目　录

1

关于 InoTouch Editor 编程软件的安装

第一章　关于 InoTouch Editor 编程软件的安装

1.1　InoTouch 系列 HMI 和 InoTouch Editor 软件的简介

1.1.1　InoTouch 系列 HMI 的简介

InoTouch 系列 HMI 可以分为标准配置产品、网络型产品配备以太网口；有的还配备音频输出；IT5104/IT5121 除了配备以太网口、音频输出外，还可选择配置视频输入等。

命名规则如下：

IT 5 070 T X
① ② ③ ④ ⑤

① 公司产品信息　　IT：汇川触摸屏（InoTouch 的缩写）
② 系列号　　5：5000 系列 HMI
③ 屏幕尺寸　　070：7 寸 HMI；100：10 寸 HMI
④ 辅助特征　　T：标准配置；　E：带以太网接口
⑤ 衍生版本号

举例说明：IT5100T，表示 IT5000 系列标准配置产品，10 寸 LCD；IT5104E，表示 IT5000 系列网络型产品配置有以太网口，10.4 寸 LCD。具体产品规格见附录。

1.1.2　InoTouch Editor 软件的简介

InoTouch Editor 是汇川技术 InoTouch 系列 HMI 编程组态软件，采用 Windows Visual Studio 风格，功能强大，简单易用具有以下特点：

1) 支持 65536 色真彩显示；

2) 支持 windows 平台矢量字体，文字大小可以自由缩放；

3) 支持 BMP，JPG，GIF 等格式的图片；

4) 支持 USB 设备，例如 U 盘、USB 鼠标、USB 键盘、USB 打印机等；

5) 支持历史数据、故障报警等，可以保存到 U 盘或者 SD 卡里面；

6) 支持 U 盘、USB 线和以太网等不同方式对 HMI 画面程序进行下载；

7) 支持配方功能，并且可以使用 U 盘等来保存和更新配方，容量更大；

8) 支持三组串口同时连接不同协议的设备，应用更加灵活方便；

9)、支持自定义启动 Logo 的功能；

10) 支持市场上绝大多数的 PLC 和控制器、伺服、变频器、温控表等，也可以为您特殊的控制器开发驱动程序

11) 支持离线仿真和在线仿真功能，极大的方便了程序的调试；

在介绍 InoTouch Editor 软件之前，先介绍一下 InoTouch Editor 软件提供的各控件的功能，稍后的章节将详细的介绍各控件的功能是如何实现的。

图 标	控件名称	功能描述
	指示灯	使用图形或者文字等显示 PLC 中某一个位的状态。
	多状态指示灯	根据 PLC 中数据寄存器不同的数据，显示不同的文字或者图片。
	位状态设定	在屏幕上定义了一个触摸控件，触摸时，对 PLC 中的位进行置位或者复位。
	多状态设定	在屏幕上定义了一个触摸控件，触摸时，可以对 PLC 中的寄存器设定一个常数或者递加递减等功能。
	切换开关	在屏幕上定义了一个触摸控件，当 PLC 中的某一个位改变时，它的图形也会改变；当触摸时，会改变另外一个位的状态。
	多状态切换开关	在屏幕上定义了一个多状态的触摸控件，当 PLC 的数据寄存器数值改变时，它的图形会跟着变化；触摸时，会改变 PLC 中数据寄存器的值。
	数值显示	显示 PLC 中数据寄存器的数值。

	数值输入	显示 PLC 中数据寄存器的数据，使用数字键盘可以修改这个数值。
	字符显示	显示 PLC 寄存器中的 ASCII 字符。
	字符输入	显示 PLC 寄存器中的 ASCII 字符，使用字母键盘可以修改这个 ASCII 字符。
	直接窗口	在屏幕上定义了一个区域，当定义的 PLC 中的位为 ON 状态时，指定编号的画面会显示在该区域。
	间接窗口	在屏幕上定义了一个区域，当定义的 PLC 数据寄存器的数据与某个画面的编号相等时，该画面会显示在该区域。
	项目选单	在屏幕上定义了一个下拉式菜单，触摸时，可以选择不同的项目，从而将不同的数据写入到 PLC 中。
	滑动开关	在屏幕上定义了一个滑动触摸控件，当手指滑动该控件时，会线性改变 PLC 中数据寄存器的数值。
	功能键	在屏幕上定义了一个功能键，可以执行画面跳转、执行宏指令等功能。
	移动图形	该控件会随着 PLC 中数值寄存器数值的改变而改变图形的状态和在屏幕上的位置。
	动画	该控件会随着 PLC 中数值寄存器数值的改变而改变图形的状态和在屏幕的位置，该位置是事先已经设定好的。
	表针	使用表针图形来显示 PLC 中数据寄存器数据的动态变化。
	棒图	使用棒状图形来显示 PLC 中数据寄存器数据的动态变化。

	趋势图	使用多点连线的方式显示 PLC 中一个或者多个数据寄存器中数据变化的趋势或者历史变化趋势。
	XY 曲线显示	PLC 中一组连续的寄存器数据为 X 轴坐标，另一组连续的寄存器的数据为 Y 轴坐标，由这些对应的坐标点连成的曲线。
	历史数据显示	使用表格的方式，显示历史数据。
	数据群组显示	显示由 PLC 中一组连续的数据寄存器中的数据组成的曲线。
	报警条	利用走马灯的方式，显示“事件登录”中的报警信息。
	报警列表	使用文字的方式显示“事件登录”中的故障信息，当故障恢复时，显示的文字消失。
	事件列表	使用文字的方式显示“事件登录”中的故障信息，可以显示故障发生的时间和恢复时间等，故障恢复时，文字不消失。
	触发式资料传输	可以手动或者根据 PLC 中某个位的状态，来执行数据的传送。
	备份	将保存到 HMI 里面的配方数据、资料采样数据或者故障报警信息等复制到指定的 U 盘或远程的计算机。
	PLC 控制	由 PLC 里面的数据寄存器或者某个位来执行指定的功能，例如画面翻页、屏幕打印、执行宏指令等。
	定时式资料传输	指定一个固定的周期，来执行数据传输。
	事件登录	定义故障发生时的文字内容和条件。

	资料取样	定时取样 PLC 的数据并保存到指定的存储器，并用于显示趋势图和历史数据显示等。
	系统信息	客户可以自定义这些由 HMI 系统本身显示的一些提示信息。
	排程	定义一个指定的时间，改变 PLC 中的一个位的状态或者改变 PLC 中某个寄存器的数据。

1.2 安装 InoTouch Editor 编程软件

1.2.1 软件来源

InoTouch Editor 编程软件由汇川控制技术有限公司自主开发的，请向您的 HMI 供应商索取，或者在深圳汇川技术网站上：http://www.inovance.cn 下载，也可以在中国工控网汇川主题上下载，获取最新软件 InoTouch Editor。

1.2.2 计算机配置要求（建议配置）

CPU：主频 1G 以上的 Intel 或 AMD 产品

内存：512MB 或以上

硬盘：最少有 500MB 以上的空闲磁盘空间

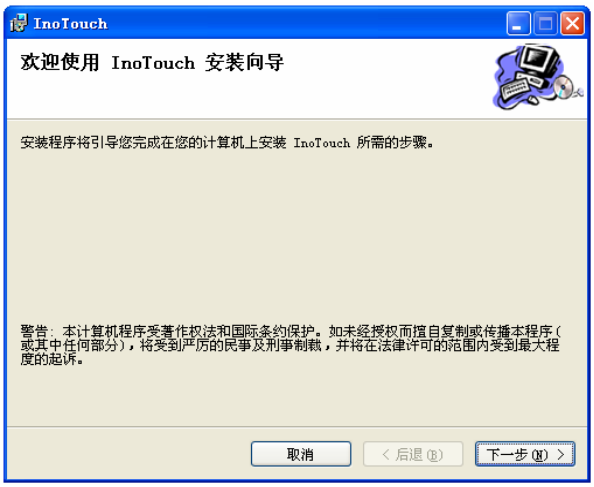
显示器：支持分辨率 1024 x 768 以上的彩色显示器

Ethernet 端口或 USB 口：上下载画面程序时使用

操作系统：Windows XP/Windows Vista/Windows 7/ Windows 2000

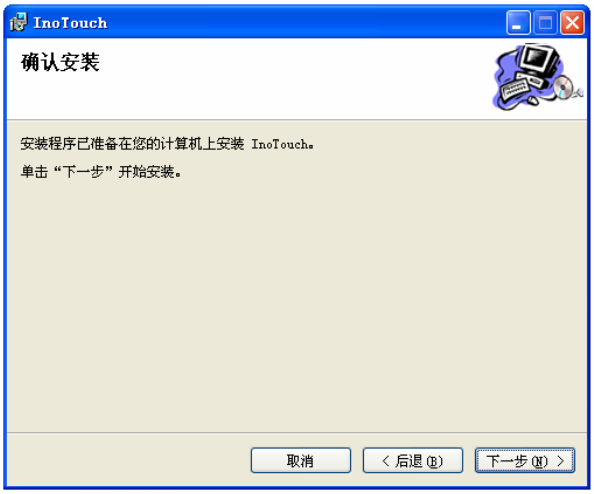
1.2.3 安装步骤

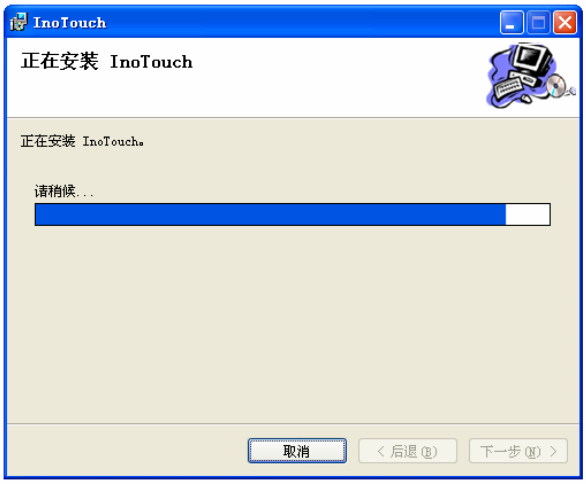
1） 将软件下载到电脑里，解压之后，点击文件内的 〔setup.exe〕文件，屏幕将显示安装窗口如下，此时根据指导提示，点选“下一步”：：

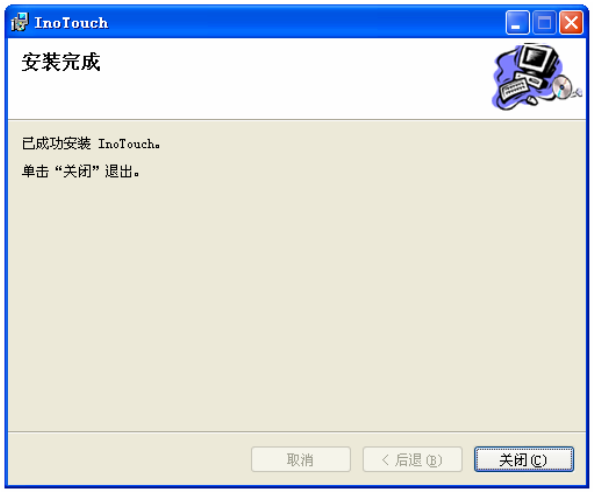


2） 选择软件安装的文件夹或选择默认路径，并点选“下一步”：

3） 根据指导提示，点选“下一步”确认安装，在安装完成后点选“关闭”即完成安装程序。







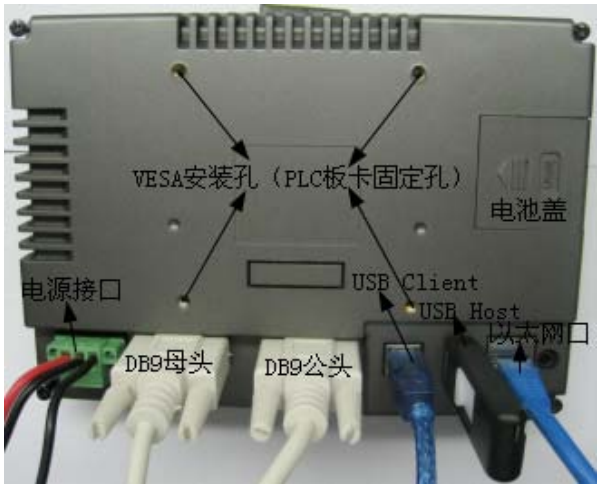
4） 要执行程序时，可以点击桌面上的  ；或者可以从菜单〔开始〕/〔程序〕/〔Inovance Control〕下找到相对应的执行程序即可。

软件目录下各选项的含义如下：

InoTouch Editor	编程软件
InoTouch Simulator	模拟器
ReleaseNotes	版本发布信息
Uninstall	卸载软件
InoTouch Editor 编程软件手册	软件手册

1.3 系统连接图

InoTouch 系列触摸屏系统的连接界面具备如下：



1）与外部设备的连接（DB9 母头/ DB9 公头）

DB9 母头：COM1 [RS485] / COM3 [RS485] / COM3 [RS232]通讯端口 9 针 D 型母座管脚排列图；这个端口用于连接具有“RS485 / RS422/ RS232 ”通讯端口的控制器。

DB9 公头：COM1/2 [RS-232] 通讯端口 9 针 D 型公座管脚排列图；这个端口用于连接具有 RS232 通讯端口的控制器。

2）USB 接口

产品外壳背面的 USB 端口：USB Client（Type B）接口，用于与 PC 连接，进行上载/下载用户组态程序和设置 HMI 系统参数，可以通过一条通用的 USB 通讯电缆和 PC 机连接；USB Host（Type A）接口，用于与 U 盘、USB 鼠标、USB 键盘及 USB 打印机等设备连接，即插即用。

3）Ethernet 以太网连接

产品外壳背面的以太网接口为 10M/100M 自适应以太网端口。端口可以用于 HMI 组态的上/下载，系统参数的设置和组态的在线仿真；可以通过以太网连接多个 HMI 构成多 HMI 联机；可以通过以太网与 PLC 等通讯；可以通过以太网口与 PC 机通讯。

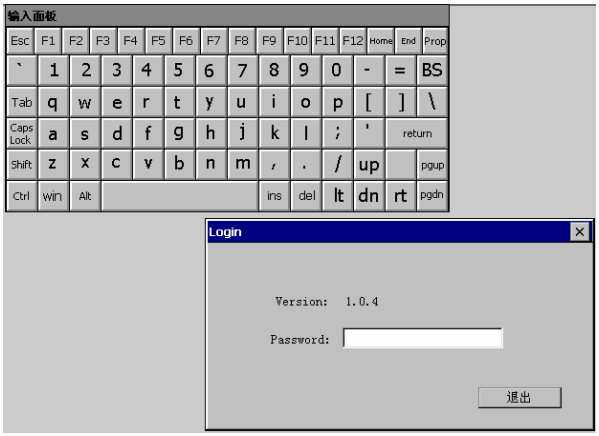
这个端口可以通过一根标准的以太网线（RJ45 直连线）与 HUB 或者以太网交换机相连，接入局域网，也可以通过一根双机互联网线（RJ45 交叉线）直接与 PC 的以太网口连接。

1.4 InoTouch 系列人机界面的系统设定

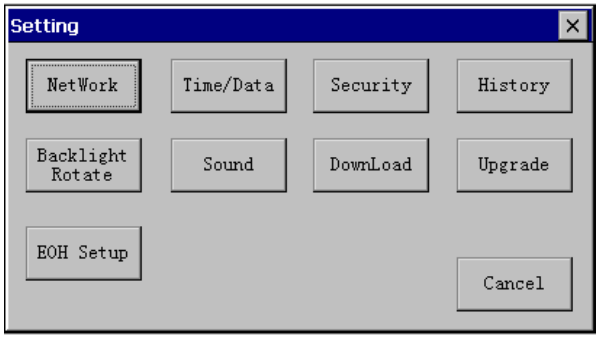
在开始对 InoTouch 系列人机界面编程之前，需要先了解 InoTouch 系列人机界面的系统设定。下面详细的介绍如何设定 InoTouch 系列人机界面的 IP 地址、日期与时间、上传/下载程序的密码、调整 LCD 亮度，以及查看系统的版本号等。

1.4.1 如何进入人机界面的系统

上电时，按压触摸面板不放，系统启动完之后，就会出现系统设置的如下画面：

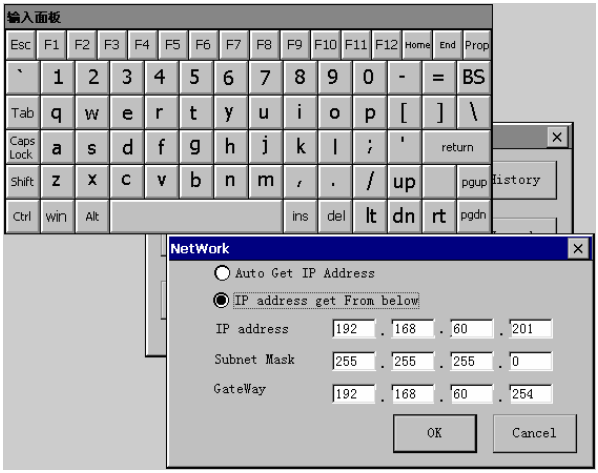


为了系统的安全，进入系统时，是需要输入密码的，输入正确的密码后，会自动进入系统设定画面。默认的密码是 6 个 1（即 111111）。如下图所示：



a、 设定人机界面的 IP 地址

单击 Network 进入人机界面 IP 地址设定界面，如下图：

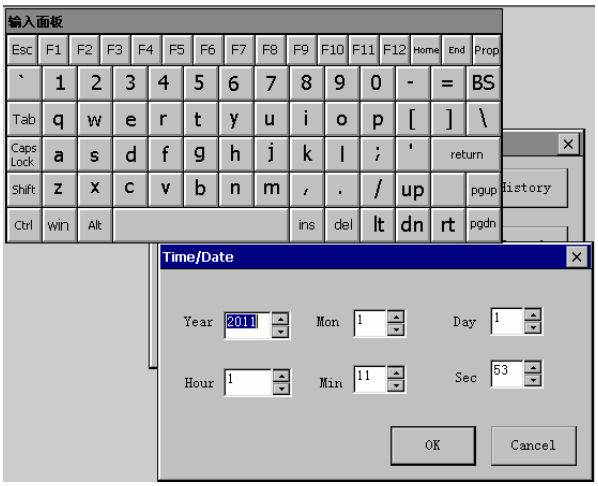


选择“Auto Get IP Address（自动获得 IP 地址）”时，会由局域网的 DHCP 服务器自动分配 IP 地址，此时 InoTouch Editor 人机界面就相当于是该局域网里面的一台计算机，IT5000 人机界面接到计算机所在的局域网时，可以勾选此选项。

选择“IP address get From below（IP 地址设定如下）”时，是手动设定人机界面的 IP 地址。此情况一般适合计算机和人机界面直接连接的情况下。手动设定 IP 地址时，请注意，与人机界面使用网线直接连接的计算机和该人机界面本身，两者都必需是手动设定静态的 IP 地址，且两者的 IP 地址必须是在同一个网段。例如，人机界面的 IP 设定为：192.168.60.201，那么计算机的 IP 地址可以设定为：192.168.60.202 等。

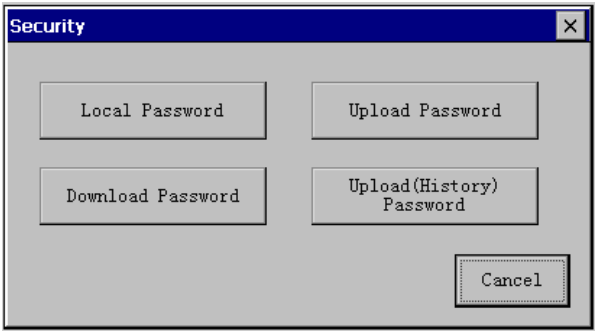
b、 设定“Time/Date(时间/日期)

单击“Time/Date（时间/日期）”，可以对人机界面设定系统时间和日期。



c、 “Security(安全设定)”

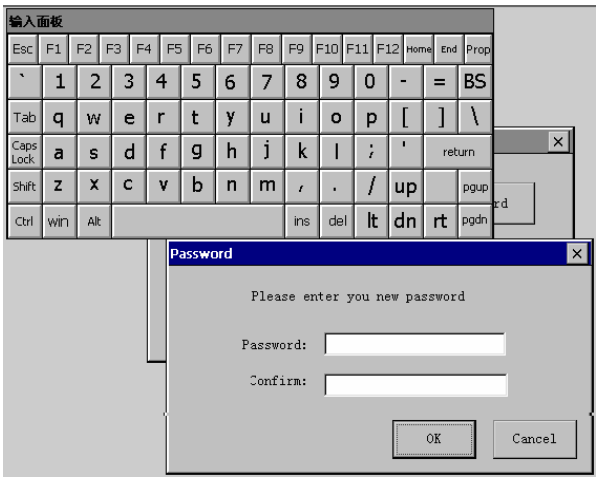
单击“Security（安全设定）”时，可以修改各种密码，如下图：



各密码的含义如下：

Local Password	进入系统设定时的密码
Upload Password	从人机界面上传画面程序时的密码
Download Password	下载人机界面画面程序时的密码
Upload（history）Password	上传保存在人机界面里面的资料取样、报警信息等文件时的密码

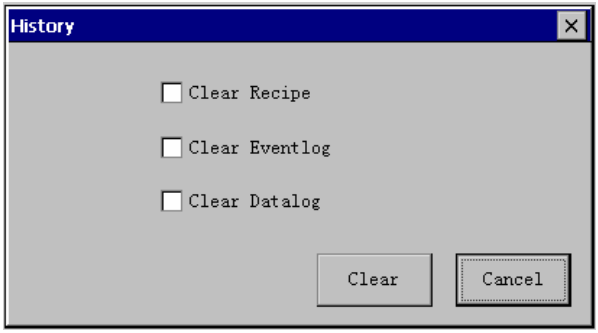
当修改任意密码时，会弹出如下对话框：



输入新的密码后，需要重新输入一次，两次输入的密码一致的话，则修改成功。

d、 “History（历史资料）”的设定：

单击“History”，将会出现如下图所示画面。



此项可以清除保存在人机界面里面的配方、取样资料和历史故障记录等。

Clear Recipe	当勾选此项时，点击清除按钮时，会清除保存在人机界面里面的配方数据。
Clear Eventlog	当勾选此项时，点击清除按钮时，会清除保存在人机界面里面的历史故障记录。
Clear Datalog	当勾选此项时，点击清除按钮时，会清除保存在人机界面里面的资料取样记录。

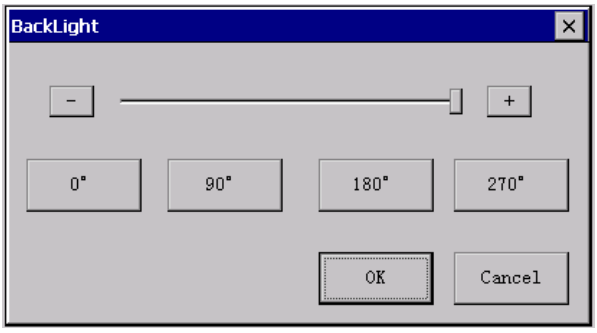
e、“Backlight（背光灯）”的设定

单击“Backlight”，会显示如下对话框，在此可以调整背光的亮度。

当用手指（或者鼠标）往左滑动这个滑块时，LCD 的背光亮度会变暗；当用手指（或鼠标）往右滑动这个滑块时，LCD 的背光亮度会变亮。

或者单击“-”，LCD 的背光亮度会变暗；单击“+” LCD 的背光亮度会变亮。

单击页面上的“0° ”、“90° ”、“180° ”和“270° ”可以让屏幕旋转您选择的角度。

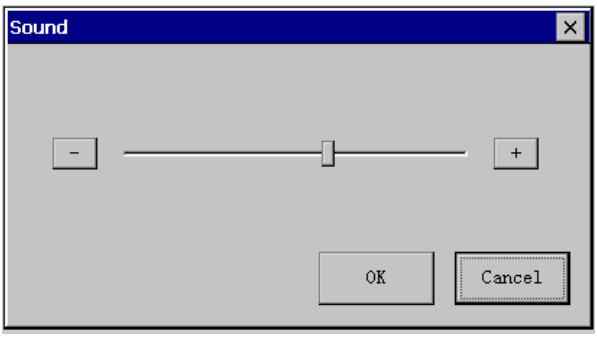


f、“Sound（声音）”的设定

单击“Sound”，会显示如下对话框，在此可以调整声音的大小。

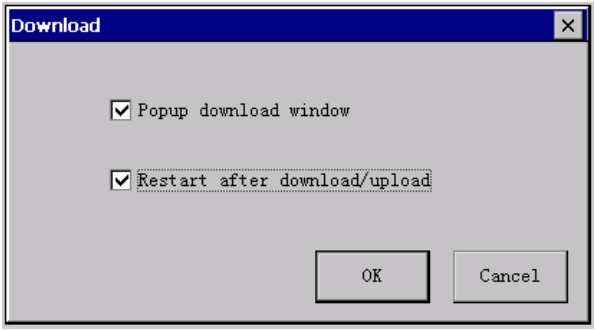
当用手指（或者鼠标）往左滑动这个滑块时，声音会变小；当用手指（或鼠标）往右滑动这个滑块时，声音会变大。

