SSP 通讯协议 V1.2

历史版本:

2010.03 V1.1

2011.12. V1.2

Smiley® Secure Protocol - SSP 是ITL开发的专用串口协议。采用的是主从模式，控制板作为主机，其他的外围设备如纸币器、硬币找零机等作为从机。

数据传输采用16位CRC效验(详细算法见附件说明)。每一个SSP的设备都有单独的产品序列码。 SSP最大的优点是支持远程升级。

特征：

串口控制
4线系统(Tx， Rx， Vcc， Gnd)
准RS232-开集驱动
16位CRC效验

1. SSP 硬件协议层:

SSP 建立在普通的串行通讯(UART)方式之上， 使用全双工三线通讯: 发送(TXD)、接收(RXD)、公共地(Gnd)。适用于任何有串行通讯(UART)方式的计算机、单片机、DSP、ARM 等系统中。

SSP 通讯参数:

波特率： 9600bps
数据位： 8Bits
起始位： 1Bit
停止位： **2Bits**
校验位： 无

特别说明:

1、**常规的串行通讯多数使用 1 个停止位，而 SSP 则使用 2 个停止位，因此，请特别注意。**

2、SSP硬件设置在串行通讯中使用 TTL 电平，如果是与普通的 MCU 可以直接连接；如果是和计算机、工控系统或类似计算机使用 232 电平通讯的，必须增加电平转换设备，否则，可能会造成硬件损坏。

TTL 电平: 高电平为MCU 的 Vcc 电平，如: 5V、3.3V 或 3V，具体要看用户所用的 MCU 决定；低电平为 0V;
232 电平: 高电平理论值为 -12V，低电平为 +12V

2. NV9、NV10、NV200 硬件接口(2*8P 结构，见图1.):

Pin1 ---- +12V

Pin2 ---- Gnd

11 -------- Rxd (方向: Host → 识币器)

15 ------ Txd (方向: 识币器 → Host)

电源说明: 电压 12V， 电流: 1.5A

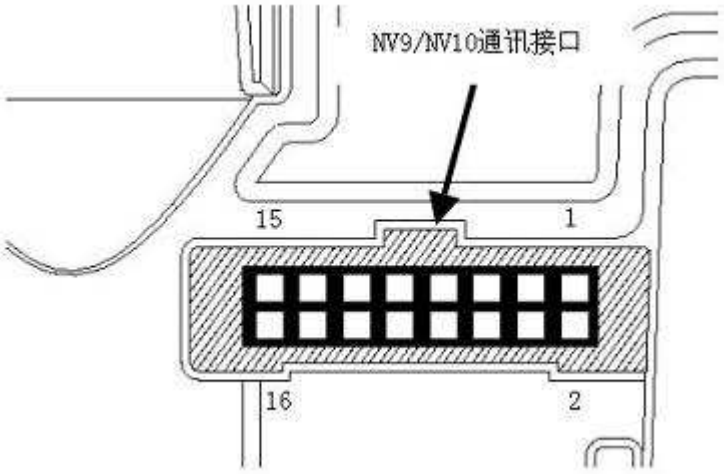

图 1. NV9、NV10、NV200 硬件接口

3. SSP 传输协议:

3.1. 通讯约定:

为说明方便，先作如下约定:

识币器一方用 slave 替代说明;

计算机或单片机一方用 host 替代说明;

3.2. 数据包格式(Packet Format):

slave 与 Host 通讯时使用下列数据通讯格式:

STX	SEQ	LENGTH	DATA	CRCL	CRCH

STX: 起始字节，为十六进制：0x7F.

SEQ: 标志位. 如果是NV9、NV10、NV200使用**0x80**和**0x00**交替使用. Hopper 使用**0x90**和**0x10**交替使用.

LENGTH: 包长度，长度不包含 STX、SEQ、CRCL，CRCH

DATA: 命令或数据

CRCL，CRCH: CRC-16 校验 (详细见附件算法)， 需要校验的数据从 SEQ 开始直到 Data 为止.

