**60路语音TDMA协议设计**

## 一、概述

本文档要参考《语音TDMA原理设计》来看。

要解决的问题两个，一个是键盘和基站之间的空中数据协议，也就是原理文档涉及到的功能模块的具体协议细节，另外一个是基站到SDK之间的通讯协议。两部分协议都有一个要求，尽量兼容原有tcpip体系的语音答题器体系，减少键盘、基站、SDK的研发工作量。由于是重新设计，协议实现不会有问题，所以核心问题是怎么尽量兼容。

键盘协议的限定条件是空中64字节，基站和SDK协议是USB数据一包最大1024字节，最快间隔1ms，实际长度可以调整。

键盘由于空中64字节，可以包含原来的32字节，所以，除了语音数据重新设计以尽量提高传输效率外，其他协议数据细节原则上保持一致，例如投票信标、提交单包数据格式等。

由于体系上需要配套使用， 5.8G芯片空中数据是带地址码的，地址码不同根本不接收，键盘带NFC，理论上可以刷卡配对传输地址码和频点。如果走空中快速配对的方式，也可以配对时候使用统一的地址码，所以技术上64字节空中数据不带配对码是可以的。但这样做有个缺点，地址码不同，如果基站频点冲突，基站相互之间是无法提示的，无人统筹安排频点的话，会有相互干扰的隐患。

所以，如果不影响语音数据传输效率很多的话，先保持配对码在64字节里面。考虑到还有组网需求，用3字节配对码+1字节组网号代替原先4字节配对码，在空中数据最前面。实践中2.4G数据层还是需要加CRC效验，所以末尾加2字节CRC。所以最终空中数据格式是：

3字节配对码 + 1字节组网号 + **1字节CMD + 57字节数据** + 2字节CRC

后面的键盘协议格式，讨论的都是从CMD开始到CRC前的数据，共58字节。

CRC从CMD开始计算，按老体系CRC16算法。

键盘和基站根据CMD，来判断后面57字节的数据格式。

另外，5.8G系统，语音情况下1个时隙（大约2.74ms）内，单个键盘可以传8个64字节包，这是和老2.4G系统不同。考虑到基站要不停接收数据和通过USB发送到SDK，尽量减少处理数据的时间。所以，基本上，键盘传过来的数据（单包和语音多包，比如0x94类结果包），原则上基站只是记录（对RAM有要求）和转发，也就是不再做打包处理！为提高传输效率，最多只做拼包处理，以64字节为一段。非语言的情况下一个时隙是1.6ms。每个大周期为28.6ms。

大周期时序分配：

语音情况下： 基站1.2ms（3帧）+10\*键盘2.74ms =28.6ms

非语音情况下： 基站 2.74ms+ 10\*键盘1.6ms+入网9.86ms=28.6ms

非语音情况下键盘可以自动选择在未满键盘的时序或是入网时序中入网。

语音情况下，未入网的键盘也自动搜索未满的键盘空隙入网。如果键盘个数是整10，没有空隙，则基站强制发送一个重传帧用来入网（pei：可取消，语音情况下可不考虑入网，都用于传语音）。

大周期最多安排9个周期，60个键盘情况下，前6个周期用于点名传输新一轮数据（最多52x8=416字节），后3个周期用于重传。点名周期的数目也是根据键盘数量来的，如果30个键盘，那就是3周期点名，重传最多6周期，重传都ok后也会终止重传。

和30路时序主要区别是，发包改成8帧，单个时间片由2.5ms改成2.74ms，然后分语音模式和非语音模式。语音模式下基站询问指令减少为3帧，1个投票2个语音点名，没有确认包。

推算60键盘会不会buf溢出，按重传最多3周期计数，共9周期是257.4ms，20ms语音压缩后32字节，411.8字节，键盘1次提交可以416字节，然后最多有3次重传机会，其实是30个键盘有1次重传机会。按理正常通讯下丢包应该有限。（V0.9协议不是可靠传输，9周期内收不齐就不收了，下一轮重新开始传新的语音数据）

备注：30路执行的时序

每周期2.5ms，每周期基站和键盘可以发7帧数据，被点名的键盘按指定的时序提交。

基站7帧是投票结果确认包相同2帧，投票信标相同2帧，（语音）点名提交或重传相同3帧，发相同是为了键盘可靠接收到。那么每个周期是1+10=11x2.5ms=27.5ms，发语音数据的时候，每帧48字节有效数据。可参考V0.9版本以前协议。

大周期是3个周期点名发数据，然后最多3个周期用于重传，重传是1次没有接收全的键盘被基站点名重新发。

## 二、键盘和基站

基站定时28.6ms发送信标，定时先发信标时隙。这个信标时隙决定了后面键盘要干什么事，要怎么处理数据和提交数据。TDMA的话，基本是指定SN号或者内部ID编号的键盘在第几个时序提交数据，然后键盘按照设定的时序提交数据。

基站的信标时隙，也一样可以在时隙内发7-8个64字节数据包，这是和2.4G体系不同的地方。一般情况下（非语言），包含投票信标、单包确认信标、指定SN号查询数据（含语音数据重发查询），考虑到干扰，可以投票信标发2次，单包确认信标发2次，指定SN号查询数据发3次（特别是语音查询，接收不到就不发语音数据就错过了，所以要多次发）。键盘处理1次即可，不用多次处理指令。语音情况下发送投票1次指定SN查询2次（1.2ms）。

投票信标同2.4G体系，用于告诉键盘现在投票状态。单包确认信标类似2.4G的确认包，我们采用简单单包数据和语音数据可以混合发的机制，单包确认信标就是告诉键盘哪个数据包已经接收到。指定SN号查询数据就是说哪些键盘可以提交数据，和提交哪些数据。