或者单击“-”，声音会变小；单击“+”，声音会变大



g、“Download（下载）”

单击“Download”，会显示如下对话框。



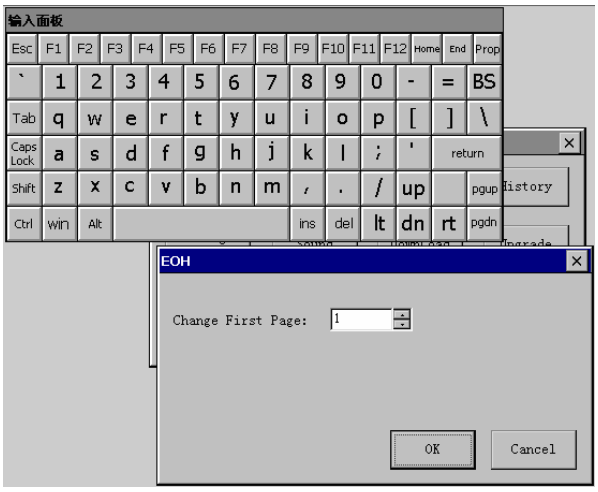
Popup download window	当勾选此项时，插入 SD 卡或 U 盘时自动弹出下载窗口
Restart after download/upload	当勾选此项时，U 盘或 SD 卡下载/上传完成时自动重新启动

h、“Update Firmware（更新固件）”

单击“Update firmware”，会显示如下对话框，在此可以使用 U 盘来更新人机界面的核心系统文件。更新方法在稍后的章节详细介绍

i、“EOH Setup（设置工程首显的页面）”

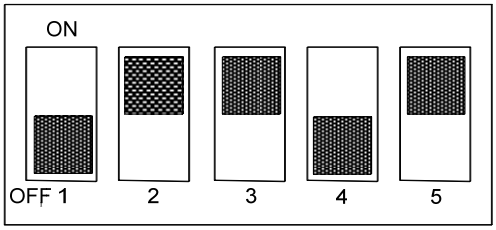
单击“EOH Setup”，会显示如下对话框，可以设置工程运行时首先显示的页面。设置完之后重新上电才生效。



1.4.2 如何将人机界面恢复到系统密码

如果不小心忘记了给人机界面设定的各种密码，例如进入系统的密码，上传下载程序的密码等，此时就无法对人机界面上下载程序了。在此情况下，只要将人机界面恢复到系统密码，则这些密码就会恢复为系统统一密码 111111，即六个 1。同时也恢复上传/下载密码为 000000，即六个 0。操作如下：

a、 将触摸屏背面的五个拨码开关，第 2 个、第 3 个和第 5 个设置为 ON，其余 2 个拨码开关设置为 OFF。如下图：



b、拨完之后请重新上电，会弹出如下对话框，询问是否要恢复为出厂设置密码。

c、请点击“yes（确认）”。

注意：此时单击“YES”，人机界面里面的画面程序和所有保存的资料，例如配方数据、资料取样数据、报警记录等将会全部被清除。

执行上述恢复系统设置的过程后，人机界面的系统密码（即 Local Password）会恢复为 111111，即 6 个 1；上传和下载程序的密码恢复为 000000，即 6 个 0。

2

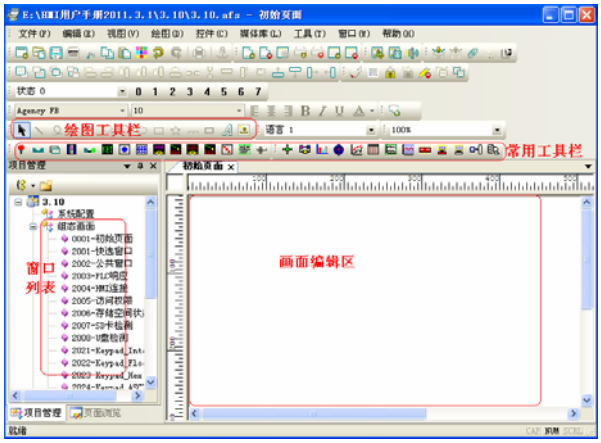
制作一个简单的工程

第二章　制作一个简单的工程

在软件安装完毕之后，要执行程序时，可以点击桌面上的  ；或者可以从菜单〔开始〕/〔程序〕/〔Inovance Control〕下找到相对应的执行程序即可。如下图，单击 InoTouch Editor，就可以进入 InoTouch Editor 软件的编程界面。

在开始编写程序前，先介绍一下 InoTouch Editor 软件的布局。

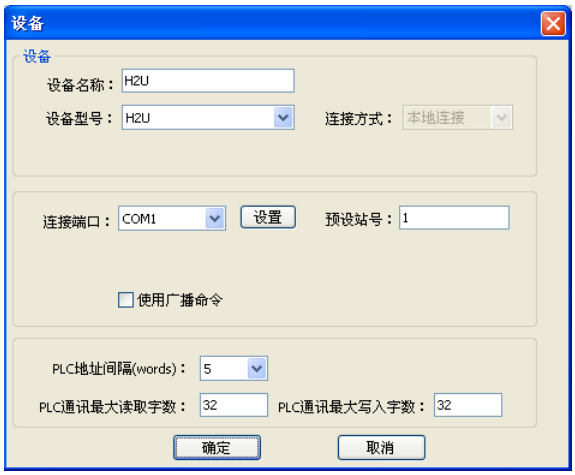
InoTouch Editor 软件打开后的结构布局如下图所示。



下面以连接汇川的 PLC 为例，说明如何制作一个简单的工程。

步骤一：首先按下工具栏上“新建工程”按钮，如下图：

编写工程名称，选择保存工程的路径，选择 HMI 型号以及屏幕类型等，设置完之后，再按下“确定”键后，将会弹出如下对话框：

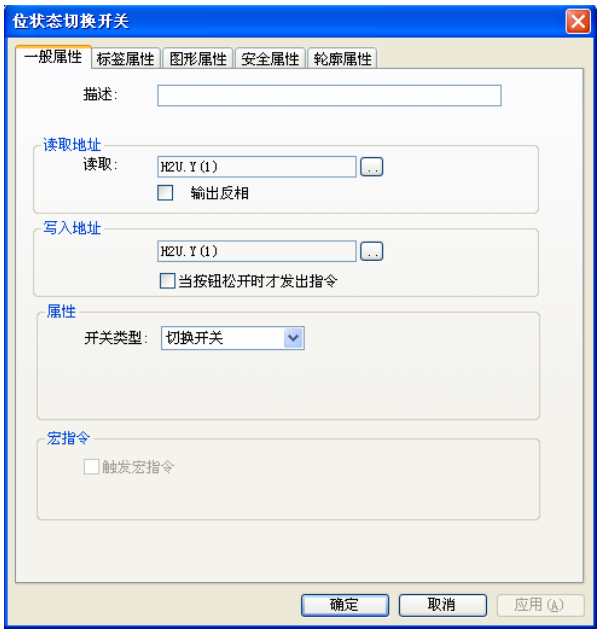


步骤二：连接汇川的 PLC H2U，如果通讯参数设置的跟 PLC 里面的通讯参数不一致的话，则单击“设置”功能即可进入修改通讯参数的界面，如下图所示。

从以上图片可以看出，通讯参数是 9600 波特率、7 位数据位、1 位停止位、偶检验，使用的是人机界面的 COM1 RS485 4W 的方式连接到汇川 PLC。

步骤三：要增加一个[位状态切换开关]控件，可按下如下图所示的控件按钮。在窗口中点击鼠标左键，就建立了“位状态切换开关”控件，如下图所示。

选择“位状态切换开关”双击或单击鼠标右键选择“属性”进行编辑，如下图：

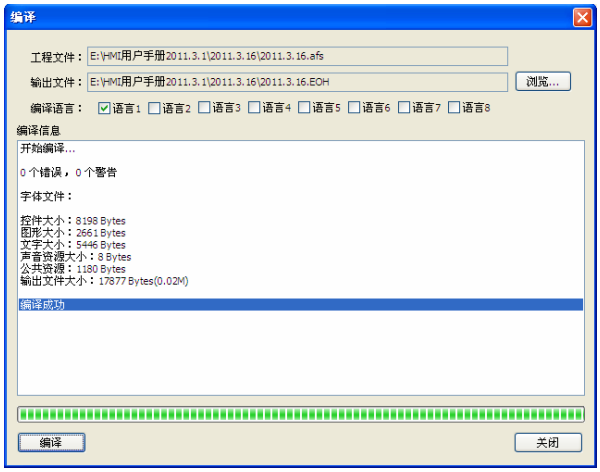


最后“初始画面”将如下图所示。

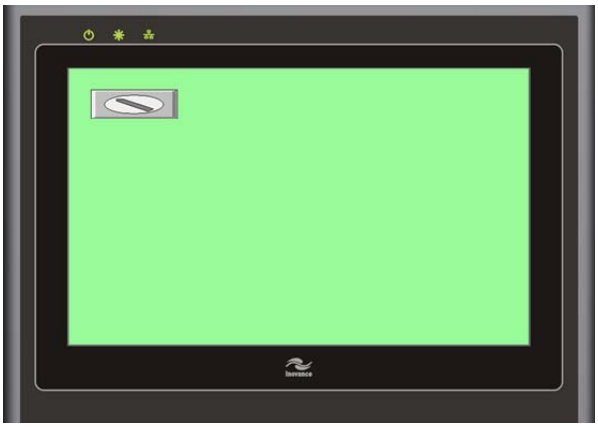
步骤四：单击  图标，保存文件。InoTouch Editor 软件编辑生成的工程文件名后缀为 afs。

存盘完成后使用者可以使用编译功能，检查画面规划是否正确，编译功能的执行按钮为  ，编译后生成的输出文件名后缀为 EOH。

假如编译结果如下图所示，并不存在任何错误，即可执行离线仿真功能。



步骤五：点击工具栏上离线仿真按钮  ，执行后画面如下图所示：

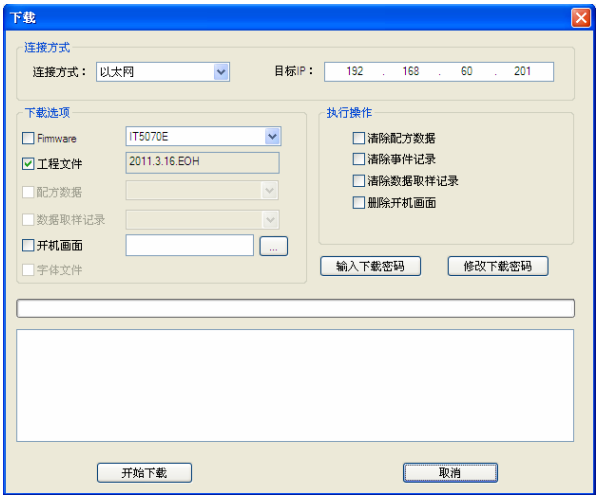


如需进行在线仿真，在接上设备后点击工具栏上的在线仿真按钮即可进行。

因此，一个简单的应用工程画面就做好了。

步骤六：把画面程序下载到人机界面里面。以以太网下载的方式为例（后面讲述 USB 安装方式及下载方式）。

在前面的章节介绍了人机界面的 IP 地址设定和下载密码设定。假如：InoTouch 系列人机界面设定的 IP 地址为：192.168.60.201，下载密码是默认的 000000（6 个 0）。那么，在下载程序前，先前已经对刚刚做的小程序做了存盘和编译的工作（这个是下载程序前必须做的操作），在此只要执行下载的动作就好。在此需要说明的是，利用网线对 InoTouch 系列人机界面下载程序的时候，计算机的 IP 地址需要与人机界面的 IP 地址在同一个网段，端口号不一样。若 IP 地址表示为（A、B、C、D）的形式，则 A、B、C 要一致，D 不能一致。例如 HMI 的 IP 地址为：192.168.60.201，计算机的 IP 地址为：192.168.60.202。单击下载按钮后，将会出现以下画面：



在 HMI 地址栏里面输入 InoTouch 系列人机界面的 IP 地址，密码框里面输入正确的下载密码，并勾选“Firmware”和“下载后启动程序画面”后，单击“下载”功能键，就可以执行下载程序的动作，

且下载完成后，InoTouch 系列人机界面会自动重新启动刚刚下载的画面程序。有关下载画面的功能，在本章只简单说明一下，在后续的相关章节，将详细介绍各种不同的下载画面程序的方法。

小结：由上所述，制作一个工程画面的基本步骤为：1、选择所使用的机型；2、选择所连接的 PLC，并设定好与 PLC 的通讯参数和连接的串口；3、利用软件提供的各种控件编辑画面程序。4、保存和编译文件；5、离线仿真，查看编写画面的布局效果。6、将画面程序下载到人机界面里面。

3

程序的编译、仿真与下载

第三章　程序的编译、仿真与下载

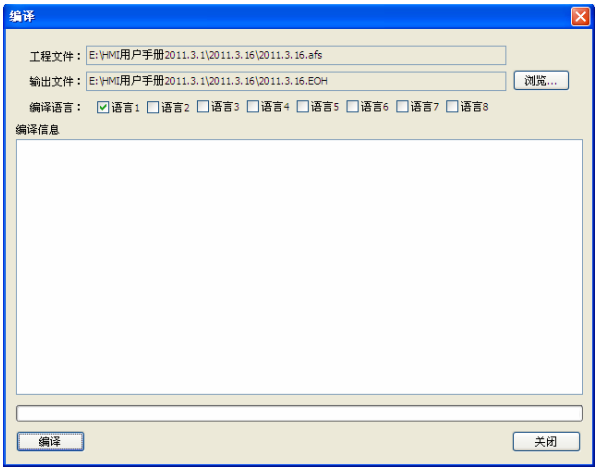
工程画面编辑好之后，一般需要看一下画面编辑的效果，预览一下画面下载到人机界面后的效果，画面的布局、颜色搭配等等。此时就需要使用到离线仿真功能。下面分别来说明一下 InoTouch Editor 工程画面常用的操作方法。

3.1 编译

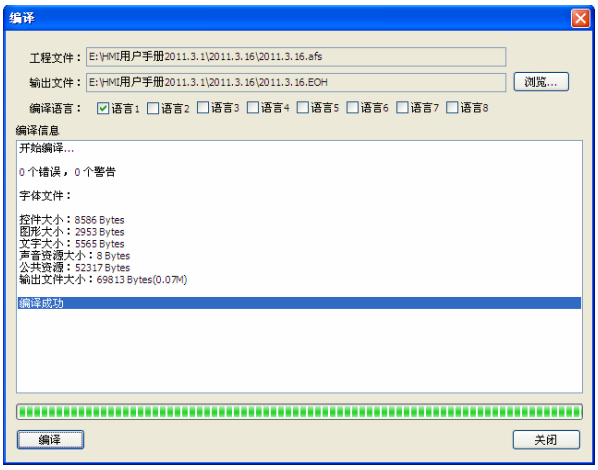
编辑好的画面，在离线仿真或者下载到人机界面之前，均需要进行“编译”这个操作。操作方法为：

a、 单击 InoTouch Editor 软件工具栏上的保存图标或者菜单“文件/保存工程”，将编辑的工程画面先保存起来。工程另存为时，会提示需要输入文件名，输入自己需要的文件名即可。文件名一般以读懂为原则。保存后的文件名后缀均为 afs，例如名字为“包装机.afs”等文件名。

b、 然后单击 InoTouch Editor 软件工具栏上的编译图标或者菜单“工具/编译”或者快捷键 F5，就会弹出“编译”对话框。



c、 单击“编译”，即可对工程进行编译，编译成功后，关闭对话框即可。如下图所示。编译后的文件后缀名均为 EOH，如“包装机.EOH”等。



如果编译有错误，双击错误行可以跳到对应的错误位置。

3.2 仿真

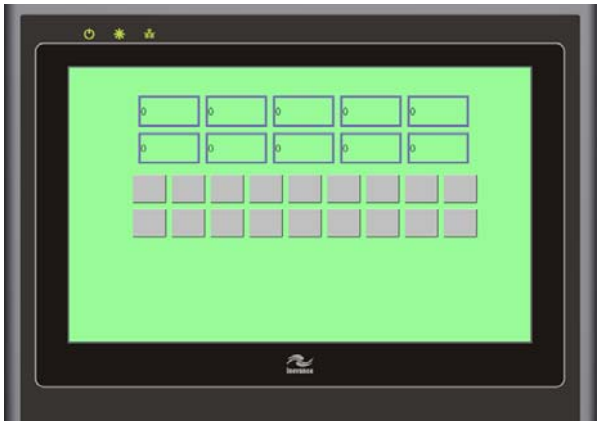
InoTouch Editor 软件提供了离线仿真和在线仿真两种仿真功能。离线仿真的功能是使用计算机来仿真为人机界面，查看编辑的工程画面布局效果等。而在线仿真可以使用计算机仿真为一台人机界面，直接操作所连接的 PLC，方便画面的调试工作，而不需要每次调试的时候将画面程序下载到人机界面里面，实际连接 PLC 来调试。下面分别说明操作步骤。

3.2.1 离线仿真

画面程序编写完，并编译生成了 EOH 文件之后，即可进行离线仿真的操作。单击 InoTouch Editor 菜单“工具/离线仿真”或者工具栏上的图标 ，即可进行离线仿真操作，显示后的画面可参考如下。

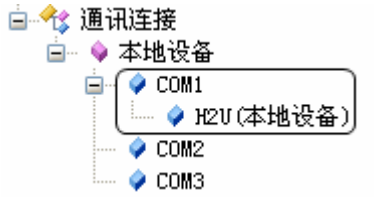
3.2.2 在线仿真

在执行在线仿真之前，先需要将计算机和 PLC 使用串口通讯线连接起来。如果 PLC 的接口是 RS485 接口，由于计算机的串口均为 RS232 口，故需要使用一个 RS485 转 RS232 的转换器。但是一般在执行 InoTouch Editor 软件的“在线仿真”功能时，都是使用 PLC 的下载线来将计算机和 PLC 下载口连接起来，就跟使用计算机给 PLC 下载程序一样的连接。然后单击 InoTouch Editor 软件菜单“工具/在线仿真”或者工具栏上的图标，正常建立连接后，将会显示 PLC 中的数据和位的状态，可参考下图。



在执行在线仿真与 PLC 建立连接时，必须在 InoTouch Editor 工程里面，事先设定好与 PLC 的各项通讯参数，只有当通讯参数设置正确后，才能够保证能够正常的执行“在线仿真”功能。

计算机通讯端口（COM1）



HMI 软件设置通讯连接（COM1）

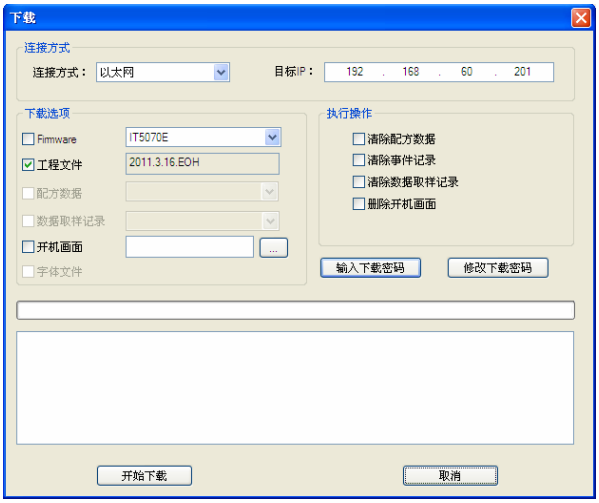
由于计算机模拟为一台人机界面，如果在 HMI 软件设置使用 COM1 口连接 PLC 的话，那么也必须使计算机的 COM1 口将计算机和 PLC 连接起来，否则将无法正确执行“在线仿真”功能。此种情况的“在线仿真”功能每次只能执行 30 分钟，超过 30 分钟后将自动断线，此时只要重新执行一次在线仿真操作，则又可以建立连接。

3.3 下载程序

前面介绍了程序的保存和编译的方法，执行以上步骤后，就可以将程序下载到人机界面里面。下载程序目前有以下几种方法：

3.3.1 使用网线下载程序

使用网线下载程序，需要知道人机界面的 IP 地址和下载密码，如果是计算机直接与人机界面连接，则计算机的 IP 地址必须设置为与人机界面的 IP 地址在同一个网段，端口号不一样。若 IP 地址表示为（A、B、C、D）的形式，则 A、B、C 要一致，D 不能一致。例如 HMI 的 IP 地址为：192.168.60.201，计算机的 IP 地址为：192.168.60.202。下载密码为初始密码 000000（即 6 个 0），单击 InoTouch Editor 软件菜单“工具/下载”或者快捷键 F7 或者工具栏上的图标，将会弹出如下对话框。



在该对话框内设置下载方式为“以太网”，并正确设置“HMI 地址”和下载密码。若是第一次给人机界面下载程序，或是第一次使用更新的软件版本， 请勾选“Firmware”，其他选项可以根据实际需要勾选。设置完之后，单击“下载”，即可将程序下载到人机界面里面。

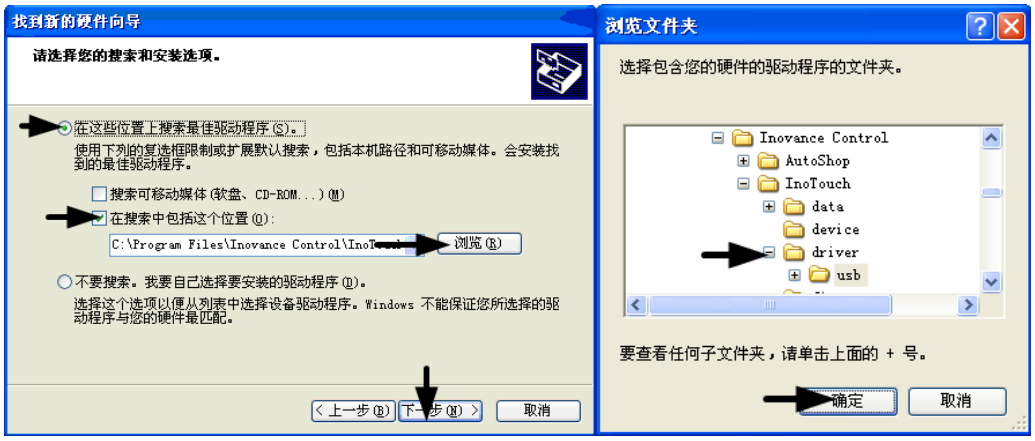
3.3.2 使用 USB 线下载程序

InoTouch 系列人机界面支持使用 USB 线下载程序。由于 USB 线下载程序的速度非常快，为了避免受到干扰，建议采用带有铜网隔离的 USB 下载线。当安装软件时，USB 驱动自动安装，若自动安装不成功，请按下面步骤手动添加。