数据包特别说明:

如果发送到识币器的数据包中包含 0x7f (起始字节 STX 除外)，必须使用 0x7f，0x7f 替代发送，注意将 0x7f替代成 0x7f，0x7f 只发生在发送过程，也就是必须在 CRC 校验完成之后，参加CRC校验的数据只能是单个0x7f。

如果接收到的数据包中除起始字节外包含有 0x7f，0x7f，必须先将 0x7f，0x7f 替换成0x7f后再进行CRC 校验。

ssp 数据包示例: 7F 00 01 07 11 88

这是 Host 发到 slave 的 Poll 命令，相关解析如下:

7F ---- STX

00 ---- SEQ ，下一次发送时将使用 0x80

01 ---- Length，因只包含 1 个字节的数据0x07(即 Poll 命令)

07 ---- Poll 命令

11 ---- CrcL (Crc 计算时是从 0x00 到 0x07 之间的数据进行的)

88 ---- CrcH

4. NV9、NV10、NV200 通讯方式：

NV9、NV10、NV200 通讯为“**被动方式**”，也就是说只有 Host 主机向识币器发送符合要求的数据后识币器才会进行相关应答。注意 2 次发送间隔最好是 200 毫秒，如果需要识币器一直工作，就需要不停的向识币器发送 Poll 命令，一旦停止发送，识币器就停止工作。

5. SSP 命令列表：

5.1 Host 发向 Slave 的常用命令列表：

功能	命令代码 (HEX)	字节数	相关解释
Reset	0x01	1Byte	使识币器复位.
Set inhibits	**0x02, data1, data2**	3Bytess	设置允许识别哪几种纸币
Display On	0x03	1 Byte	打开纸币器面板显示灯
Display Off	0x04	1 Byte	关闭纸币器面板显示灯
Set-up Request	**0x05**	1 Byte	读取识币器版本及通道与纸币对应关系等
Host Protocol Version	0x06, VerNo	2Bytes	主机协议版本，第一字节是指令，第二字节是协议版本
Poll	**0x07**	1 Byte	Poll 命令
Reject	0x08	1 Byte	拒绝接收当前的纸币
Disable	**0x09**	1 Byte	关闭识币器命令
Enable	**0x0a**	1 Byte	开放识币器命令
Program Firmware/currency	0x0b	1 Byte	用于远程升级
Get Serial Number	0x0c	1 Byte	读取识币器系列号
Unit data	0x0d	1 Byte	读取识币器软件版本
Channel Value Data	0x0e	1 Byte	通道数
Channel Security data	0x0f	1 Byte	
Channel Re-teach data	0x10	1 Byte	
Synchronisation	**0x11**	1 Byte	同步命令
Update Coin Route	0x12	1 Byte	
Last Reject Code	0x17	1 Byte	退币原因
Hold	0x18	1 Byte	保持纸币
Enable Protocol Version Events	**0x19**	**1 Byte**	**必须在发送同步之后发此命令，才能在取走钱箱后返回正确的命令。 2010/06/12**
Get Bar Code Reader	0x23	1 Byte	
Configuration Set Bar Code Reader	0x24	4 Bytes	
Configuration Get Bar Code Inhibit	0x25	1 Byte	
Set Bar Code Inhibit	0x26	2 Bytes	
Get Bar Code Data	0x27	1 Byte	
Manufactures Extension	0x30, Cmd, Data	3 Bytes	

同步命令：将 SEQ/Slave ID 重设为 0x80(NV9、NV0、NV200) 或 0x90 (Hopper)

Unit Data Request: 纸币器型号:一位字节

Firmware 版本: 4个字节的 ASCII 字串

国家代码: 3个字节的 ASCII 字串(详细见附录)

货币扩展位数: 3 个字节的整型数

协议版本: 1 个字节的整型数 (Page16)

Channel Value Request: 货币通道值，单字节指令，返回一个通道号.
举例：纸币器有通道1，2，4，6，7.
纸币器返回 07，01，02，00，04，00，06，07.
实际的货币价值是用通道数乘以通道倍数。