基站发完信标时隙后，就是10个周期的接收时序，指定了时序的键盘，依次提交。一样，每个时隙，键盘最多可以提交7-8个数据包，也可以说没数据了。

### 2.1数据查询和语音传输

#### 2.1.0点名查询指令

就是指定10个（V2.0支持不足10个）SN号查询数据，包含了语音数据的指定包重传。

指令格式：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | GETCMD | 0x11 指定SN号查询数据，TDMA体系 |
| 2 | SEND\_SEQ | 发送序列号，1起， 1到7 （实际实现是~~5、6、7~~—60路体系改发1、2）  每个时隙内查询包会发送~~3次~~（60路是2次）以提高键盘收到的概率，为了时间同步，必须让键盘知道自己接收的是第几次，然后才能进行准确的延时。 所有键盘时间的基准就是数据查询和语音传帧。收到一帧后就开始进行计时。 |
| 3 | HOP | 0没跳频，1-4表示当前模块编号，1是主模块  （键盘能判断跳频模式是否启用，也能在不同步重新锁频收到模块编号不对时候，快速推断自己的模块所在频点） |
| 4 | ASKN | （V2.0开始）原先00，表示老版本的10个点名  低4位是01-10的时候，表示动态数量点名，只有ASKN个键盘应答，应答时间也只有ASKN个  高4位00的时候，表示按标准周期（送8片的时间）间隔应答  01表示使用缩短间隔模式1应答，用于送简短数据，比如应答录音是否开始 |
| 5 | ATTRIB1 | 对应教育键盘协议里面的ATTRIB1字节，主要是关机时间  （用于解决科大讯飞版本设置关机时间启动了投票信标） |
| 6-8 |  |  |
| 9-12 | SN1 | 第1个时序的键盘的SN号，4字节，高位前，全00表示空  SN号是这个SN1的键盘，在第1个时序做应答 |
| 13 | SN1-CMD1 | SN1的键盘的命令码  00，表示不需要做语音重发，键盘发什么数据都可以（实际上不论CMD1值如何，键盘都可以在自己的时间片里面发单包结果）  >1时候，用bit表示，对应bit=1，表示刚才的语音数据包的第几包需要重新发  例如值=3时候，表示第1、第2包都要重传 |
| 14-18 | SN2+CMD2 | 5字节，第2个键盘的SN号和控制命令 |
| 。。。 | 。。。 |  |
| 。。。58 | 。。。CMD10 | 一直到第10个键盘的SN号和控制命令  第58字节是最后1个数据 |
| 59-60 | CRC | 空中数据最后2字节是CRC |

键盘在对应时序，可以提交单包数据，也可以加上语音数据提交，不超出时序时间即可。

如果提交单包数据，按3.4章节描述。

**V0.9提交语音数据格式**如下：（V0.8 48字节的有效语音数据，ANSCMD=0x95）

（V0.9 取消了ID和CRC，多4字节，ANSCMD改0x97）

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | ANSCMD | 0x97 语音数据包，60个TDMA体系  （0x95是V0.8格式，0x96是组网） |
| 2-5 | SN | 键盘的SN号，4字节，高位前 |
| 6-7 | PACKH | ~~语音数据包的段号，可以理解为语音周期号，高位前~~  ~~确定了语音压缩格式，每一周期代表多少时间的语音数据，以及每一周期多少字节的数据，是确定的~~  ~~0000表示录音停止或未开始录音，FFFF可以表示实时语音流？~~  ~~如果高4位做别用，100ms一个周期的话可以409.6秒语音~~  用第6个字节表示语音包的序号，~~0-255 循环~~ 0表示录音没开始或停止---实际实现0是发心跳包代替，1-255循环，表示在录音。  第7个字节的~~Bit7表示语音停止，为1表示正在录音~~。  Bit6表示压缩类型，暂0  低6位表示本帧VOCDATA可用的字节数量0-63 |
| 8 | SLOTN | ~~低4位是本次周期语音包的片号，0起~~  ~~高4位是压缩格式，暂0~~  ~~根据格式，基站能推知一个周期内应该有多少包数据，所以可以知道没收到哪些，安排重发~~  高4位是本周期总片数。  低4位是当前片号，（代码实现是）1-8  基站可以根据这个判断本周期内哪帧的语音没有收齐需要重传  目前speex压缩，30路语音，一个周期大概100ms是150字节，4片数据就提交完，一个时序可发7包，键盘还可以安排发送2次 |
| 9-60 | VOCDATA | 语音数据，52字节 |
|  |  |  |

基站如果收到语音数据包，根据数据格式，就知道这个周期，键盘哪片语音数据没收到，然后就安排特定的查询重发周期，让指定键盘重发。这个重发周期，基站可以根据键盘数量、丢包情况灵活处理，例如只有一两个键盘丢包，那么可以给这两个键盘更多的时序保证重发多次。但总体上，下一个周期的语音数据要传输了，前面传输不上的就不要了，语音是可以丢包的。

#### 2.1.1语音可靠传输格式

传输方案，先沿用V0.9的最多6点名+3重传，然后键盘加大buf，然后键盘可以要求基站特殊多安排重传（信号差传得慢，buf快不够了），甚至优化成1个点名周期可以安排多个时间片给键盘。如果键盘buf够大也不用太优化。

类似A1手写数据可靠传输，基站认为键盘每次都送8片，每次都问键盘送上次没收到的片数，键盘端判断上次没发的不会重发，比如上次只发了4片，然后基站自然要求后4片重发，键盘自判不处理。重发，是把要求重发的片挪到前面发，例如要求重发1、3、5，那么上次的1、3、5片数据是新一轮的1、2、3片（注意packno还是原来的），然后后面5片可以取新数据发送。如果重发值是00，自然是都可以发新数据。

键盘提交语音数据格式如下：（48字节的语音数据，可靠传输）

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | ANSCMD | 0x98语音可靠数据包，TDMA体系  0x9B 图片可靠数据包，60路同时传输 |
| 2-5 | SN | 键盘的SN号，4字节，高位前  （如果用入网编号可1字节，也不用ID了，多出5字节，8包多40字节相当于多发20ms语音数据） |
| ~~6-7~~ | ~~ID~~ | ~~键盘编号ID，高位前，如果高位=0xFF，表示这是最后一片数据，低位是VOCDATA实际长度，以适配传文件要求字节数不一定是48的倍数~~ |
| 6 | ID0 | 用于传实时丢包率的1个字节，4包才组成1个数据，具体参考程序 |
| 7 | RECPOS | 录音编号，新一轮录音递增，1开始 |
| 8 | SLOTN | （SDK不需要处理）  ~~最高位Bit7=1，键盘语音缓存区紧张，请求多安排重发周期点名~~（做语音停止信息包冲突了，所以变成Bit4）  ~~Bit6-bit4是键盘这次一共发多少有效包，0-7，加1就是包数~~（键盘不传，基站就当8包来处理重发，键盘自己判断是否需要重发，因为键盘知道上次我发了几包）  低4位是当前片号，实际是0-7，共8片  基站可以根据这个判断本周期内哪帧的语音没有收齐需要重传  （键盘没有数据提交，也要求发1包用来表示心跳，SLOTN=0xFF，基站自动丢弃不送SDK）  Bit4=1表示语音buf告警，缓存紧张，基站多扫描我（2.1.2节协议） |
| 9-10 | PACKNO | 后面VOCDATA的数据包的序号，48字节1个序号，依次增加，高位在前，1起，理论上可以传65535x48=3072K字节，实际2048即可传96K=60秒speex语音  PACKNO是发送的时候增加的，重传的时候不要变，  SDK按位置=PackNox48保存即可组成完整文件  V1.21 PACKNO用低15位，高位作为录音变化标志（字节9的高位），这样SDK好判断是新的录音，解决连续2个录音间隔很短，停止心跳丢失，SDK不好判断的问题 |
| 11-58 | VOCDATA | 语音数据，48字节，键盘够48字节才合成1片提交，不够先暂存不发送。  录音停止时候，最后1包可能不足48字节，填00，键盘可靠送出后，发普通心跳状态包，SDK认为录音停止，关闭文件。（也可以用SLOTN标志位，或者PACKNO高位） |
| 59-60 | CRC |  |

