东莞纳普电子科技有限公司
地址：广东省东莞市松山湖科技产业园区博士创业园 8 号 2 楼
http://www.napui.com

纳普仪表通讯规约

注意：

本规约中使用的数据用十进制或十六进制表示，数据后面带“**H**”的为十六进制数据，不带“**H**”为十进制数据

（本规约中所有示例中，均假定仪表的通讯地址为**1**）

通讯规约版本号： **Ver 2013.7.2**

2013年7月2日星期二

目 录

东莞纳普电子科技有限公司
地址：广东省东莞市松山湖科技产业园区博士创业园 8 号 2 楼
http://www.napui.com

单相仪表通讯规约

此规约仅适用于纳普公司的以下产品，请核实规格型号后使用。

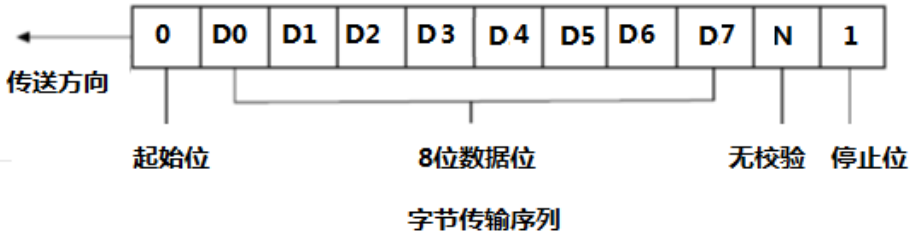
PM9805

纳普公司的以下产品用户可选配具有通讯功能（规约同上）

PM9800、PM9801、PM9804、PM9804A、PM9813、PM9815、PM9840

一、 通讯字节格式说明：

8 位数据、1 位起始位、1 位停止位、无校验。先传低位，后传高位，位传送方式与微机一样（先传低位，后传高位）



字节传输序列

二、 通讯帧格式：

2.1 上位机发送格式： 55H— Address —Command—CS

代码说明备注

55H: 帧起始符，表示计算机发出命令帧起始位置，51H＝01010001B

Address: 仪表地址，仪表的通讯地址，单字节，范围为0~255

Command: 命令码，标识计算机要读取的内容命令，单字节（见通讯命令码）

CS： 校验码，从功能码到校验码之前的所有字节的算术和的256 的模，即：

CS=(55H+Address+Command) AND FFH

2.2 仪表回送格式 ：**AA—Address —Command—Data—CS**

代码说明备注

AA: 帧起始字符，标识仪表回送的数据帧的起始位置，AAH＝10101010B

Address： 仪表地址，见2.1 仪表地址**Address**

Command： 命令码，与计算机读取的内容命令码一致

Data(4*n):数据域，仪表回送数据，为4字节浮点数，低前高后。见备注1

CS 校验码: 从帧起始符到校验码之前的所有字节的算术和的256 的模，即：

CS=(AAH+Adrress+Command+ Data) AND FFH

备注**1**---数据域：仪表根据相应的命令回送的数据，其字节数根据仪表类型及命令的不同而不同。仪表回送的每个量为四字节的浮点数，低字节在前，高字节在后，所以数据域Data的字节个数为4 的倍数。