5.2 Host 接收来至 Slave 的常用命令列表：

功能	命令代码 (HEX)	字节数	相关解释
Stacking	**0xcc**	1Byte	压币
Bar Code Ticket Acknowledge	0xd1	1Byte	
Note cleared from front at reset	0xe1, ChNo	2Bytes	
Note cleared into cash box at reset	0xe2, ChNo	2Bytes	
Cash Box Removed	**0xe3**	1Byte	钱箱被取走
Cash Box Replaced	**0xe4**	1Byte	钱箱已放回
Bar Code Ticket Validated	0xe5, route No	2Bytes	
Fraud Attempt, n	0xe6	2Bytes	识币器发现有欺骗行为（钓鱼）
Stacker Full	**0xe7**	1Byte	钱箱满，需要将纸币取走
Disabled	**0xe8**	1Byte	识币器处理关闭状态，需要向识币器发送 Enabile 命令
Unsafe Jam	0xe9	1Byte	识币器非安全卡币
Safe Jam	0xea	1Byte	识币器安全卡币
Stacked	0xeb	1Byte	堆叠纸币结束
Rejected	**0xec**	1Byte	拒绝接收纸币执行完毕
Rejecting	**0xed**	1Byte	拒绝接收纸币
Credit, n	0xee, ChNo	2Bytes	确认纸币所在通道号（纸币被识别）建议使用该指令作为纸币器有效收币讯号。
Read, n	**0xef, ChNo**	2Bytes	读取纸币所在通道号
OK	**0xf0**	1Byte	识币器已执行完毕
Slave Reset	0xf1	1Byte	识币器正在复位
Command not known	0xf2	1Byte	如果有无效指令发给纸币器是，纸币器返回该指令。
Wrong number of	0xf3	1Byte	参数错误

parameters			
Command cannot be processed	0xf5	1Byte	指令无法执行
Software Error	0xf6	1Byte	
Checksum Error	0xf7	1BYte	
FAIL	0xf8	1Byte	
Key Not Set	0xfa	1Byte	

6. 识币器常规的工作流程:

6.1. 初始化串口:

根据用户所用的系统设置通讯参数并打开通讯口。OpenSSPComPort()

通讯参数如下:

通讯波特率: 9600 bps
数据位: 8Bits
停止位: 2Bits　　// 注意: 常规的通讯是使用 1Bit 的停止位使用， SSP必须使用2个停止位
校验位: 无

6.2. 发送 0x11 号命令查找识币器是否连接:

发送一个同步命令到识币器， 如果串口上接有识币器且连接正常，应该会应答 OK 命令，发送同步时，置 SEQ 为 0x80，以后的发送依次为 0x00，0x80， 0x00，0x80 **交替**进行。

示例: W: 7F 80 01 11 65 82　　// 发送 Synchronisation 命令
R: 7F 80 01 F0 23 80　　// 识币器应答 OK

6.3. 发送 0x05 号命令读取识币器通道配置情况:

发送 Request 命令，了解识币器各通道配置情况，比如配置成是哪个国家的纸币系统，有多少个通道，每个通道对应的纸币面额是多少等。示例如下:

W: 7F 00 01 05 1E 08　　// 发送 Request 命令
R: 7F 00 1F F0 00 30 33 31 39 43 4E 59　　// 识币器应答包
00 00 01 **07 01 02 05 0A 14 32 64** 02
02 02 02 02 02 02 00 00 00 03 54 52

数据包解析:

7F ---- STX
00 ---- SEQ
1F ---- 数据长度， 共 31 个字节
F0 ---- OK
00 ---- 数据包类型(Unit Type)
30 33 31 39 ---- 控制软件版本(Firmware Version) “0319”
43 4E 59 ------- 国家代码(Country Code),“CNY”
00 00 01 ------- 乘数变量(Value Multiplier)，将决定通道对应的纸币面额，此为 *1
07 ---- 识币器所使用的最大通道数(Highest used channel)， 示例中最大 7 个通道，即能识别 7 种纸币
01 02 05 0A 14 32 64 ---- 每个通道所对应的纸币面额，示例中的通道与纸币面额间的对应关系如下:
通道号: 1 通道，2 通道，3 通道，4 通道，5 通道，6 通道，7 通道
纸币面额: 1 元， 1 元， 5 元， 10 元， 20 元， 50 元， 100 元
02 02 02 02 02 02 02 00 00 00 03 ---- 与我们应用没有太大关系，可以不理

54 ---- CrcL
52 ---- CrcH

注意：1. 识币器识别纸币时并不是直接返回是多少面额的纸币，而是返回被识别的纸币是在对应的通道上，至于是金额就必须知道每个通道与纸币面额的对应关系。因此，执行 Request 命令，读取通道对应的纸币面额非常重要。另外还可以读出通道配置情况， 并与预先要求的通道配置进行比较， 以防错误的使用其他配置的识币器.