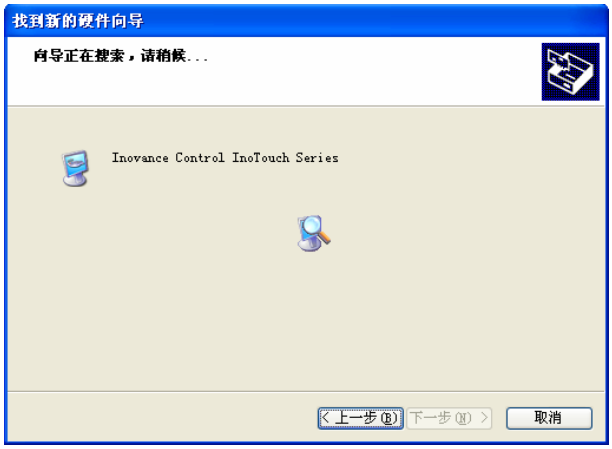
1）USB 线驱动的安装

步骤 1：将 USB 线一端接入 InoTouch 系列人机界面的 USB 口，一端接入计算机的 USB 口，给人机界面上电后，将会弹出找到新硬件对话框，请选择“从列表或指定位置安装” ，如下图所示，并单击“下一步”。

步骤 2：“在这些位置上搜索最佳驱动”选项上勾选第二项“在搜索中包含这个位置”，并浏览在 InoTouch Editor 软件的安装根目录下面“driver USB”这个文件夹，USB 线的驱动保存在这个文件夹中，例如位置为： “C:\Program Files\Inovance Control\InoTouch\driver\usb”，如下图所示。

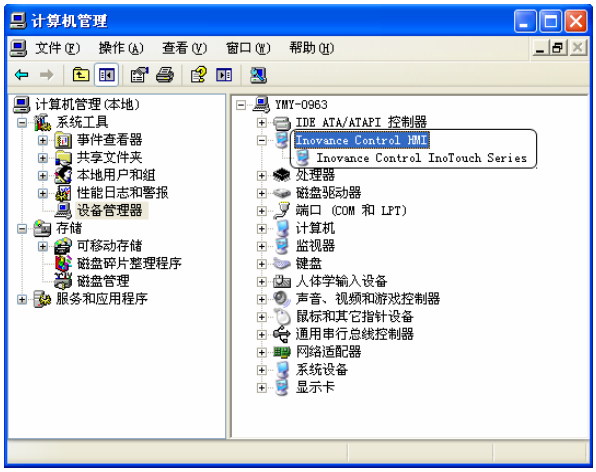


步骤 3：此时按下“下一步”按键，电脑会自动安装这个驱动，如下图所示。



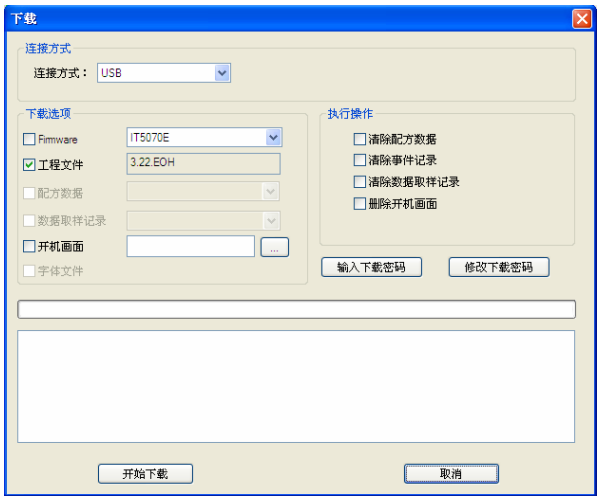
步骤 4：当出现“完成找到新硬件向导”对话框时，表示驱动已经安装完成。

步骤 5：单击“完成”后，在“设备管理器”里面，就可以找到这个“Inovance Control HMI”的设备，如下图所示。

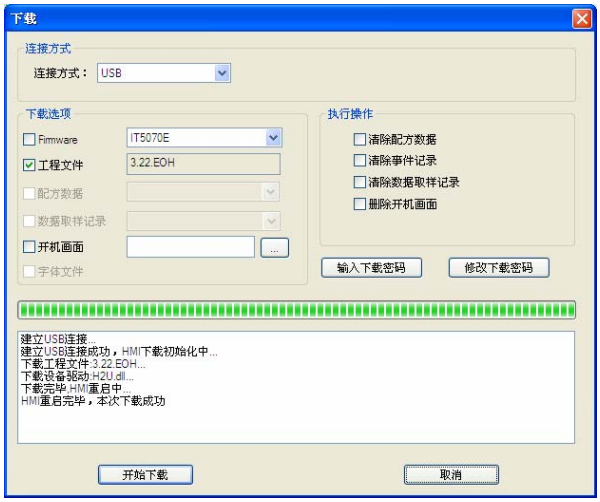


2) 使用 USB 线下载程序

安装好 USB 线驱动之后，就可以使用 USB 线下载程序了。单击 InoTouch Editor 软件菜单“工具/下载”或者工具栏上的图标或者快捷键 F7，将会弹出如下对话框。



因为是使用 USB 线下载，故选择“USB”这项。此时只需要点击“开始下载”，即可下载程序。如下图。



4

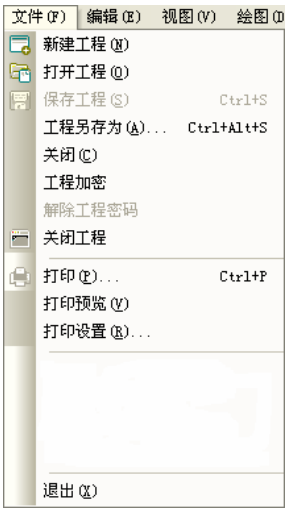
InoTouch Editor 软件的使用

第四章　InoTouch Editor 软件的使用

本章将详细讲解 InoTouch Editor 软件的常规操作，以帮助大家快速的掌握 InoTouch Editor 软件的使用。

4.1 文件

单击“文件”菜单，将会显示如下选项：



[新建工程]

单击“新建工程”，表示新建一个 InoTouch Editor 画面程序。所有 InoTouch Editor 能够编辑的画面程序均以 .afs 结尾。

[打开工程]

单击“打开工程”，表示打开一个已经编辑好的 InoTouch Editor 画面程序。

[保存工程]

单击“保存工程”，表示将当前正在编辑的画面程序保存起来。

[工程另存为....]

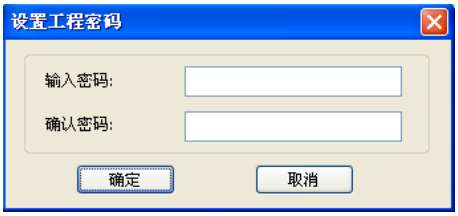
单击“工程另存为....”，表示将目前打开的 InoTouch Editor 画面程序以另外的一个文件名重新保存起来。

[关闭]

单击“关闭”，表示将当前显示的页面关闭。

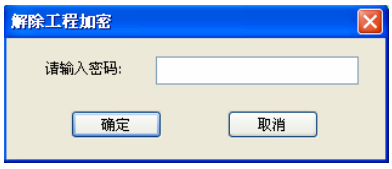
[工程加密]

单击“工程加密”，弹出如下对话框，输入您想设置的密码，点击“确定”，工程密码设置成功。



[解除工程加密]

单击“解除工程加密”，弹出如下对话框，输入正确的密码，点击“确定”，工程密码解除。



[关闭工程]

单击“关闭工程”，表示将当前打开的 InoTouch Editor 画面程序关闭。

[打印]

单击“打印”，表示使用与目前使用的计算机连接的打印机，以图片的方式打印出正在编辑的画面程序。

[打印预览]

按下此功能键，表示将目前正在编辑的画面程序，以图片的方式显示出来，此显示效果即为打印出来后的效果图。

[打印设置]

按下此功能键，表示对目前计算机上所连接的打印机进行打印画面的设置。例如设置纸张的型号，打印方向等。

以上有关“文件”菜单的说明，与一般 Windows 操作系统中文件的操作相同，故不做详细的说明。

4.2 编辑

单击菜单“编辑”，将会出现如下图所示选项。下面将逐个介绍各项的功能。

[撤消] 

撤消的命令一般是用来对编辑画面时，最后一个动作的取消。例如你删除的一个控件，按下“撤消”后，刚刚删除的控件又会被恢复到画面上。

[重做]

重做的命令是对撤消动作的取消。例如你确实想删除刚刚被“撤消”的一个控件，你再按下“重做”后，则刚刚被恢复到画面的控件又被删除了。

[剪切]，[复制] 和[粘贴]

剪切一般是将选中的控件剪切下来，然后再复制到另外一个窗口里面去；而复制的功能一般就是将选中的控件复制到同一窗口的另外的位置或者另外的窗口里面，同时保留选中的控件。当事先已经执行了“剪切”或者“复制”命令后，再执行“粘贴”的命令，则会将刚刚选择的控件复制到当面打开的窗口里面。

[批量复制]

批量复制的功能是编辑 InoTouch Editor 画面时常用的一个功能。它表示将选中的控件复制多个同样类型的控件在同一个画面上。当选中一个控件时，单击“编辑/批量复制”，则会显示如下对话框。

位间隔：表示字位类型的地址间隔。（例如：LW_BIT0:0 ，位间隔为 1，LW_BIT0:1）

根据需要，编写以上画面的内容，然后单击“确定”后，则当前画面上就会出现多个整齐排列的同样类型的控件。有关以上对话框内的各项参数说明如下：

[需复制的数量]

X 轴数量和 Y 轴数量，这两个参数决定了复制后的控件的总数。其中 X 轴的数量表示水平方向上控件的个数，Y 轴的数量表示垂直方向上控件的个数。

[间距] 和 [间隔]

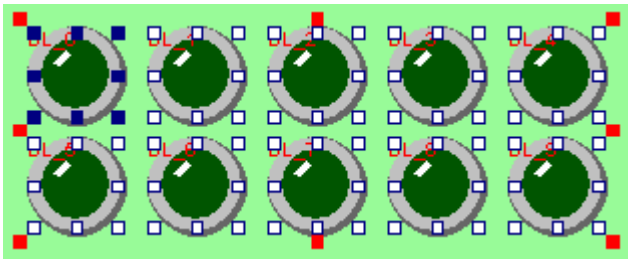
当选择“间距”时，复制后的控件中，前面一个控件的左边和下一个控件的左边的距离由后面的“X 轴间距”方框里面设置的数来决定，而上一排控件的顶端和下一排控件的顶端的距离，是由“Y 轴间距”方框里面设置的数来决定。

当选择“间隔”时，复制后的控件中，前面一个控件的右边和下一个控件的左边的距离由后面的“X 轴间距”方框里面设置的数来决定，而上一排控件的底端和下一排控件的顶端的距离，是由“Y 轴间距”方框里面设置的数来决定。

[放置方式]

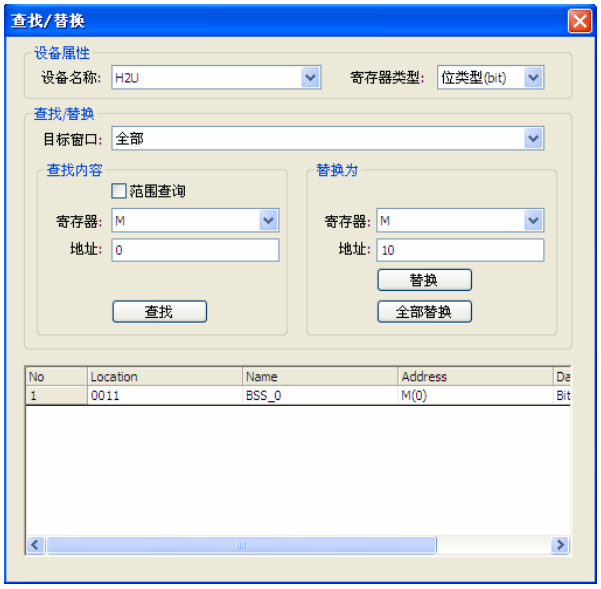
地址间隔表示的是复制后这些控件中地址数据的递加值。例如选择“地址间隔”的值为“1”，然后勾选“水平增加”，表示的是复制后的多个控件，其设备地址是水平递加 1 的关系。当勾选“垂直增加”时，表示的是复制后的多个控件，其设备地址是垂直递加 1 的关系。

综上所述，由上图批量复制设定的参数来看，可以举例，总共复制了 10 个相同类型的控件，水平方向有 5 个，垂直方向有 2 排。水平方向上相邻两个控件的间隔是 10，垂直方向上两排控件的间隔是 5，设备地址是沿着水平方向递加 1 的。如下图所示：



[查找/替换]

按下“寻找/替换”功能键，将会弹出如下对话框



[设备名称]

表示需要寻找 InoTouch Editor 编辑画面中连接的指定的控制器的设备类型。

[目标窗口]

选择需要寻找的窗口范围。

[查找内容]

表示设定要查找/替换的寄存器和起始地址、终止地址。

[替换为]

设定将符合查找条件找到的寄存器和地址，替换为制定的寄存器和地址。

当设定好以上参数后，单击对话框中的“查找”按钮，则在符合查找范围的窗口内查找符合条件的控件。若找到后，会在底部的白色框内显示找到的控件的窗口编号（Location）、控件编号（Name）和设备类型及地址（Address）。当找到符合条件的控件时，双击该找到的控件，画面则会自动的跳转到该控件所在的窗口并标示是该窗口中的哪个控件。

单击“替换”或者“全部替换”时，表示将符合条件找到的控件，替换为参数设定中的控件设备类型和地址。

综上所述，以上对话框的设定中，当单击替换时，表示将目前编辑的画面中所有窗口的所有为 M0 的控件全部替换为 M10。

[选择下一个]和[全选]

“选择下一个”是选择下一个控件。

“全选”是选择页面上所有的控件。

[位置锁定] / [解除锁定]

使用位置锁定功能后，可以将选定的一个或者多个控件位置固定，无法移动，且其形状和大小也无法改变。

1) 选择需要固定的一个或者多个控件；

2) 单击图标位置锁定，则选择的控件将会被固定住；

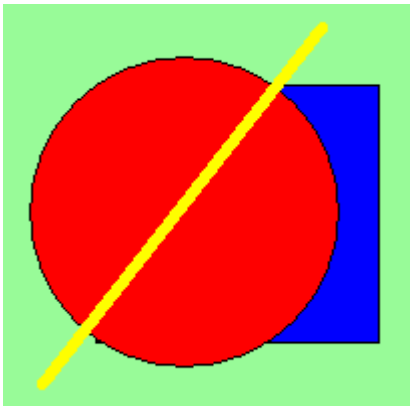
3) 选择已经被固定的控件，再次单击图标解除锁定，则可以将该控件解除固定，又可以移动和改变大小了。

[删除]

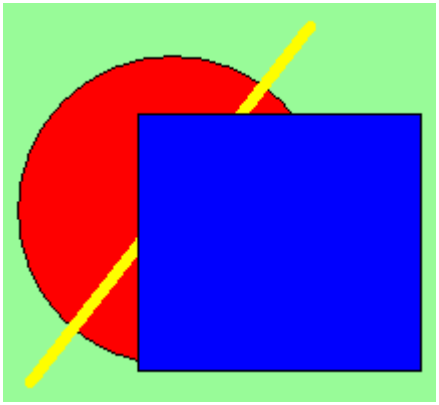
当需要删除一个或者多个控件时，先选择这一个或者多个控件，按下“删除”按键或者键盘上的“Delete”键，则可以将选中的控件删除掉。当然，删除的控件使用前面说明的“撤消”功能键又可以恢复。

[图层]

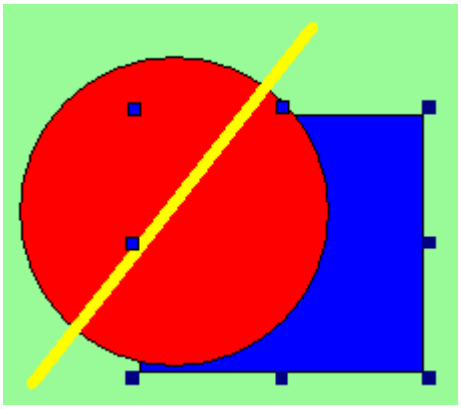
图层的概念是相对多个叠加在一起的控件来产生的。为了更好的说明图层的概念，下面以一个圆、长方形和直线为例来说明。如下图所示。



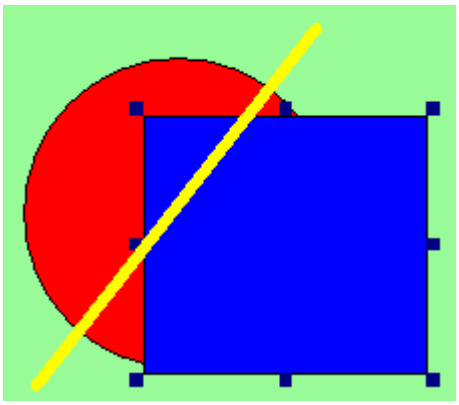
1） 先选中蓝色的长方形，再单击编辑菜单下的图层，然后“移到最顶层”，或者单击工具栏中的图标，则显示的效果如下图所示。可以看出，蓝色长方形就盖住了圆和直线。



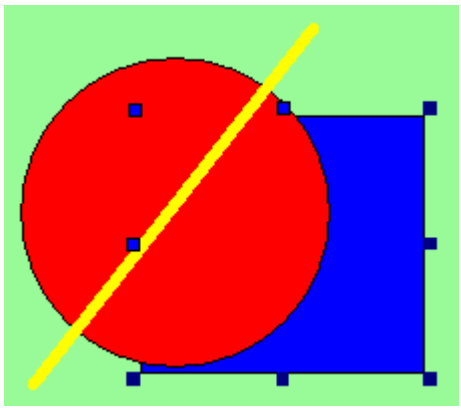
2） 接着单击“移到最底层”或者单击工具栏上的图标，则显示的效果如下图所示。可以看出，蓝色长方形在圆和直线的后面了。



3） 接着单击“移上一层”或者工具栏中的图标，显示的效果如下图所示。可以看出，蓝色长方形往上移了一层，盖住了圆，而没有盖住直线。

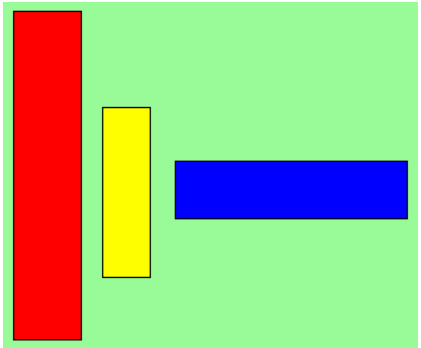


4） 接着单击“移下一层”或者工具栏中的图标，显示的效果如下图所示。可以看出，蓝色长方形此时又移动到了圆和直线的后面了。



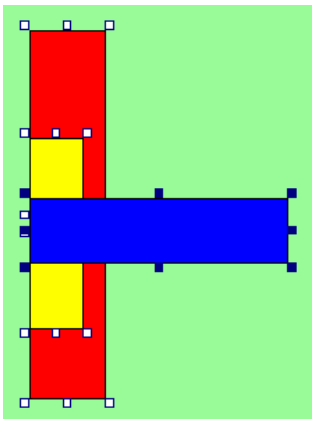
[对齐]

对齐也是平常编辑画面时常用的一个工具。可以让选中的两个或者两个以上的控件快速的排列整齐。选择控件时，最后一个被选中的控件四周会被不同于其他选中控件的颜色区分开，一般是使用蓝色来做标志区分。同时，做对齐动作时，会以这个控件为基准，与之排列成各种对齐方式。下面以红黄蓝三个不同颜色的长方形来分别加以说明，便于大家更好的理解。三个图形如下图所示。



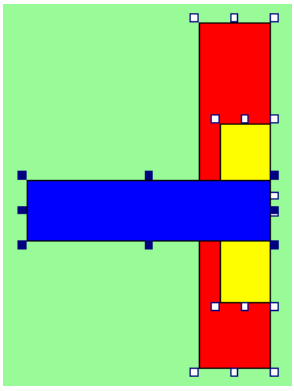
1） 左对齐

先选中三个控件。可以单击鼠标左键，使用拖动的方式选择该三个控件，或者按住键盘上的 Ctrl 键不放，然后分别单击三个控件，也可以将三个控件选中。然后单击该“左对齐”的图标，显示的结果如下图。表示的是以蓝色的长方形为基准，三个控件左边对齐。最后按下“撤消”，这三个控件又恢复到之前的排列位置。



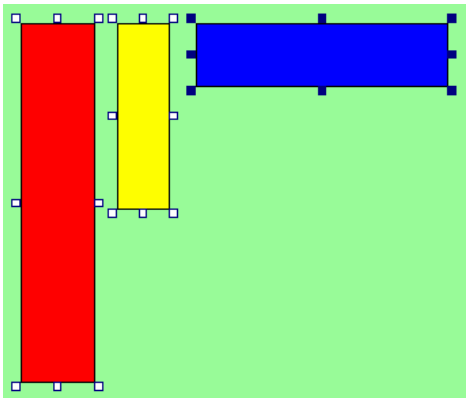
2） 右对齐

同样方法选中该三个控件，然后单击右对齐图标，则三个控件排列后的图形如下图所示。表示是以蓝色长方形为基准，三个控件右边对齐。最后按下“撤消”，这三个控件又恢复到排列前的位置。



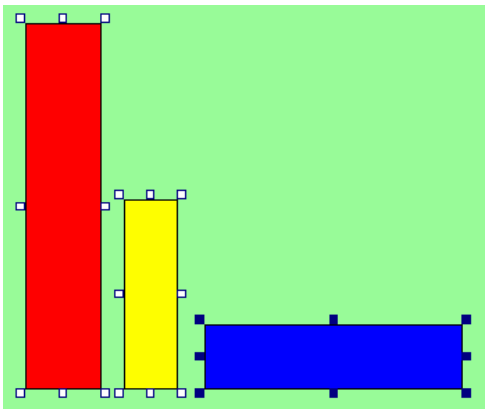
3） 顶部对齐

选择该三个控件后，然后单击图标，表示三个控件要顶端对齐。如下图所示。从排列图来看，是以蓝色长方形为基准，三个控件的顶端对齐排列的。最后按下“撤消”，则这三个控件又恢复到排列前的位置。



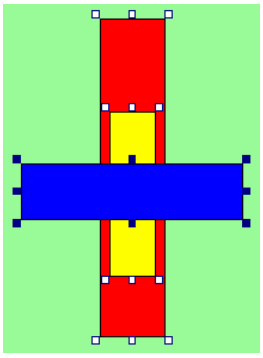
4） 底部对齐

选择该三个控件后，再单击图标，表示该三个控件要底端对齐。如下图所示。从排列图来看，是以蓝色长方形为基准，三个控件的低端对齐来排列的。最后按下“撤消”，则这三个控件又恢复到排列前的位置。



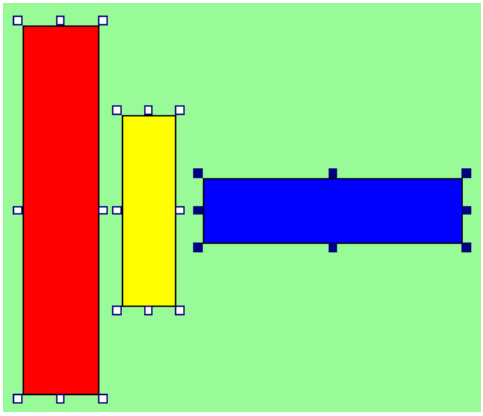
5） 水平居中对齐

选中该三个控件，并点击图标，表示将三个控件水平居中来排列。排列后的图形如下。由此可以看出，是以蓝色长方形为基准，三个控件水平居中来排列的。最后，按下“撤消”，则三个控件有恢复到排列前的位置。



6） 垂直居中对齐

选中三个控件，单击图标，表明要将三个控件以垂直居中方式排列。排列后的位置见下图。可以看出，此时是以蓝色长方形为基准，三个控件垂直居中排列。最后，单击“撤消”按钮，三个控件又恢复到排列前的位置。

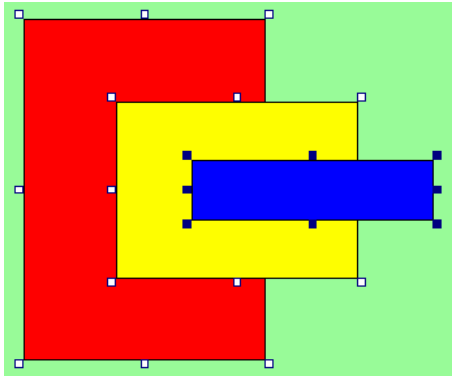


[对象尺寸调整]