#### 2.1.2单周期重传数据

就是指定1个SN号查询数据，而且只用1个周期（1个时间片）应答。

实践中，有个别键盘语音录音中buf很紧张，需要特殊安排多的周期传输数据，减少丢语音的概率。考虑到要切换回标准的10个SN的数据周期，还有数据包片段的确认方法，最简单的还是沿用键盘1次送8包的方式。这样，所有键盘收到这条指令时候，如果不是点名到自己，可以不休眠，或者只休眠1个周期（平常2.1节指令都是10周期）。而收到点名的键盘，马上应答8包数据。基站根据键盘应答是否buf还继续紧张来控制是否继续对他继续询问，如果不紧张了，可以切换回10个SN应答模式。理论上也不会对1个键盘连续点名很长时间，比如最多10次就转回正常模式一轮。

（备注：2.1.0节支持单个键盘点名后，本指令其实可以用2.1.0节替代）

指令格式：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | GETCMD | 0x15 指定SN号查询数据，TDMA体系  （0x12用做无线升级广播了，0x13是老竞争信标，0x14是80包连续） |
| 2 | SEND\_SEQ | 发送序列号，60路体系发1、2。  小周期编号，1开始，用于跳频时候计算换频时间 |
| 3 | ~~PACKH~~ |  |
| 4-8 |  | 未使用 |
| 9-12 | SN1 | 第1个时序的键盘的SN号，4字节，高位前，全00表示空  SN号是这个SN1的键盘，在第1个时序做应答 |
| 13 | SN1-CMD1 | SN1的键盘的命令码  00，表示不需要做语音重发，键盘发什么数据都可以（实际上不论CMD1值如何，键盘都可以在自己的时间片里面发单包结果）  >1时候，用bit表示，对应bit=1，表示刚才的语音数据包的第几包需要重新发  例如值=3时候，表示第1、第2包都要重传 |
| 后面 |  | 暂时没意义 |

收到SN点名的键盘，应答数据包格式，同2.1-A节。

#### 2.1.3带压缩方式语音包-含丢包率

和原来0x98语音数据包类似，只是为了加上语音压缩方式，占用了原先ID的位置。

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | ANSCMD | 0x9D语音可靠数据包，TDMA体系 |
| 2-5 | SN | 键盘的SN号，4字节，高位前 |
| 6 | INFOTYPE | 低4位代表语音压缩编码类型：  01-Opus  高4位不是0表示特殊信息：  F—丢包率信息 |
| 7 | CODER | 码率，暂1=12.8Kbps |
| 8 | SLOTN | Sdk不用处理 |
| 9-10 | PACKNO | 同0x98指令 |
| 11-58 | VOCDATA | 语音数据，同0x98指令，丢包率包时候是特殊信息 |
|  |  |  |

丢包率包的时候：（INFOTYPE高4位是F）

|  |  |  |
| --- | --- | --- |
| 9-10 | PACKNOMAX | 键盘上传的packno的最大值，理论上SDK就应该收到这么多  根据这个值SDK即可计算丢包率 |
|  |  | （以下信息是调试信息，SDK可转成文本附加信息输出） |
| 11-12 | ADC | 键盘ADC次数，单位20ms，高位在前 |
| 12-13 | CODEN | 键盘调用语音压缩次数 |
| 14-15 | ALERT | 键盘buf不足告警的最大时间 |
|  |  |  |

#### 2.1.4 可不满10个键盘的点名查询

用于指定不满10个键盘的点名。在做T2Soc的时候，单通道键盘容量可能是25个，老协议每组10个，第3组就空了5个周期，虽然T2 5.8G基站优化补填了5个，但对T2Soc不是最优化的。如果个数可以随意，最后1个周期可以根据实际键盘数量，然后键盘buf告警就独立用单独点名，对T2soc多模块的方案更好。

而且基站点名算法可以改成有哪些键盘录音了就只点哪些键盘，而不是老版本依次点名，如果键盘没开始录音或者已经传输完，依次点名就有点浪费时序。新方案在录音键盘不是全部的情况下（大部分使用情况不会同时录音），相对依次点名，部分录音键盘的传输时间更多，提高了可靠性。同时实时流（语音麦克风）的时候，语音延时也会大大改善，因为只有1键盘录音。

个数变化，键盘休眠算法改起来简单，主要是基站点名算法，要处理好在录音、录音结束和未开始录音的键盘的查询，也就是要满足开始录音的键盘怎么加入录音点名队列。

V2.0版本开始支持，参见2.1.0节的修改。

#### 2.1.5语音停止信息包

在语音0x98数据包上修改，用于通知SDK录音结束，并带丢包率等数据。

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | ANSCMD | 0x98语音可靠数据包，TDMA体系 |
| 2-5 | SN | 键盘的SN号，4字节，高位前  （如果用入网编号可1字节，也不用ID了，多出5字节，8包多40字节相当于多发20ms语音数据） |
| 6-7 | ID |  |
| 8 | SLOTN | 0xF1 语音停止包 |
| 9-10 | PACKNO |  |
| 11 | RECPOS | 录音编号，录音1次加1。原语音数据第1个字节 |
| 12-15 | PUSH | 压栈语音字节数，等同于录音数据字节数，4字节，高位前 |
| 16-19 | POP | 出栈语音字节数，等同于传输到基站字节 |
| 20-23 | DELAY | 录音停止时候，到最后一包的传输时间，单位20ms |
| --58 |  |  |
| 59-60 | CRC |  |

### 2.2组网和白名单

#### 2.2.1 组网设计【T2 5.8G 老版本】

组网申请是指，键盘开机后，向基站报到。这主要关系到TDMA的效率，如果一个键盘没开机，或者已经配对到另外基站了，基站不知道，还一直查询，那么信道资源就被占用了。

键盘开机后，先在原先频点接收基站信息，然后判断前3字节是否是自己的配对码，如果是，判断组网号，组网号相同，就不用申请。组网号不同，表示基站开始了新一轮使用，需要申请入网。

组网申请类似白名单申请过程。

键盘用教育协议3.2.4节登录信息，ANSTYPE=0x94，TYPE=11。---实际T2键盘用的是0x96指令（0x95是语音数据），格式和标准语音包一样，只是命令码0x96，然后基站没有做应答，只是开始扫描对应键盘SN了，然后键盘知道入网了。

基站用教育协议2.3.2节基站带确认信息应答，MSGTYPE=1格式可确认多个SN。（暂时不自动分配ID。键盘也不用取配对码、频点等信息，暂时用NFC获取）

~~键盘组网申请和基站应答，都在组网时序2.5ms内完成，基站的应答，可以固定在第7个内部时间片位置，这样前面6片做申请。~~

A1手写，由于要支持白名单，所以还是按设计来做，不用T2的0x96申请。基站的应答，如果是组网，可以放在组网2.5ms里面的第7片，用0x1A应答（键盘提交申请后需要监听）；也可以放基站信标7片里面，2个确认包里面第1个改0x1A

白名单频点，先暂定0。。。

#### 2.2.2 T2Soc版本组网

T2Soc版本，基站是多模块，并要适配配对、白名单和遥控开机。

一种方法是沿用T2 5.8G方式，那么刷卡配对时候，基站就得主动分配不同的频点，键盘跳到具体频点用0x96指令申请入网（在点名周期的空周期，例如10个点名，后5个点名是空）。

