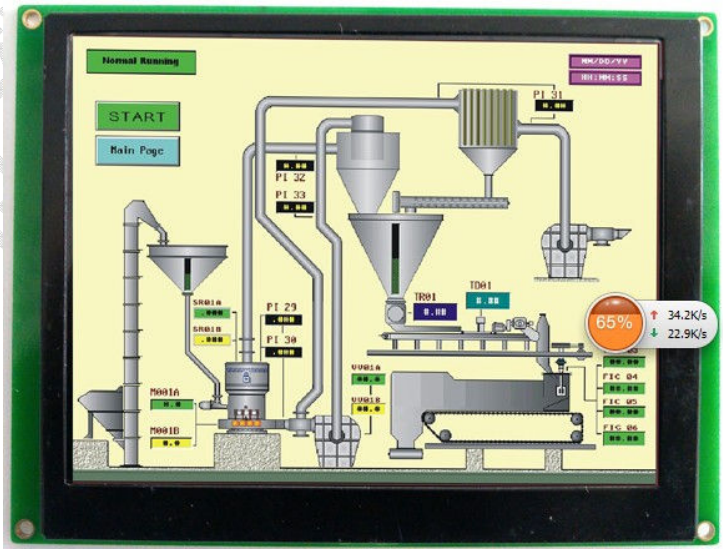
目 录

随着时代的变迁，科技进步，显示部分对一个产品也显得越来越重要，好的产品肯定会有一个良好的人机交互体验，而随着彩屏的大量使用，以前传统的单色屏譬如1602，128*64，320*240等产品已经不能满足产品需求和客户体验；然而对于像8051,PIC,AVR,大部分ARM7来说都是不支持TFT外设的，即使部分ARM7,ARM9支持，若要解决高分辨率显示，图片字库存储，触控菜单操作，GUI以及美工素材等细节问题，依然存在一定的开发难度。

出于以上原因，为了减少客户的开发难度，串口屏成为了客户不错的选择，它已经做好了显示的底层部分，彻底将用户控制和显示分离出来。用户无需更改核心控制代码， 只需要在原来的代码基础上增加串口发送和接收函数，即可让自己的产品快速升级到真彩屏，提高市场竞争力。



深圳市新雁飞科技推出的串口屏是集合TFT驱动，图片存储，自带字库，以及各种控件于一体的智能显示终端。产品设计将成本和功能做了一个良好的结合，以其高性价比为核心竞争力，功能力求实用但是并不堆砌。产品的核心功能主要由上位机+控件+指令三部分构成，我司上位机设计力求清晰明了，把握重点，让用户使用时能快速上手，控件是用户常用功能的具体体现，由上位机设置，由指令或者触摸对其操作，而指令的补充也可以让用户对屏的控制变得更加灵活。

对于大部分产品而言，产品显示可以看做是多幅画面构成，用户对于新雁飞串口屏开发的核心思路是图片+控件+指令，用户首先准备图片素材，通过上位机放置控件，将素材图片放置到上位机工程中，根据用户需要通过上位机放置控件，下载图片和相关配置文件，之后在下位机中通过指令对其进行数据传输和控制。

1 串口说明

1.1 串口工作模式

新雁飞科技所有标准HMI产品均采用异步、全双工串口（UART），串口模式为 8n1，即每个数据传送采用10个位：1个起始位，8个数据位（低位在前传送，LSB），1个停止位。

▲上电时，如果终端的/O0引脚为高电平或者浮空状态，串口波特率由用户预先设置，范围1200-921600bps，具体设置方法参考0xE0指令,屏通信速度不要超过115200bps,下载图片时建议用最高921600bps来下载。

1.2 数据帧架构

新雁飞HMI的串口数据帧由4个数据块组成，如下表所述：

数据块	1	2	3	4
举　例	0xAA	0x70	0x01	0xCC 0x33 0xC3 0x3C
说　明	帧头	指令	数据，最多512字节	帧结束符（帧尾）

1.3 通信帧缓冲区

新雁飞 HMI 有一个高达 512 字节的通信帧缓冲区，只要通信缓冲不溢出，用户可以连续传送数据给 HMI。

新雁飞 HMI 有一个硬件引脚（用户接口中的“BUSY 引脚”)指示了 FIFO 缓冲区的状态，正常时，BUSY 引脚为高电平（RS232 接口为负电平），当 FIFO 缓冲区可用空间小于 64 字节时，BUSY 引脚会立即变成低电平（RS232 接口为正电平）,此时应该暂停往 HMI 发送数据，避免数据溢出错误，HMI 在两种情况下 BUSY 都会提示忙碌，第一是缓冲区不足，第二是执行需要较长执行时间的指令时，因为执行指令时间较长(特别是图片切换指令)，容易出现后面指令相应不及时的情况，所以此时 HMI 用忙碌提示用户暂缓指令输入。

对于一般的应用，由于新雁飞 HMI 的处理速度很快，用户用不着判断 BUSY 信号状态，但是对于短时间需要传送多个数据帧的应用，比如一次需要刷新上百个屏幕参数，建议客户使用 BUSY 信号来控制串口发送，当 BUSY 信号为低电平时，就不要发送数据给 HMI。

如果用户使用 HMI 过程中，出现“丢帧”现象，即某些数据没有显示出来或者显示错误乱码

可能就是缓冲区溢出丢帧了，这时需要用示波器检查 BUSY 信号是否有跳变，如果有跳变，则需要减慢发送数据，或者增加硬件检测 BUSY 信号判忙处理。

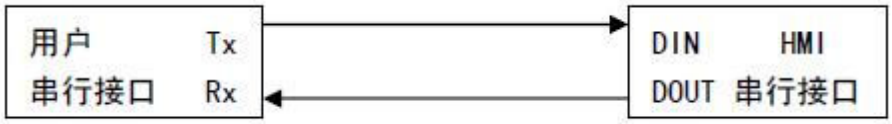
1.4 字节传送顺序

新雁飞 HMI 的所有指令或者数据都是 16 进制(HEX)格式；对于字型(2 字节或者多字节)数据，总是采用高字节先传送(MSB)方式。

比如：x 坐标为 100，其 HEX 格式数据为 0x0064,传送给 HMI 时，传送顺序为 0x00,0x64。

▲ 下行（Tx） 用户发送数据给 HMI，数据从 HMI 用户接口的“DIN 引脚”输入。

▲ 上行（Rx） HMI 发送数据给用户，数据从 HMI 用户接口的“DOUT 引脚”输出。



2 指令速查表

类别	指令	说明
握手	0x00	查看版本信息,握手后才能下载和修改配置。
显示参数配置	0x40	设置调色板。
	0x41	设置字符间距。
	0x42	取色到背景色调色板。
	0x43	取色到前景色调色板。
	0x44	设置光标显示模式。
文本显示	0x54	16×16 点阵 GB2312 内码字符串显示
	0x55	32×32 点阵 GB2312 内码字符串显示
	0x6F	24×24 点阵 GB2312 内码字符串显示
	0x64	文字放大和通透显示
置点	0x50	背景色置多个点（删除点）。
	0x51	前景色显示多个点。
	0x74	动态曲线快速置点。
线段和多变形	0x56	把指定点用前景色线段连接。
	0x5D	把指定点用背景色线段连接。
	0x75	快速显示连续的同底垂直线段（频谱）。
	0x76	快速显示折线图。
	0x78	偏移量连线。
圆弧和圆域	0x57	反色/显示多个圆弧或圆域。
矩形框	0x59	前景色显示多个矩形框（显示矩形框）。
	0x69	背景色显示多个矩形框（删除矩形框）。