在编辑画面程序时，有时候也会需要将两个或者多个不同的控件调整为相同的大小的功能，使之画面控件更加美观。同样的，选择的多个控件中，最后选中的控件会与其他被选中的控件四周的颜色不一样，一般是以蓝色来显示，且调整后的尺寸，是以该控件为基准。下面同样以上面所列举的几个长方形为例说明。

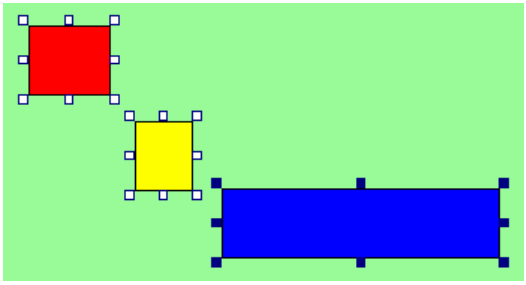
1） 等宽

选中上例中的三个不同颜色的长方形。然后再选择图标，最后显示的效果图如下所示。从图上可看出，是以蓝色的长方形为基准，三个长方形的宽度变为相等。最后，按下“撤消”功能键，这三个控件的宽度又恢复到改变前的宽度。



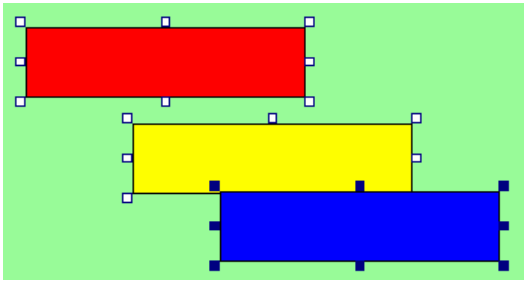
2） 等高

再次选择这三个控件，然后单击图标，将这三个控件设定为高度相同，执行后的效果图如下图所示。可以看出，是以蓝色长方形高度为基准，三个长方形的高度都变为一样了。单击“撤消”功能键后，三个控件的尺寸又恢复到改变前的状态。



3） 等大小

选中这三个长方形，然后单击等大小的图标，这三个控件会变为大小一致，改变后的大小见下图。可以看出，是以蓝色长方形为基准，三个长方形的大小都变为一样了。



[对象位置微调]

微调的作用是更精细的调整画面上控件放置的位置，它提供了往上、往下、往左、往右四个方向的调整工具按钮。图标分别是：往上、往下、往左、往右。操作步骤为：

1) 选择需要微调的一个或者多个控件

2) 用鼠标单击需要调整方向的图标，单击一次移动一个像素的位置。

[对象组合/对象拆分]

使用群组功能，可以将两个或者两个以上的控件组合在一起。这样，可以很方便的对这几个控件执行删除、复制、移动等操作，就好像只操作一个控件一样方便。操作过程如下：

1） 选择需要群组的控件

2） 点击图标，则选中的控件就会被群组在一起。

3） 如果需要取消群组功能，则选择已经被群组的控件，然后单击图标，则选中的控件就会从群组中分离，每个控件可以单独的执行各种操作了。

[新增页面/删除页面]、[新增设备/删除设备]、[HMI 系统配置]

对于新增页面/删除页面、新增设备/删除设备、HMI 系统配置会在后面相关的章节做详细的说明。

4.3 绘图

InoTouch Editor 人机界面工程画面中，是以“图形化”的方式来显示所控制的机器的各种信息和状态。这些图形可以由 InoTouch Editor 软件的图库自带，编程时调用，也可以通过外部添加进来。同时 InoTouch Editor 软件也提供了一套绘图工具，先来介绍一下如何使用这个绘图工具来绘制各种需要的图形。

InoTouch Editor 软件的绘图工具提供了直线、圆、弧、多边形等绘图工具，通过这些工具绘出来的图形，可以很方便显示在工程画面中，也可以添加到 InoTouch Editor 软件的图库中，以备后续工程继续使用。

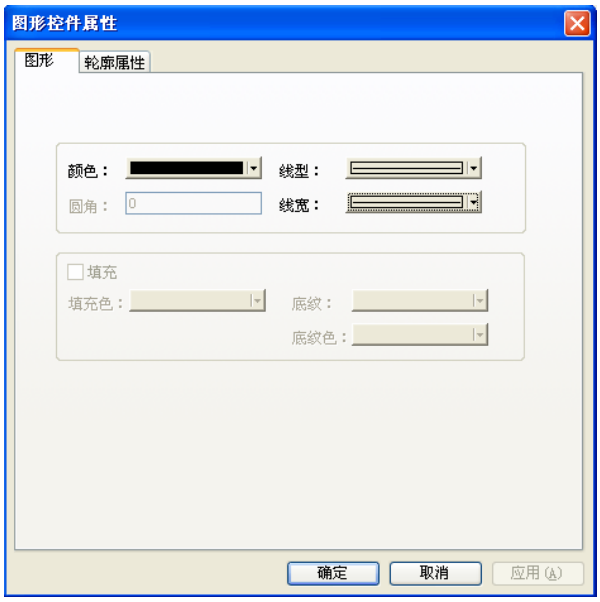
4.3.1 画直线

直线包含了三个属性：直线的线型、线宽和颜色。设定了这三个属性后，就可以画出一条直线出来。步骤如下：

a、 单击 InoTouch Editor 菜单上的“绘图/直线”或者工具栏上的图标，此时在画面工作区单击一下鼠标左键，表示直线的起始点。拖动鼠标一定距离后，再单击一下鼠标左键，表示直线的结束点。画面上就出现了一条直线。

b、 选中刚刚绘出的直线后，可以将直线移动至需要的位置；也可以拖动鼠标改变直线的长短、方向。

c、 选中这条直线，双击或单击鼠标右键选择“属性”进行编辑，如下图所示：



d、 单击“颜色”后面的按钮，将会弹出颜色列表，选择需要的颜色后，会立即显示在这个地方；

e、 单击“线型”后面的按钮，将会弹出线条的形状，是连续的实线还是虚线等；

f、 单击“线宽”后面的按钮，将会弹出线条的宽度，选择所需要的宽度；

4.3.2 手画线

该工具提供了一个可以绘制任意轨迹的线条功能

a、 单击 InoTouch Editor 软件菜单“绘图/手画线”，或者单击工具栏上的图标，此时在编辑画面区域单击鼠标左键，则表示了该手画线的起始位置，在根据需要在编辑画面区域移动鼠标，会看到鼠标移动的轨迹上都会有线条出现，单击鼠标，完成一条手画线。如果需要再画多条手画线的话，重复前面的操作即可；

b、 双击这个手画线或单击鼠标右键选择“属性”进行编辑，可以更改它的颜色、线宽等属性。

4.3.3 折线

使用折线工具可以画出一条连续的直线，从而形成一个新的图形。

a、 单击 InoTouch Editor 软件菜单“绘图/折线”，或者单击工具栏上的图标， 在画面编辑区域单击鼠标左键表示绘制直线的开始，然后根据自己的需要拖动鼠标，将会在鼠标的轨迹上出现一条直线，再单击一下鼠标的左键，第一条直线此时结束绘制，同时以第一条直线的末端为起点开始另外一条直线的绘制，重复刚刚的操作步骤。最后将鼠标移到你想画的位置，单击鼠标右键后，则形成了有这几条直线组成的一条折线。

b、 双击刚刚绘制的折线图形或单击鼠标右键选择“属性”进行编辑，可以修改图形的颜色、线宽等属性。

4.3.4 画圆弧

利用画圆弧工具可以绘制一条圆弧形曲线出来。

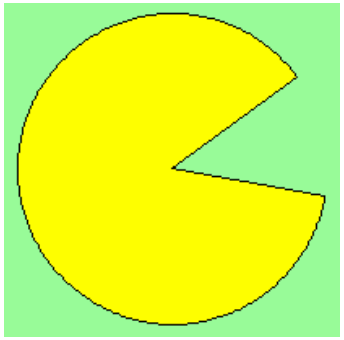
a、 单击 InoTouch Editor 软件菜单“绘图/圆弧”或者单击工具栏上的图标， 在画面编辑区域单击鼠标左键，表示开始绘图，然后拖动鼠标，会看到一个圆形轨迹的图形在变化，达到预想的大小时，再单击鼠标左键一次，此时会从圆心出现一条直线，使用鼠标将该直线移到需要起始绘图的位置并单击左键，再使用鼠标将直线移动到需要结束的位置，再单击一次鼠标左键，这样就完成了一个圆弧的绘制。

b、 双击绘制好的圆弧形曲线或单击鼠标右键选择“属性”进行编辑，可以修改这条弧形曲线的颜色、线宽等属性。

4.3.5 扇形

使用扇形绘图工具可以绘出需要的扇形图形，只是它的显示角度是在 1°到 360°之间。

a、 单击 InoTouch Editor 软件菜单“绘图/扇形”或者单击工具栏上的图标，在画面编辑区域单击鼠标左键，表示开始绘图，然后拖动鼠标，会看到一个圆形轨迹的图形在变化，达到预想的大小时，再单击鼠标左键一次，此时会从圆心出现一条直线，使用鼠标将该直线移到需要起始绘图的位置并单击左键，再使用鼠标将直线移动到需要结束的位置，再单击一次鼠标左键，这样就完成了一个扇形的绘制，如下图所示。



b、双击刚刚绘制的扇形图或单击鼠标右键选择“属性”进行编辑，可以修改该扇形图的相关属

性。

4.3.6 圆/椭圆

圆/椭圆工具可以绘制出椭圆或者圆形图形。绘制圆/椭圆由四个参数决定：线宽、颜色、线型和内部填充。

a、 单击 InoTouch Editor 软件菜单“绘图/圆”或者工具栏上的图标，在画面编辑区域点击鼠标左边，并拖动鼠标，此时在画面上会看到形成了一个圆。单击鼠标的左键，则结束了这个圆的绘制；

b、 双击这个圆或单击鼠标右键选择“属性”进行编辑，则可以修改这个圆的填充颜色、线宽、大小等属性。

c、 如果圆/椭圆的内部需要填充，请勾选“填充”，并选择填充的颜色、底纹和底纹色；

d、 在“颜色”属性，选择圆/椭圆边框的颜色和线条的宽度；

4.3.7 矩形

利用矩形绘图工具可以绘制出需要的长方形或者正方形。绘制后的图形，由矩形边框宽度、颜色、大小、填充颜色等参数来决定。

a、单击 InoTouch Editor 软件菜单“绘图/矩形”或者单击工具栏上的图标，在画面编辑区域单击鼠标左键，然后再拖动鼠标到另外一个位置，立即会看到画面上画出了一个矩形图形，再次单击鼠标左键，则矩形图形显示在画面上了；

b、双击刚刚绘制的矩形图形或单击鼠标右键选择“属性”进行编辑，则可以修改边框颜色、宽度、

大小、填充颜色等属性。

4.3.8 多边形

使用多边形物件可以绘制多边形，同样的，多边形的形状由边框的颜色、宽度、大小、填充的颜色等属性来决定。

a、单击 InoTouch Editor 软件菜单“绘图/多边形”或者单击 InoTouch Editor 软件工具栏上的图标， 在画面编辑区域单击鼠标左键，表示开始多边形控件的绘图；此时拖动鼠标到另外一个位置，再单击一下鼠标左键，就绘制了多边形的一条边；再拖动鼠标到另外的一个位置，再单击一下鼠标左键，又绘制了多边形的一条边；如此重复操作，即可绘制多个边。

b、多边形的边都绘制完成之后，单击鼠标的右键，则刚刚绘制的直线会自动的首尾连接起来，从而形成了一条多边形。

c、双击刚刚绘制的多边形图形或单击鼠标右键选择“属性”进行编辑，可以修改多边形的各属性。

4.3.9 刻度

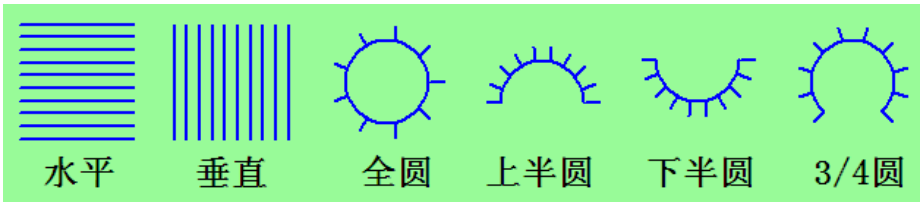
使用刻度工具，可以绘制出需要的格式的刻度图形到画面上，刻度图形是由刻度轮廓大小、

线条颜色、刻度样式和分割数量来决定的。

a、单击 InoTouch Editor 软件菜单“绘图/刻度”或者单击 InoTouch Editor 软件工具栏上的图标，此时鼠标会变成“+”形状；在画面编辑区域，单击鼠标左键，并拖动鼠标到另外一个位置，再单击鼠标左键一次，此时会显示一个默认的水平方向的刻度在画面上；

b、双击刚刚绘制的刻度图形或单击鼠标右键选择“属性”进行编辑，可以显示刻度图形的属性对

话框，在此可以根据需要修改刻度图形的颜色、线宽、刻度的样式等。刻度的样式主要有以下几种类型；



c、选择需要等份分割的刻度数量，默认为 10。除了水平和垂直两个方向，对于其他方向的刻度可以设定刻度指针的长度；

d、单击“确定”即完成了刻度图形的绘制。刻度图形一般与表针、趋势图、棒图等控件配合使用，可以更加形象的显示 PLC 中数据的状态。

4.3.10 边框

利用绘制边框工具可以绘制出需要的长方形或者正方形边框。绘制后的图形，由边框宽度、颜色、大小、填充颜色等参数来决定。

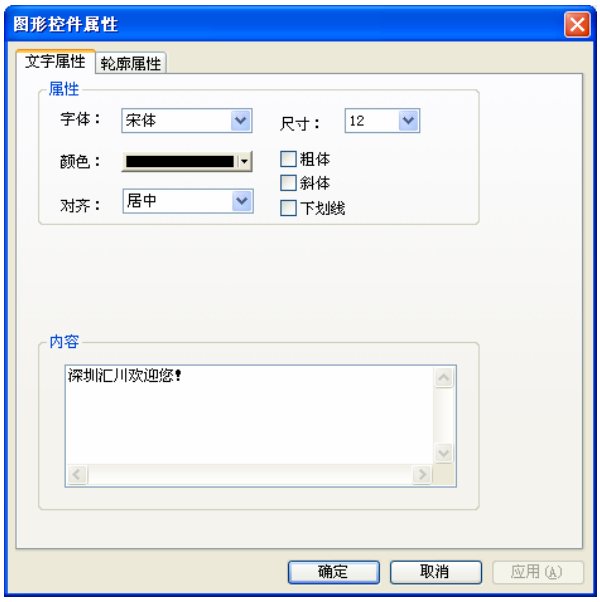
a、 单击 InoTouch Editor 软件菜单“绘图/边框”或者单击工具栏上的图标，在画面编辑区域单击鼠标左键，然后再拖动鼠标到另外一个位置，立即会看到画面上画出了一个边框图形，再次单击鼠标左键，则边框图形显示在画面上了；

b、 双击刚刚绘制的边框图形或单击鼠标右键选择“属性”进行编辑，则可以修改边框颜色、宽度、边框类、填充颜色等属性。

4.3.11 文字

InoTouch Editor 软件支持 Windows 的所有字体，且文字均为矢量字体。这样，你就可以使用任意你喜欢的字体，且可以自由平滑的放大和缩小，均不会产生锯齿波现象。各字体可以选择采用粗体、斜体或者下划线等方式。

a、单击 InoTouch Editor 软件菜单“绘图/文字”或者单击工具栏的图标，在画面编辑区域单击鼠标左键，就显示Text，双击此图标或单击鼠标右键选择“属性”进行编辑，如下图；



b、从字体列表中选择需要的字体，文字颜色、字体大小、对齐方式等属性；

c、在“内容”输入框内，就可以输入需要的文字。以键盘上的回车键换行。

d、输入完成后，单击“确定”，然后移动鼠标并单击鼠标左键可将该文字物件放置在画面上合适的位置。

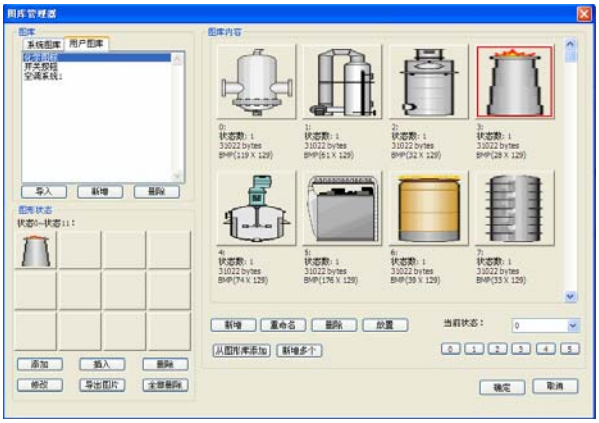
4.3.12 图片

InoTouch Editor 软件图库中提供了一部分的图片。InoTouch Editor 软件支持的图片显示颜色为 65536 色，支持的图片格式为 BMP、JPG 和 GIF 三种格式。除了软件的图库中自带的图片外，用户还可以将外部图片添加到软件的图库中，以方便多次使用，例如设备的一个整体照片、公司 logo 的图片等等。

使用位图库中的图形

a、 单击 InoTouch Editor 软件“绘图/图片”或者单击工具栏上的图标，在画面编辑区域单击鼠标左键，就显示，双击此图标或单击鼠标右键选择“属性”进行编辑，如下图；

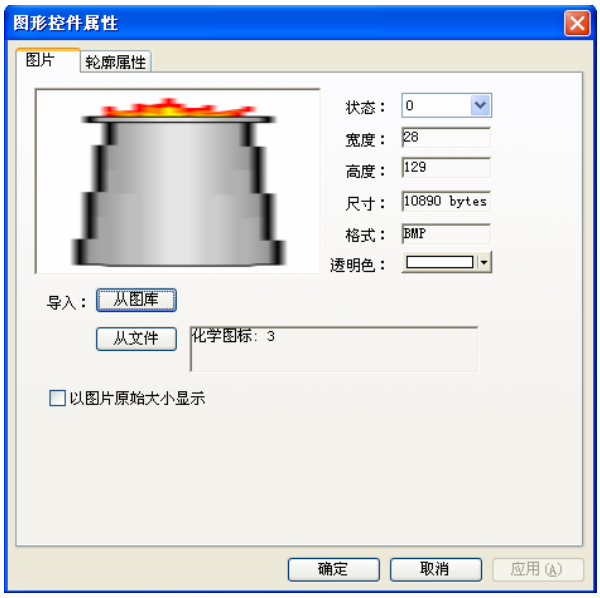
b、单击“从图库”按键，将会显示图形库的对话框，如下图所示；



c、单击左上角图库名称，即可显示该图库下的图形，通过移动右边的滚动条还可以查看该图库里面所有的图形；

d、从图库中选中一张图形，被选中的图形其所有的状态将会在对话框的左下角图形状态栏内显示出来。

e、单击“确定”选中这张图形，将会回到刚刚的图形对话框，刚刚选中的图形将会出现在该对话框中，如下图所示；



f、再单击“确定”，刚刚选中的图形就会显示在画面上。

g、单击“从文件”按键，将会显示打开对话框，选择你所要的图片。

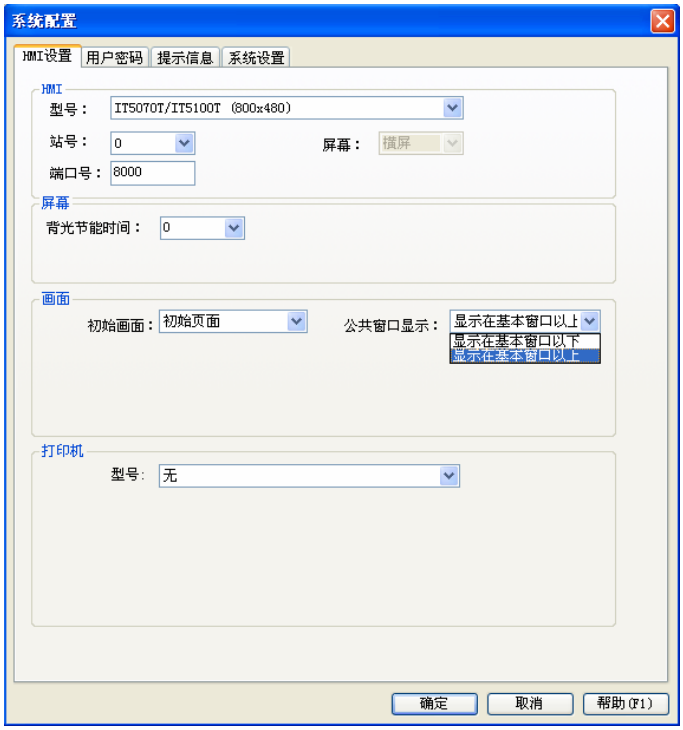
小结：本章主要介绍了 InoTouch Editor 软件的画面编辑和绘图，在做画面编辑时，对各控件常见的编辑整理工作。例如，多个控件的排列，使之更加的整齐美观等；绘图可以绘制自己想要的图形，也是矢量图。

5

系统参数

第五章 系统参数

打开 InoTouch Editor 软件后，单击菜单“编辑/HMI 系统配置”，则可以打开系统参数的设定对话框，如下图所示。系统参数设定里面总共包含了 [HMI 设置]、[用户密码]、[提示信息]和[系统设置]总共四个部分。下面将分别对这几个部分进行说明。



5.1 HMI 设置

[HMI 设置]用来设定所使用的人机界面的型号、站号、端口号、屏幕、画面等有关的设定。各项设定的说明如下：

[HMI 型号]

型号可选择，根据所使用的 HMI 型号来选择。

[HMI 站号]

选择 HMI 所使用的站号，如无特殊目的，采用默认值即可。

[端口号]

设定 HMI 所使用的通讯端口号码，如无特殊目的，采用默认值即可。

[背光节能时间]

当未操作人机界面的持续时间超过此设定值时，将关闭背光灯，设定的时间单位为分钟。关闭背光灯后只需碰触屏幕触控区域的任意一个地方，即可重新打开背光灯。当设定值选择“0”时，人机界面将不使用背光节能的功能。对于每天 24 小时连续操作的机器，为了延长背光灯的使用寿命，一般建议设定此项参数。

[初始画面]

选择人机界面载入程序后，显示的起始页面。

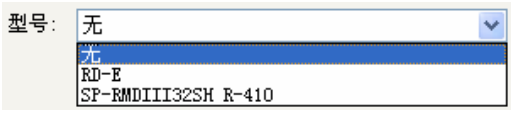
[公共窗口显示]

公共窗口(2002 号窗口)内的控件会出现在工程画面的每个基本窗口中，此选项用来选择公共窗口内的控件是出现在基本窗口原来控件的上层，或下层。

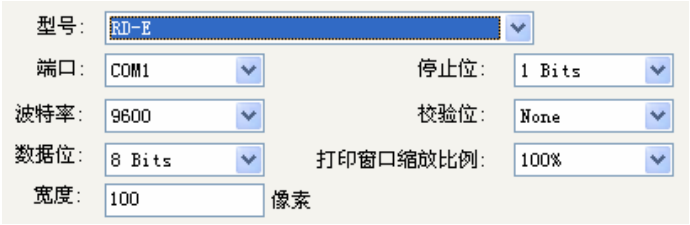
[打印机]

[型号]

显示目前支持的打印机类型，打印机需使用串口连接。

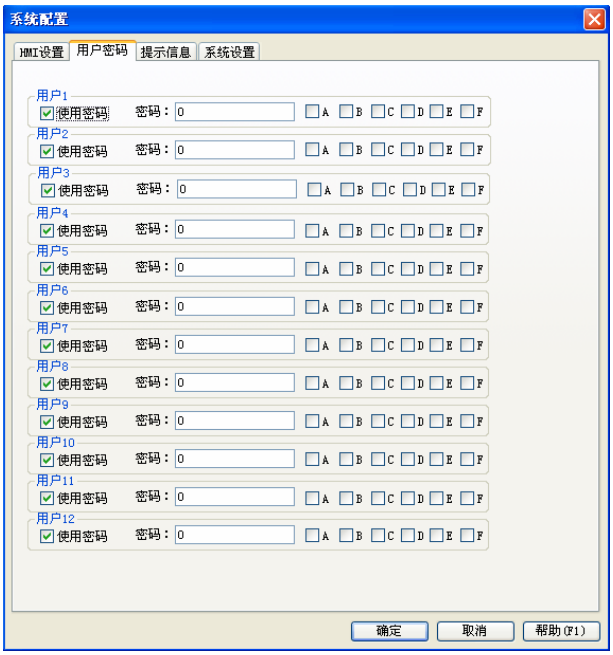


使用串口连接的打印机需正确设定串口的通讯参数。当打印机的型号为 RD-E 时，需正确设定宽度等，此设定值不可以超过打印机每行可以打印的像素，否则将造成错误的打印结果。