2. 此命令只须操作一次就行了，除非再次运行程序；

6.4. 发送 0x02 号命令设置允许识别哪几种纸币：

发送 Set inhibits 命令，配置需要开放/关闭的相应通道。示例如下

```
W: 7F 80 03 02 FF 00 27 A6  // 发送 Set inhibits
R: 7F 80 01 F0 23 80        // 识币器应答 OK
```

此示例开放所有通道，但实际应用可能只开放某些通道，比如只识别 10，20，50，100 元的纸币，其他通道全部关闭。因此，弄清楚如何开放哪个通道非常重要。在 Set inhibits 命令中，02 后面的两个字节就是控制相应的通道，其中第一个字节的最高位 Bit7 必须为 1，其他每个位对应于一个通道，因此，最大可以控制 15 个通道。

示例中只有 7 个通道，因此，只须使用 1 个字节(Byte)的控制位就可以了，即 02 后面这一个，再后面 1 个字节不用，置 0x00， 因此第一个字节与每个通道号及所对应纸币的关系如下：

通道号	7 通道	6 通道	5 通道	4 通道	3 通道	2 通道	1 通道
控制位	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
金额	100 元	50 元	20 元	10 元	5 元	2 元	1 元
值	1	1	1	1	0	0	0

如果需要开放某个通道，就将该通道位置 1，否则，置 0，假设只需要识别 10，20，50，100 元纸币，则 Bit6=1，Bit5=1， Bit4=1， Bit3=1， Bit2=0， Bit1=0， Bit0=0，最高位必须置 1，则得到如下二进制结果：(1111 1000)B ，即十六进制的 0xf8， 因此需要发送的控制串为：02 f8 00

```
W: 7F 80 03 02 F8 00 24 34   // 发送 Set inhibits
R: 7F 80 01 F0 23 80         // 识币器应答 OK
```

特别说明：

1. **发送 Set inhibits 命令时， 必须在识币器处于 Disable 状态下进行， 否则， 可能造成设置的参无效；**
2. **Set inhibits 命令也可能在纸币识别过程中修改。比如限定顾客只能投入 180.00 元纸币操作如下：**
   a. 开始时开放全部通道：02，ff，00 如果顾客插入了 1 张 100 元的纸币后， 则发送 0x09 号命令关闭识币器；
   b. 发送 Set inhibits 命令关闭 100 元纸币：02 bf 00， 再发送 0x0a 号命令及 poll 命令进行纸币识别；如果顾客又插入了 1 张 50 元的纸币；
   c. 这时只准顾客使用 20 元以下的纸币， 则发送 0x09 号命令关闭识币器；
   d. 发送 Set inhibits 命令关闭 100 元，50 元纸币：02 9f 00， 再发送 0x0a 号命令及 poll 命令进行纸币识别；如果顾客又插入了 1 张 20 元的纸币；

e. 这时只准顾客使用 10 元以下的纸币， 则发送 0x09 号命令关闭识币器；

f. 发送 Set inhibits 命令关闭 100 元，50 元， 20 元纸币：02 8f 00， 再发送 0x0a 号命令及 poll 命令进行纸币识别， 等待用户插入纸币；

g. 接着类推， 使用 Set inhibits 命令步步逼近的方式接收指定面额的纸币；当然还要在显示上给予相应的提示。

6.5. 发送 0x0A 号命令允许识币器识别纸币：

```
W: 7F 00 01 0A 3C 08    // 发送 Enable 命令
R: 7F 00 01 F0 20 0A    // 识币器应答 OK
```

6.6. 循环发送 Poll 命令，等待接收纸币：

(注意：每 2 个发送数据包之间要有 100 到 200ms 的延时， 以便给识币器有足够的时间处理相关的事件)：

```
W: 7F 00 01 07 11 88
R: 7F 00 01 F0 20 0A

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA   // OK， 有纸币进入

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76  // OK， 有纸币进入

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA  // OK， 有纸币进入

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 04 D7 F6  // OK， 纸币处于第 4 通道

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2     // Stacking(压币)

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22     // Stacking(压币)

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2     // Stacking(压币)

W: 7F 00 01 07 11 88
R: 7F 00 05 F0 EE 03 CC EB E1 18   // Credit， n； Stacking(压币)； Stacked

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 CC EB B6 82        // Stacking， Stacked， 到此处 1 张纸币接收就算处理结束

W: 7F 00 01 07 11 88
R: 7F 00 01 F0 20 0A

W: 7F 80 01 07 12 02
```