三角形	0x58	前景色显示多个三角形或填充的三角形。
椭圆	0x53	前景色显示多个椭圆或填充的椭圆。
区域操作	0x52	背景色清屏。
	0x5A	以背景色填充矩形区域。
	0x5B	以前景色填充矩形区域。
	0x5C	多个指定区域反色。
	0x60	水平卷动。
	0x61	垂直卷动。
	0x62	水平卷动，最右边区域用背景色填充。
	0x63	垂直卷动，最下边区域用背景色填充。
图片显示	0x70	显示一幅全屏图像。
	0x71	从指定图片剪切图标粘贴到当前显示页。
触摸屏操作	0x72	触摸屏松开后，最后一次数据上传（可 0xE0 指令关闭）
	0x73	触摸屏按下后，数据上传（可 0xE0 指令设置）
	0x79	触控界面自动切换模式下，触摸屏松开时，预设键码自动上传。
	0x78	触控界面自动切换模式下，触摸屏按下时，预设键码自动上传。
蜂鸣器控制	0x79	蜂鸣器鸣叫一声。
背光控制	0x5E	关闭背光或设置触控背光模式。
	0x5F	打开背光或 PWM 方式调节背光亮度。
参数配置	0xE0	配置用户串口速率，触摸屏数据上传格式，显示模式，上电保存。
睡眠功能	0xE7	设置触摸屏进入睡眠模式，点击触摸唤醒触摸屏。
指令开关睡眠功能	0xE8	指令开启或者关闭睡眠模式
查询当前所在图片编号	0xE9	查询当前所在图片的编号
图片下载	0xF7	下载图片数据。
按钮控件下载	0xF2-0xF9	下载配置文件。
打开关闭触摸功能	0xE1	用来打开或者关闭触摸功能，控制触摸是否可以点击
控件操作	0x80	对控件进行操作

3 指令集说明（指令说明都是 16 进制表达方式，省略了 0x）

3.1 握手指令（0x00）

Tx：AA 00 CC 33 C3 3C

Rx：终端型号 ，终端版本

新雁飞 HMI 上电初始化需要 0.5-2 秒左右的时间（取决于用户的电源容量和上电速率），在上电初

始化未完成之前，不会响应用户指令，有几个操作需要握手后才能执行，1：参数配置；2：图片下载；3：按钮控件下载；4：编辑框控件下载；5：进度条控件下载；6：滑动条控件下载；7：勾选框控件下载；这在上位机操作时有体现，首先要先握手才能进行下载操作，其他主要指令则无需先握手。

3.2 设置当前调色板（0x40）

Tx：AA 40 <FC> <BC> CC 33 C3 3C

Rx：无

▲ <FC>前景色调色板，2 字节（16 bit, 65K color）,复位默认值是 0xFFFF(白色)

▲ <BC>背景色调色板，2 字节（16 bit, 65K color）,复位默认值是 0x001F(蓝色)

▲ 16bit 调色板定义是 5R6G5B 模式，如下表所示：

16bit 调色板位定义																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Define	R4	R3	R2	R1	R0	G5	G4	G3	G2	G1	G0	B4	B3	B2	B1	B0
	红色 0xF800					绿色 0x07E0						蓝色 0x001F				

一旦设定好，除非重新设定，就会一直保存下来，直到 HMI 硬件断电复位后恢复默认值。

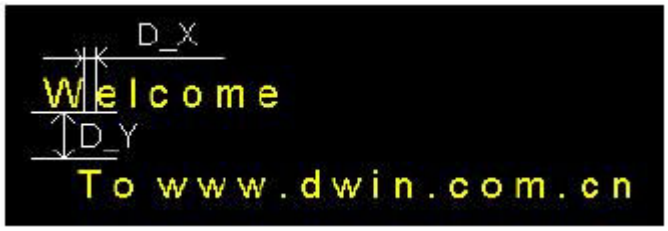
3.3 设置字符显示间距（0x41）

Tx：AA 41 <D_X> <D_Y> CC 33 C3 3C

Rx：无

▲ <D_X> X 方向的字符间距（列间距），取值范围 0x00-0x3F,复位默认值是 0x00。

▲ <D_Y> Y 方向的字符间距（列间距），取值范围 0x00-0x1F,复位默认值是 0x00。



一旦设定好，除非重新设定，就会一直保存下来，直到 HMI 硬件断电复位后恢复默认值。

3.4 取指定位置颜色（0x42，0x43）

Tx：AA <CMD> <X> <Y> CC 33 C3 3C

Rx：无

▲ <CMD> 0x42 为取指定位置颜色到背景色调色板；0x43 为取颜色到前景色调色板。

▲ <X> <Y> 指定位置的坐标（2 节字节表示）。

举例：

AA 42 00 20 02 00 CC 33 C3 3C

取 x=32 (0x0020) y=512(0x0200)位置的颜色到背景色调色板。

3.5 光标显示（0x44）

Tx：AA 44 <Cursor_EN> <X> <Y> <Cursor_Width> <Cursor_Height> <Cursor_Blink_En> <Blink_Time> CC 33 C3 3C

Rx：无

▲ <Cursor_EN>

0x01 光标显示打开，光标将在（x,y）位置显示；

0x00 光标显示关闭。

▲ <X> <Y> 是字符位置，光标在其右下角。

▲ <Cursor_Width> 是显示光标的宽度，取值范围 0x01-0x0F；

▲ <Cursor_Height> 是显示光标的高度，取值范围 0x01-0x0F。

▲ <Cursor_Blink_En> 光标闪烁打开关闭，0x01 打开否则关闭。

▲ <Blink_Time> 光标闪烁周期，取值范围 0x01-0xFF。

当禁止光标显示时(Cursor_EN=0x00),指令中的其它参数再设置就无任何作用。

3.6 文本显示（0x54, 0x55, 0x6F）

3.6.1 文字显示（0x54, 0x55, 0x6F）

Tx：AA <CMD> <X> <Y> <String> CC 33 C3 3C

Rx：无

▲ <CMD>

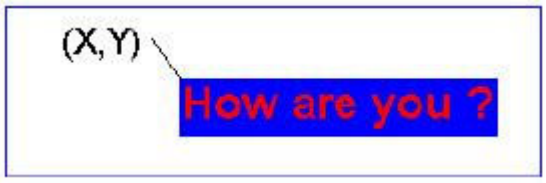
0x54 显示 16*16 点阵 GB2312 的汉字字符串；

0x55 显示 32*32 点阵 GB2312 的汉字字符串。

0x6F 显示 24*24 点阵 GB2312 的汉字字符串

▲ <X> <Y> 显示字符串的起始位置（第一个字符左上角坐标位置）

▲ <String> 要显示的字符串，显示颜色由 0x40 指令设定，显示字符间距由 0x41 设置，遇到行末会自动换行。



举例：

AA 55 00 80 00 30 48 6F 77 20 61 72 65 20 79 6F 75 20 3F CC 33 C3 3C

从(128,48)位置开始显示字符串”How are you?”。

3.6.2 文字旋转, 放大, 通透, 全屏水平, 垂直旋转（0x64，0x65）

Tx：AA 64 < (Enlarge_Status), (Transtatus),(Largx) > CC 33 C3 3C

Rx：无

▲ Enlarge_Status：是否放大标志用 Largx 设置放大倍数,0：不放大,1：放大。

▲ Transtatus：通透标志,让文字背景色和背景图片相同，0：不通透，1：通透。

▲ Largx：放大倍数，范围：01—04，01 表示不放大。

举例：

AA 64 01 01 02 CC 33 C3 3C

文字放大 2 倍，通透显示，使用该指令的方法是，先发送该指令，然后再发送文字显示指令（54，55，6f）指令，通透模式下文字的背景色会被现有背景过滤，在该模式下不适合用在同一位置写文字的方式来更新文字，因为原有文字会被当作背景而产生叠加效果，这里用户需要注意。