2.3 传送的速率：波特率=9600。

仪表地址 ：见仪表机箱地址条码，（所有仪表地址均在0-255之间）

三、 通讯命令及仪表回送数据

3.1 通讯命令码：仪表命令码见下表：

功能码	命令码	命令码含义	仪表种类
55H（读）	10H	读取基本电参数：V， I， P， FREQ， PF	所有仪表

功能码	命令码	命令码含义	仪表种类
52H（写）	待定	待定	待定

3.2 仪表回送数据：仪表应答/回送见下表：

序号	命令码	仪表应答内容
1	10H	V/I/P/F/PF：电压真有效值，电流真有效值，有功功率，频率，功率因数
2	…	……

3.3 通讯具体格式：

3.3.1.1 主机发送： 55H+Address+10H+CS

含义：主机向仪表发送读取所有电参数命令

3.3.1.2 仪表回送数据：

AAH+Address+10H+数据（V，I，P，FREQ，PF）+CS

每个参数4 个字节，回送5 个参数，共回送5 x 4+4=24 个字节

含义：仪表向主机回送所有电参数

3.4 注释：

3.4.1 仪表回送数据中的“V”—表示电压，单位V,显示最小值的小数点位0.01V

例如：电压显示100.68V，5.47V，0.01V

3.4.2 仪表回送数据中的“I”—表示电流，单位A,显示最小值的小数点位0.01mA

例如：电流显示10.68 A，9.999 A，999.9mA，1mA显示为1.00mA，0.1mA显示为0.10mA

3.4.3 仪表回送数据中的“W”—表示功率，单位W,显示最小值的小数点位0.01W

例如：功率显示10.68KW，9999W, 999W, 99.99W，0.01W

3.4.4 仪表回送数据中的“F”—表示频率，单位Hz,显示最小值的小数点位0.01Hz

例如：频率显示49.98Hz，60.01Hz，100.1Hz，，400.1Hz

3.4.5 仪表回送数据中的“PF”—表示功率因数,显示最小值的小数点位0.001

例如：功率因数显示0.997，1.000，0.576

3.4.12 浮点数据格式如下：共四个字节，32 位数据

31	**30---23**	22---0
符号位	指数	尾数

附录1-1：IEEE754 单精度浮点格式：

IEEE 单精度格式由三个字段组成：23 位小数 f ； 8 位偏置指数 e ；以及 1 位符号 s。这些字段连续存储在一个 32 位字中（如附录A 表1 所示）。0:22 位包含 23 位小数 f，其中第 0 位是小数的最低有效位，第 22 位是最高有效位； 23:30 位包含 8 位偏置指数 e，第 23 位是偏置指数的最低有效位，第 30 位是最高有效位；最高的第 31 位包含符号位 s。

格式	S	e[30:23]	f[22:0]
位	31	30－23	22－0

表1：单精度浮点数据存储格式

附录1 表2 显示一侧的三个组成字段 s、e 和 f 的值与另一侧的单精度格式位模式表示的值之间的对应关系；u 意味着无关，即指示字段的值与确定特定单精度格式位模式的值无关。

单精度格式位模式	值
0 < e < 255	$(-1)^S \times 2^{e-127} \times 1.f$ （正规数）
e = 0;f ≠ 0（f 中至少有一位不为零）	$(-1)^S \times 2^{-126} \times 0.f$ （次正规数）
e = 0; f = 0（f 中的所有位均为零）	$(-1)^S \times 2^{-126} \times 0.0$（有符号的零）
s = 0; e = 255; f = 0（f 中的所有位均为零）	+INF （正无穷大）
s = 1; e = 255; f = 0（f 中的所有位均为零）	－INF （负无穷大）
s = u; e = 255;f ≠ 0（f 中至少有一位不为零）	NaN （非数）

表2：单精度格式位模式表示的值

注意，当 e < 255 时，为单精度格式位模式分配的值是使用以下方法构成的：将二进制基数点插入到紧邻小数最高有效位的左侧，将一个隐含位插入到紧邻二进制点的左侧，因而以二进制位置表示法来表示一个带分数 （整数加小数，其中 0 ≤ ò3 小数 < 1）

如此构成的带分数称为单精度格式有效数字。之所以称为隐含位的原因是，在单精度格式位模式中没有显式地指定其值，但偏置指数字段的值隐式指定了该值。

对于单精度格式，正规数和次正规数的差别在于正规数有效数字的前导位（二进制点左侧的位）为 1，而次正规数有效数字的前导位为 0。在 IEEE 754 标准中，单精度格式次正规数称为单精度格式非规格化数。