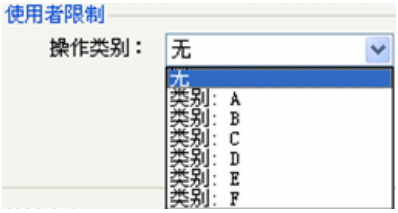


5.2 用户密码

系统参数中的[用户密码]设定页用来设定每个用户的密码，并规划每个用户可操作的控件类别，在 InoTouch Editor 中， 控件被划分为“无”与“A~F”等共 7 个类别。用户的密码必须是由 0~9 的数字所组成，InoTouch Editor 最多可规划 12 个用户。



人机界面程序运行时，用户在成功输入密码后，InoTouch Editor 会依照用户的设定内容决定用户可以操作的控件类别。在工程文件中，控件的类别被区分为“无”与“类别 A”至“类别 F”共 7 种，控件可以设定所属的类别，参考下图。类别属于“无”的控件，任何用户均可操作使用。

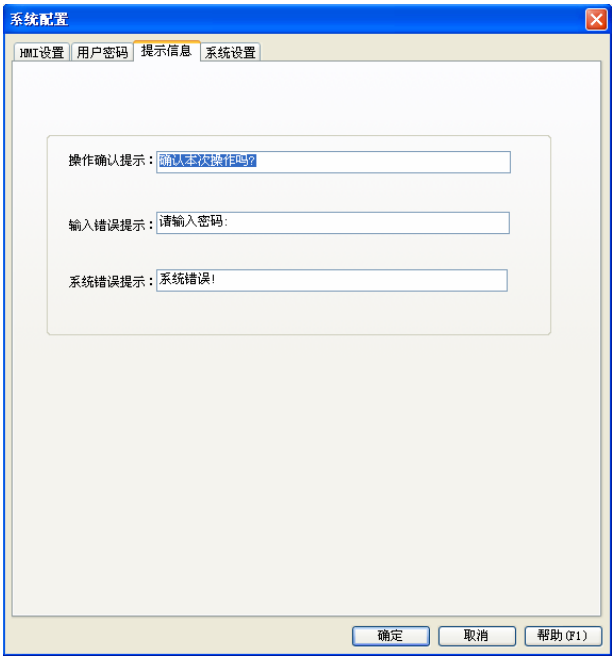


由上图的“用户密码”设定页，可以看出，用户 1 的密码是 11，该使用者允许操作的控件类别属于“无”与“类别 A”的控件。而用户 12 的密码是 666，该用户可以操作“无”和“类别 A”到“类别 F”所有类别的控件，说明该用户是机器的最高管理员了。详细说明请参考 [控件安全防护]。

注意：不能使用数字全部为 0 的密码。

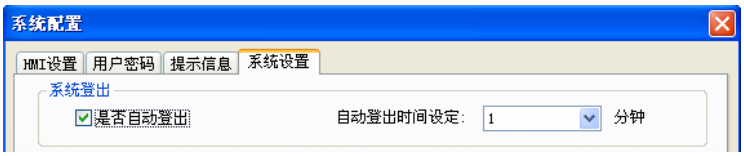
5.3 提示信息

当使用提示信息时，就可以选择“操作确认提示”“输入错误提示”“系统错误提示”三种。



5.4 系统设置

系统设置是把系统设置成多长时间不操作，就会自动登出，返回到登录页面。



6

窗　口

第六章　窗口

窗口是 InoTouch Editor 人机界面画面程序的一个基本元素，也是很重要的一个元素。有了窗口后，画面上的各种物件、图形、文字等信息才可以显示在人机界面上。一般的工程中，不会只有一个窗口，所以一般一个工程需要创建多个窗口。InoTouch Editor 提供了 1~2000 总共 2000 个窗口的创建和编辑功能。而每一个工程具体能够使用的窗口数量，由所有创建窗口编译之后，输出文件（EOH 文件）容量不大于 InoTouch Editor 人机界面所能输出文件（EOH 文件）容量大小所决定。例如，InoTouch 系列人机界面的画面编译之后，输出文件（EOH 文件）容量超出 16MB，就会报警，则所编写的画面 EOH 文件容量只要不超过 16MB，则可以尽可能多的创建多个画面窗口。

6.1 窗口类型

在 InoTouch Editor 软件中，依照功能与使用方式的不同，可将窗口分为下列三种类型：

⑴ 基本窗口；⑵ 公共窗口；⑶系统讯息窗口；依次说明如下：

6.1.1 基本窗口 (base window)

这是最常见的窗口，一般当作主画面的用途之外，也被用在：

a. 底层画面，可提供其它窗口作为背景画面。

b. 键盘窗口。

c. [功能键]控件所使用的弹出窗口。

d. [间接窗口]与[直接窗口]控件所使用的弹出窗口。

e. 保护屏幕画面。

基本窗口必须是与屏幕的大小一样。也即，基本窗口的分辨率需要与所使用的人机界面的分辨率一致。

6.1.2　公共窗口 (common window)

2002 窗口为预设的公共窗口，此窗口中的物件也会出现在其它基本窗口中，因此通常会将各窗口共享的物件或者称为相同的物件放置在公共窗口中。例如产品的一个 logo 图标，或者某一个共用的按键等。

6.1.3　系统讯息窗口(system message window)

2003 窗口、2004 窗口、2005 窗口与 2006 窗口为系统预设的系统提示讯息窗口。其中 2003 窗口为“PLC　响应”窗口，当人机界面与 PLC 或者控制器通讯中断时，系统将自动弹出此窗口在人机界面当前打开的窗口上。

2004 窗口为“HMI 连接”窗口，当人机界面无法连接到远程的人机界面时，系统将自动弹跳出此窗口。

2005 窗口为“访问权限”窗口，当使用者的操作权限不足以操作正要操作的控件时，会依组件

的设定内容，决定是否弹跳出此窗口作为警示用途。

2006 窗口为“存储空间状态”窗口，当 HMI 内存或 U 盘上的可用空间不足以储存新的数据时，系统将自动弹跳出此窗口。使用者也可以使用下列的系统保留暂存器检视 HMI 内存、U 盘上目前可用的储存空间。

[LW 9072]　HMI 目前的可用空间(单位 K bytes)

[LW 9074]　SD 卡目前的可用空间(单位 K bytes)

[LW 9076]　U 盘目前的可用空间(单位 K bytes)

当发生空间不足的情形时，可以利用下列的系统寄存器检视是哪一个类型的储存装置空间不足。

[LB 9035] HMI 可用空间不足警示

[LB 9036] SD 卡 可用空间不足警示

[LB 9037] U 盘 可用空间不足警示

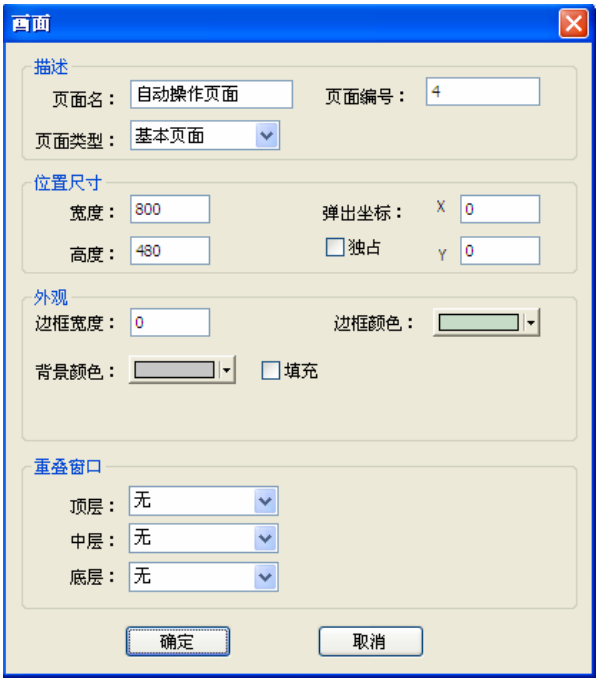
窗口 2003 到窗口 2006 里面的内容提示，可以自己根据实际需要来修改。例如，窗口 2003 的显示内容“PLC 响应”，可以改为“人机界面与 PLC 通讯中断！”等实际内容的提示。其他几个窗口的内容可以比照修改，方便操作人员容易读懂和识别故障信息。

6.2 窗口的建立、设定与删除

InoTouch Editor 软件中建立的窗口可以通过左边的窗口列表来查看，如下图所示。下面说明如何建立与设定这些窗口。

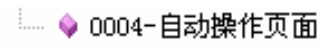
6.2.1 窗口的建立

建立窗口有三种方式，第一种方式是在窗口树状图上选择要建立的窗口，并按下鼠标的右键，在窗体出现后选择[添加页面]将出现设定对话窗，在完成各项设定并按下确认键后，即可建立新的窗口，参考下图。



[页面名]

页面编号后所显示的名称，参考下图。一般以容易读懂和记住为原则。例如：手动操作页面，自动操作页面等名称，清楚明了。



[页面编号]

页面编号，从 1~2000。

位置尺寸

[宽度]、[高度]　窗口大小。一般基本窗口的分辨率与所选用的人机界面的分辨率一样。例如，使用的人机界面是 IT5070T，它的分辨率是 800*480。那么这个新建立的窗口的宽就设定为 800，高设定为 480。

[X]，[Y] 基本窗口也可当作弹出窗口，[X]与[Y]用来设定基本窗口在屏幕弹跳出的坐标位置。坐标原点在屏幕的左上角。

[独占] 如勾选此选项，当基本窗口作为弹跳窗口并出现时，在此基本窗口未关闭前，将无法操作其它窗口。当基本窗口被作为键盘窗口时，自动具有此项属性。

外观

[边框宽度] 此设定是对该窗口设定一个边框的宽度。范围是 0~10，默认设定是 0。

[边框颜色] 设定边框的颜色。可以在颜色列表里面选择一个自己需要的颜色。

[背景颜色] 设定窗口的背景颜色。可以单击颜色列表，选择一个合适的颜色。

重叠窗口

[底层]、[中层]、[顶层]　每一个基本窗口最多可以再选择其它三种基本窗口作为背景，从[底

层]开始到[顶层]结束，这些背景窗口内的控件将在基本窗口中依序出现的。系统默认均为无底层窗口。

第二种建立的窗口方式是使用 InoTouch Editor 菜单上的[编辑]，选择[新增画面]后可以得到“一个新的窗口”对话窗，

第三种单击工具栏上的  图标。

6.2.2 窗口属性的设定

InoTouch Editor 更改窗口的属性，在各窗口中，在未选择到任何控件时按下鼠标的右键并在窗体出现后选择[属性]，或者在项目管理列表中，选择要修改的窗口，点击右键选择[属性] 。

6.2.3 窗口的开启、关闭与删除

要开启已存在的窗口可以使用鼠标双击窗口树状图上的窗口编码，就可以打开你选择的窗口。

要关闭可以点击窗口的“X”close 

。

删除存在的窗口，在窗口树状图上先选择要删除的窗口，并按下鼠标的右键，在窗体出现后选择[删除页面]，就可以删除选择的窗口。

6.3 基本窗口的使用

上面讲到了在 InoTouch Editor 编辑工程画面时，对各种窗口类型的一般操作。那么在工程画面运行时，如何使用基本窗口呢？下面来一一说明。

6.3.1 打开一个基本窗口

前面提到，基本窗口一般用来作为满屏幕的窗口，也可以作为“功能键”控件调用的弹出窗口。使用“功能键”控件时，可以打开一个满屏的基本窗口也可以调用一个弹出窗口。

1） 调用基本窗口

使用功能键打开基本窗口的时候，会将当前打开的基本窗口关闭，并切换到指定的基本窗口。设定步骤如下：

a) 单击“控件/功能按键”，或者单击工具栏中的图标  ，在窗口中单击左键，就建立了一个功能键。

b) 选择刚刚建立的“功能键”双击或点击右键选择属性，可以设置功能键的属性。

c) 在“一般属性”对话框里面选定“切换基本窗口”，并在“窗口编号”框后面选定需要切换的目标窗口编号；

d) 在“图形”选项，选择图库里面的一个图片；

e) 在“标签选项”里面，填写需要的文字，并选定该文字使用的字体；

f) 单击设定对话框底部的“确定”按钮，就完成了属性设定。

这样，就做好了一个切换基本窗口的功能键，执行离线仿真时，就会发现，单击这个按键，画面就会切换到指定编号的基本窗口。

2）调用弹出窗口

基本窗口也可以作为弹出窗口用。此时，基本窗口一般不会是满屏的（也可以是满屏的）。弹出窗口可以显示在基本窗口的上面，具体它显示在窗口的什么位置，由在建立这个窗口时，设定作为弹出窗口时的 X 轴和 Y 轴的坐标来决定。

a) 单击“控件/功能键”，或者单击工具栏中的图标  ，在窗口中单击左键，就建立了一个功能键。

b) 选择刚刚建立的“功能键”双击或点击右键选择属性，可以设置功能键的属性。

c) 在“一般属性”对话框里面选定“弹出窗口”，并在“窗口编号”框后面选定需要弹出的目标窗口编号；

d) 在“图形”选项，选择图库里面的一个图片；

e) 在“标签选项”里面，填写需要的文字，并选定该文字使用的字体；

f) 单击设定对话框底部的“确定”按钮，就完成了属性设定。

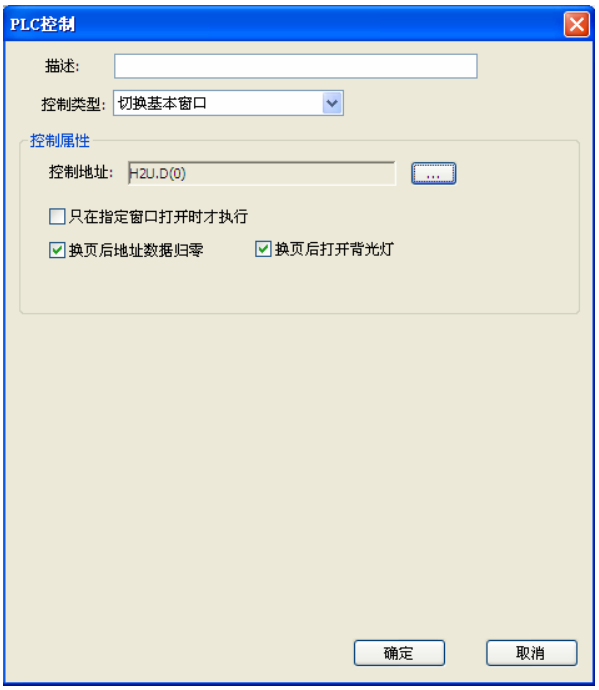
这样，就做好了一个弹出窗口的功能键，执行离线仿真时，就会发现，单击这个按键，当前画面上就会弹出一个指定的画面在该画面上层，并且该弹出窗口的顶端有一个“窗口控制条”，拖动这个“窗口控制条”，可以将该窗口移动到目前窗口范围的任意位置。

3) 使用 PLC 来调用基本窗口

使用 InoTouch Editor 控件菜单下的“PLC 控制”控件，也可以来执行打开基本窗口的功能。该功能是定义一个 PLC 中的数据寄存器来切换基本窗口。此时，人机界面会不断的扫描 PLC 中该寄存器的数值，当这个数值与工程画面中的某一个画面编号相同时，人机界面就会自动的切换到这个画面窗口，同时会将这个画面编号的值传送给这个寄存器的下一个寄存器中。例如，在“PLC

控制”这个控件中定义的是 PLC 中的 D0 来切换基本窗口，当 D0=11 时，人机界面会自动的切换到 11 这个编号的基本窗口画面，并且将数据 11 传送给 D1，以确认窗口切换完毕。设定步骤如下：

a) 打开 InoTouch Editor 菜单左侧项目管理下的“PLC 控制对象表”，显示 PLC 控制设定对话框如下图。



b) 设定由连接的哪个 PLC 来切换窗口，并在“触发地址”项的“设备类型”和“地址” 里面填上相应的设备类型和地址。并设定地址的数据格式。一般设置为 16-bit unsigned 格式。数据格式根据 PLC 中数据格式来设定。

c) 单击“确定”按键，结束设定，并关闭这个对话框。这样会看到新增了一个 D0 这个地址来切换基本窗口。当 PLC 的 D0 寄存器得到一个数据，且与工程中某一个画面编号相同时，人机界面会自动的切换到这个画面上。当 PLC 的 D0 寄存器中得到的数据，在工程中无法找到与之相同编号的画面时，则人机界面将不做任何切换画面的动作。

6.3.2 直接窗口和间接窗口

如果想使用 PLC 来弹出小的窗口到一个基本窗口上，有两个控件可以实现这个功能，即“直接窗口”控件和“间接窗口”控件。

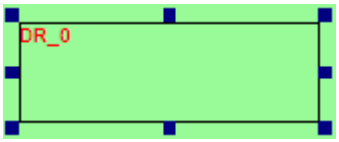
直接窗口控件是定义 PLC 中的某一个位的状态来执行窗口的弹出。当人机界面检测到这个位的状态为 ON 时，则事先定义好的窗口，就会显示在该“直接窗口”控件放置的位置显示出来，并且窗口显示的大小与该控件的“轮廓”一致。

间接窗口控件时定义 PLC 中的某一个寄存器，由这个寄存器中的数据来弹出一个窗口。与“PLC 控制”中的“切换基本窗口”属性一样，当这个寄存器中的数据与某一个画面的编号相同时，该画面将会显示在这个“间接窗口”控件放置的位置，且显示的画面大小与该控件的“轮廓”一致。

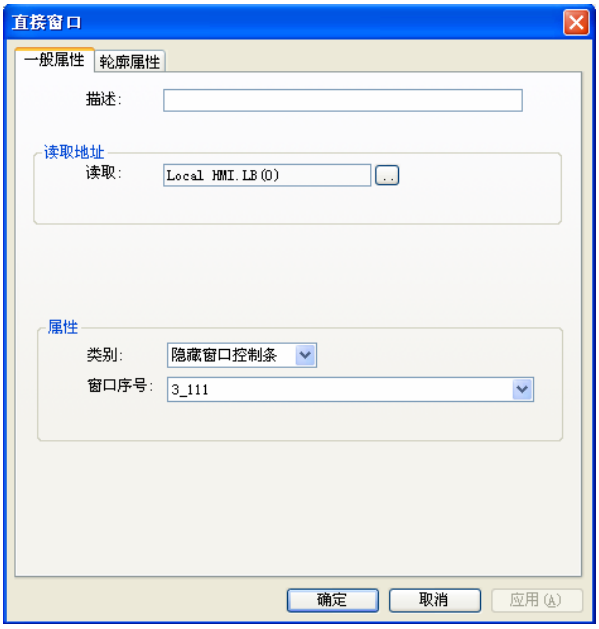
同样的，作为“直接窗口”和“间接窗口”控件来显示的窗口，一般要设定这些窗口的大小要小于满屏的基本窗口。

6.3.3 显示“直接窗口”

1）打开 InoTouch Editor 软件菜单的“控件/直接窗口”，或者工具栏上的图标  ，在窗口中点击鼠标左键，就建立了“直接窗口”控件，如下图所示。



2）选择此控件双击左键或者点击右键选择属性，就可以设置属性了，如下图。



3）选择“读取地址”， PLC 中需要控制直接窗口弹出的“设备类型”和“地址”。

4）在“窗口序号”中选择需要弹出的窗口编号；若需要弹出窗口可以随意在基本窗口里面移动，在“类别”中就选择“显示窗口控制条”，否则选择“隐藏窗口控制条”；

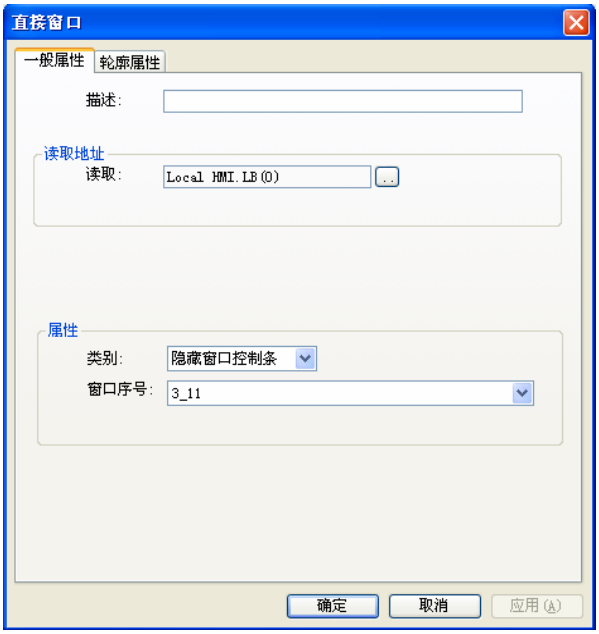
5） 单击“确定”，结束设定，并单击鼠标左键，可以将该控件放置到编辑的画面上；

6） 使用鼠标左键可以拖动该控件的位置，并可以改变其轮廓至合适的大小。

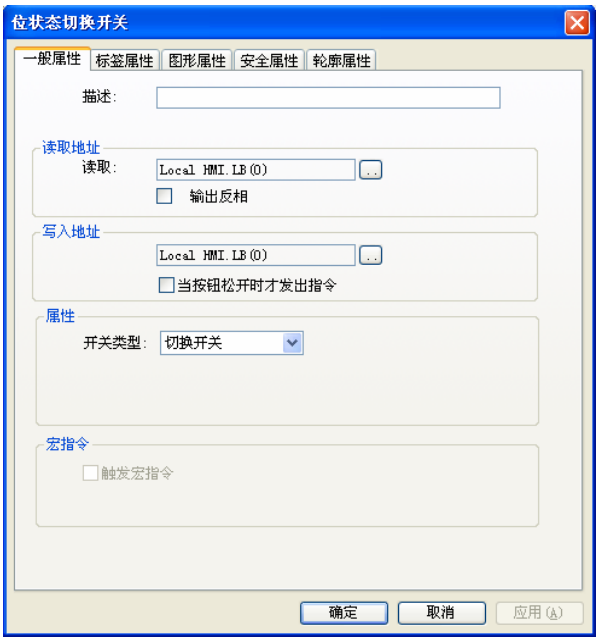
这样就做好了一个“直接窗口”控件。

下面以一个简单的范例来说明“直接窗口”原件的使用。

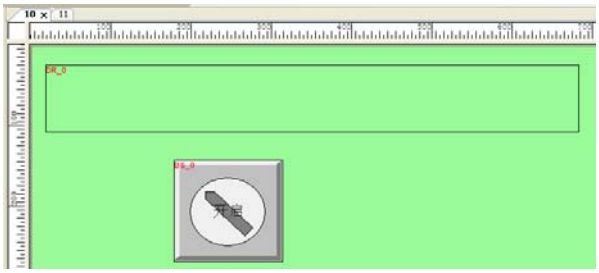
a、 假设已经建立好了两个窗口，一个是窗口 10，一个是窗口 11。在窗口 10 上，建立一个“直接窗口”控件，选择此控件双击左键或者点击右键选择属性，就可以设置属性了，如下图所示。