Tx：AA 65 <(V_Status),(H_Status) > CC 33 C3 3C

Rx：无

▲ V_Status：整幅图是否垂直旋转 180 度标志,0:不旋转;1：旋转。

▲ H_Status：整幅图是否水平旋转 180 度标志,0:不旋转;1：旋转。

举例：

AA 65 01 00 CC 33 C3 3C

全图沿垂直方向旋转 180 度。

3.7 点显示（0x50，0x51，0x74）

3.7.1 设置当前调色板（0x50，0x51）

Tx：AA <CMD> <(x0,y0) (x1,y1) ……(xi,yi)> CC 33 C3 3C

Rx：无

▲ <CMD>

0x50 背景色显示点；

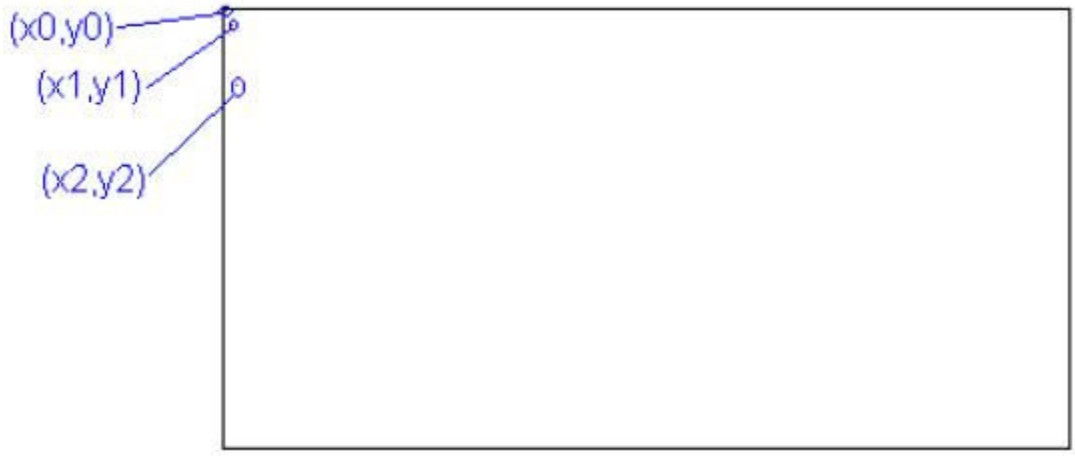
0x51 前景色显示点。

▲ <（x0, y0）(x1, y1) ……(xi, yi)>要显示的点坐标，一帧串口数据最多显示 128 个点。

举例：

AA 51 00 00 00 00 00 03 00 06 00 05 00 20 CC 33 C3 3C

以前景色显示 3 个点，坐标位置为(0, 0), (3, 6), (5, 32), 00 00 00 00 ：（0, 0）；00 03 00 00 05 00 20：（5，32）显示结果如下：



画点显示示意图

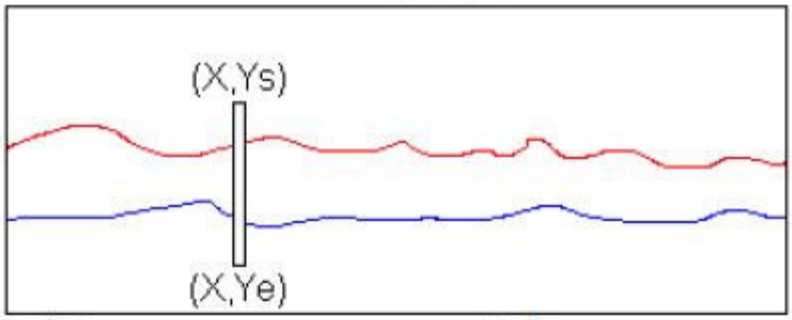
3.7.2 动态曲线显示（0x74）

Tx：AA 74 <X> <Ys><Ye><Bcolor><(Y0,Fcolor0), (Y1,Fcolor1) ……(Yi, Fcolori)> CC 33 C3 3C

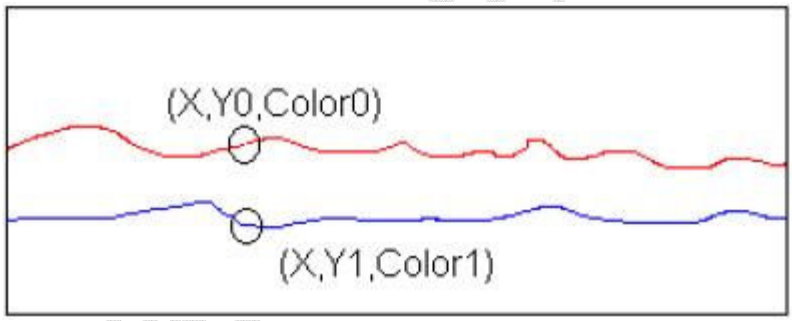
Rx：无

本条指令主要用来方便用户在一个视窗中快速显示多条变化（动态）的曲线，终端按照下面的数据来处理指令：

第 1 步：用<Bcolor>颜色擦除从(X,Ys) 到(X,Ye)的垂直线，把原来的显示内容清空：



第 2 步：在（X,Yi）位置用<Fcolori>颜色置点。



本指令并不会改变用户预设的调色板属性！

3.8 连线显示（0x56，0x5D，0x75，0x76，0x78）

3.8.1 指定点连线（0x56，0x5D）

Tx：AA <CMD> <(x0,y0), (x1,y1) ……(xi, yi)> CC 33 C3 3C

Rx：无

▲ <CMD>

0x56： 用前景色（0x40 指令设置）把指定点用线段连接；

0x5D： 用背景色（0x40 指令设置）把指定点用线段连接；

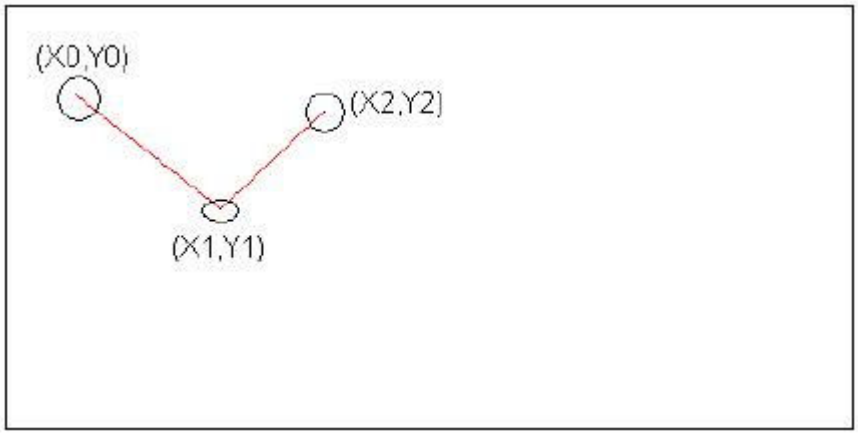
▲ <（x0, y0）(x1, y1) ……(xi, yi)>是连线点的坐标。

举例：

AA 56 00 28 00 32 00 78 00 70 00 B1 00 3A CC 33 C3 3C

用前景色把 3 个点（40，50），（120，112），（177，58）连线，00 28 00 32 为（40，50）坐标。00 78 (

70 为（120，112）坐标。00 B1 00 3A 为（177,58）坐标。显示结果如下：



3.8.2 频谱显示（0x75）

Tx：AA 75 <(x,y)>, <H_max>, <H0……Hi> CC 33 C3 3C

Rx：无

▲ <(x,y)> x 为频谱的 X 轴起点坐标，每显示一根频谱线后，x=x+1； y 为频谱的水平基准位置；每根谱线的 Y 轴起始和终止坐标分别为 y 和(y-Hi)。

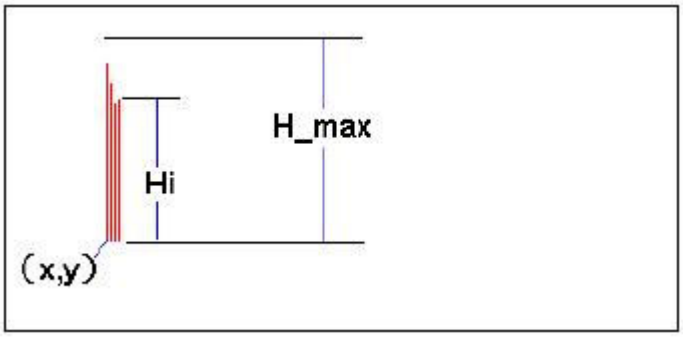
▲ <H_max>谱线的最大高度，

如果 H_max=0x01-0xFF,则谱线高度 Hi 也是 1 字节的变量；

如果 H_max=0x00,则后续两字节为 Hmax,Hi 为两字节的变量。

▲ <H0……Hi>是单根谱线的高度，1 字节或者两字节。

显示谱线颜色由 0x40 调色板设定，显示谱线时，谱线（Hi 高度）会以前景色显示，空余谱线（H_max-Hi）会以背景色(擦除)显示。



3.8.3 折线图显示（0x76）

Tx：AA 76 <x>, <x_dis>, <Y0……Yi> CC 33 C3 3C

Rx：无

▲<x> 折线图的 X 轴起点坐标，每连线一点后，x=x+x_dis;

