数	说明
enable	是否开启
addr[64]	abc.com
port	端口号
username[64]	用户名
passwd[64]	密码

4.1.13 声音文件信息

```
typedef struct{
    char filename[64];
    char type[16];
    int filesize;
    JBOOL modify;
}SoundFileInfo_t;
```

参数	说明
filename[64]	文件名称（单纯的名称不包含路径）
type[16]	pcm,g711a,g711u,aac
filesize	文件大小
modify	是否可修改

4.2 枚举定义

4.2.1 报警状态

```
typedef enum {
    ALARM_STOP,       //报警停止
    ALARM_START,      //报警开始
    ALARM_PULSE,      //脉冲式
    ALARM_MAX
} AlarmStatus_e;//报警状态
```

4.2.2 通道类型

```
typedef enum {
    ALARM_CHN_TYPE_VIDEO_INPUT, // 视频输入通道
    ALARM_CHN_TYPE_ALARM_INPUT, // 外部输入通道
    ALARM_CHN_TYPE_DEVICE,      // 设备自身报警
    ALARM_CHN_TYPE_MAX
} AlarmChnType_e; //通道类型
```

5. 帮助

5.1 环境准备

- 详见 2.1.3 章节

5.2 编译运行demo

- 详见 2.1.3 章节

5.3 FAQ