R: 7F 80 01 F0 23 80

注意：1. 识币器返回的数据包中可能包含多个命令字，因此，拆分时必须逐个按命令的字节数向后读取命令，直到数据包结束为止；

2. 终止识币器对纸币操作时， 必须先检测识币器是否处理空闲状态，即连续 2 到 3 次发送 Poll 命令后接收到的数据只含有一个字节的 0xf0；

3. 如果在几个连续的 Poll 命令发送和接收过程中都有 Credit 指令，则只当作一个来处理，其余的丢掉，注意必须是连续的；

4. 当识别完毕一张纸币后再次进入循环时，最好是先发一个同步命令，再继续进行。

6.6. 关闭识币器工作：

如果纸币读取完毕后，本次不需要再读取纸币，最好关闭识币器，发送 Disable 命令，然后不再向识币器发送任何数据就可以了。

注意：发送 Disable 命令，必须先检测识币器是否处理空闲状态，即连续 2 到 3 次 发送 Poll 命令后接收到的数据只含有一个字节的 0xf0;

如果有纸币插入， 必须先将纸币处理完毕(接收或拒收)后， 检测到识币器空闲才能发送 Disable 命令， 退出操作。

7. 日志文件的编写：

完整的日志文件应该包含发送到识币器和从识币器接收到的数据，但是对于发送 Poll 后只返回 0xf0 这样的数据，建议只保留一次就行，随后重复的 Poll 和收到的 0xf0 可以不用保存，因为，这样的数据非常多，又没有太多用处. 如果 Poll 后除 0xf0 外还有其他数据，还是要保存到日志文件. 这里所说的重复数据是指连续的完全一样的数据。

如: W: 7F 80 01 07 12 02
R: 7F 80 01 F0 23 80

W: 7F 00 01 07 11 88
R: 7F 00 01 F0 20 0A

上以示例中的上两个数据是实质是一样的，都是发送 Poll 命令，只返回 OK，形式上不一样是因为 SEQ 不相同，因此在日志文件中只须保存 “W: 7F 80 01 07 12 02”和 “R: 7F 80 01 F0 23 80”，更详细一点的还包含发生的日期及时间，方便异常状态下分析数据用。

附录 1:识币器纸币数据国家代码表:

国家名称	缩写代号	国家名称	缩写代号
Algeria	DZD	Japan	JPY
Arab (Un. Emir)	AED	Jordan	JOD
Argentina	ARA	Kenya	KES
Australia	AUD	Kuwait	KWD
Austria	ATS	Lebanon	LBP
Bahrain	BHD	Luxembourg	LUF
Belgium	BEF	Malaysia	MYR
Brazil	BRE	Mexico	MXP
Bulgaria	BGL	Morocco	MAD
Canada	CAD	Netherlands	NLG
China	CYN	Netherlands Ant.	ANG
C.I.S	RBL	New Zealand	NZD
Columbia	COP		
Cuba	CUP	Nigeria	NGN
Cyprus	CPY	Norway	NOK
Czechoslovakia	CSK	Oman	OMR
Denmark	DKK	Philippines	PHP
Egypt	EGP	Poland	PLZ
Finland	FIM	Portugal	PTE
France	FRF	Qatar	QAR
Germany	DEM	Rep. of Croatia	HRD
Great Britain	GBP	Rep. of Slovenia	SIT
Greece	GRD	Rumania	ROL
Hong Kong	HKD	Saudi Arabia	SAR
Hungary	HUF	Singapore	SGD
Iceland	ISK	South Africa	ZAR
India	INR	Spain	ESP
Indonesia	IDR	Sri Lanka	LKR
Iran	IRR	Sweden	SEK
Iraq	IQD	Switzerland	CHF
Ireland	IEP	Syria	SYP
Israel	ILS	Tunisia	TND
Italy	ITL	Turkey	TRL
		United States	USD