▲ <x_dis>x 坐标的增量;

▲ <Y0……Yi>折线图的顶点坐标，使用前景色连线显示。

本指令的功能同 0x56 基本相似，只是 X 坐标为 HMI 自动计算，提高了连线速度。

3.8.4 按照偏移量连线（0x78）

Tx：AA 78 <x,y>, <dx0,dy0>, <dx1,dy1>,……<Yi> CC 33 C3 3C

Rx：无

▲<x,y> 连线的起点坐标;

▲ <dxn,dyn>1 字节的 x,y 偏移量，最高位(.7)为符号位，“1”表示负;

本指令用来绘制多线段，用户确定起点后续线段只要输入 X,Y 坐标偏移量就可以自动绘制线段。

3. 9 圆弧曲线显示（0x57）

3.9.1 圆弧或圆域显示（0x57）

Tx：AA 57 (<Type_0>, <x_0>,<y_0>,<R_0>……(<Type_i><X_i><Y_i><R_i>) CC 33 C3 3C

Rx：无

▲ <Type_i> 格式控制

0x01 前景色显示（0x40 指令设定）指定的圆弧;

0x05 背景色显示（0x40 指令设定）指定的圆弧（清除该圆弧）;

0x03 用前景色（0x40 指令设定）填充指定的圆域。

0x06 用背景色（0x40 指令设定）填充指定的圆域。

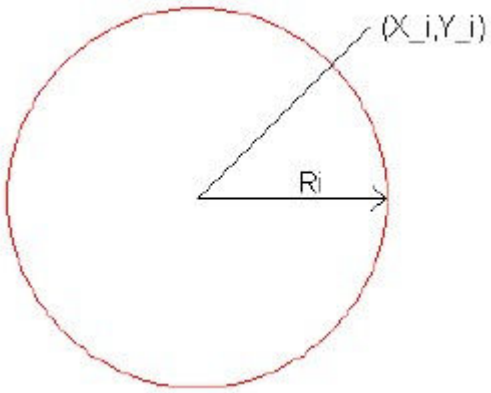
▲ <X_i> <Y_i> 圆弧或圆域的圆心坐标;

▲ <R_i> 圆弧或圆心的半径，0x01-0xFF。

举例：

AA 57 01 00 40 00 50 30 CC 33 C3 3C

用前景色显示一个圆弧，圆心是（64，80），半径是 48。显示结果如下：



3. 10 区域显示

3.10.1 矩形框或矩形区域显示（0x59, 0x69, 0x5A, 0x5B, 0x5C）

Tx：AA <CMD> (<Xs_0> <Ys_0> <Xe_0> <Ye_0>) ……(<Xs_i> <Ys_i> <Xe_i> <Ye_i>) CC 33 C3 3C

Rx：无

▲ <CMD>

0x59 以前景色（0x40 指令设置）显示矩形框，显示线宽是一个点阵；

0x69 以背景色（0x40 指令设置）显示矩形框，显示线宽是一个点阵；

0x5A 以背景色（0x40 指令设置）填充矩形区域；

0x5B 以前景色（0x40 指令设置）填充矩形区域；

0x5C 把指定的矩形区域反色显示（XOR,0xFF 操作），再次反色将复原。

▲ <Xs_i > <Ys_i> <Xe_i> <Ye-i> (Xs_i,Ys_i) 是矩形框或矩形域的左上角坐标，(Xe_i,Ye_i)是矩形框或矩形域的右下角坐标。

举例：

AA 5C 00 40 00 40 00 80 00 80 CC 33 C3 3C

左上角坐标(64,64)和右下角坐标(128,128)定义的矩形区域反色，指令执行后效果如下：

3.11 全屏清屏(0x52)

Tx：AA 52 CC 33 C3 3C

Rx：无

使用背景色(0x40 指令设定) 把全屏填充(清屏)。

3.12 指定区域平移(0x60，0x61，0x62，0x63)

Tx：AA <CMD> (<Xs_0> <Ys_0> <Xe_0> <Ye_0> <N_0>)……(<Xs_i> <Ys_i> <Xe_i> <Ye_i> <N_i>) CC 33 C3 3C

Rx：无

- ▲ <Type_i> 格式控制
  - 0x60 水平卷动；
  - 0x61 垂直卷动；
  - 0x62 水平卷动，最右边区域用背景色填充。
  - 0x63 垂直卷动，最下边区域用背景色填充。
- ▲ <X_i> <Y_i> <Xe_i> <Ye_i> 选择区域的左上角和右下角坐标。
- ▲ <N_i> 移动区域点阵数，水平：0x01—0xFF;垂直：0x01—0xFF；

3.13 图片或剪切图片显示(0x70，0x71)

3.13.1 图片显示（0x70）

Tx：AA 70 <Pic_ID> CC 33 C3 3C

Rx：无

<Pic_ID>保存在 HMI Flash 存储器的图片索引 ID(对应 0xE2 指令),两字节表示。

举例：

AA 70 00 00 CC 33 C3 3C 显示保存在 HMI 中的第 0 幅图片。

AA 70 01 00 CC 33 C3 3C 显示保存在 HMI 中的第 256 幅图片。

3.13.2 剪切图片显示（0x71）

Tx：AA 71 <Pic_ID> <Xs> <Ys> <Xe> <Ye> <X> <Y> CC 33 C3 3C

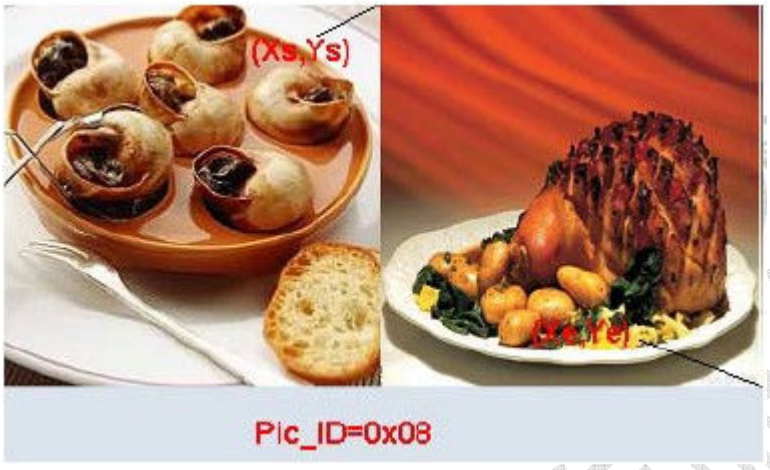
Rx：无

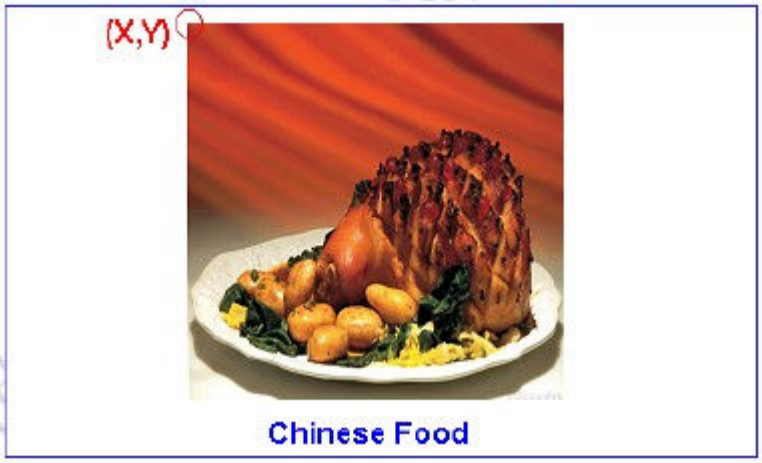
- ▲ <Pic_ID> 保存在 HMI Flash 存储器的图片索引 ID，两字节表示。
- ▲ <Xs> <Ys> <Xe > <Ye > (Xs, Ys) 是要剪切区域在原来图片的左上角坐标，(Xe, Ye)是右下角坐标。
- ▲ <X> <Y> (X, Y) 是剪切下来的图片在当前屏幕显示位置的左上角坐标。

