Neodun 驱动与签名机通信协议

20170730：

上位机驱动 driver简称D

单片机签名机 signer简称S

之前的简称c s 比较临时，废弃不用

D2S 表示上位机发往单片机的报文

S2D 表示单片机发往上位机的报文

## 报文通用设计

报文长度 64字节

后文说第几字节，一律从零开始计数

第0 1字节 用作报文识别信息，byte1 byte2

第2 3字节 用作响应识别串号， 是一个小头在前的 uint16

第 62 63 字节 用作报文验证位，是对报文0~62字节做crc32运算之后的整数，以小头在前存储，去前两个字节。

第 4~61 共58个字节用作报文数据区，由每条报文独自约定

关于丢包，上位机向单片机发送，丢包是异常，可以捕获。

## 数据传输报文

数据传输采取被动式设计，发送方发起传输指令，接收方自己根据需要获取，至获取完毕后，接收方发出确认指令。

数据传输类报文 第一字节都是01

异常通知和收到确认这样的报文，第一字节都是 00

协议响应数据从 0x01a0开始

协议错误响应数据从 0x01e0开始

标准错误响应从 0x0001开始

1.D2S 0x0101 向签名机发送数据包

第2 第 3 字节，串号取随机数（除了是针对特定指令的确认和异常通知是取相等串号，其余全是随机数，后文不再赘述）。

第4 -7 字节 一个 小头在前的 uint32，表示传输数据大小

第8~39 字节，32个字节的 hash256 数据，表示传输数据的特征码

2.S2D 回复，串号都是 D2S0x0101的串号

S2D 0x0001 //报文出错，报文最后两个字节 和 报文内容验证不匹配

无数据，表示报文数据区全0，后文不再赘述

S2D 0x01e0 //协议出错，数据容量太大，无法缓存

无数据

S2D 0x01e1 //数据接收完毕，但是hash256 不匹配

无数据

S2D 0x01a0 //数据接收完毕，并且hash256 匹配

第4~7字节是一个uint32，表示对该数据的分配编号。

传输完成之后，D2S 可以再发送其他控制指令，指定用制定编号的缓冲区数据进行操作。单片机上最多同时保持四个缓冲区数据（暂定）。

3.S2D请求数据

S2D 0x0110 请求数据，将原数据按照 50 byte 分片，最后一片不足50byte 补0

第4~5 字节是一个uint16，表示请求数据的开始片数

第6~7 字节是一个uint16，表示请求数据的结束片数

第8~39字节，hash256 数据，请求内容的特征码

4.D2S响应请求

D2S 0x0102 （回复协议）

2 3字节的串号是 S2D 0x0110的串号

4~5字节是一个uint16，表示发送哪一片的数据

6~55字节是数据

D2S 0x0103（回复协议）

2 3字节的串号是 S2D 0x0110的串号

请求的数据全部发送完毕

5.S2D数据确认

收到0x0103后，signer判断数据是否接收完整，不完整继续发送数据请求

完整了做判断后回复 0x01a0 或者0x01e1

数据确认的是报文D2S 0x0101，所以用D2S 0x0101 的串号

只有一条协议是另一条协议的回复协议，才用他的串号。

单片机向上位机发送数据

上述协议S2D 变成D2S，D2S变成S2D 即可