在单精度格式正规数中 23 位小数加上隐含前导有效数位提供了 24 位精度。

附录1 表3 中给出了重要的单精度存储格式位模式的示例。最大正正规数是以 IEEE 单精度格式表示的最大有限数。最小正次正规数是以 IEEE 单精度格式表示的最小正数。最小正正规数通常称为下溢阈值。（最大和最小正规数和次正规数的十进制值是近似的；对于所示的数字来说，它们是正确的。）

通用名称	位模式（十六进制）	十进制值
+0	**0 0 0 0 0 0 0 0**	**0.0**
-0	**8 0 0 0 0 0 0 0**	**-0.0**
1	**3 f 8 0 0 0 0 0**	**1.0**
2	**4 0 0 0 0 0 0 0**	**2.0**
最大正规数	**7 f 7 f f f f f**	**3.40282347e+38**
最小正正规数	**0 0 8 0 0 0 0 0**	**1.17549435e-38**
最大次正规数	**0 0 7 f f f f f**	**1.17549421e-38**
最小正次正规数	**0 0 0 0 0 0 0 1**	**1.40129846e-45**
+∞	**7 f 8 0 0 0 0 0**	无穷
-∞	**F f 8 0 0 0 0 0**	负无穷
非数	**7 f c 0 0 0 0 0**	**NaN**

表3:单精度存储格式位模式及其 IEEE 值

附录1-2：IEEE754 单精度浮点手工转换样例：

一、概述：单精度浮点数据由四个字节组成，仪表在通讯中，回送这四个字节时分2 种方式：一种是高字节在前，低字节在后；另一种是低字节在前，高字节在后，下面手工转换，都假设四字节的浮点数据高字节在前，低字节在后。四字节共32 位

二、**IEEE754** 单精度浮点格式及计算（共四字节32 位）

1、IEEE754 单精度浮点格式（共四字节32 位，从高到低）

二进制位	32	31 24	23 1
说明	符 号(1 位) 其数值用 S 表示	指 数（8 位） 其数值用 E 表示	尾 数（23 位） 其数值用 F 表示

2、格式说明：

A、第32 bit 为符号位，为0 则表示正数，反之为负数，其读数值用S 表示；

B、第31～24 bit 共8 位为幂数(2的幂数)，其读数值用E 表示；

C、第23～1 bit 共23 位作为系数, 视为二进制纯小数, 假定该小数的十进制值为F

D、转换后的十进制浮点数据以FData 表示；

3、转为为十进制浮点数据公式为：

$$FData=(-1)^S * (1 + F) * 2^{(E - 127)}$$

三、浮点数据实例：（高字节在前）

十进制	十六进制	二进制数据
220.5	43-5C-80-00	0100 0011 0101 1100 1000 0000 0000 0000
380.6	43-BE-4C-CD	0100 0011 1011 1110 0100 1100 1100 1101
50.25	42-49-00-00	0100 0010 0100 1001 0000 0000 0000 0000
0.999	3F-7F-BE-77	0011 1111 0111 1111 1011 1110 0111 0111
1.0	3F-80-00-00	0011 1111 1000 0000 0000 0000 0000 0000

四、手工转换实例：

1、**220.5=43-5C-80-00** 进行转换

A、220.5=43-5C-80-00 用二进制表示如下：

0	1 0 0 0 0 1 1 0	1 0 1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
32	31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
		1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
S	幂数部分 E	纯小数部分 F

B说明：

a、 符号位：为0，即S=0

b、指数部分：E = 86H = 134

c、 纯小数部分：F= $\frac{1}{2^1}+\frac{1}{2^3}+\frac{1}{2^4}+\frac{1}{2^5}+\frac{1}{2^8}$ =0.5+0.125+0.0625+0.03125+0.00390625

=0.72265625 注：只有小数部分的1，3，4，5，8 位为1，其他为0

d、转换后的数据 Fdata=$(-1)^S*(1+f)*2^{(E-127)}=(-1)^0*(1+0.72265625)*2^{(134-127)}=1.72265625*2^7=1.72265625*128=220.5$