T2soc采用了新平台的方式，主模块频点有独立的组网周期，有组网应答，组网应答包括切换频点（这样可多模块多频点），也可以拒绝（白名单）。然后主模块也会随机定时在特定频点广播基站信息，也就是基站在什么频点，是配对模式还是白名单等信息，这样键盘离线后直接到特定频点监听，不用遍历频点，离线锁频比较快。

这样，刷卡配对只有1个配对码，基站容易处理。

同时，主频点才安排组网周期，副频点副模块上不安排组网周期，负载均衡、白名单都是主模块协调完成。由于基站是模块结构，所以主模块还需要通知副模块有键盘SN加入(副模块好点名键盘)，离线的时候，副模块也要通知到主模块（主模块空出连接资源，新键盘可加入），这是比较特殊的地方。

具体来讲，就是基站主模块会在特定频点广播基站连接信标，基站在什么频点，配对码是什么等信息，1秒广播2次以上，多基站之间规避同时发送。

键盘开机后监听，听到基站组网信标，获取到主频点，跳到主频点监听组网周期。

组网周期是组网信标和确认信标，之间10个时间片，要入网的键盘，在10个周期里面随机提出入网申请。组网确认信标里面，会通知键盘是否能入网，在什么频点。如果键盘没收到确认包指示信息，下一个组网周期继续申请。

组网周期会耗费12个时间片时间，所以语音模式下可能1秒内只安排2个组网周期，非语音模式就安排多一些。

#### 2.2.3 基站连接信标

指令格式：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | GETCMD | 0x51 基站连接信息（基站配对码已经在前面） |
| 2 | NET\_SEQ | 组网序号，如果变化，键盘重新入网 |
| 3 | NETMODE | 基站组网模式，1配对 2白名单 |
| 4-7 | FREQ1-FREQ4 | 主频点，副频点2，副频点3，副频点4，  0的话不启用 |
| 8 | HOP | 0没跳频，1-4表示当前模块跳频频点编号，1表示在主频点 |
|  |  | 基站名称等，暂时没意义 |
| 59-60 | CRC |  |

#### 2.2.4 组网信标和确认

组网信标指令格式：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | GETCMD | 0x52 组网信标 |
| 2 | NET\_SEQ | 组网序号，如果变化，键盘重新入网 |
| 3 | NETMODE | 基站组网模式，1配对 2白名单 |
| 4-- | NetInSlotN | 第几个组网小周期，1开始，用于跳频时候，键盘好计算换频点时间 |
|  | CONFIGS | 基站常用特性，类似原先基础信标内容，可设置关机时间等 |
| 59-60 | CRC |  |

入网确认信标指令格式：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | GETCMD | 0x53 入网确认信标 |
| 2 | TYPE | 暂1，表示格式1 |
|  |  | 空 |
| 9-12 | SN1 | 4字节SN， 第1个键盘的SN |
| 13 | CMD1 | 第1个键盘的入网结果  0 失败 1 成功在主频点 2入网ok，但换频点2 3换频点3 4换频点4 9系统容量满 10没在白名单 |
| 14-18 | SN2-CMD2 | 第2个键盘的SN和入网结果  每个键盘5字节，1次可确认10个键盘入网（但入网效率做不到那么高） |
|  | 。。。 | 第3个到第9个SN和入网结果 |
| --58 | SN10-CMD10 | 第10个键盘的SN和入网结果 |
| 59-60 | CRC | 暂时没意义 |

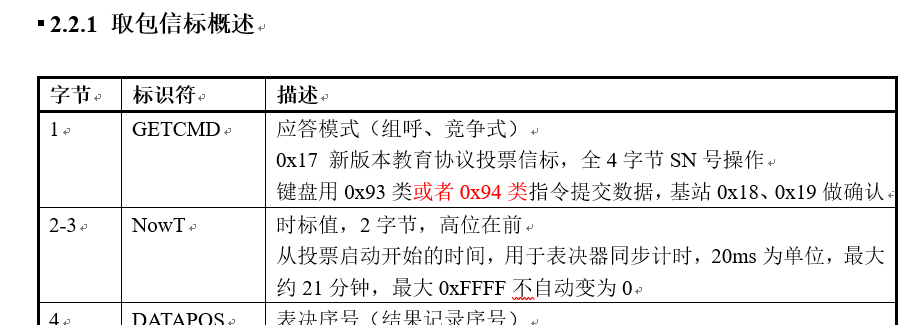
#### 2.2.5 键盘入网申请

入网确认信标指令格式：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | GETCMD | 0x96 入网申请 |
| 2 | TYPE | 暂1，表示格式1 |
| 3-6 | SN1-SN4 | 4字节SN |
| 7 | HW | 键盘硬件版本 |
| 8-10 | SW1-3 | 软件版本，3字节 |
|  |  |  |
| 59-60 | CRC | 暂时没意义 |

### 2.3投票信标

按教育新版协议的2.2.1节，只是最后2字节效验可放到58-59位置。



### 2.4单包上传

如果是键盘提交单题结果，例如单选多选，按教育协议3.2.11节，只是CRC挪位置



如果是键盘提交测验、字符串结果，暂时按教育协议，挪CRC位置，后期可优化利用完64字节



对于以上两种提交结果，如果有键盘提交了，基站接收到了，就记下SN，在下一个周期，注意是下一个周期的时序里面的确认信标，基站用确认包格式告诉键盘上一个单包数据提交我收到了，这也是

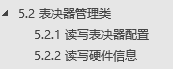
确认包指令格式：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | GETCMD | 0x18 确认包，指令码和2.4G的一样，但长度加长，可以放10个键盘的SN号 |
| 2-8 |  | 暂填00，空 |
| 9-12 | SN1 | 被确认的第1个键盘的SN号，4字节，高位前，全00表示空  SN号是这个SN1的键盘，就知道我上次提交数据成功了 |
| 13 | CMD1 | SN1的键盘的命令码，暂时00 |
| 14-18 | SN2+CMD2 | 5字节，第2个键盘的SN号和控制命令 |
| 。。。 | 。。。 |  |
| 。。。58 | 。。。CMD10 | 一直到第10个键盘的SN号和控制命令  第58字节是最后1个数据 |
| 59-60 | CRC | 空中数据最后2字节是CRC |

后期有1个可优化的地方，键盘提交单包数据都是不带结果提交编号SENDPOS的，如果加上，CMD1的地方可以作为SENDPOS回传，键盘就更容易判断结果是否提交成功，SDK也好判断，是否结果前面已经提交过，过滤掉。