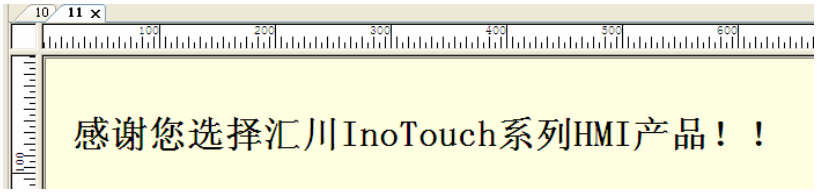
b、 由上图可以看出，是使用 LB0 这个位来控制窗口 11 的显示和关闭。且隐藏了窗口控制条。这样，再做一个“位状态切换开关”控件，读取地址和写入地址均设定为 LB0，“开关类型”设置为“切换开关”。并将状态 0 的文字标签设置为“开启”，状态 1 的文字标签设置为“关闭”，如下图所示。



c、 最后，放在画面 10 上的位置如下图所示。

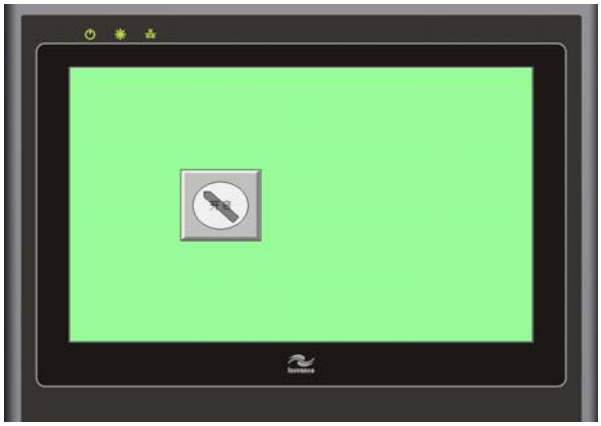


d、 窗口 11 的内容如下图所示。



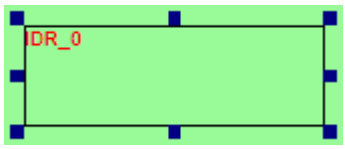
e、 执行离线仿真，单击一次“位状态切换开关”控件，就可以将 11 号窗口显示在“直接窗口”原件的位置和轮廓大小，如下图所示。

f、 此时，再按一次“位状态切换开关”控件，将会把 11 号窗口关闭，在“直接窗口”原件的位置上消失。如下图所示。

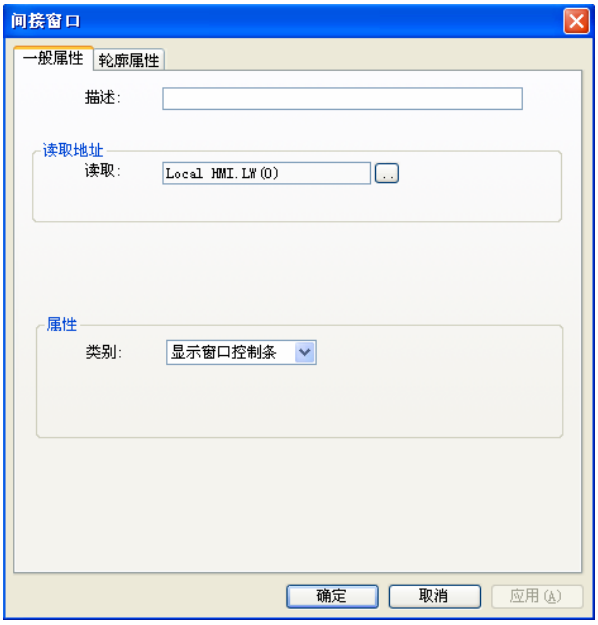


6.3.4 显示“间接窗口”

1）打开 InoTouch Editor 软件菜单的“控件/间接窗口”，或者工具栏上的  图标，在窗口中点击鼠标左键，就建立了“间接窗口”控件，如下图所示。



2）选择此控件双击左键或者点击右键选择属性，就可以设置属性了，如下图。



3）选择“读取地址”， PLC 中需要控制间接窗口弹出的“设备类型”和“地址”。

4）选择 PLC 中该数据寄存器的数据格式。

5）如果需要弹出的窗口可以在基本窗口里面移动到任意位置，则在“属性/类别”里面选择“显示窗口控制条”，否则选择“隐藏窗口控制条”。

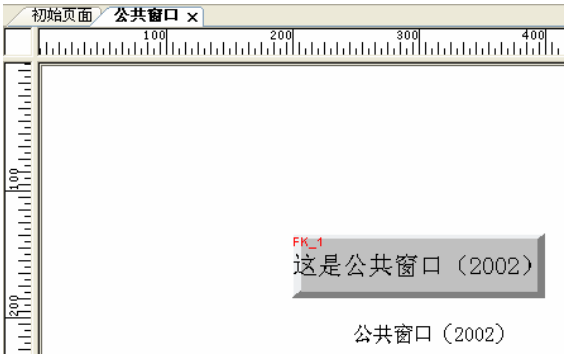
6）单击“确定”结束设定， 可以使用鼠标将该控件移动到合适的位置，或者将轮廓调整为合适的大小。

这样，就建立了一个“间接窗口”显示控件。由上图的设定可以看出，是由 LW0 这个寄存器来控制“间接窗口”的显示的。当 LW0 的值=11 时，窗口 11 将显示在该控件放置的窗口位置上，当 LW0 的值=12 时，窗口 12 将显示在该控件放置的窗口位置上，依此类推。当然，这个数值编号的窗口必须存在，就会正常显示。例如如果 LW0 的值=20，如果 20 号窗口不存在，则该窗口不会显示出来。

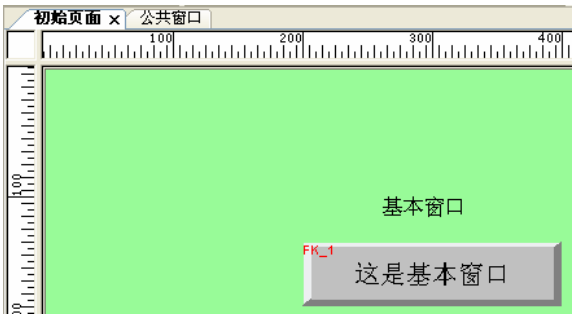
6.3.5 公共窗口的使用

在工程画面中，有时候需要某一个信息，不论目前打开的是哪个基本窗口，都需要显示出来。这样可以将这些共同一致要显示的控件放置在公共窗口即可。

当新建任意一个 InoTouch Editor 软件的工程画面时，公共窗口（也即窗口 2002）会自动的建立在工程中。由于放置在公共窗口的物件在任何满屏的基本画面均会显示出来，为了不至于这些物件挡到，一般这些物件放置在公共窗口的顶部或者底部等位置。若是相同位置的话，会根据“HMI 系统配置/HMI 设置/公共窗口显示”设定来决定。下面以两张设定后的效果图来说明。

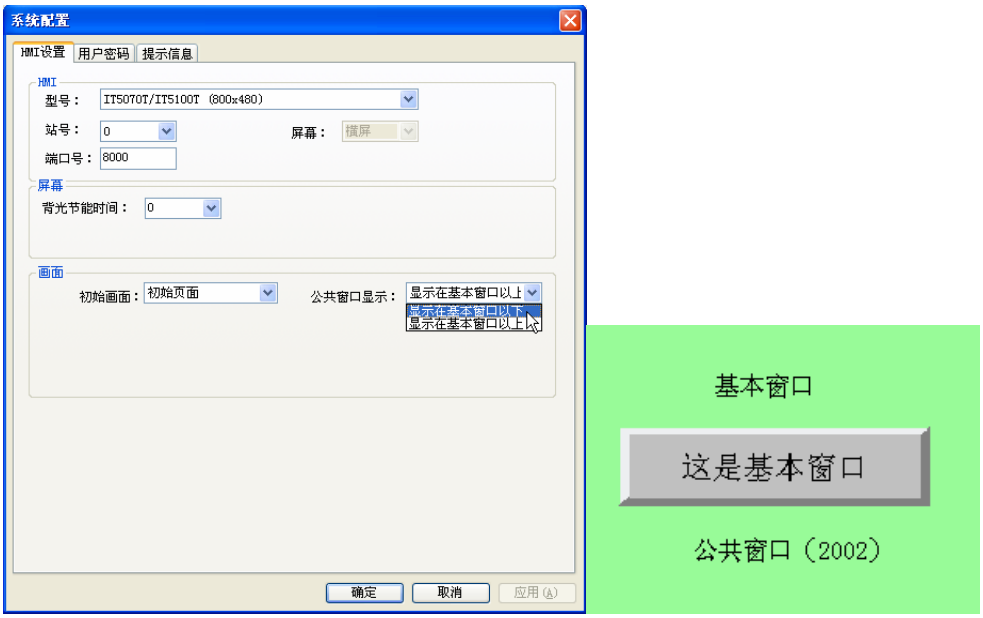


公共窗口画面内容



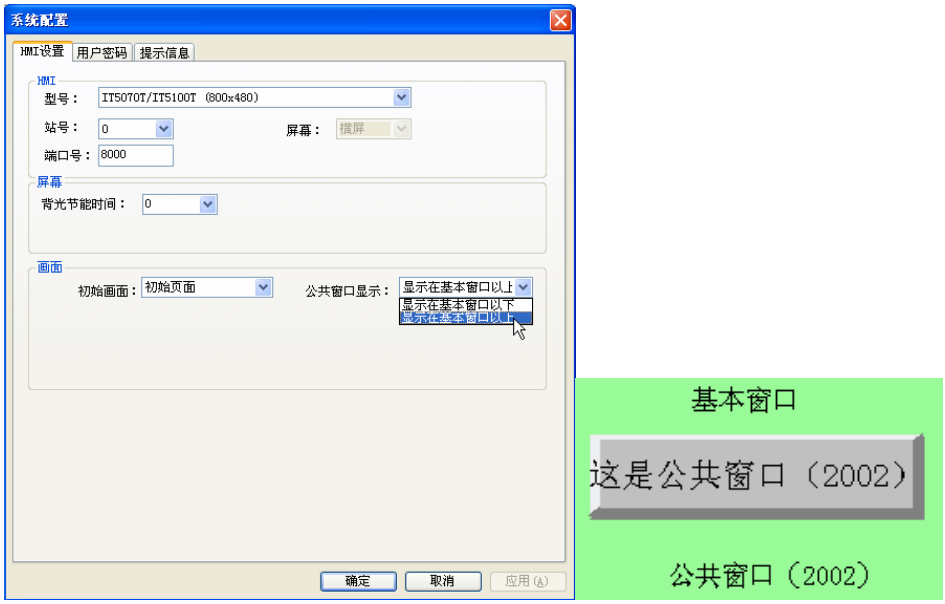
基本窗口的内容

当 HMI 系统设置中的公共窗口属性设置为“显示在基本窗口以下”时，得到的显示结果如下图所示：



这时可以看到，相同位置的物件，基本窗口的物件在上层，公共窗口的物件在下层。

当系统参数中的公共窗口属性设置为“显示在基本窗口以上”时，得到的显示结果如下图所示：



这时可以看到，相同位置的物件，基本窗口的物件在下层，公共窗口的物件在上层。

注意：窗口内控件重叠时，只有最上面的触控有效！

小结：通过以上叙述，了解到 InoTouch Editor 软件编辑画面时，窗口的概念和各种窗口的类型。其实，InoTouch Editor 工程画面就是由各种窗口来组成的。常用的是基本窗口，而弹出窗口、直接窗口、间接窗口等，这些是由 InoTouch Editor 软件提供的控件，注意，他们是 InoTouch Editor 软件提供的控件，而非本身的窗口。只是通过这些控件，可以将需要的窗口显示在基本窗口上，且这些来源窗口一般要小于基本窗口的大小。而使用公共窗口等，可以编辑出类似 Windows 操作方格的画面结构。多种系统讯息窗口，且讯息可以自己定义，提高了系统维护的便捷性。另外需要注意一下“直接窗口”控件和“间接窗口”控件的区别。“直接窗口”控件是使用一个位地址来控制指定的窗口显示出来；“间接窗口”控件是使用寄存器的值来控制窗口显示出来。

7

图形库、声音库的建立与使用

第七章 图形库、声音库的建立与使用

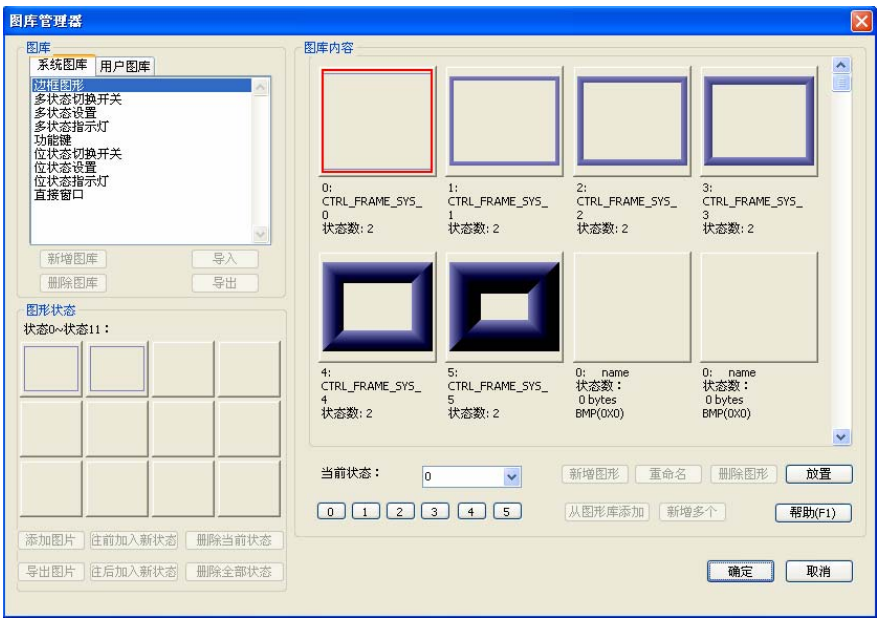
InoTouch Editor 软件提供的图库，分为系统图库和用户图库。系统图库是 InoTouch 软件系统提供的图形，只能使用，不能修改；用户图库：您可以根据自己的喜欢建立您自己的图形图库。

每个图形库最多可包含 256 个状态，也就是说，在图形库的某一个图库同一个位置最多可以使用256张向量图或者图形。那什么是“状态” 呢？状态的概念可以理解为不同的向量图或者图片。通常使用“位状态指示灯”来显示，PLC 中某一个位的状态是 ON 还是 OFF，使用“多状态指示灯”来显示 PLC 中的寄存器的不同数据或者机器的不同工作进程等等。在显示这些不同的状态时，就是使用不同的图形，有时候还搭配不同的文字标签来显示。所以，在指示 PLC 中的某个位的状态或者某个寄存器数据不同状态时，至少要有两张不同的图形。下面将说明如何建立图形库。

系统图库的使用请参考《控件的一般属性》章节。

7.1 用户图库的建立

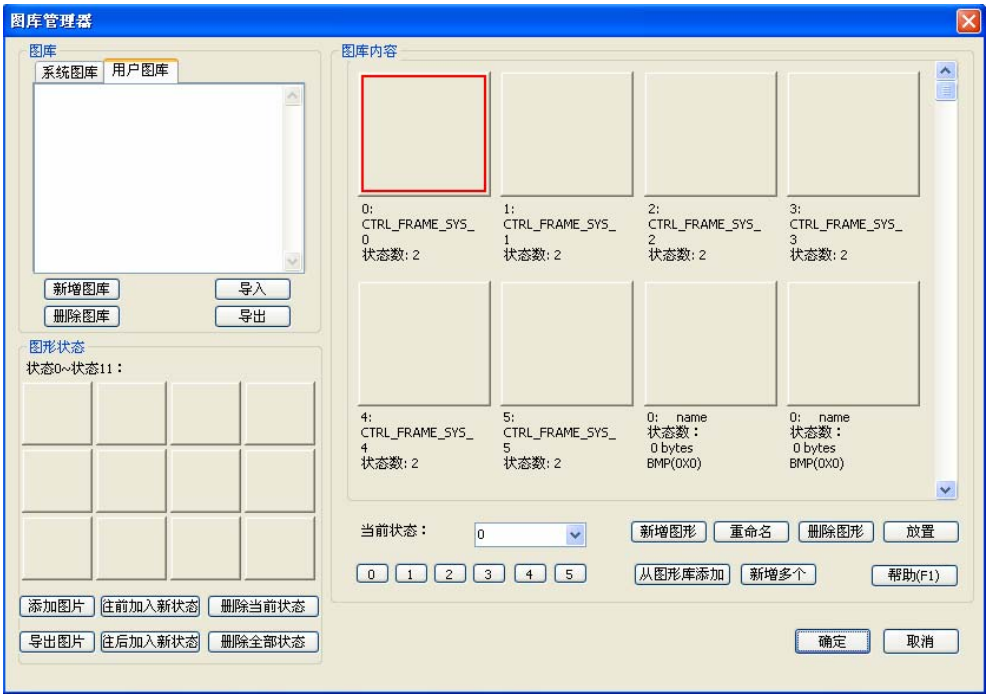
打开 InoTouch Editor 软件菜单的“媒体库/打开图库”，或者工具栏上的图标  ，参考下图：



[系统图库]

系统图库是 InoTouch 软件系统提供的图形，只能使用，不能修改；要选择使用哪一类图形需点选图库名称，选择图形使用即可。

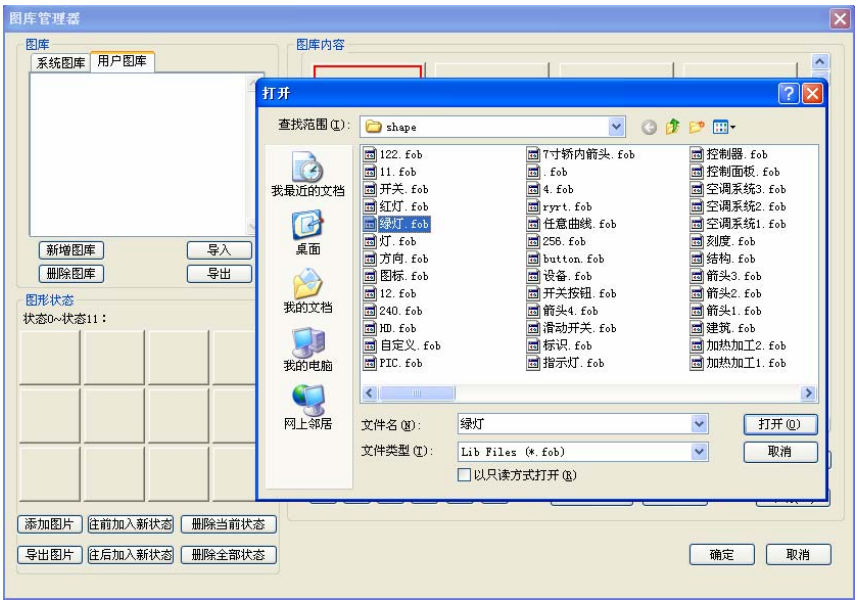
点击用户图库，见如下图：



a、图库

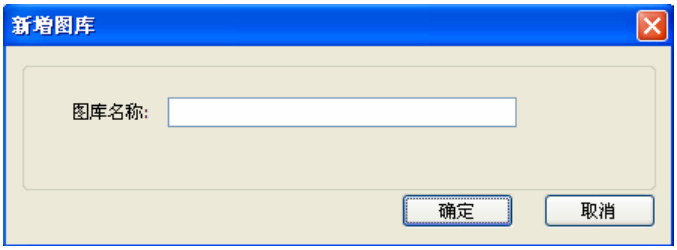
[导入]

按下该按钮后可出现下图的画面，可选择要加入此工程画面的图库。当点击一个图库的名称时，可将该图库里面的图形导入到用户图库中。



[新增图库]

按下该按钮后可出现下图的画面，可用来增加一个空的图库。图库名称可以自己命名。



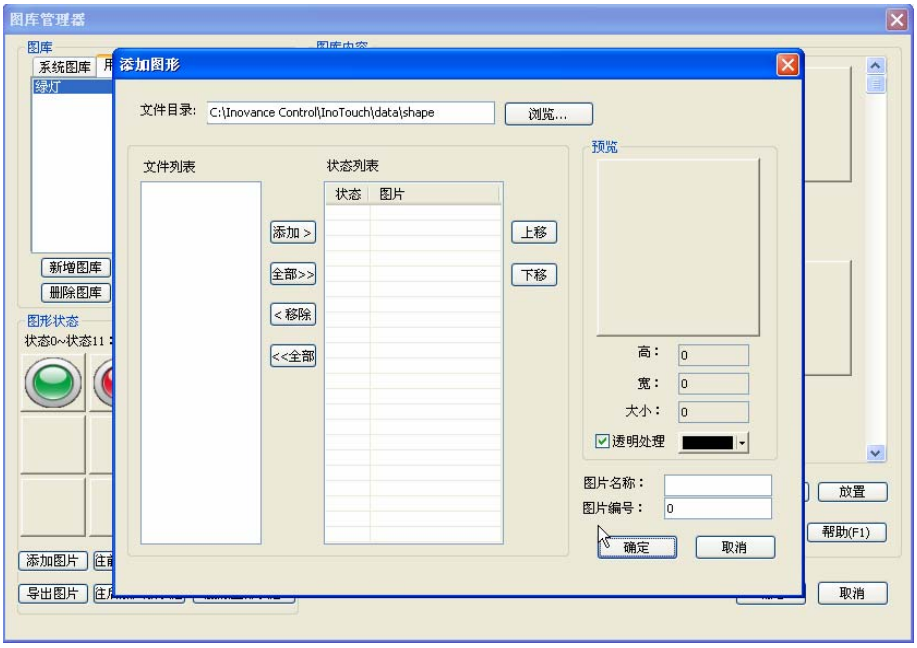
[删除图库]

选择要删除的图库名称，按下该按钮后，即可将[用户图库]中显示的图库从此工程画面中剔除。注意只是将选择的图库从当前的工程中删除，但是 InoTouch Editor 软件的根目录文件夹 shape 里面并没有删除这个图库文件。

b、图库内容

[新增图形]

增加自己需要或喜欢的图形。按下该按钮后可出现下图的画面，选择浏览，找到图形的文件，确定之后，文件列表就会显示图形的名称，选择需要加入此图库的图形，点击“添加”可将该图形添加到状态列表中。



[全部>>]

全部是指将文件列表所有的图形全部添加到状态列表中。

[<移除]

从状态列表中删除所选图形。

[<<全部]

删除状态列表中所有的图形。

[上移]/ [下移]

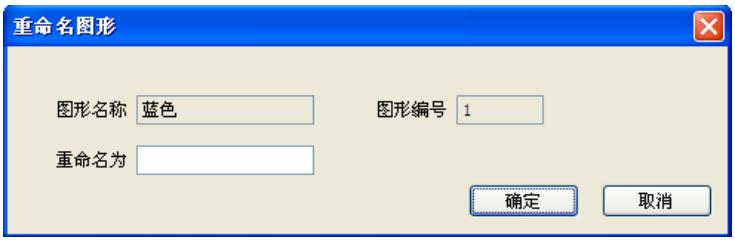
将选择的图形上移或下移。

[透明处理]

该颜色是透明的看不见的，即不显示。如将红色设成透明处理颜色，那么所有的红色将不显示出来。

[重命名]

按下该按钮后将出现下图的对话窗，可重新命名目前选择的图形。

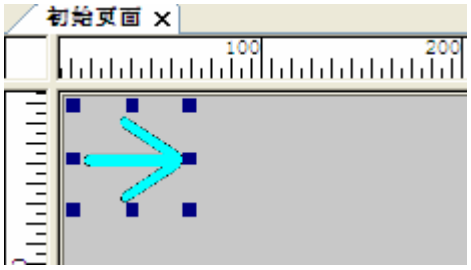


[删除图形]

用来删除所选择图形的全部状态。也即将这个图库的图形清空。

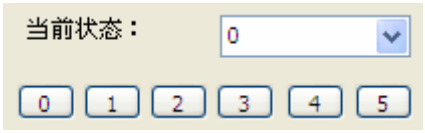
[放置]

可将目前选择的图形输出到使用中的窗口上，如下图所示。可点击图片移到想放的位置。



[当前状态]

选择用户图库当前要显示的状态，单击“当前状态”后的状态列表，可以更改显示哪个状态时的图形。当窗口中未显示图形时，表示该图形不存在，或此图形在当前的状态并未被定义。



[从图形库添加]