附录 2: ssp 所用的 Crc 算法:

```
#define FALSE    0x00
#define TRUE     0x01

#define uchar    unsigned char
```

```
#define uint     unsigned int

uchar CrcL,CrcH;

const uint CrcTable[8*32]=
{
    0x0000, 0x8005, 0x800F, 0x000A, 0x801B, 0x001E, 0x0014, 0x8011,
    0x8033, 0x0036, 0x003C, 0x8039, 0x0028, 0x802D, 0x8027, 0x0022,
    0x8063, 0x0066, 0x006C, 0x8069, 0x0078, 0x807D, 0x8077, 0x0072,
    0x0050, 0x8055, 0x805F, 0x005A, 0x804B, 0x004E, 0x0044, 0x8041,
    0x80C3, 0x00C6, 0x00CC, 0x80C9, 0x00D8, 0x80DD, 0x80D7, 0x00D2,
    0x00F0, 0x80F5, 0x80FF, 0x00FA, 0x80EB, 0x00EE, 0x00E4, 0x80E1,
    0x00A0, 0x80A5, 0x80AF, 0x00AA, 0x80BB, 0x00BE, 0x00B4, 0x80B1,
    0x8093, 0x0096, 0x009C, 0x8099, 0x0088, 0x808D, 0x8087, 0x0082,
    0x8183, 0x0186, 0x018C, 0x8189, 0x0198, 0x819D, 0x8197, 0x0192,
    0x01B0, 0x81B5, 0x81BF, 0x01BA, 0x81AB, 0x01AE, 0x01A4, 0x81A1,
    0x01E0, 0x81E5, 0x81EF, 0x01EA, 0x81FB, 0x01FE, 0x01F4, 0x81F1,
    0x81D3, 0x01D6, 0x01DC, 0x81D9, 0x01C8, 0x81CD, 0x81C7, 0x01C2,
    0x0140, 0x8145, 0x814F, 0x014A, 0x815B, 0x015E, 0x0154, 0x8151,
    0x8173, 0x0176, 0x017C, 0x8179, 0x0168, 0x816D, 0x8167, 0x0162,
    0x8123, 0x0126, 0x012C, 0x8129, 0x0138, 0x813D, 0x8137, 0x0132,
    0x0110, 0x8115, 0x811F, 0x011A, 0x810B, 0x010E, 0x0104, 0x8101,
    0x8303, 0x0306, 0x030C, 0x8309, 0x0318, 0x831D, 0x8317, 0x0312,
    0x0330, 0x8335, 0x833F, 0x033A, 0x832B, 0x032E, 0x0324, 0x8321,
    0x0360, 0x8365, 0x836F, 0x036A, 0x837B, 0x037E, 0x0374, 0x8371,
    0x8353, 0x0356, 0x035C, 0x8359, 0x0348, 0x834D, 0x8347, 0x0342,
    0x03C0, 0x83C5, 0x83CF, 0x03CA, 0x83DB, 0x03DE, 0x03D4, 0x83D1,
    0x83F3, 0x03F6, 0x03FC, 0x83F9, 0x03E8, 0x83ED, 0x83E7, 0x03E2,
    0x83A3, 0x03A6, 0x03AC, 0x83A9, 0x03B8, 0x83BD, 0x83B7, 0x03B2,
    0x0390, 0x8395, 0x839F, 0x039A, 0x838B, 0x038E, 0x0384, 0x8381,
    0x0280, 0x8285, 0x828F, 0x028A, 0x829B, 0x029E, 0x0294, 0x8291,
    0x82B3, 0x02B6, 0x02BC, 0x82B9, 0x02A8, 0x82AD, 0x82A7, 0x02A2,
    0x82E3, 0x02E6, 0x02EC, 0x82E9, 0x02F8, 0x82FD, 0x82F7, 0x02F2,
    0x02D0, 0x82D5, 0x82DF, 0x02DA, 0x82CB, 0x02CE, 0x02C4, 0x82C1,
    0x8243, 0x0246, 0x024C, 0x8249, 0x0258, 0x825D, 0x8257, 0x0252,
    0x0270, 0x8275, 0x827F, 0x027A, 0x826B, 0x026E, 0x0264, 0x8261,
    0x0220, 0x8225, 0x822F, 0x022A, 0x823B, 0x023E, 0x0234, 0x8231,
    0x8213, 0x0216, 0x021C, 0x8219, 0x0208, 0x820D, 0x8207, 0x0202
};

//-----------------------------------------------------------------------
void UpdateCrc(const uchar num)
{
    uint  table_addr;

    table_addr=(num ^ Crch);
    Crch=(CrcTable[table_addr] >> 8) ^ CrcL;
    CrcL=(CrcTable[table_addr] & 0x00FF);
}
```

```
//-----------------------------------------------------------------------
void ResetCrc(void)
{
  CrcL=0xFF;
  CrcH=0xFF;
}
```

示例：对 7F 80 03 02 FF FF 进行校验，假设 uchar 数组 WrBuf[] 存放的数据是 7F，80，03，02，FF，FF 等数据，该数据包的 Crc 计算如下：

```
int i0, s_len;

ResetCrc();
for(i0=1;i0<6;i0++) UpdateCrc(WrBuf[i0]);      // 计算 Crc，0x7f 不参加 Crc 计算
WrBuf[i0++]=CrcL;
WrBuf[i0++]=CrcH;
s_len=i0;

for(i0=0;i0<s_len;i0++) SendChar(WrBuf[i0]);    // 数据发送到识币器
…
```