### 2.5单包下载

一期项目暂不实现优化，按老流程，暂用键盘协议5.2节，可改长度和CRC位置。



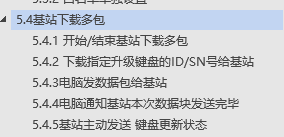
### 2.6多包下载

T2 5.8G：

键盘沿用多包下载的6.2节和6.3节，但每包数据改为32字节



基站和SDK协议沿用基站协议5.4节。



【T2Soc版本更新为】

由于支持64字节包，所以更新为每包48字节，1次广播最大24包，实际22包可1024字节，键盘询问应答变成3字节24bit。为区别，改为0x41指令。

#### 2.6.1 进入和退出多包下载

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | CMD | 0x41 下载多包类指令-24包，每包48字节 |
| 2 | DOWNCMD | 1 进入下载 0 退出 |
| 3 | DATAPOS | 下载的DataPos，新下载就变化1次 |
| 4-5 | KEYID | 表决器编号，2字节，高位在前  一般要指定表决器编号使指定键盘进入或退出下载状态  FFFF，是用序列号指定键盘 |
| 6-9 | SN4 | SN号，4字节 |
| 10 | DOWNTYPE | 多包类型 30=固件 |
| 11 | DOWNNO | 用于同类型数据下载多个，一般是0 |
| 12-14 | TOTALLEN | 3字节下载内容总长度，单位字节，高位前 |
| 15-16 | PACKCRC | 下载数据的CRC，效验计算从下载内容首字节到TOTALLEN  高位前，0000的话不效验 |
|  |  |  |
| 59-60 | CRC16 |  |

表决器回应结果状态：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | CMD | 0xC1 下载多包类指令应答 |
| 2 | DOWNCMD | 1 进入下载 0 退出 |
| 3 | DATAPOS | 下载的DataPos，新下载就变化1次 |
| 4-5 | KEYID | 表决器编号，2字节，高位在前  FFFF，是用序列号指定键盘 |
| 6-9 | SN4 | SN号，4字节 |
| 10 | STATUS | 1 成功 2失败 |
|  |  |  |
| 59-60 | CRC16 |  |

#### 2.6.2 广播式下载数据包

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | CMD | 0x41 下载多包类指令 |
| 2 | DOWNCMD | 2 下载具体数据包，必须是前面进入下载模式的才执行  3下载具体数据包，所有键盘收取 |
| 3 | DATAPOS | 下载的DataPos，新下载就变化1次  如果DOWNCMD=2，键盘判断和下载启动的Datapos相同才处理 |
| 4-6 | LEN | 下载内容总长度，暂00可不管 |
| 7 | DOWNTYPE | 多包类型 |
| 8 | DOWNNO | 用于同类型数据下载多个，一般是0 |
| 9 | PACKH | 数据段编号，0-255  如果固件大于256K用DownNo=1表示第2段256K |
| 10 | PACKL | 数据片编号， 0-21，共22片，1024字节 |
| 11-58 | PACKDATA | 48字节的数据，最后1包只有16字节，1次可广播1024字节 |
| 59-60 | CRC16 |  |

备注：

1. 投票器先判断PACKH是否发生变化，发生变化就表示新的16片数据下载开始了，要把标志24片段下载成功状态的OKBITS全置1；
2. 然后计算地址，把PACKDATA的48字节数据写到指定位置，地址=PACKH\*1024+PACKL\*48；
3. 然后把OKBITS中对应PACKL的比特位置0表示已经对应片段下载成功，用于应答下面的下载状态询问指令

#### 2.6.3 查询下载状态

询问下载成功状态指令结构：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | DOWNCMD | 0x41 下载多包类指令 |
| 2 | DOWNCMD | 4 询问接收状态 |
| 3 | DATAPOS | 下载的DataPos |
| 4-5 | KEYID | 表决器编号，2字节，高位在前  FFFF，是用序列号指定键盘 |
| 6-9 | SN4 | SN号，4字节 |
| 10 | DOWNTYPE | 多包类型 |
| 11 | DOWNNO |  |
| 12 | PACKH | 数据段编号，是前面下载数据时候的PACKH |
|  |  |  |
| 59-60 | CRC16 |  |

表决器回应下载成功状态：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | DOWNCMD | 0xC1 下载多包类指令应答 |
| 2 | DOWNCMD | 4 下载状态应答 |
| 3 | DATAPOS | 下载的DataPos |
| 4-5 | KEYID | 表决器编号，2字节，高位在前  FFFF，是用序列号指定键盘 |
| 6-9 | SN4 | SN号，4字节 |
| 10 | DOWNTYPE | 多包类型 |
| 11 | DOWNID | 数据包标识码 |
| 12 | PACKH | 数据段编号，对应询问的PACKH |
| 13-15 | OKBITS\_L  OKBITS\_M  OKBITS\_H | 下载成功状态，共24Bit，代表0-22号数据片的接收状态，Bit=0表示下载成功  注意是低位字节在前，例如OKBITS\_L=3，第1、2数据片段没接收成功 |
|  |  |  |
| 59-60 | CRC16 |  |

#### 2.6.4 空包

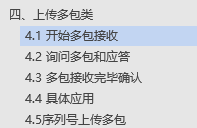
用于等待SDK的下一段数据：不用应答！！

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | DOWNCMD | 0x41 下载多包类指令 |
| 2 | DOWNCMD | 5 等待 |
| 3 | DATAPOS | 下载的DataPos |

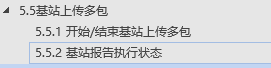
### 2.7多包上传

一期项目不用实现，也暂不优化。

键盘协议用键盘协议第四章，SDK也沿用老流程。



二期实现基站取多包，键盘和基站之间的协议，还是上面协议，但基站和SDK之间，使用《通讯协议-教育新版-基站部分-V0.94》里面的基站取多包协议：



由于基站取键盘数据太快了，2.5ms可以取6段，USB透传实测SDK会丢包，所以基站内部转成2.9节的多包数据传输格式，当成投票结果传输处理。

### 2.8手写数据传输

手写数据和语音数据查询的区别是，手写不安排重传周期（基站自动取消），查询指令里面告诉键盘上次哪些包没收到，bit位表示，和语音一样。但键盘应答数据时候，先送上次没收到的笔迹片段，再送新笔迹数据片段，最多~~7~~8片。

（V0.7创始，可靠传输，但数据可以不传满。V1.1后和语音可靠传输格式同，主要是SLOTN含义修改成相同）

指令格式：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | GETCMD | 0x11 指定SN号查询数据，TDMA体系 |
| 2 | SEND\_SEQ | 发送序列号，1起， 1到7  每个时隙内查询包会发送3次以提高键盘收到的概率，为了时间同步，必须让键盘知道自己接收的是第几次，然后才能进行准确的延时。 所有键盘时间的基准就是数据查询和语音传帧。收到一帧后就开始进行计时。 |
| 3 | ~~TYPE~~ | ~~01 表示是手写（语音是00）~~（V1.1后和语音可靠传输模式同） |
| 4-8 |  | 暂填00，空 |
| 9-12 | SN1 | 第1个时序的键盘的SN号，4字节，高位前，全00表示空  SN号是这个SN1的键盘，在第1个时序做应答 |
| 13 | SN1-CMD1 | SN1的键盘的命令码  用bit表示，对应bit=1，表示刚才的笔迹数据包的第几片需要重新发  例如值=3时候，表示第1、第2包都要重传  实际实现最多~~7~~8片，基站自动预置0x7F  键盘端加个处理，比如，如果上次没传第5片，即使基站说没收到第5片，也不需要重传 |
| 14-18 | SN2+CMD2 | 5字节，第2个键盘的SN号和控制命令 |
| 。。。 | 。。。 |  |
| 。。。58 | 。。。CMD10 | 一直到第10个键盘的SN号和控制命令  第58字节是最后1个数据 |
| 59-60 | CRC | 空中数据最后2字节是CRC |