举例：

AA 71 00 08 01 90 00 00 03 1F 01 90 00 C8 00 14 CC 33 C3 3C

把第 8 幅图片的(400,0) (799,400)的区域剪切下来，并显示到当前屏的(200,20)位置，结果如下：





3. 14 背光亮度控制(0x5E，0x5F)

3.14.1 背光关闭（0x5E）

Tx：AA 5E CC 33 C3 3C

Rx：无

3.14.2 打开背光到最大亮度（0x5F）

Tx：AA 5F CC 33 C3 3C

Rx：无

3.14.3 调节背光亮度（0x5F）

Tx：AA 5F <PWM_T> CC 33 C3 3C

Rx：无

▲ <PWM_T> 背光亮度 PWM 控制设定值，取值 0x00-0x3F,0x00 亮度最低，0x3F 亮度最高。

3.15 触摸屏操作(0x72，0x73，0x78，0x79)

3.15.1 触摸位置自动上传（0x72，0x73）

HMI 设置为非自动模式下时,当按压触摸屏时，HMI 将自动以如下格式上传触摸位置坐标。

Tx：无

Rx：AA 73 <X> <Y> CC 33 C3 3C 当按压触摸屏时上传，1 次或多次（可由 0xE0 指令设置。）

AA 72 <X> <Y> CC 33 C3 3C 当离开触摸屏时上传，仅 1 次（可由 0xE0 指令设置。）

▲ <X> <Y> 触摸位置坐标，与屏幕分辨率对应。

3.15.2 触摸位置自动上传（0x78，0x79）

HMI 设置为自动模式即用户启用了触控，键控界面处理功能(0xE0 指令，上位机设置)，并启用了触控键码回传功能，则当点击有效的触控区域时，HMI 会自动上传用户预先设置的 2 字节触控键码（0x1E,0x1B 配置文件定义）。

Tx：无

Rx：AA 78（79） <Touch_Code> CC 33 C3 3C

注：0x78 对应触摸屏被按压时(0x73)，0x79 对应触摸屏抬起时(0x72)；自锁按钮按下时对应 78，弹起时对应 79.

3.16 工作模式配置(0xE0)

Tx：AA E0 <Bode_Set> <Para1> CC 33 C3 3C

Rx：AA E0 0D <BaudRate> < Para1 >CC 33 C3 3C

▲ <Bode_Set>设置串口通信波特率，设置如下（新雁飞屏出厂默认是 0xFF）：

Bode_Set	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0xFF
波特率	1200	2400	4800	9600	19200	38400	57600	921600	115200

▲ <Para1>配置触摸屏处理模式，设置如下：

Para1	Bit_description
.7	0= 点击触摸屏后，松开触摸时，自动上传 0x72 指令。 1= 点击触摸屏后，离开触摸时，不上传 0x72 指令。
.6	0= 点击触摸屏后，会以 100mS 的间隔定时自动上传 0x73 指令,直到触摸屏松开。

	1= 点击触摸屏后，自动上传 0x73 指令。
.5	0= 点击触摸屏后，HMI 不进行触控界面的切换： 1= 点击触摸屏后，HMI 自动按照下载的配置文件进行触控界面的切换。
.3	蜂鸣器自动伴音开关： 0= 关闭蜂鸣器自动伴音，在自动模式下，点击有效区域时不鸣叫。 1= 打开蜂鸣器自动伴音，在自动模式下，点击有效区域时鸣叫。
.2	.
.1	.
.0	.

3.17 蜂鸣器控制(0x79)

Tx：AA 79 <On_Time> CC 33 C3 3C

Rx：无

▲ <On_Time>0x01-0xFF，蜂鸣器鸣叫时间长度，单位为 10mS。

蜂鸣器鸣叫指定时间。

3.18 睡眠模式控制(0xE7)

Tx：AA E7 <Slee_Set> CC 33 C3 3C

Rx：无

▲ <Sleep_Set>当 Sleep_Set 为 1 时，LCD 进入睡眠模式，此时模块功耗很低，当点击触摸时候唤醒模块，唤醒屏回送 aa,e7,ff,ea,cc,33,c3,3c 指令，提示用户屏被唤醒，注意：第一次点击是唤醒模块，后面的触摸动作才是有效的。

3.19 指令开关睡眠模式(0xE8)

Tx：AA E8 <Slee_Set> CC 33 C3 3C

Rx：无

<Sleep_Set>当 Sleep_Set 为 1 时，LCD 进入睡眠模式，Sleep_Set 为 0 时,LCD 退出睡眠模式，并向用户发送 aa,e8,ff,ea,cc,33,c3,3c 指示屏已经退出睡眠模式

3.20 查询当前图片编号(0xE9)

Tx：AA E9 <无> CC 33 C3 3C

Rx：AA E9 PicID CC 33 C3 3C

<PicID>返回当前图片编号，2 个字节，高字节在前，低字节在后，主机通过发送该指令来确定画面是否切换成功。

3.21 控件操作指令(0x80)

控件是新加入的功能，目的是为了更好的方便用户开发，控件主要包含：编辑框控件，进度条控件，滑动条控件，勾选框控件，按钮控件。这里主要介绍如何对控件进行读写。

3.21.1 编辑框控件的读写

编辑框示意图

编辑框控件是重要的用户输入输出窗口，用户通过该控件可以方便的读写（输入输出），如数据的实时记录展示等，关于对编辑控件的操作我们分两部分解释，首先是对编辑框写入操作，此操作可通过 3 种方式对其写入，1：按钮控件写入，用户通过点击按钮将按钮控件值输入给编辑框，按钮控件值和其对应写入的编辑框都是通过上位机进行设置；2：编辑框自带键盘输入，为了方便用户，我们为编辑框植入两种键盘，点击编辑框即可弹出键盘对其进行输入操作；3：指令输入，即通过指令对其输入，我们紧接着就是介绍指令的格式和其对应的意思，而对编辑框当前数值的读操作则只能通过指令，方法为发送指令告诉 HMI 需要读写的编辑控件，然后编辑控件发送当前值给用户，用户根据返回值处理数据。

写控件指令格式如下：

Tx：AA 80 <PicID>< ControlType>< Control_ID> <WriteorRead> <Writetype> <data0><data1>… CC 33 C3 3C

Rx：无

<PicID>当前图片编号，2 个字节，高字节在前，低字节在后，每个控件都有其所在的图片，当控件所在图片编号不是当前所在的图片时，控件不会马上显示数据，而只有在图片刷新的时候才会显示出来。例如：当前图片是 02，而用户指令输入 aa 80 00 01 后控件数据不会显示，只有在图片重新刷新的情况下才会显示。

< ControlType>控件的类型，1 代表的是编辑框控件，1 个字节。

< Control_ID>控件的编号，此参数同上位机设置的编号相同，2 个字节。

<WriteorRead>控件读写标志，1 个字节，此参数代表对控件的操作是读还是写，1：写操作，0：读操作

<Writetype>当前一个参数（WriteorRead）设置为写入时，此参数代表写入的方式，我们提供两种方式写入，如果该参数为 1 则是一串数据写入，0 则是在原来数据上增加一个字符。 两种方式各有用处，用户可以根据需要来选择。

<data0,1…>后续参数是用户要输入的数据，数据是以标准 ASCII 码来表示。

例如：AA 80 00 01 01 00 05 01 31 32 33 CC 33 C3 3C ，该控件编号为 5，其所在图片编号为 1，对其写入 123(31,32,33 是 1，2，3 对应的 ASCII 16 进制度码)这 3 个字符，写入方式为 1 串字符。

