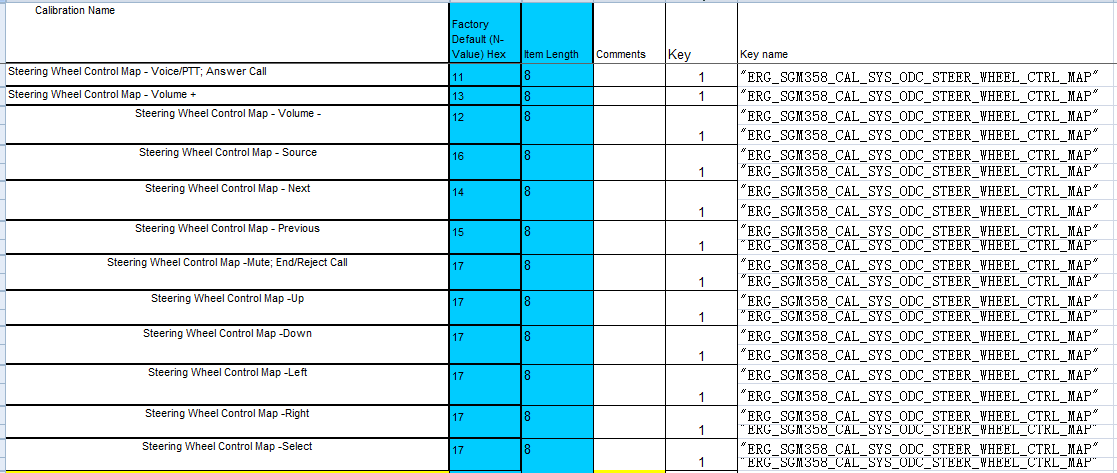
Tool Requirement

以下图为例：



# 1．生成头文件A给PATAC，相同内容的 ”Key”列 自动生成如下2个结构，生成bin的xls文件统一自动生成1个头文件A：diagCalibrationKeyMap\_pub.h

// 结构体和联合体名字命名规则：”u\_” 和 Keyname 字段去掉” ERG\_SGM358\_CAL\_”/后的内容组合起来的字符串

typedef struct s\_SYS\_ODC\_STEER\_WHEEL\_CTRL\_MAP

{

// calibration name 列Steering Wheel Control Map - Voice/PTT; Answer Call

// 遇到空格转成下划线，解析到无法识别的字符就停止，这里假设”;”无法解析

char Steering\_Wheel\_Control\_Map\_-\_ Voice/PTT; // 注1

char Steering\_Wheel\_Control Map\_-\_Source;

...

char Steering\_Wheel\_Control\_Map\_-\_Select;

} SYS\_ODC\_STEER\_WHEEL\_CTRL\_MAP;

union u\_SYS\_ODC\_STEER\_WHEEL\_CTRL\_MAP

{

char buffer[13]; // 相同Keyname 的Item Length 总和,用于存放keydata

SYS\_ODC\_STEER\_WHEEL\_CTRL\_MAP map; // 用于将buffer映射到xls的每1行

}

注1：

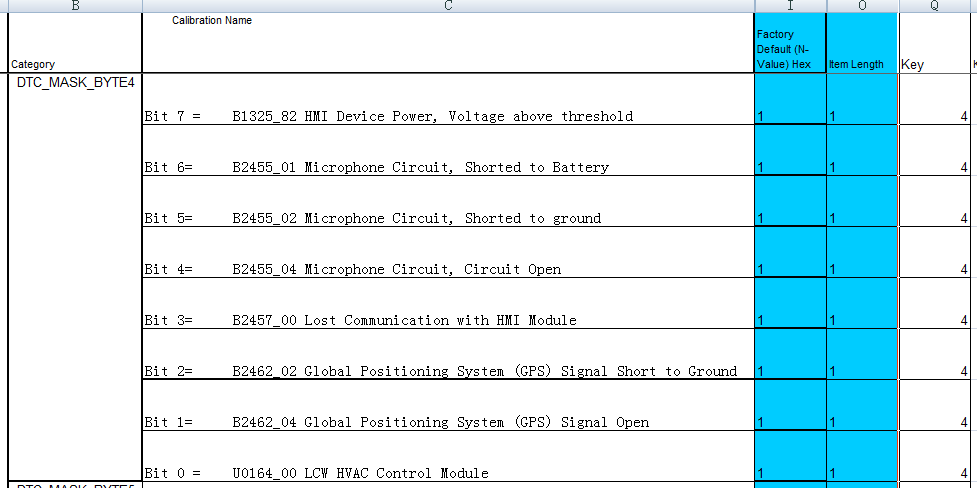
// 数据类型根据Item Length 定

// 8 -> char

// 16-> char [2]

// 2048 -> char [256]

// 1 ->



struct s\_KEYNAME\_BYTE4//结构体名称使用key name+BYTE+x ，x为第x个字节

{

char Bit\_7 :1;

char Bit\_6: 1;

char Bit\_5:1;

char Bit\_4:1;

char Bit\_3:1;

char Bit\_2:1;

char Bit\_1:1;

char Bit\_0:1;

} KEYNAME\_BYTE4;

注2： keyname列为空的行，不体现到该头文件中。

# 2. 生成头文件B，Conti内部使用。用于Key列和Key Name列的映射。文件B：diagCalibrationKeyNameMap.h

Key列内容后续会改成2 bytes 长。首字节表示属于哪个xls文件，第2个字节表示xls文件中的第几个key。

文件内容为：

typedef struct s\_diagKeyid2KeyName

{

char Key\_id[2]; // key列内容

char \*pKey\_name; //key name 列内容

} diagKeyid2KeyName;

diagKeyid2KeyName keyid2keyname\_table[] =

{

{0301, "ERG\_SGM358\_CAL\_SYS\_ODC\_STEER\_WHEEL\_CTRL\_MAP" }

…

};

注意：

1. 所有要求解析的文件统一生成1个头文件
2. Key为0的数据项不做体现，那是calibration header部分，不作为key

# 3．生成的binary文件内容调整。

Binary 文件结构：

|Key列为0行的value部分（固定14个字节）| -> 对应xls文件头部分

|keyid （2字节）|key length（2字节）|KeyValue| -> 对应xls 数据项部分

原先的keyid字段从1->2字节，为保证总文件长度是偶字节。

Checksum字段计算改动，把checksum字段后所有的字节都纳入计算范围。而不是只统计KeyValue域的内容。