键盘在对应时序，可以提交单包数据，也可以加上笔迹数据提交，不超出时序时间即可。

如果提交单包数据，按3.4章节描述。

**提交笔迹数据格式**如下：（48字节的有效笔迹数据）

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | ANSCMD | ~~0x97~~ 0x99笔迹数据包，TDMA体系 |
| 2-5 | SN | 键盘的SN号，4字节，高位前 |
| 6-7 | ID | 键盘编号ID |
| 8 | SLOTN | （V1.1和2.1语音可靠传输格式同）  高4位=0，~~笔迹数据的小格式，原先是~~本周期总片数，0等于1片  低4位是当前片号，实际是~~1-7，共7片~~0-7  基站可以根据这个判断本周期内哪帧的语音没有收齐需要重传  （键盘没有数据提交，也要求发1包用来表示心跳，SLOTN=0xFF）  SLOTN位置和语音不同，是为了好查看数据，SDK可以只处理PACKNO后的数据，SDK实际不需要处理SLOTN的值 |
| 9-10 | PACKNO | 后面VOCDATA的数据包的序号，依次增加，高位前，1起（0作为停止，可在提交ok键时候表示），理论上可以传3042K字节  PACKNO是发送的时候增加的，重传的时候不要变  比如这次应答只有3个PENDATA包，那么PACKNO在上次PACKNO是依次加1，本次应答只增加3 |
| 11-58 | PENDATA | 笔迹坐标数据，48字节，FF表示空  笔迹数据暂定3字节1个小段，X、Y和状态（含力度），最多16个坐标数据，如果以后XY坐标是双字节，可以带9个坐标数据  键盘提交数据先先全部填FF，SDK碰到FF数据了，就表示后面数据是无效的 |
| 59-60 | CRC |  |

键盘可以参考如下流程，实现笔迹数据的缓存，以及传输和重传：

1、笔迹数据单独1个buf或者队列保存，基本是3字节一个笔迹数据，最好做了去重；

2、另外开一个传输buf，7个数组，对应7个SLOTN片段，每个数组50字节，2字节PACKNO号码和48字节笔迹数据；

3、新的笔迹数据查询指令到，（对重传的处理，放下一步骤，这里只举例初始），从笔迹buf中取数据填到传输buf，能填多少片填多少，最多7片，新填了数据的片，给一个新的PACKNO号（每次+1），注意新填数据的片段，笔迹数据先填满0xFF；然后填到传输buf的笔迹数据可以释放空间了；时间到，执行无线发送；

4、基站下次笔迹数据查询指令到，如果有说哪1片没收到，例如第3和5片没收到，那么把传输buf中第3片和第5片的数据，挪到第1和第2片，注意里面的PACKNO不要变，然后后面5片从笔迹buf中取新数据填；然后发送，自然上次的3、5片再次重发

### 2.9多包上传数据

基站取多包时候，老协议键盘应答0xA0的数据，转成**提交键盘多包数据格式**如下：（16字节或48字节的键盘多包数据）

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | ANSCMD | ~~0x98~~ 0x9A多包上传数据包，TDMA体系 |
| 2-5 | SN | 键盘的SN号，4字节，高位前 |
| 6-7 | ID | 键盘编号ID |
| 8 | PACKMODE | 高4位是对应键盘的PACKTYPE，低4位是PACKID |
| 9-10 | PACKNO | 后面PACKDATA的数据包的序号，1起（按笔迹数据），高位前  如果最高位是1，表示后面数据48字节，否则16字节有效 |
| 11-58 | PACKDATA | 多包数据，最大48字节，一般16字节 |
| 59-60 | CRC |  |

### 2.10图片数据传输（60路同时）

**图片数据或其他可以作为文件的数据**，和语音可靠传输方式一样，数据是不能丢和错漏的。有2种传输方式，一种是60路同时传输，就是本小节讨论的，另外一种方式是单独1个键盘连续传输，走2.11节协议。

由于方法和2.1-A节语音可靠传输方式一样，这里定义图片数据类型是0x9B即可。考虑到文件可能需要准确的长度，最后1包要指示有效数据长度，所以最后1包的键盘编号ID位置用于放最后1包实际长度，参见2.1-A节。

传图片的时候，基站投票信标的投票模式是17。

### 2.11图片数据传输（单独1路）

如果只指定1个或多个键盘提交图片，60路同时传输慢了。本方式用于全部带宽给1个或几个键盘。

基站和键盘的交互流程如下：

1、基站发确认包、投票信标包、和2.1节的点名包2次，和语音模式是一样的，投票信标让键盘知道当前是单路拍照模式，确认包是用于键盘提交上传图片申请后的确认，点名是让键盘按时序应答，可提交图片上传申请或者0x97心跳包，重点是键盘不要直接送图片数据，而是先发传图片申请

2、基站收到键盘上传申请后，会通知SDK但SDK只记录，然后自动转单路图片传输流程，也就是SDK不用管收取

3、基站指定SN询问键盘数据，并告诉上次哪些包没有收到，键盘把上次没收到的包重新发送1次，如果还有时间（例如可发80包但没满），后面可以发新的数据

4、基站重复步骤3直到键盘传输完最后1包，然后基站继续询问下1个SN或者回到步骤1；

5、步骤3，基站发送4包，确认包、投票信标包，和2次0x14点名包

6、图片上传申请是图片数据的一种特殊格式，键盘如果后继收到0x14点名包，表示上传申请提交ok，自己清除状态

基站指定SN号查询数据，并指定包重传。指令格式：

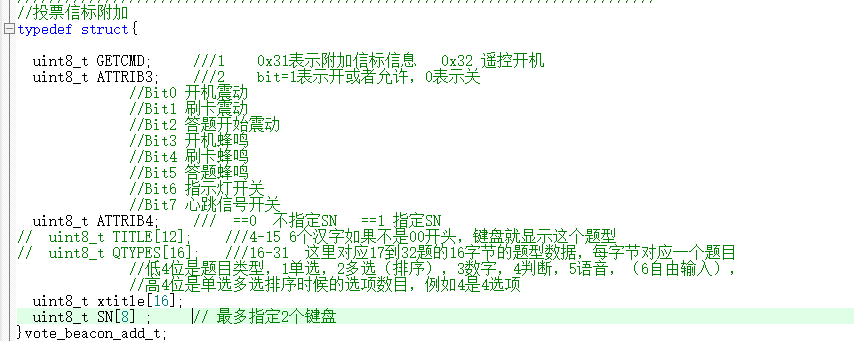
|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | GETCMD | 0x14 指定SN号查询多包数据  （0x12用做无线升级广播了，0x13是老竞争信标） |
| 2 | MODE | 暂0 |
| 3-6 | SN | 要传数据的键盘的SN号，4字节，高位前 |
| 7 | TXSLOT | 给多少个时间片发送数据（5.8G每个时间片发8包）  暂最大10 |
| 8-17 | RESEND\_BITS | 上次80包数据，那些包要重传，对应bit=1表示要重传  共10字节，第1字节代表前8包，第2字节代表第8-16包，依次类推  每1字节8个bit，分别对应80包的1包，第1字节的bit0代表第1包，第2字节的bit0代表第9包 |
| 。。。 |  |  |
| 59-60 | CRC | 空中数据最后2字节是CRC |