附录 3：拒收原因：

代码	拒绝原因	拒绝原因
0x00	Note Accepted	纸币接收
0x01	Note length incorrect	纸币长度错误
0x02	Reject reason 2	
0x03	Reject reason 3	
0x04	Reject reason 4	
0x05	Reject reason 5	
0x06	Channel Inhibited	通道禁用
0x07	Second Note Inserted	第二张纸币被放入
0x08	Reject reason 8	
0x09	Note recognised in more than one channel	纸币被认为有多个通道面额
0x0a	Reject reason 10	
0x0b	Note too long	纸币太长
0x0c	Reject reason 12	
0x0d	Mechanism Slow / Stalled	机械故障
0x0e	Striming Attempt	有钓鱼行为
0x0f	Fraud Channel Rejec	假币
0x10	No Notes Inserted	没有纸币放入
0x11	Peak Detect Fail	
0x12	Twisted note detected	纸币变型或过旧

0x13	Escrow time-out	禁用超时
0x14	Bar code scan fail	无法读取条形码
0x15	Rear sensor 2 Fail	后端传感觉器无效
0x16	Slot Fail 1	
0x17	Slot Fail 2	
0x18	Lens Over Sample	
0x19	Width Detect Fail	纸币宽度错误
0x1a	Short Note Detected	纸币过短

附录 4：避免读到多个 Credit 的建议：

1. 程序读到第一个 Credit(0xee) 之后， 记录纸币所在通道号(nn)， 进行第 2 步操作；
2. 循环 5 次 Poll 发送和接收的操作；
3. 在 5 个循环中如果 Poll 之后返回的数据中没有"0xee nn" 这样的指令， 则退出循环， 返回纸币通道号；
4. 如果 Poll 之后返回的数据中存在 "0xee nn" 则不作任何处理， 继续循环操作；
5. 循环结束正常返回纸币通道号。

附件5：对于 Credit 特殊解析：

通常情况下 Credit(0xee) 命令紧跟在 OK 命令之后， 如：“7F 00 05 F0 **EE 02** CC EB F6 98”， 但是也有特殊情况， 如：“7F 80 04 F0 CC **EE 03** 6D 2A”， 因此不能从一个固定的地址上读取 Credit 命令. 因为识币器的返回数据中包含多个指令， 必须在解析数据时必须逐一进行: 首先看返回的数据中含有多少个有效的数据， 对照 SSP 返回指令表逐个向后处理， 如果是单字节的， 处理完毕后长度加一， 如果是双字节的， 则在处理完毕后长度加二，直到长度大于等于有效数据长度为止。

附件6：通讯超时定义：

通讯超时定义如下: 指从 Host 端发送数据到 slave 端开始计时, 等待 slave 返回数据为止等待时间。建议设置成 500ms, 也就是说发送数据后如果等待 500ms 还没有数据返回, 则说明存在通讯超时错误。

附件7：关于同步时 SEQ 是否必须从 0x80 开始的问题：

对于同步时 SEQ 是否必须从 0x80 开始的问题， 经测试不是必须的， 但是如果同步时SEQ 如果是 0x00， 则同步之后的下一条指令的 SEQ 必须从 0x00 开始， 之后才开始是 0x80 / 0x00 交替进行。 因此，为方便编程和方便使用，在本文中都是使用同步时从 SEQ 从 0x80 开始， 之后是 0x00 / 0x80 将进行。

附件8：一次完整的测试数据：

PortOpened(); // 9600bps, StopBits:2, Parity:No, DataBits:8

W: 7F 80 01 11 65 82
R: 7F 80 01 F0 23 80

W: 7F 00 01 05 1E 08
R: 7F 00 1F F0 00 30 34 30 35 43 4E 59 00 00 01 07 01 02 05 0A 14 32 64 02 02 02 02 02 02 02 00 00 00 04 38 69