读编辑框控件指令格式如下：

Tx：AA 80 <PicID> < ControlType> < Control_ID> <WriteorRead> CC 33 C3 3C

Rx：AA 80 < ControlType> <PicID> <ControlID><Data0><Data1>…CC 33 C3 3C

< ControlType>控件的类型，1 代表的是编辑框控件，1 个字节。

<PicID>当前图片编号，2 个字节，高字节在前，低字节在后，该参数是当前图片的编号。

< Control_ID>控件的编号，此参数同上位机设置的编号相同，2 个字节。

<Data0,Data1..>用户通过上位机设置控件是返回控件里的字符串对应的 ASCII 码或者是其对应的数值，如果是返回十进制数值的话，则整数和小数部分各占 4 个字节。

例如：用户输入 AA 80 00 01 01 00 04 00 CC 33 C3 3C

HMI 返回 AA 80 01 00 01 00 04 <编辑框里的数据> CC 33 C3 3C

3.21.2 进度条控件的读写

进度条示意图

进度条控件是被设计用来表达某事进行状态的工具，该控件被广泛应用在各种工程设计中。用户既可以通过触摸屏改变其数值大小，也可通过指令改变数值来改变其当前进度，指令格式如下：

Tx：AA 80 <PicID>< ControlType>< Control_ID> <WriteorRead> <Writetype> <data0><data1><data2> <data3> CC 33 C3 3C

Rx：无

<PicID>当前图片编号，2 个字节，高字节在前，低字节在后，只有控件所在的图片在当前图片时可以直接写控件。

< ControlType>控件的类型，2 代表的是进度条控件，1 个字节。

< Control_ID>控件的编号，此参数同上位机设置的编号相同，2 个字节。

<WriteorRead>控件读写标志，1 个字节，此参数代表对控件的操作是读或者写，1：写操作，0：读操作

<data1>--<data4>给控件赋值，4 个字节的有符号，高字节在前，低字节在后，赋值范围-9999~~+9999，进度条本身的极值范围用户可通过上位机进行设置，例如用户将进度条设置为-20--90 为该进度条的范围，则使用指令输入-20 则进度条为空进度，输入 90 为满进度。

例如：用户输入 AA 80 00 02 02 00 03 00 00 00 20 CC 33 C3 3C，对进度条编号为 3，其所在图片编号是 2，发送 32 的数据，假如进度条范围为 0---100，则进度到 32%位置。

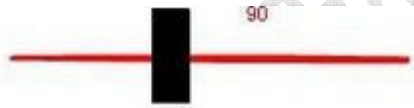
读进度条控件指令格式如下：

Tx：AA 80 <PicID> < ControlType> < Control_ID> <WriteorRead> <Read_type> CC 33 C3 3C

Rx：AA 80 < ControlType> <PicID> <ControlID><Data0><Data1>…CC 33 C3 3C

＜ControlType>控件的类型，2 代表的是进度条控件，1 个字节。
<PicID>当前图片编号，2 个字节，高字节在前，低字节在后，只有控件所在的图片在当前图片时可以直接读进度条的数值。
<WriteorRead>控件读写标志，1 个字节，此参数代表对控件的操作是读或者写，1：写操作，0：读操作
<Read_type>当<writeorRead>标志为读时，该参数有效，参数为 1：返回的是百分比；0：数值。
<Data0><Data1>..返回的数值，当用户设置<Read_type>为 1 时，返回一个字节，为 0 时返回四个字节，高位在前，低位在后。
例如：用户输入 AA 80 00 03 02 00 01 00 00 CC 33 C3 3C，读取编号为 1，所在图片编号为 3 的控件值。
HMI 返回 AA 80 02 00 03 00 01 <进度条当前值> CC 33 C3 3C

3.21.3 滑动条控件的读写



滑动条示意图

滑动条控件可以通过其滑动来形象的表达数值大小的改变，滑动条的范围以经固定为 0~100,用户既可以通过触摸屏改变其数值大小，也可通过指令改变数值大小，其写指令格式如下：
Tx：AA 80 <PicID>< ControlType>< Control_ID> <WriteorRead> <Writetype> <data0><data1> CC 33 C3 3C
Rx：无
<PicID>当前图片编号，2 个字节，高字节在前，低字节在后，只有控件所在的图片在当前图片时可以直接写控件。
＜ControlType>控件的类型，3 代表的是滑动条控件，1 个字节。
＜Control_ID>控件的编号，此参数同上位机设置的编号相同，2 个字节。
<WriteorRead>控件读写标志，1 个字节，此参数代表对控件的操作是读还是写，1：写操作，0：读操作
<data1><data2>给控件赋值，2 个字节，高字节在前，低字节在后，赋值范围 0~100。
例如：用户输入 AA 80 00 02 03 00 05 01 00 10 CC 33 C3 3C 则表示对控件编号为 5，其所在图片编号为 2 的滑动条输入数值 16。

读滑动条控件指令格式如下：
Tx：AA 80 <PicID> ＜ControlType> ＜Control_ID> <WriteorRead> CC 33 C3 3C
Rx：AA 80 ＜ControlType> <PicID> <ControlID><Data0><Data1>CC 33 C3 3C

＜ControlType>控件的类型，3 代表的是滑动条控件，1 个字节。
<PicID>当前图片编号，2 个字节，高字节在前，低字节在后，只有控件所在的图片在当前图片时可以直接读滑动条的数值。
<WriteorRead>控件读写标志，1 个字节，此参数代表对控件的操作是读还是写，1：写操作，0：读操作
<Data0><Data1>返回的数值，返回的是两个字节，高位在前，低位在后。

3.21.4 钩选框控件的读写

钩选控件示意图

钩选框控件可以形象的表达某项目参数是否选择，其值只用 0 或者 1 表达,其写指令格式如下：
Tx：AA 80 <PicID>< ControlType>< Control_ID> <WriteorRead><data0><data1> CC 33 C3 3C
Rx：无
<PicID>当前图片编号，2 个字节，高字节在前，低字节在后，只有控件所在的图片在当前图片时可以直接写控件。
＜ControlType>控件的类型，4 代表的是钩选框控件，1 个字节。
＜Control_ID>控件的编号，此参数同上位机设置的编号相同，2 个字节。
<WriteorRead>控件读写标志，1 个字节，此参数代表对控件的操作是读还是写，1：写操作，0：读操作
<data1><data2>给控件赋值，2 个字节，高字节在前，低字节在后，其值只能事 0 或者 1，1：选中，0：没有选中。

读钩选框控件指令格式如下：
Tx：AA 80 <PicID> ＜ControlType> ＜Control_ID> <WriteorRead> CC 33 C3 3C
Rx：AA 80 ＜ControlType> <PicID> <ControlID><Data0><Data1>CC 33 C3 3C
＜ControlType>控件的类型，4 代表的是钩选框控件，1 个字节。
<PicID>当前图片编号，2 个字节，高字节在前，低字节在后，只有控件所在的图片在当前图片时可以直接读钩选框的数值。
<WriteorRead>控件读写标志，1 个字节，此参数代表对控件的操作是读还是写，1：写操作，0：读操作
<Data0><Data1>返回的数值，返回的是两个字节，高位在前，低位在后。

3.21.5 按钮控件的读写

按钮控件是触摸屏功能的基本功能体现，用户通过上位机设置好按钮即可通过点击触摸屏来实现其基本功能，我们提供 3 种按钮控件用以用户操作，1：切换按钮，用以实现画面的自动切换；2：自锁按钮(上位机里的名字是点动按钮)，点击后按钮按下并不弹起，只有再次点击时才弹起；3：局部切换按钮，点击后会将任意张图片的局部显示在现在显示的画面上，其包含的按钮同样有效。与其他控件一样，用户同样可

以用指令读或写按钮的当前状态，其写指令格式如下：

Tx：AA 80 <PicID>< ControlType>< Control_ID> <WriteorRead> <data>CC 33 C3 3C
Rx：无
<PicID>当前图片编号，2 个字节，高字节在前，低字节在后，只有控件所在的图片在当前图片时可以直接写控件。
< ControlType>控件的类型，0 代表的是按钮控件，1 个字节。
< Control_ID>控件的编号，此参数同上位机设置的编号相同，2 个字节。
<WriteorRead>控件读写标志，1 个字节，此参数代表对控件的操作是读还是写，1：写操作，0：读操作
<data>给控件赋值，1 个字节，赋值 1 代表按钮按下，0 代表按钮弹起。