键盘应答数据格式如下：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | ANSCMD | 0x9C图片多包上传格式，1次传输>8的多包 |
| 2-5 | SN | 键盘的SN号，4字节，高位前 |
| 6 | STATUS | V1.22 高位作为图片变化标志，这样SDK好判断是新的图片文件  低7位暂时空 |
| 7 | SLOTMAX | 本次连续发送多少包数据，1起（SDK不用处理）  如果是0，表示本包是最后1包，~~同时，SLOTN值是本包有效数据长度，因为最后1包不一定是48字节（SDK要处理）~~---键盘编程上先验证前面所有的包基站都可靠收到后，才发最后1包，这样基站好退出收包流程，--但键盘端没有收到确认不好退出。（取消长度是整个文件前面加头，包含了长度，SDK可以处理。）  如果是FF，是图片上传申请  ~~如果是FE，代表键盘要提前终止发送~~（实现是键盘PACKNO从0开始） |
| 8 | SLOTN | 后面数据是本次发送第多少包数据，1起（SDK不用处理） |
| 9-10 | PACKNO | 后面VOCDATA的数据包的序号，48字节1个序号，依次增加，高位在前，1起，理论上可以传65535x48=3072K字节， 每段数据的PACKNO是固定的，重传的时候也不会变，  SDK按位置=PackNox48保存即可组成完整文件  ~~V1.21 PACKNO用低15位，高位作为图片变化标志（字节9的高位），这样SDK好判断是新的图片文件~~（但满足不了原图传输可能>1.5MB） |
| 11-58 | VOCDATA | 片段数据，48字节 |
| 59-60 | CRC |  |

### 2.12 自定义标题和指定SN作答

是投票信标支持64字节，第33字节开始当扩展信标用



### 2.13 遥控开机

### 2.14 换信道重传

做T2Soc项目中，键盘端有时候通讯不畅（可能持续2、3分钟），导致某一两个键盘，测试90秒时候丢包不达标。这个通讯不畅，和位置有一点关系，本方案是尝试通讯不畅的键盘，自动换到另外的一个专用通讯模块进行通讯。底层逻辑一是专门的通讯信道来补救，提高成功的概率，第二是期望通过接收天线位置的变化和频点的变化，通讯效果好一点。但如果键盘还是通讯不畅，可能还是会失败。

这里面要处理一个问题，重传信道，怎么知道键盘需要单独重传（进入），和什么时候退出。键盘端判断倒是简单，根据buf是否超限即可。一种是原基站通讯模块和重传模块之间通过spi通讯相互告知，但准确退出最好是有确定的ack过程，如果只是判断数据包里面的标志，重传信道还不一定能准确收到buf不告警标志。另外一种方法，键盘和重传信道自主处理。最终考虑方法是，每一个重传周期（时长一般按语音大周期）前加1个带确认的竞争周期，键盘竞争提交申请重传和退出重传申请，然后剩余的时间，跑1对1的8包重传，也可以2.1.4节的1对多重传。

所以键盘自己判断buf不足了，就切换到重传信道频点（例如副模块4），提交重传申请，模块收到在确认包中确认，并在重传周期中对键盘重传点名。如果键盘没收到确认，下次继续申请。如果键盘认为不需要单独重传了，申请退出，并被确认后，回原先信道继续通讯，如果没收到确认，下次继续申请。

如果实测可行，可考虑不另开信道，而利用原来3信道，模块1的在模块2上重传，模块2的在模块3上重传，节约信道资源。但4信道方案可能容易实现单基站90个键盘甚至100个键盘（3路各30个，重传信道再带10个）。

#### 2.14.1 重传申请信标和确认

入网确认信标指令格式：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | GETCMD | 0x55 重传指令 |
| 2 | TYPE | 1 是信标，2是确认包 |
| 3--8 |  | 空 |
| 9-12 | SN1 | 4字节SN， 第1个键盘的SN |
| 13 | CMD1 | 1 收到 |
| 14-18 | SN2-CMD2 | 第2个键盘的SN和确认  每个键盘5字节，1次可确认10个键盘 |
|  | 。。。 | 第3个到第10个SN和确认 |
|  |  |  |

#### 2.14.2键盘申请

为节省指令资源，暂用入网确认信标指令格式，TYPE值不一样：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | GETCMD | 0x96 入网申请 |
| 2 | TYPE | 2 申请重传 3退出重传 |
| 3-6 | SN1-SN4 | 4字节SN |
|  |  | 原始绑定模块编号，1-4 |
|  |  |  |

### 2.15 跳频

支持跳频模式是否启用。

2.2.3节基站连接信标加跳频标志，新入网键盘能快速跳到主模块频点入网。

2.1.0节语音点名指令加跳频标志。投票信标等暂时不用加。

基站跳频主流程比较简单，每个大周期前先换频点，再发送信标。

键盘收到0x11点名，就启动跳频定时器（相当于同步定时器了），定时换频点。这个定时器是多次触发的，周期比跳频周期短一点（例如2ms），即使下个周期干扰原因没收到0x11指令，键盘也能换频点，而且是提前一点的。

如果0x11点名中自己是第10个应答，为避免出现应答发送中变频，关定时器，设置发送后换频标志，发送后换频，并开跳频定时器。

键盘一段时间后，没收到自己模块的0x11指令，同步变失步，重新锁频。

锁频可以到基站连接频点听主模块现在频点，但录音时候可能不发组网，同时组网周期比较长，所以不用这种方法。

用慢周期换频等待的方法。换频周期用基站换频循环周期（比如T2soc=4x70ms）时候，理论上一个慢周期，总能收到模块的信号。收到时候，老同步代码起效果，相当于同步上了。如果是多模块系统，理论70ms就能收到1个模块的0x11指令，然后根据里面的模块编号，快速推断出自己的模块频点，然后马上切，避免换频周期慢错过，但跳频周期还是不改，用慢的。

如果是单模块跳频，换频周期改2个跳频周期，例如140ms，先跳到预估频点的下一个频点等待。后面的频点变化用反向可提高搜索速度？现在代码没采用，因为频点少差别不大。

如何和1秒收不到重置无线的操作兼容？可暂时取消。

入网组网信标和单独点名信标，加内部第几轮小点名编号，用于键盘计算下个跳频周期。比如单独点名，不加的话，键盘按64ms换频时候正好在发送数据，换频失效。

## 三、基站和SDK

总体上继续沿用SDK 1问10答1确认的方式，也就是SDK查询投票和语音数据时候，先询问，然后基站最多连续发10个应答，然后SDK发送确认包告诉刚才收到哪些包。

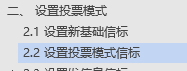
基站使用高速USB，最快可以1ms传1024字节，所以基站传数据要有效利用这1024字节，而不是老系统的64字节。