W: 7F 80 01 09 35 82
R: 7F 80 01 F0 23 80

W: 7F 00 01 07 11 88
R: 7F 00 04 F0 E8 E8 E8 4C BE

W: 7F 80 03 02 FF 00 27 A6
R: 7F 80 01 F0 23 80

W: 7F 00 01 0A 3C 08
R: 7F 00 01 F0 20 0A

W: 7F 80 01 07 12 02
R: 7F 80 01 F0 23 80

... // 重复 Poll -->OK 数据

W: 7F 00 01 07 11 88
R: 7F 00 01 F0 20 0A

W: 7F 80 01 07 12 02
R: 7F 80 01 F0 23 80

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88

R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 02 C0 4A

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 05 F0 EE 02 CC EB F6 98

W: 7F 80 01 07 12 02
R: 7F 80 01 F0 23 80

W: 7F 00 01 07 11 88
R: 7F 00 01 F0 20 0A

... // 重复 Poll -->OK 数据

W: 7F 00 01 07 11 88

R: 7F 00 01 F0 20 0A

W: 7F 80 01 07 12 02

R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88

R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02

R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88

R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02

R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88

R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02

R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88

R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02

R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88

R: 7F 00 02 F0 ED 6E 22

W: 7F 80 01 07 12 02

R: 7F 80 02 F0 ED 51 A2

W: 7F 00 01 07 11 88

R: 7F 00 02 F0 ED 6E 22

W: 7F 80 01 07 12 02

R: 7F 80 02 F0 ED 51 A2

W: 7F 00 01 07 11 88

R: 7F 00 02 F0 ED 6E 22

W: 7F 80 01 07 12 02

```
R: 7F 80 02 F0 EC 54 22

W: 7F 00 01 07 11 88
R: 7F 00 01 F0 20 0A

... // 重复 Poll -->OK 数据

W: 7F 00 01 07 11 88
R: 7F 00 01 F0 20 0A

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 03 C5 CA

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22
```

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 05 F0 EE 03 CC EB E1 18

W: 7F 80 01 07 12 02
R: 7F 80 01 F0 23 80

... // 重复 Poll -->OK 数据

W: 7F 80 01 07 12 02
R: 7F 80 01 F0 23 80

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 01 C9 F6

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 05 F0 EE 01 CC EB C9 90

W: 7F 00 01 07 11 88
R: 7F 00 01 F0 20 0A

... // 重复 Poll -->OK 数据

W: 7F 80 01 07 12 02
R: 7F 80 01 F0 23 80

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02

R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 04 D7 F6

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 05 F0 EE 04 CC EB 8D 90

W: 7F 00 01 07 11 88
R: 7F 00 01 F0 20 0A

... // 重复 Poll -->OK 数据

W: 7F 80 01 07 12 02
R: 7F 80 01 F0 23 80

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 05 D1 CA

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 05 F0 EE 05 CC EB 99 18

W: 7F 80 01 07 12 02

R: 7F 80 01 F0 23 80

... // 重复 Poll -->OK 数据

W: 7F 80 01 07 12 02
R: 7F 80 01 F0 23 80

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 06 D8 76

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 05 F0 EE 06 CC EB A6 10

W: 7F 00 01 07 11 88
R: 7F 00 01 F0 20 0A

... // 重复 Poll -->OK 数据

W: 7F 80 01 07 12 02
R: 7F 80 01 F0 23 80

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 06 DB CA

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 05 F0 EE 06 CC EB A5 18

W: 7F 80 01 07 12 02
R: 7F 80 01 F0 23 80

... // 重复 Poll -->OK 数据

W: 7F 00 01 07 11 88
R: 7F 00 01 F0 20 0A

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88

R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 07 DD F6

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 05 F0 EE 07 CC EB B1 90

W: 7F 00 01 07 11 88
R: 7F 00 01 F0 20 0A

... // 重复 Poll -->OK 数据

W: 7F 80 01 07 12 02
R: 7F 80 01 F0 23 80

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 00 CF CA

W: 7F 00 01 07 11 88
R: 7F 00 03 F0 EF 00 CC 76

W: 7F 80 01 07 12 02
R: 7F 80 03 F0 EF 07 DE 4A

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88
R: 7F 00 02 F0 CC A8 22

W: 7F 80 01 07 12 02
R: 7F 80 02 F0 CC 97 A2

W: 7F 00 01 07 11 88

R: 7F 00 05 F0 EE 07 CC EB B2 98

W: 7F 80 01 07 12 02
R: 7F 80 01 F0 23 80

... // 重复 Poll -->OK 数据

W: 7F 00 01 07 11 88
R: 7F 00 01 F0 20 0A

W: 7F 80 01 07 12 02
R: 7F 80 01 F0 23 80

W: 7F 00 01 09 36 08
R: 7F 00 01 F0 20 0A

ClosePort();

End.