2、**50.25＝42-49-00-00 进行转换**

A、50.25＝42-49-00-00 用二进制表示如下：

0	1 0 0 0 0 1 0 0	1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
32	31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
		1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
S	幂数部分 E	纯小数部分 F

B 说明：

a、号位：为 0，即 S=0

b、指数部分：E = 84H = 132

c、纯小数部分：$F=\frac{1}{2^1}+\frac{1}{2^4}+\frac{1}{2^7}=0.5+0.0625+0.0078125=0.5703125$

d、转换后的数据 Fdata=$(-1)^S*(1+f)*2^{(E-127)}=(-1)^0*(1+0.5703125)*2^{(132-127)}=1.5703125*2^5=1.72265625*32=50.25$

3、**0.999＝3F-7F-BE-77 进行转换**

A、0.999＝3F-7F-BE-77 用二进制表示如下：

0	0 1 1 1 1 1 1 0	1 1 1 1 1 1 1 1 0 1 1 1 1 1 0 0 1 1 1 0 1 1 1
32	31 30 29 28 27 26 25 24	23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
		1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
S	幂数部分 E	纯小数部分 F

B 说明：

a、位：为 0，即 S=0

b、指数部分：E = 7EH = 126

c、纯小数部分：

$$F=\frac{1}{2^1}+\frac{1}{2^2}+\frac{1}{2^3}+\frac{1}{2^4}+\frac{1}{2^5}+\frac{1}{2^6}+\frac{1}{2^7}+\frac{1}{2^8}+\frac{1}{2^{10}}+\frac{1}{2^{11}}+\frac{1}{2^{12}}+\frac{1}{2^{13}}+\frac{1}{2^{14}}+\frac{1}{2^{17}}+\frac{1}{2^{18}}+\frac{1}{2^{19}}+\frac{1}{2^{21}}+\frac{1}{2^{22}}++\frac{1}{2^{23}}$$

=0.5+0.25+0.125+0.0625+0.03125+0.015625+0.0078125+0.00390625+0.0009765625+0.00048828125+0.000244140625+0.0001220703125+0.00006103515625+0.00000762939453125+0.00000381469726562 5+0.00000190734863281 25+0.00000047683715820 3125+0.00000023841857910 15625+0.00000011920928955 078125=0.99800002574920654296875

d、转换后的数据

Fdata=$(-1)^S*(1+f)*2^{(E-127)}=(-1)^0*(1+0.99800002574920654296875)*2^{(126-127)}=1.99800002574920654296875/2=0.99900001287460327148437 5\approx0.999000$

注：之所以在最后添加了“≈”符号，因为单精度浮点数据有效位数为 7 位

附录2：数据转换例程

一、概述：本文是以C51(Franklin C)的格式来给出“浮点转换为字节数据”、“字节转换为浮点数据”，“长整形转换为字节数据”、“字节数据转换为长整形数据”等，本文使用的编译器为Franklin C，使用其他类型的编译器时(例Keil C)，只须修改浮点数据转换
例程中填入字节的顺序即可。同时，本文的例程，同样可以在Turbo C++上直接运行。产生同样的效果
二、数据转换例程：

```
/*=========================
公共变量
==========================*/
union
{
   unsigned char uc[4];
   long lda;
   unsigned long ul;
   float fda;
}un_4b;
union
{
   unsigned char uc[2];
   int ida;
   unsigned int ui;
}un_2b;
long lda;
int ida;
float real;
unsigned char uca[4];
unsigned char ucb[2];
```

1、 浮点数据转换为字节数据

```
/*================================================
浮点数据转换为字节数据
入口数据：real 放入要转换的浮点数据;
出口数据：转换的四字节数据在uca[]中
顺序是从低(uca[0])到高(uca[3])
real=1.0;转换为字节uca[0]-uca[3]=0,0,0x80,0x3f
================================================*/
void FtoB(void)
{
   un_4b.fda=real;
```

```
    uca[0]=un_4b.uc[0];
    uca[1]=un_4b.uc[1];
    uca[2]=un_4b.uc[2];
    uca[3]=un_4b.uc[3];
}
```