把另外一个图形库里面的图形全部添加到当前图库。

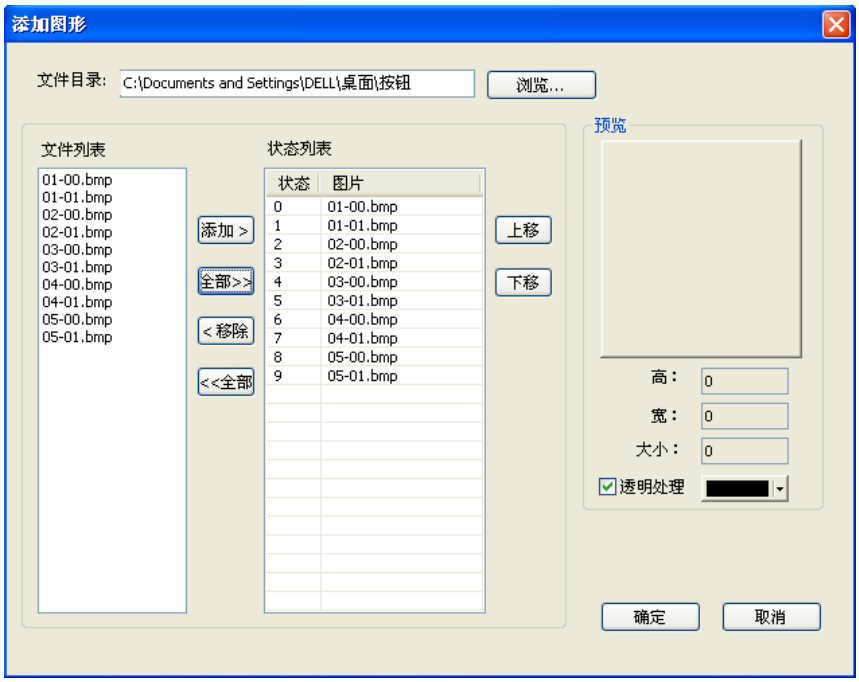
[新增多个]

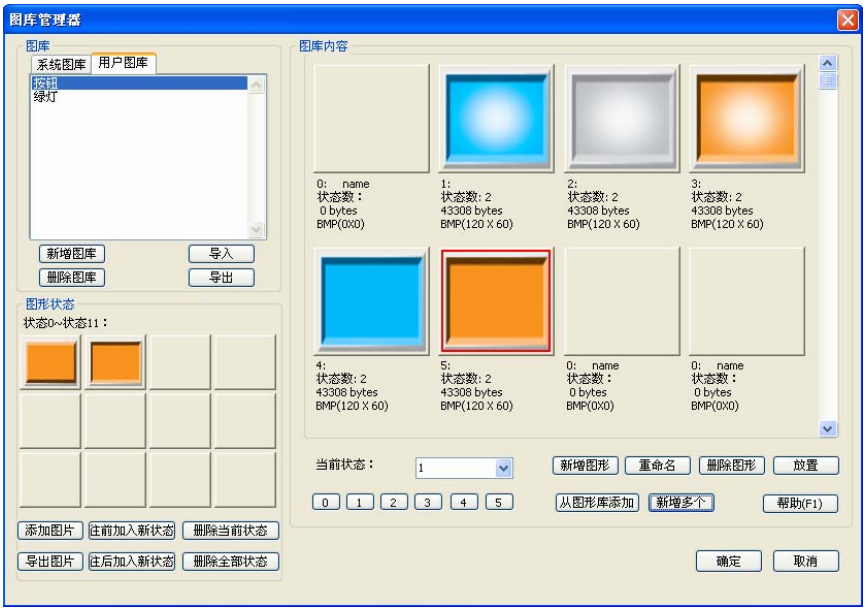
可以同时增加多个图库多个状态，但是需要先把图形名称编辑好，按照一定命名规则，把多个位图（JPG、BMP、GIF）一次添加到当前图库中。

命名规则如下：

XX-##.jpg，其中 XX 表示图形 ID，占 2 个字符，##表示图形状态号，占 2 个字符，图形 ID 和状态号之间用一个字符隔开。

例如：现在有 10 张位图，分别把它命令为 01-00.jpg、01-01.jpg、02-00.jpg、02-01.jpg、03-00.jpg、03-01.jpg、04-00.jpg、04-01.jpg、05-00.jpg、05-01.jpg





通过【新增多个】按钮，就可以一次把这些位图一次分别添加到当前图库的 ID 号位 1、2、3、4、5 的图形中，提高效率。

c、图形状态

[添加图片]

往选定的图形库中增加一个新的图形。

[删除当前状态]

用来删除当前所选择位置的图形所显示的状态。

［删除全部状态］

用来删除目前所选择位置的图形的全部状态。

［导出图片］

单击该按钮，可将目前选择的图形输出到指定的位置，让使用者可以获得原始图形。

[往前加入新状态]

在目前所显示的图形状态前面加入一个新状态。例如目前选定的该图片为状态 1，那么新增的图形将会为状态 0 的图形。同时，该位置的图形总状态数也增加了一个。

[往后加入新状态]

在目前所显示的图形状态后面加入一个新状态。例如目前选定的该图片为状态 1，那么新增的图形将会为状态 2 的图形。同时，该位置的图形总状态数也增加了一个。

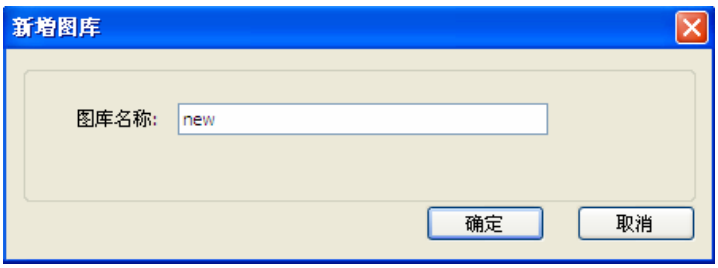
1）新建一个图片式的用户图库

下面说明如何新建立一个新的用户图库，并在此图库中加入一个具有两个状态的图形。一般来说，加入到图形库中的图片可以使数码相机拍的照片，或者 photoshop、Windows 下的“画图”工具等形成的图片。图片格式可以使 BMP、JPG 和 GIF 三种格式。

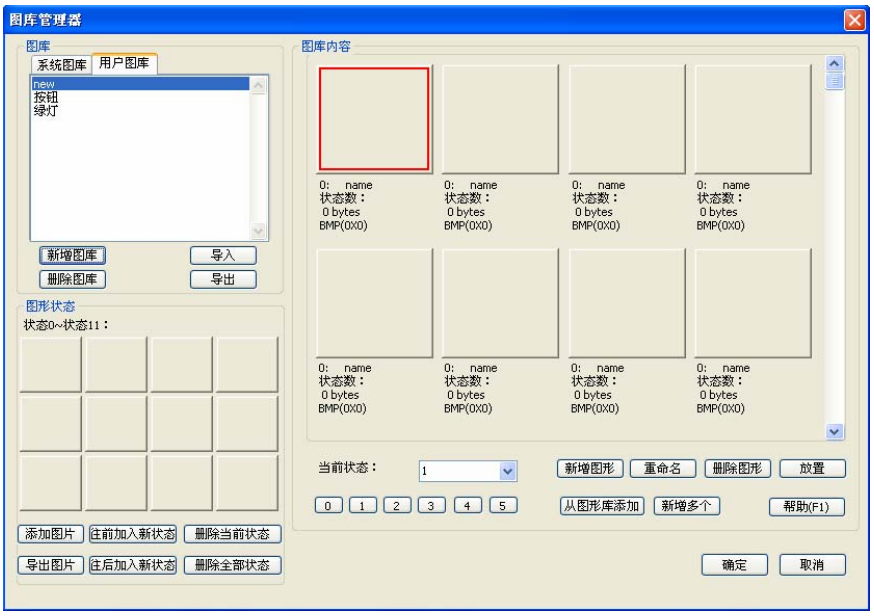
注意：添加图形到图形库时，该图形的分辨率不要超过使用机型的分辨率。

步骤一：新增图库

按下[新增图库]后，在对话窗中输入新的图形库名称。



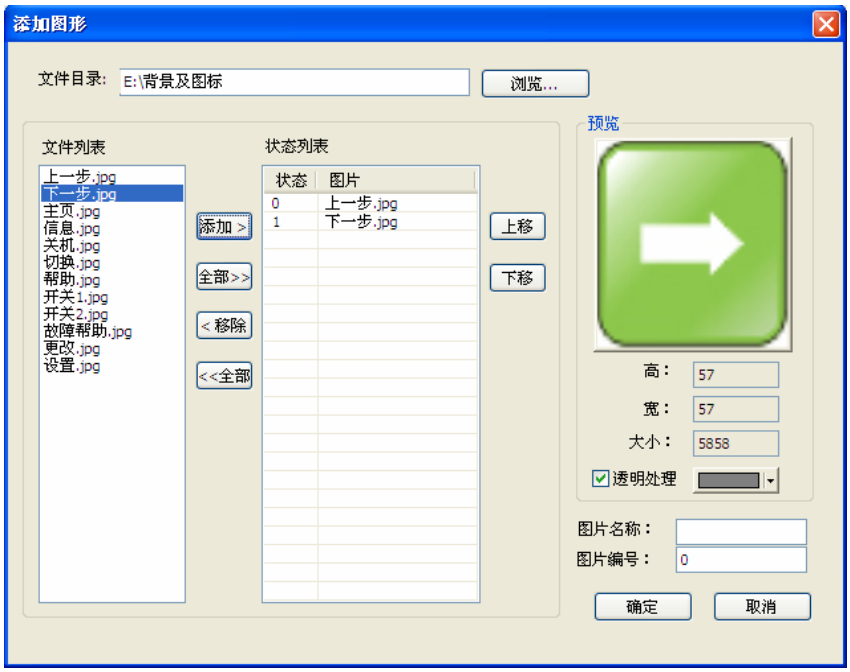
此时可以发现图形库管理对话窗中增加一个新的图形库“new”，且此新的图形库中并未包含任何图形，见下图。



步骤二：选择需要添加的图形

先用绘图工具准备好要加入的图形；假设现在要将下面的两个图形分别用来表示状态 0 与状态 1。

首先按下[新增图形]按键，可出现下图的对话窗，这时浏览图形所在文件，“确定”之后，文件列表中选择要使用的图形，添加到状态列表中，最后按下“确定”。



在完成上述的各项动作后，即建立一个完整的图形，可参考下图。这时在图形管理对话框中

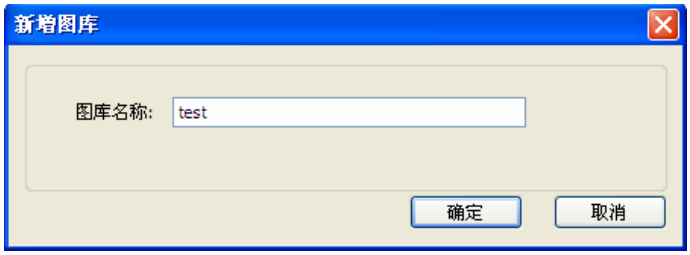
可以发现新加入的图形“Lamp”，由图片信息中也可看出此图片为 JPG 形式，且包含两个状态。

2）新建一个矢量图形的用户图库

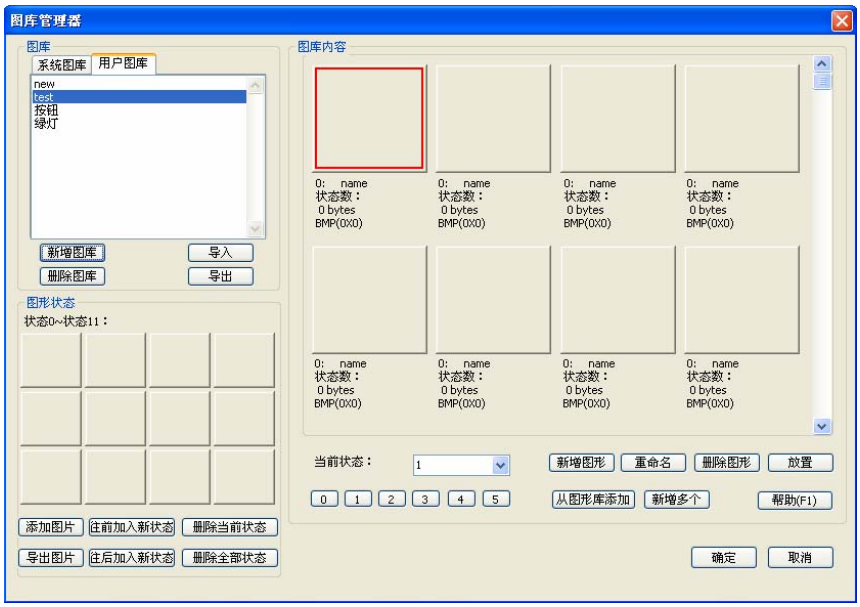
InoTouch Editor 软件也可以添加矢量图到用户图库里面来使用。下面说明如何新建立一个新的矢量图用户图库，并在此图库中加入一个具有两个状态的矢量图。

步骤一：新增图库

按下[新增图库]后，在对话窗中输入新的用户图库名称。



此时可以发现用户图库管理对话窗中增加一个新的用户图库“test”，且此新的用户图库中并未包含任何矢量图，参考下图。

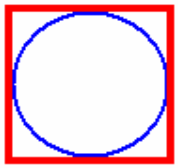


这样就新建了一个名称为 test 的矢量图用户图库，单击“确定”关闭对话框。

步骤二：选择需要增加的矢量图

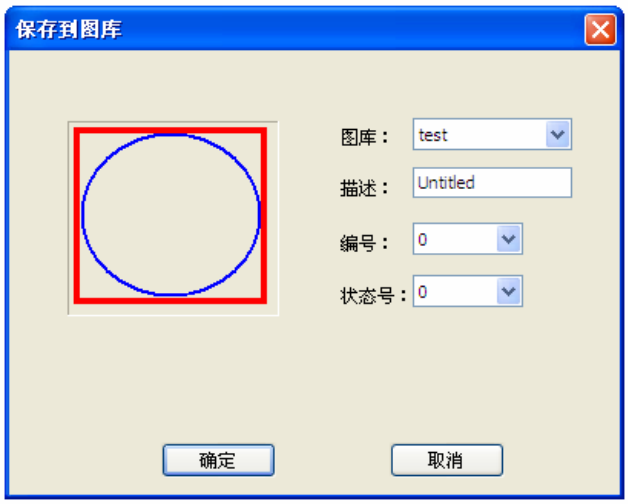
先使用 InoTouch Editor 软件提供的画图工具画出需要的图形。例如，使用“矩形”和“椭圆”画图工具，画出的图形如下图所示。

然后拖动鼠标将这个图形圈选起来，或者使用“群组”功能将其组成“一个”图形。



接着按下工具列上的图标 

，或者 InoTouch Editor 软件菜单“媒体库/保存到图库”按钮，可以得到下面的对话窗：



对话框上各项参数的说明如下。

[图库]

选择目前的图形是要加到哪一个用户图库中，目前选择“test”。

[描述]

矢量图的名称，可以不写

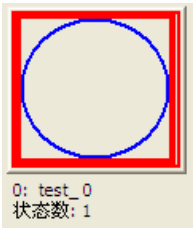
[图型编号]

选择目前的向量图形要加到“test”向量图库中的哪一个位置矢量图中。每一个用户图库有0~55 总共 56 个编号的位置。在此设定为 0，表示添加到编号为 0 的位置中。

[状态]

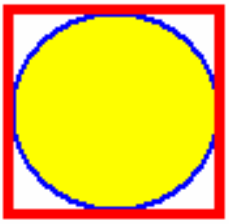
已经选定了向量图形保存的位置为编号 0。每一个编号位置的向量图形状态最多可以到 256 个状态。在此选择为 0，表示该向量图形作为状态 0 时的图形。

上图的信息也显示“ test” 用户图库中编号 0 的矢量图，目前的状态(state 0)并未定义任何边框与内部。在按下确认键后可以发现图形已经加入到用户图库中，见下图。由下图也可看出编号 0 的矢量图只具有一个状态。

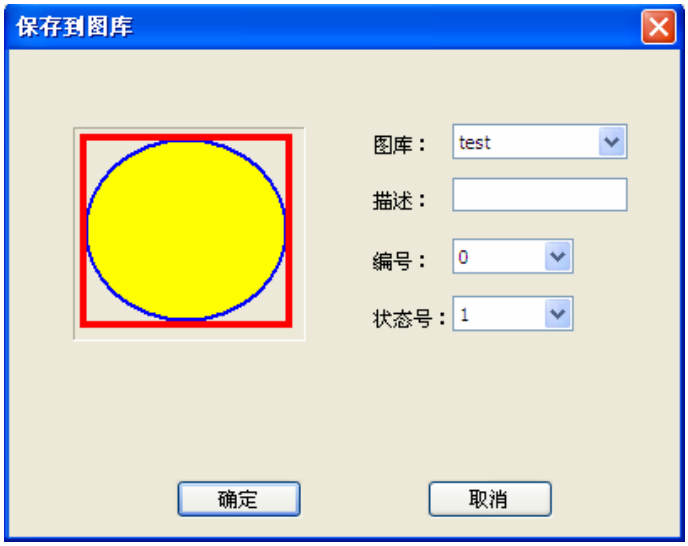


步骤三：添加矢量图的另一种状态

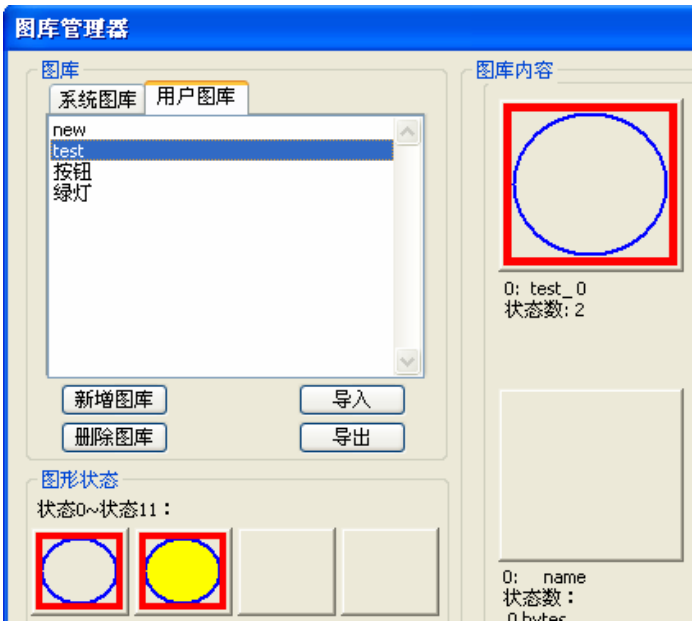
使用步骤二相同的方式，可以添加新的矢量图形到用户图库中。假设新的矢量图形如下图所示。



新加入的图形选择作为编号为 0 的位置图形，且状态为 1，则设定如下图。



在完成上述的各步骤操作后，即建立了一个新的矢量图用户图库，并且在位置 0 添加了两种状态的图形，结果如下图所示。



7.2 声音库

目前只有人机界面配置有音频输出口，接上音箱后，就可以根据需要在程序中设置播放声音 InoTouch Editor 软件支持的声音文件为 WAV 格式，这些格式的文件是保存在声音库中的。同图形库的图形一样，也可以对保存在声音库中的各声音文件进行添加新的声音文件、删除已有的声音文件等等操作。

打开声音库

单击 InoTouch Editor 软件菜单上的“图库/声音库”或者工具栏上的图标，则弹出的对话框如下：

[新增声音库...]

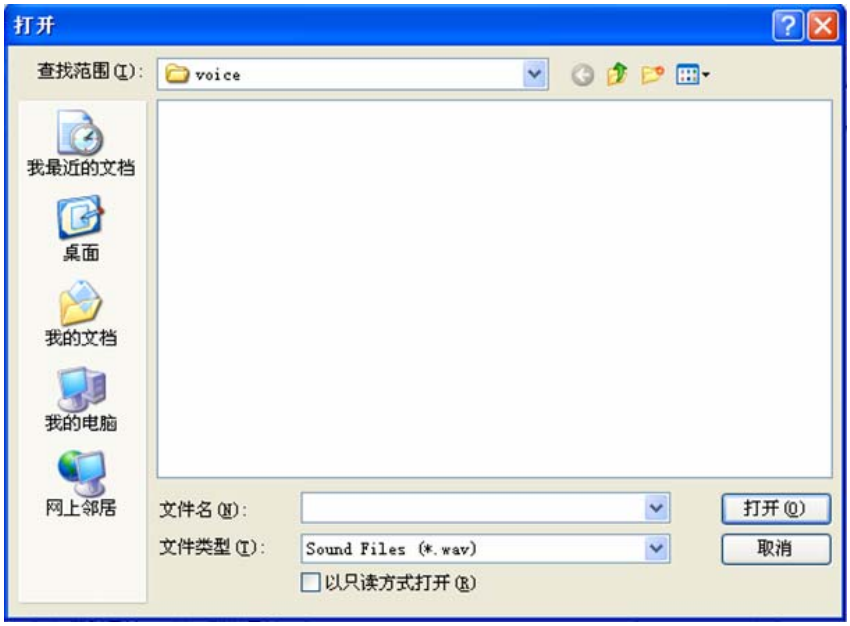
单击该按钮，将会新建一个声音库，操作方法与新建用户图库等方法是一模一样，在此就不在累述。

[删除声音]

单击该按钮，将会把目前打开的声音库中选择的声音从当前声音库中删除。

[新增声音...]

先录制或者生成一个 WAV 格式的文件保存到计算机中。单击[新增声音...]，将会弹出如下对话框。



浏览找到保存的 WAV 文件，单击“打开”后，就会将该声音文件添加进来。

[播放声音]

先选择当前打开的声音库中的某一个声音文件，再单击该按钮时，将会把这个声音文件播放出来。也即可以先听一下这个声音文件的声音效果。

[导出声音...]

单击该按钮，将会把当前选中的声音文件导出到计算机中，且文件也是为 WAV 格式。

8

文字标签库与多国语言显示

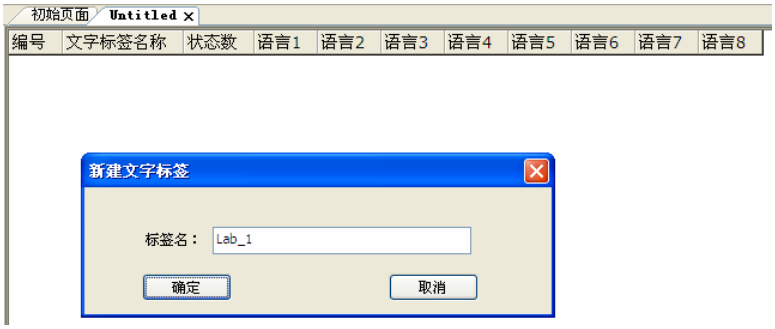
第八章　文字标签库与多国语言显示

有些情况下，一个工程画面中，需要显示多种语言文字。特别是做出口设备的厂家，工程画面中会用到多种不同的语言文字，以适应机器出口后能够让当地的操作工人看懂画面上的各种文字信息。例如，一个工程画面中需要使用到中文、英文、俄罗斯文等文字显示。如果将相同的画面做三幅不同的文字显示，这样画面编辑的工作繁琐且效率低。

因此，使用 InoTouch Editor 软件提供的“文字库”功能，就可以轻松的实现多语言文字显示和切换。InoTouch Editor 软件目前支持最多 8 种语言文字的显示，这些语言文字可以从系统支持的语言文字中任意挑选 8 种即可。事先将需要的文字建立在“文字标签库”中，在编辑画面时选择需要的“文字标签”即可显示不同的语言文字。下面来详细说明如何实现这个功能。

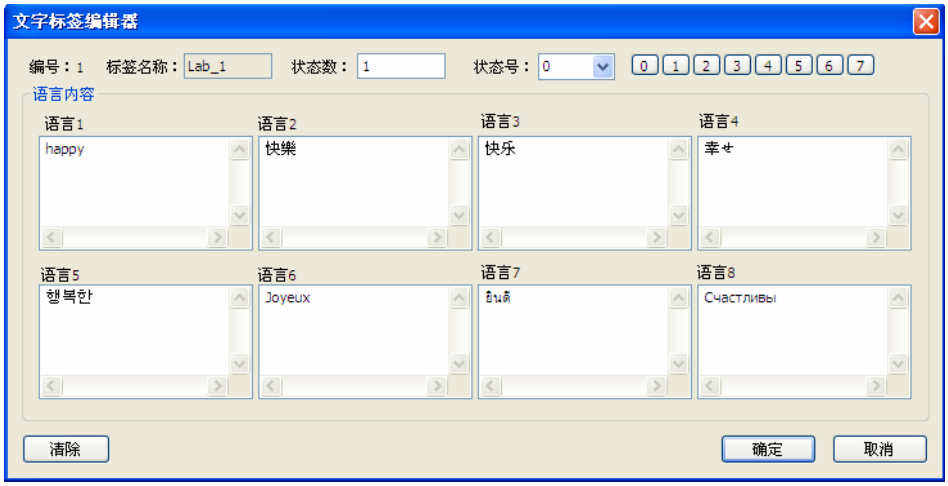
8.1 文字标签库的相关操作说明

单击 InoTouch Editor 软件左侧项目管理下面的“文字标签库”，即会弹出如下对话框。



选择“确定”，就会弹出如下对话框。

编辑各语言的文字内容，见如下对话框。编辑完成后单击“确定”回到“文字标签库”页面。



初始页面 | 文字标签库

编号	文字标签名称	状态数	语言1	语言2	语言3	语言4	语言5	语言6	语言7	语言8
1	Lab_1	1	happy	快樂	快乐	幸せ	Felice	Joyeux	Glücklich	Счастливы

[状态数]

显示文字标签库中当前状态的文字内容，最多可以有 256 中状态的文字。

[新增标签项]

在“文字标签库”页面单击鼠标右键，选择“新增标签项”表示新增加一个文字标签。

[编辑标签项]

对选择的文字标签的文字内容进行编辑。

[删除标签项]

单击此功能键将会把目前选中的文字标签库删除。

[全部删除]

单击此功能键将会把文字标签库中的所有文字标签均会删除。

[导出...]