读按钮控件指令格式如下：
Tx：AA 80 <PicID> < ControlType> < Control_ID> <WriteorRead> CC 33 C3 3C
Rx：AA 80 < ControlType> <PicID> <ControlID><Data>CC 33 C3 3C
< ControlType>控件的类型，0 代表的是按钮控件，1 个字节。
<PicID>当前图片编号，2 个字节，高字节在前，低字节在后，只有控件所在的图片在当前图片时可以直接读按钮当前的状态。
<WriteorRead>控件读写标志，1 个字节，此参数代表对控件的操作是读还是写，1：写操作，0：读操作
<Data>返回的数值，返回的是 1 字节，1 代表弹起，0 代表按下。

3.22 控件详细说明

上面已经对各种控件的读写操作指令做了说明，此部分对部分控件进行了更加详细的说明。

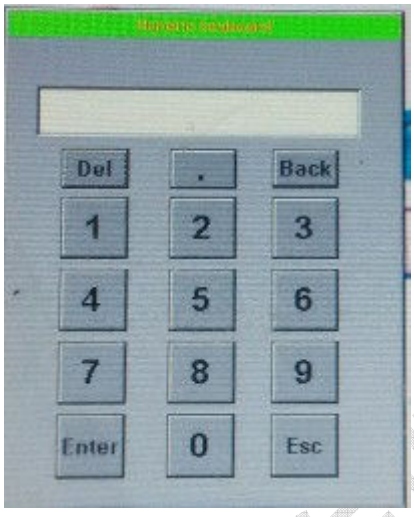
3.22.1 编辑框控件

所有的控件都由上位机生成，操作方法在上位机使用说明中详细介绍，编辑框控件可以作为用户重要的输入输出窗口，它主要是为用户显示数据和字符，一共能显示 12 个字符，用户既可以通过指令将数据写入，也可以用 HMI 自带键盘实现输入，而用户想读出编辑框控件的值则只能通过指令读取。编辑框控件通过上位机来确定其类型，其有以下几种类型：

（1） 普通编辑框，点击该区域后既不弹出数字键盘也不弹出字母键盘，仅仅通过指令来对其进行读写操作。

（2） 普通密码编辑框，不弹出字母键盘，指令输入其在控件上显示的都是”*” ，但是用指令可以将数据读出，应当注意的是读出的都是 ascii 码值，另外密码编辑框不支持 10 进制数据返回。

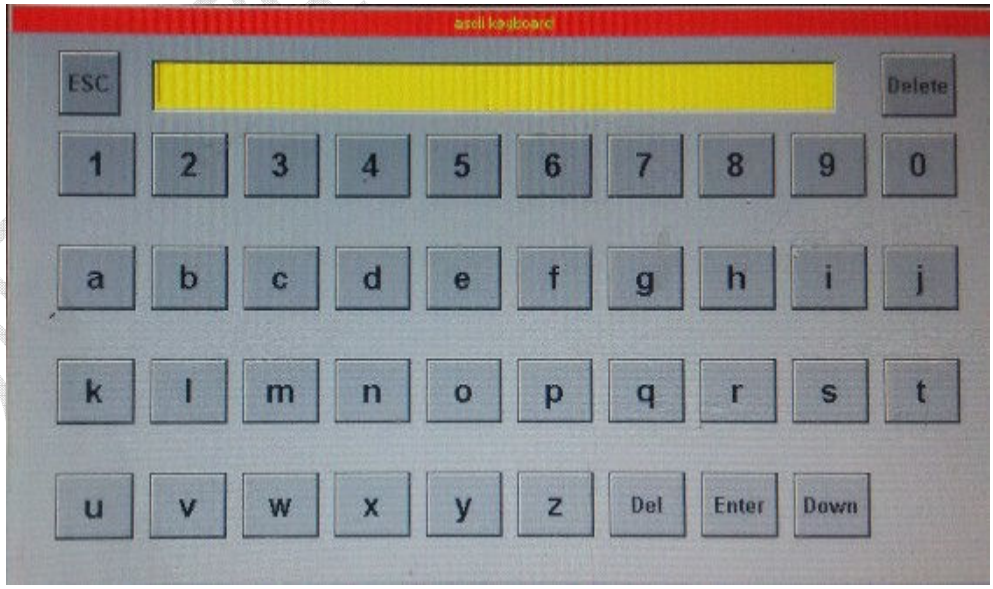
（3） 数字键盘编辑框，点击后弹出数字键盘，可以通过数字键盘往控件里写入数据，同时也可通过指令对其进行读写，键盘如下图所示：



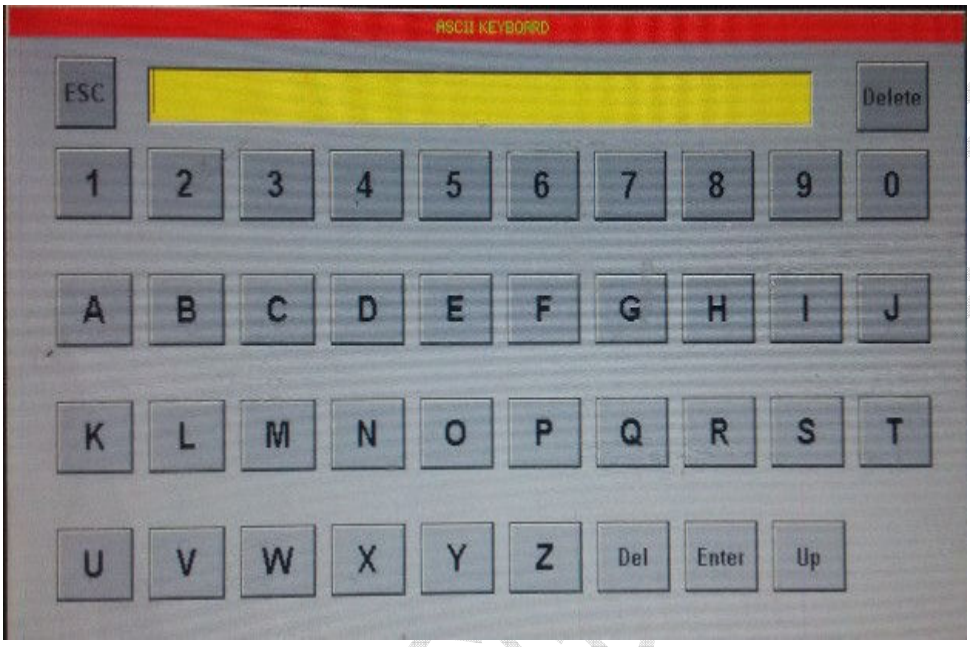
数字键盘

注：“Del”为将所有输入清除，“Back”为清除前一个输入字符，“Enter”为将已经输入的数据输入到指定的编辑框中（通过上位机指定），“Esc”关闭数字键盘。