2、 字节数据转换为浮点数据

```
/*=================================================
字节数据转换为浮点数据
入口数据：要转换的四字节数据在uca[]中
顺序是从低(uca[0])到高(uca[3])
出口数据：real 存放的为已转换的浮点数据;
数据uca[0]-uca[3]=0,0,0x80,0x3f 转换为浮点real=1.0
=================================================*/
void BtoF(void)
{
    un_4b.uc[0]=uca[0];
    un_4b.uc[1]=uca[1];
    un_4b.uc[2]=uca[2];
    un_4b.uc[3]=uca[3];
    real=un_4b.fda;
}
```

3、 长整形数据转换为字节数据

```
/*=============================================================
长整形数据转换为字节数据
入口数据：要转换的长整形放在lda 中
出口数据：转换完的四字节数据在uca[]中
顺序是从高(uca[0])到第(uca[3])
长整数据lda=1000 转换的字节数据uca[0]-uca[3]=0xe8,0x03,0,0
=============================================================*/
void LtoB(void)
{
    un_4b.lda=lda;
    uca[0]=un_4b.uc[0];
    uca[1]=un_4b.uc[1];
    uca[2]=un_4b.uc[2];
    uca[3]=un_4b.uc[3];
}
```

4、 字节数据转换为长整形数据

```
/*=============================================================
字节数据换为长整形数据转
入口数据：转换完的四字节数据在uca[]中
顺序是从高(uca[0])到第(uca[3])
出口数据：转换完毕的长整形放在lda 中
```

字节数据uca[0]-uca[3]=0xe8,0x03,0,0 转换的长整形数据lda=1000
==========*/

```
void BtoL(void)
{
  un_4b.uc[0]=uca[0];
  un_4b.uc[1]=uca[1];
  un_4b.uc[2]=uca[2];
  un_4b.uc[3]=uca[3];
  lda=un_4b.lda;
}
```

5、 整形数据转换为字节数据

```
/*==========
整形数据换为字节数据
入口数据：要转换的整形放在ida 中
出口数据：转换完的 2 字节数据在ucb[]中
顺序是从高(ucb[0])到第(ucb[1])
要转换的整形数据ida=1000,转换的字节数据ucb[0]-ucb[1]=0xe8,0x03
==========*/
void ItoB(void)
{
  un_2b.ida=ida;
  ucb[0]=un_2b.uc[0];
  ucb[1]=un_2b.uc[1];
}
```

6、 字节数据转换为整形数据

```
/*==========
字节数据转换为整形数据
入口数据：要转换的 2 字节数据在ucb[]中
顺序是从高(ucb[0])到第(ucb[1])
出口数据：转换完毕的整形放在ida 中
字节数据ucb[0]-ucb[1]=0xe8,0x03 转换的整形数据ida=1000
==========*/
void BtoI(void)
{
  un_2b.uc[0]=ucb[0];
  un_2b.uc[1]=ucb[1];
  ida=un_2b.ida;
}
```

附录3：通讯样例：（本样例中定义仪表地址为1）

3.1通讯数据实例——读取电参数

3.1.1 主机发送： 55 03 10 68 (55起始，03地址，10命令，68校验)

3.1.2 仪表回送：

字节位置	1	2	3	4－7	8－11	
字节名称	AAH	3	10H	ECH 6AH 66H 43H	00H 00H 00H 00H	
说明	帧头	地址	命令	电压数据 230.4V	电流数据 0.000	
字节位置	12－15			16－19	20－23	24
字节名称	00H 00H 00H 00H			8AH 52H 48H 42H	00H 00H 00H 00H	22H
说明	功率数据 0.0			频率数据 50.08	功率因数 0.000	校验和