在底层数据上，对于不大于255的数据包，还是沿用F5 AA AA开头，然后1字节长度LEN，然后是有效数据包和CRC的包结构。

对于大于255的数据，沿用T1语音数据包格式，改F5 AA AB开头，然后2字节长度LEN，高位前，然后再是数据和CRC。

### 3.1启动投票

暂按基站协议2.2节和2.1节。



### 3.2读取结果

沿用SDK 1问10答1确认的方式，也就是SDK查询投票和语音数据时候，先询问，然后基站最多连续发10个应答，然后SDK发送确认包告诉刚才收到哪些包。

#### 3.2.1 基站询问结果

指令和老体系一样，但也可以改变长度和CRC位置：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | BASECMD | 0x60 基站类指令 |
| 2 | BASEID | 指定的基站编号，一般是01 |
| 3 | CMDTYPE | 基站命令类型  5 读取基站结果 |
| 4-29 |  | 无意义 |

#### 3.2.2 基站应答

基站应答最多10个数据包，每个包带编号MsgNo，最终应答包用到。

这里把引导头和长度字节也列出来：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
|  | **F5 AA AB** | 固定引导头，注意是F5 AA AB |
|  | **LEN** | 2字节长度，后面数据的字节数，高位前  长度根据实际结果包的个数计算 |
| 1 | BASECMD | 0xE0 基站类指令应答 |
| 2 | BASEID | 应答的基站的编号 |
| 3 | CMDTYPE | 应答类型  5 基站结果上传 |
| 4 | MSGNO | 数据包序号  基站最多依次输出10个数据包，第1个数据包MSGNO=1，第2个为2，依次类推，如果不足10个数据包，MSGNO=0xFF表示没有数据了（或者完全没数据，第一包MsgNo就FF） |
| 5 | MSGTYPE | 基站数据包类型  30 T2语音体系结果提交（统一60字节，语音数据是60字节）  （21-28是教育体系模式已经占用） |
| 6 | MSGNUM | 有效结果数目，表示后面有多少个60字节的键盘数据可以分解  理论上根据长度也可推算 |
| 7 |  | 空1字节，00 |
| 8-67 | MSGDATA1 | 第1个键盘数据，60字节  这60字节包含了键盘提交上来的原始数据，从ANSCMD开始  第1个字节是0x94的话，就按普通投票数据解析  第1个字节是0x95的话，就按T2体系语音包解析，格式见2.1节 |
| 68- | MSGDATA2 | 第2个键盘数据，60字节 |
|  | 。。。。 | 第3个键盘数据。。。 |
|  | MSGDATA16 | 最多可以到第16个键盘数据 |
|  | CRC |  |

#### 3.2.3 SDK确认

SDK给基站刚才发过来的结果包，收到哪些MsgNo的包，这样，基站可以丢弃SDK已经收到的数据，没收到的下一次询问再发。

指令和老体系一样，但也可以改变长度和CRC位置：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | BASECMD | 0x60 基站类指令 |
| 2 | BASEID | 指定的基站编号 |
| 3 | CMDTYPE | 基站命令类型  6 确认基站结果 |
| 4 | MSGNO1 | 第1个成功接收数据包的MSGNO |
| 。。。 |  | 第2个到第9个成功接收数据包的MSGNO，如果没有，对应字节填0xFF |
| 13 | MSGNO10 | 第10个成功接收数据包的MSGNO，如果没有，对应字节填0xFF |
| 14-29 |  | 无意义 |

### 3.3基站设置

暂沿用基站协议5.2章节，主要是频点、配对码、基站SN号读写。



### 3.4单包下载

参考2.5节。

### 3.5多包下载

参考2.6节。

### 3.6多包上传

参考2.7节。

### 3.7基站调试信息

以往基站或键盘的调试信息，只能串口、RTT实时打印，需要连接线和特殊工具才能获取，这里提供一种直接通过USB上报的方法，就是把调试信息当一种类型的投票结果传给SDK，SDK直接解码显示到log显示界面，这样，调试很方便。参见3.2.2节基站的应答数据包，1包共60字节，平常传32字符信息没问题。

调试信息指令格式：

|  |  |  |
| --- | --- | --- |
| **字节** | **标识符** | **描述** |
| 1 | GETCMD | 0x9F Log文本信息 |
| 2 | TYPE | 暂1，表示格式1 |
| 3 | SOURCE\_ID | 来源，0-4，比如0是主控，1是模块1,2是模块2，最多模块4  SDK那边可以分开不同的tab页显示 |
| 4-5 | TIME | 时间，2字节，高位前，单位20ms，一般是投票启动时为0  SDK转换成分、秒、毫秒显示 |
| 6-- | LogText | 开始是Log文本信息，SDK按文本显示 |
|  |  |  |
| 59-60 | CRC | 暂时没意义 |

## 【版本信息】

V0.5 概述，2.1-2.4章节，投票和语音

V0.6 修改语音数据包格式做小调整，补充2.6-2.7节，都按老体系流程和协议，暂不做优化

V0.7 增加2.8节手写数据查询、手写数据格式；2.2增加组网应答格式，含白名单申请；2.2节增加白名单说明

V0.8 2.7节多包上传修改成基站取多包上传，而且键盘多包数据转成0x98类结果，2.9节

V0.9 戴雨语音改成60路，时序针对语音做了调整，语音发8包，然后语音数据包可以发52字节，原先48字节，取消了ID和CRC

V1.0 增加了2.1-节语音数据可靠传输模式，根据T2-60代码和基站代码，完善了V0.9版本时序的细节描述

V1.1 增加2.10节图片格式描述，并修改2.1-A节支持最后1包有效数据长度可变；2.9节多包上传改数据类型；完善原来没记录的自定义标题、指定键盘作答

V1.2 支持指定单个键盘上传图片数据，增加2.11节单路多包上传方式

V1.21 语音包packno加上新录音变化标志，解决两个录音间隔很短的时候，心跳没传上，sdk不好判断是新录音开始

V1.3 针对键盘语音buf告警，设计允许键盘多发几个时间片

V2.0 T2Soc优化

1、2.6节多包下载，优化成1包48字节，同时一次广播24包

2、组网模式修改，支持多个频点，可负载均衡

3、增加调试信息结果包，3.7节

4、增加语音压缩格式、录音丢包率的实现，2.1.3节

5、优化2.1.0节点名查询指令，可指定动态数量的键盘点名

V2.2 跳频支持

1、2.2.3基站连接信标增加跳频属性

2、2.1.0点名查询增加跳频属性

3、增加2.15节跳频说明

V2.3 补充T2李傲寒语音传输完善对应的协议修改

1、增加2.1.5节录音停止包说明

2、由于录音停止SlotN用了0xF1，和buf告警高位1冲突了，修改buf告警到bit4，相当于0x08用于告警，原先0x80

3、语音0x98数据包ID0位置做了实时丢包率，ID1位置变成了录音编号，录音1次就递增

4、2.1.0语音点名指令增加关机时间设置