（4） 字母键盘编辑框，点击后弹出字母键盘，和数字键盘一样，既也可以通过字母键盘往控件里写入数据，同时也可通过指令对其进行读写，键盘如下图所示：



小写字母键盘



大写字母键盘

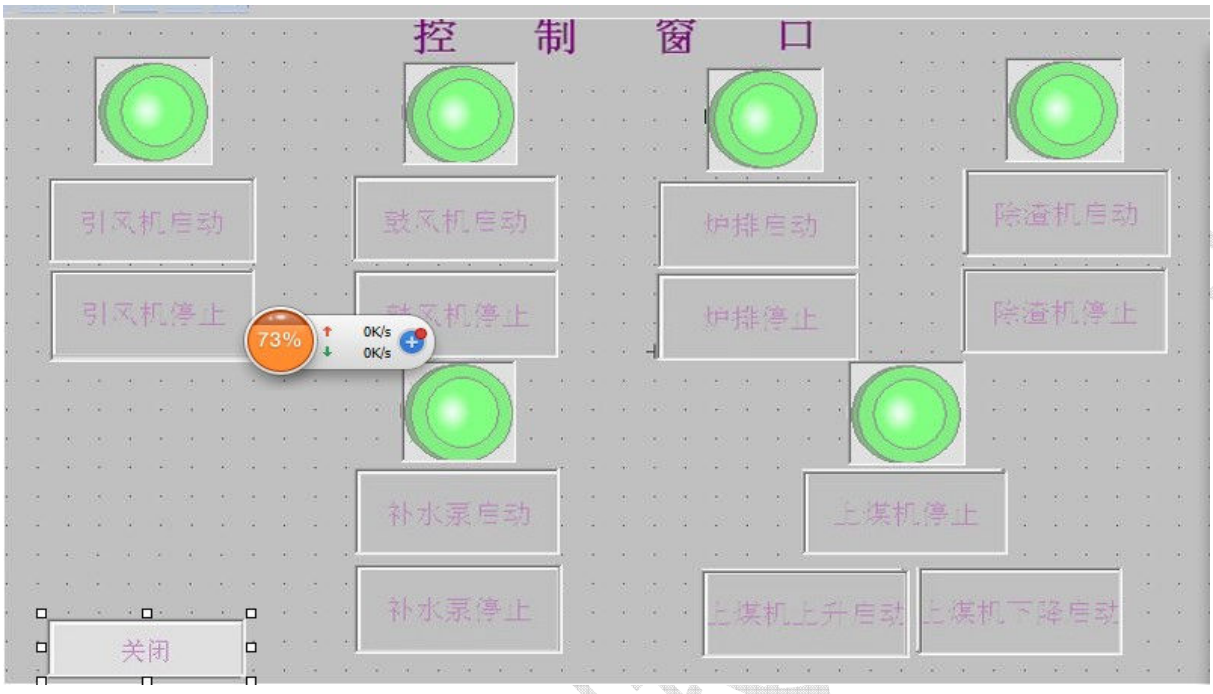
注：“ESC”是关闭键盘，“Delete”是删除前一个输入字符，“Del”是清除之前输入的字符，“Enter”是将字符输入到对应的编辑框里（上位机设置对应哪个编辑框），“Down”“Up”是大小写字符切换，点击后字符会在大小写之间随意切换。另外需要注意的是，当弹出键盘（数字或者字母键盘）时只能点击键盘上的按键，其他按钮都是无效的。

（5） 字母键盘密码编辑框，点击后也弹出字母键盘，和字母键盘编辑框不同的是，当通过键盘或者指令往编辑框里输入数据时，显示出来的都是“*”，可以且只能通过指令将数据读出来。

同过上位机用户可以设置编辑框的背景颜色，其自身的编号，字体的大小，颜色，读取编辑框内容时返回的是 asc 码还是 10 进制的数据，编辑框的内容即使图片切换后依然会保留。下图是编辑框在上位机上对应的属性栏。

背景颜色：编辑框的背景色；前景颜色：字体显示的颜色；弹出位置：这个属性只有在选择有键盘弹出时有效，并指明弹出的位置；位置，高度，宽度：分别代表了编辑框所在位置以及本身的大小；小数点数：在返回类型为 10 进制数时有效，并确定小数的位数；当前图片编号：指控件所在图片的编号；字体类型：选择字体大小；组件类型：根据客户需求选择编辑框为上面文字介绍的任意一种。

3.22.2 按钮控件



按钮普通（弹起）状态下的图片素材

按钮按下状态的图片素材

为了让用户做出更加多样的按钮，按钮控件是通过图片的方式来制作的，如上图，两副图基本相同，区别在于按钮部分的颜色，因为另一幅图表达的是按钮按下时的状态，当然在不需要按下状态时一副图片就可

以了。切换按钮的属性栏目如下：

文档设置中文	
Button类型	切换画面按钮
编辑控件编号	0
初始化状态	未点击
触控键码	1
当前图片编号	3
动画图片编号	2
高度	41
宽度	132
目标矩形区域	0, 0, 0, 0
目标图片编号	0
位置	29, 433
粘帖本区域	0, 0
组件编号	80

切换按钮属性设置栏

前面图片部分的“关闭”按钮即是我们说的切换按钮，它的属性栏目主要由触控键码，按钮所在图片编号，动画图片编号（按钮按下后的图片编号），目标图片编号以及组建编号组合而成。用户设置好参数后，点击触摸部分画面按照目标图片自动切换，这个就是切换按钮。

自锁（点动）按钮属性栏如下：

文档设置中文	
Button类型	点动按钮
编辑控件编号	0
初始化状态	未点击
触控键码	10
当前图片编号	3
动画图片编号	2
高度	77
宽度	79
目标矩形区域	0, 0, 0, 0
目标图片编号	-1
位置	60, 36
粘帖本区域	0, 0
组件编号	74

自锁(点动)按钮属性栏

自锁按钮与切换按钮的区别在于点击后并不切换画面，所以目标图片编号为-1，-1 在上位机中即代表不切换画面，点击按钮后按钮按下，手离开触摸后按钮不弹起，再次点击按钮，此时按钮处于按下状态，点击后按钮弹起。

局部剪切按钮是按钮点击后剪切某幅画面的在当前画面相同位置显示，此按钮可以被用来弹出用户自己制

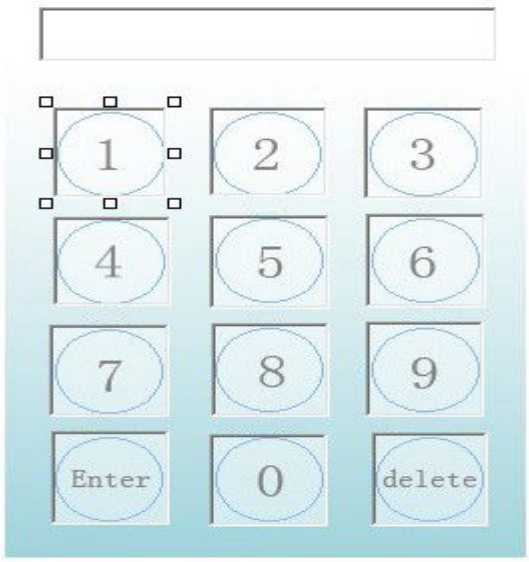
作的键盘，窗口等，其属性栏如下：

Button类型	局部剪切按钮
编辑控件编号	0
初始化状态	未点击
触控键码	30
当前图片编号	0
动画图片编号	-1
高度	24
宽度	49
目标矩形区域	5, 91, 794, 391
目标图片编号	38
位置	545, 223
粘帖本区域	5, 91
组件编号	71

局部剪切按钮属性栏

局部剪切按钮的属性栏区别前两者（切换按钮和自锁按钮），点击后同样不会切换，触控键码，当前图片编号，目标图片编号设置与前两者相同（上图的动画图片编号设置为-1，因为该按钮没有制作动画图片），与前两种按钮最大的区别在于目标图片编号这里对应的是需要剪切显示的图片，目标矩形区域是用户框选的区域，如我们点击某按钮，弹出一个用户自制的键盘，用这个按钮就可以轻易实现。

ASC 码图片按钮如下图所示，我们需要做一个键盘，点击后向图片上的编辑框控件中写入相应的数据，这时就要用到 ASC 码按钮了，键盘如下图：



键盘示意图

我们以按钮“1”为例，看到其属性栏目如此设置，如下图：

Button类型	asc码图片按钮
编辑控件编号	7
初始化状态	未点击
触控键码	49
当前图片编号	36
动画图片编号	37
高度	58
宽度	57
目标矩形区域	0, 0, 0, 0
目标图片编号	-1
位置	74, 106
粘帖本区域	0, 0
组件编号	17

ASC 按钮属性栏示意图

该属性同切换按钮大致相同，区别在于需要设置编辑控件编号，该属性对应的是按钮点击后往哪个编辑框输入数据，触控键码我们设置为 49，因为数字“1”的 ASCII 码十进制表达就是 49，因为点击后不切换画面，所以目标图片设置为-1，点击该按钮后即可往对应的编辑框里输入“1”这个数字了。

4 修订记录

日期	修订记录	内核版本	文档版本
2012.8.12	首次发布	---	Ver1.0
2013.7.4	增加 e8,e9 指令,修改 e7 指令	---	Ver1.1
2014.6.12	增加 80 控件指令，修改部分指令，改为 V1.2	---	Ver1.2

有关新雁飞 HMI 的最新资料，欢迎访问我们的网站：

www.xinyanfeitech.com