单击此功能键，将会把文字库中所有的文字标签以 Excel 的方式保存起来，文件名的结尾为.xls 格式。这样可以将工程中建立的文字标签保存起来，重复使用。

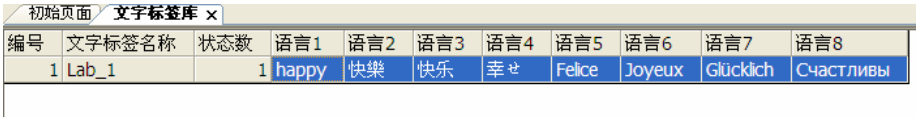
[导入...]

将保存在计算机上的文字标签库导入到目前的工程中。

[字体]

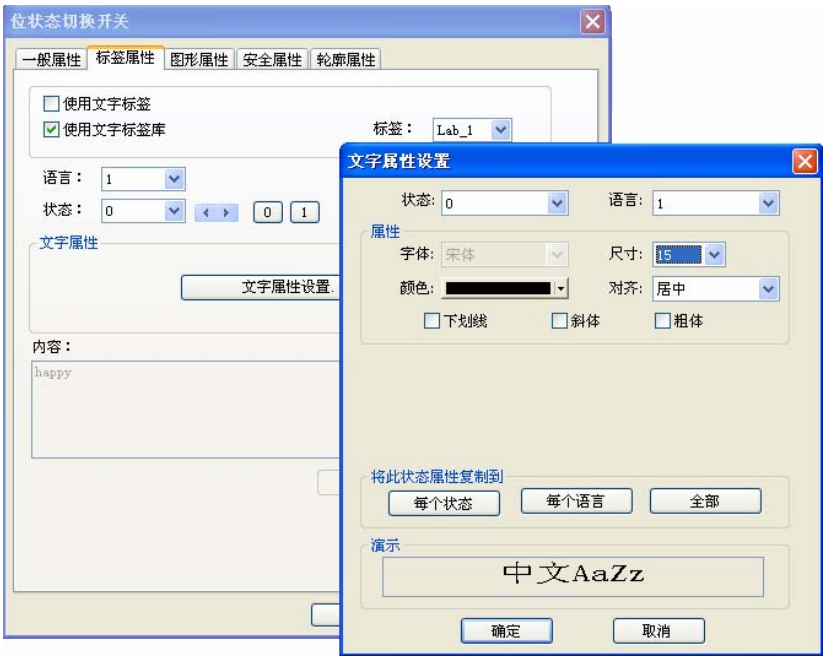
设置各语言文字的字体。在“文字标签库”页面单击鼠标右键，选择 “文字”，即可对各语言文字设定字体，见下图。单击“确定”后回到“文字标签库”页面。

建立后的文字标签库如下图所示。



8.2 文字标签库的使用

当在文字标签库里面建立有文字标签后，在控件的“标签”属性栏中，就会有“使用文字标签库”的选项。勾选此项时，表明使用“文字标签库”。



[将此状态属性复制到]

a.、每个状态

当按下此按键时，将会把当前状态下设置的各属性复制到其他状态中。即所有的状态属性都

一样。例如：字体颜色、对齐方式、字体大小等属性。

b、每个语言

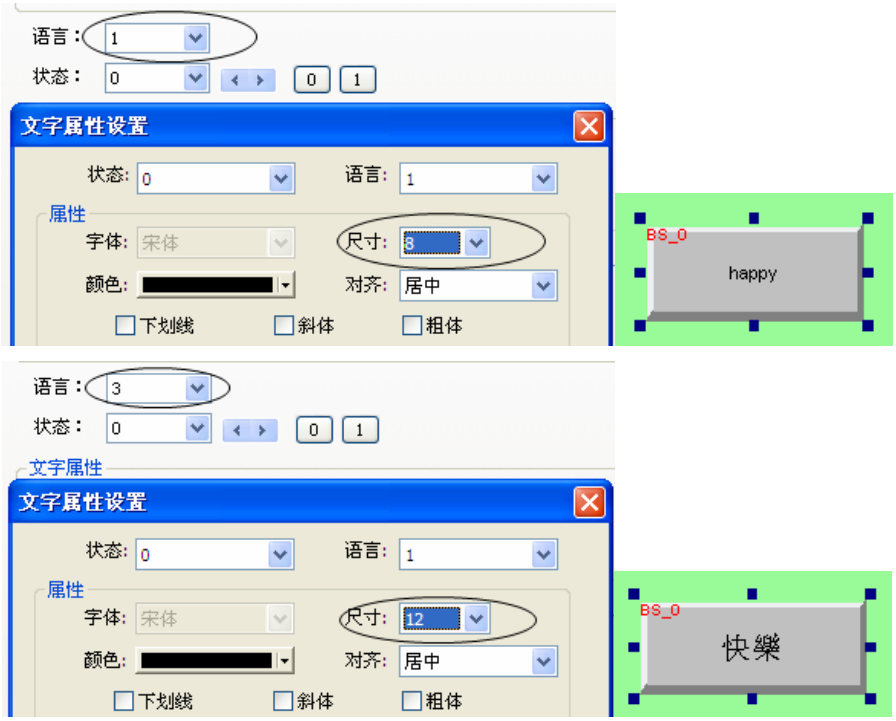
当按下此按键时，将会把当前语言设置的各属性复制到其他状态中。即所有的语言属性都一样。例如：字体颜色、对齐方式、字体大小等属性。

c、全部

当按下此按键时，将会把当前状态和显示语言的属性设置，复制到所有状态、所有语言显示文字的属性中。也即所有的状态和语言显示属性都一样。例如：字体颜色、对齐方式、字体大小等属性。

在“标签”后的文字标签库中选择需要的文字标签，此时可以看到在“字体”和“内容”中，有之前建立文字标签时选定的字体和文字内容，只是在此处不可改而已。

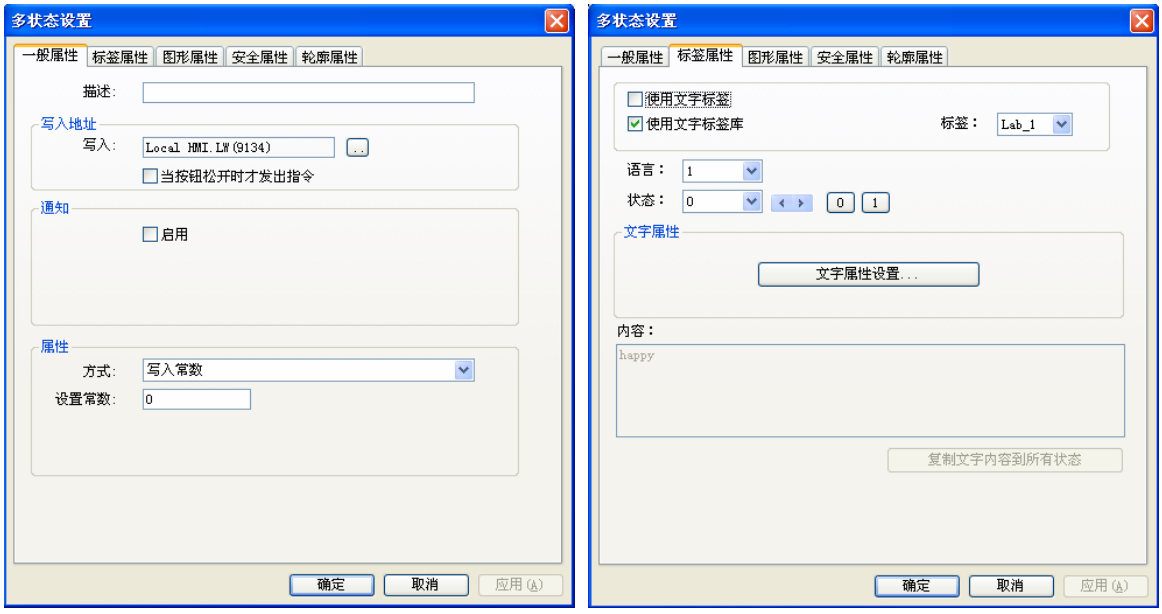
由于字体不一样，且语言文字内容也不一样，有时候就会遇到文字长度不一样的情况。例如：“happy”和“快乐”这两种语言文字当作为英文和中文对照文字使用时，前者的文字所占的空间明显要比后者要大。InoTouch Editor 提供了不同的语言文字其字体可以设置为不同字体大小的功能。利用这个功能，可以将占空间大的文字字体改小的方式，可以达到不同语言文字在同一个位置显示时空间匹配的问题。假如语言 1 的文字为“happy”，语言 3 的文字为“快乐”，那么在控件的“标签”属性对话框中，分别设定语言 1 的字体大小为 8，语言 3 的字体大小为 12，即可。设定后的图如下所示。



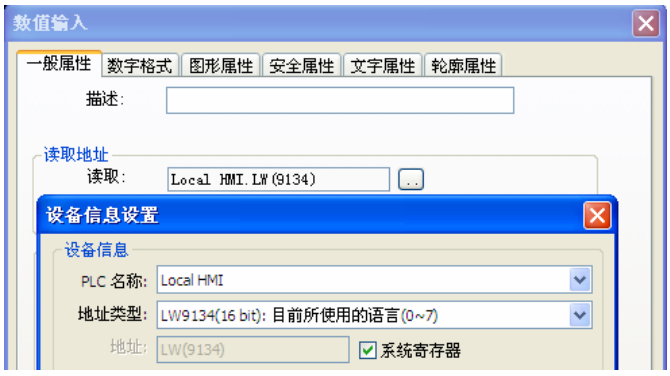
8.3 多国语言的显示

建立了文字标签库，且在控件的“标签”中使用了文字标签库。那么如何在 IT5070T 的工程画面中来显示不同的语言文字呢？InoTouch Editor 软件是通过改变“系统保留字”LW9134 这个寄存器的内容来实现显示不同的语言文字的。LW9134 的有效设定范围为 0~7，分别对应的文字标签中“语言 1~语言 8”共 8 中语言文字。也就是说，当 LW9134=0 时，控件的“标签”中显示的是“文字标签库”中语言 1 中设置的文字内容，LW9134=1 时，显示的是“文字标签库”中语言 2 中设置的文字内容，以此类推。当 LW9134 的值超过 0~7 这个范围时，以最后显示的语言文字内容为准。

以上面的说明制作的名称为 Lab_1 的“文字标签库”为例，新建一个“多状态设置”控件，写入地址”设置为“LW9134”，将其“标签属性”选项中勾选“使用文字标签”并选中“Lab_1”作为其要显示的文字内容，设定后的对话框如下，并将该控件放置在画面中。

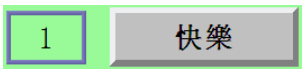


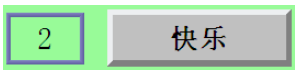
此时在画面上做一个数值输入控件，该数值输入控件的设定如下图所示。

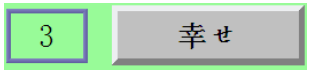


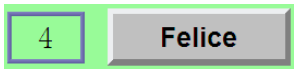
然后将工程画面保存，编译（编译时“语言 1~语言 8”都选勾），并执行离线仿真，当给 LW9134 输入不同的数据时，显示的文字内容也不相同。详见下图所示。

LW9134=0 时， 

LW9134=1 时， 

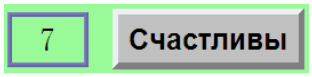
LW9134=2 时， 

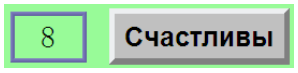
LW9134=3 时， 

LW9134=4 时， 

LW9134=5 时， 

LW9134=6 时， 

LW9134=7 时， 

LW9134=8 时， 

（注：大于 7 时，以最后显示的语言文字内容为准。）

小结：综上所述，建立文字标签库的目的是为了工程画面中能够显示多种不同的语言文字。不同语言文字的显示是通过改变系统保留字 LW9134 这个寄存器的值来实现的。为了使用方便，可以在画面上分别做几个“多状态设置”控件，“写入地址”设置为“LW9134”，在“属性”里面选择“写入常数”，这样在画面上就可以很方便的选择显示哪种语言文字了。

9

地址标签库的建立与使用

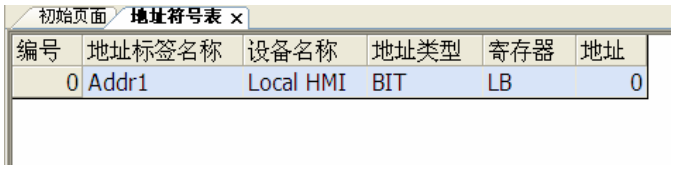
第九章　地址标签库的建立与使用

一般在设计 PLC 程序时，有时候为了很方便的读懂 PLC 的程序，往往使用“注释”的方式，来说明使用的各地址的具体含义，例如“M0”中间寄存器为“开机”，“M1”为“停机”等具体的文字来说明这些软控件在程序中的具体作用，可以让读程序的人很快的就读懂程序，也方便其他人来修改程序。同样的，InoTouch Editor 软件也提供了这个功能，就是“地址标签库”。

9.1 地址标签库的建立

单击 InoTouch Editor 软件左侧项目管理下面的“地址符号表”，即会弹出如下对话框。

选择“确定”之后，就建立了一个“地址符号表”的页面，如下图所示。



[新增地址标签项]

在“地址符号表”页面单击鼠标右键，选择“新增地址标签项”表示新增加一个地址标签。

[编辑地址标签项]

对选择的地址标签的内容进行编辑。

[删除地址标签项]

单击此功能键将会把目前选中的地址标签库项删除。

[删除所有标签项]

单击此功能键将会把地址标签库中的所有地址标签均会删除。

[导出…]

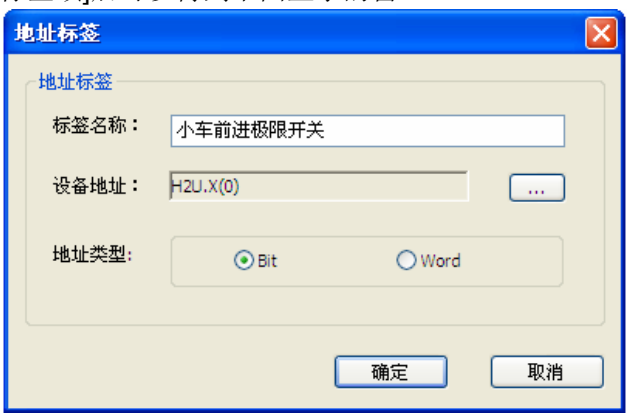
单击此功能键，将会把地址符号表中所有的地址标签以 Excel 的方式保存起来，文件名的结尾为.xls 格式。这样可以将工程中建立的地址标签保存起来，重复使用。

[导入...]

将保存在计算机上的地址标签库导入到目前的工程中。

了解“地址标签库”中各按键的功能后，就可以自己根据工程实际需要，定义各地址的地址标签。下面说明如何建立自定义的地址标签库

在按下[新增地址标签项]后可以得到下图显示的窗口。



[标签名称]

在此输入设定地址的地址标签内容。最多 100 个字符，可以是任意字符内容。

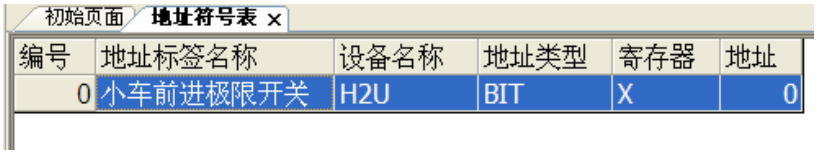
[设备地址]

该名称均来自刚开始建立设备清单时建立的设备名称。表明对哪个 PLC 的软元件地址来定义地址标签。

[地址类型]

即地址模式，可选择“位”或“字”两种类型。

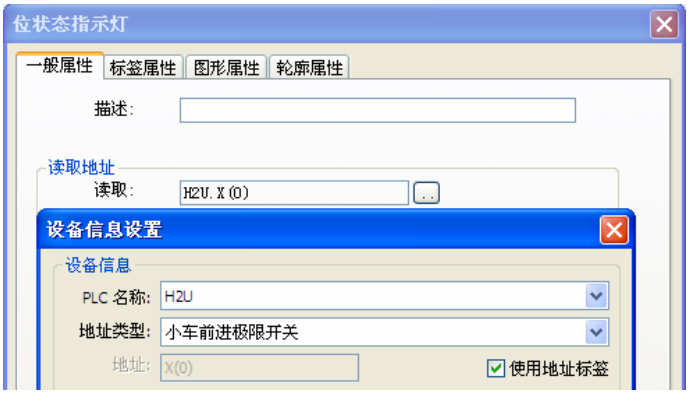
完成以上各项设定后按下确认键，即可在地址标签中发现一条新的标签，参考下图。



重复以上操作，可以建立多个不同的地址标签。

9.2 地址标签库的使用

完成地址标签库的建立，并在控件的属性页中选择与这些标签有关的 PLC 后，即可发现存在“使用地址标签”的选项，勾选此选项后即可使用这些地址标签，参考下图。



此时[地址类型]具有下图所显示的选项：

在设定完成后，点击工具栏上的  图标，即可发现控件所使用的地址标签名称显示在控件上，如下图所示。

这样，选定“小车前进极限开关”这个地址标签，将跟设定“设备类型”为“X”，设备地址为“0”的效果是一样的。其他的使用依此类推。

10

控件的一般属性

第十章　控件的一般属性

控件是放置在工程画面中用来显示所连接的 PLC 或者控制器的相关信息；或者控制所连接的 PLC 或者控制器来执行相关的动作。只有对各控件进行了正确的设置后，这些信息显示或者控制动作就会能够正常的显示和正确的执行。下面先来说明一下所有控件一般属性的设置。

控件一般属性的设置内容包含下面项目：

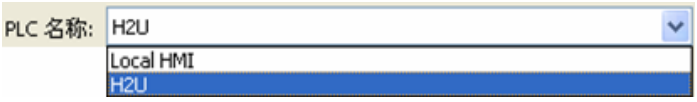
选择 PLC 设备 / 读写地址(reading and writing address)设定；

向量图库(shape library)与图形库(picture library)的使用；

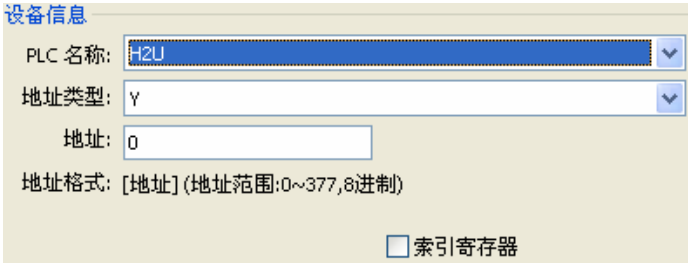
标签内容设置(text) / 轮廓调整(profile)。

10.1 选择 PLC

某些控件的使用需选择要操作的 PLC 对象，如下图所示。[PLC 名称]用来表示要控制的 PLC，下图显示目前存在的 PLC 名称有“Local HMI”与“H2U”，这些 PLC 名称来自“新增设备”。



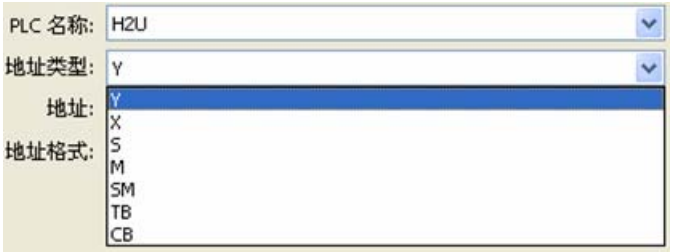
10.2 读写地址设定



上图可以看出一般地址的设定包含下列项目：

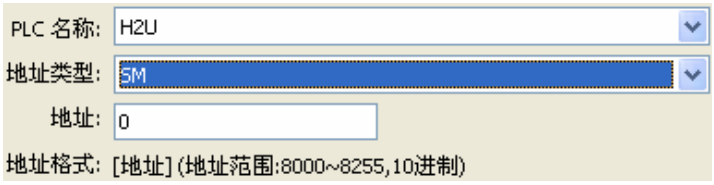
[地址类型]

选择地址类型，当选择不同的 PLC 时，将出现不同的地址类型。



[地址]

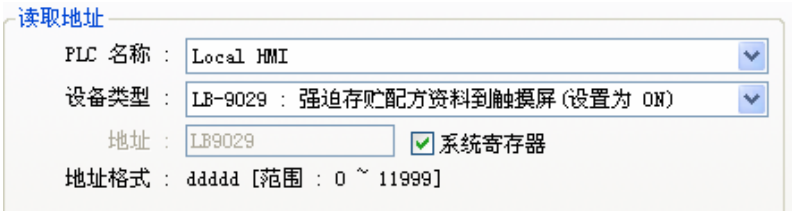
设定读写的地址。如果不清楚该如何填写地址格式时，在该画面上已经提供了地址格式的提示。如下图所示。



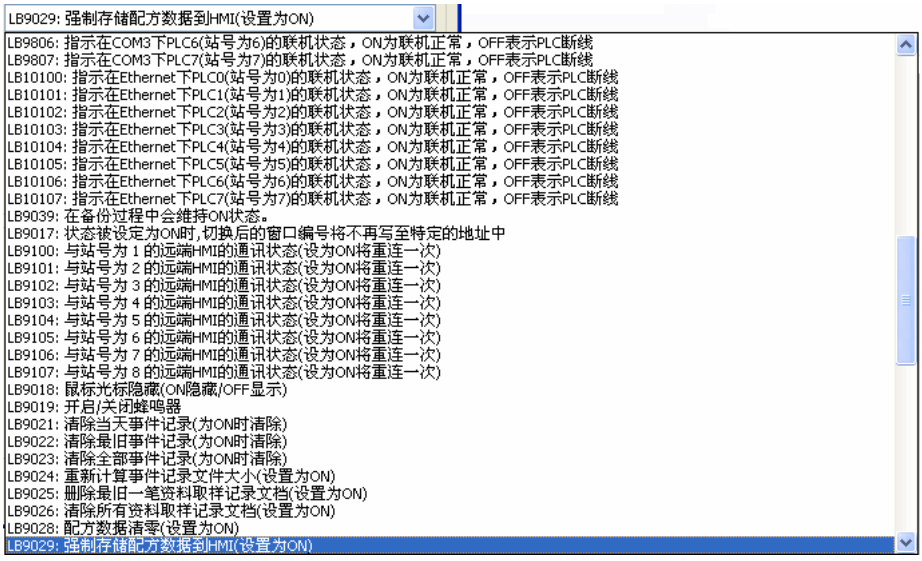
由上图可以看出，当选择“地址类型”为 PLC 的 SM 时，在“地址”这一栏中的格式为：[范围：8000~8255，10 进制]。

[系统寄存器]

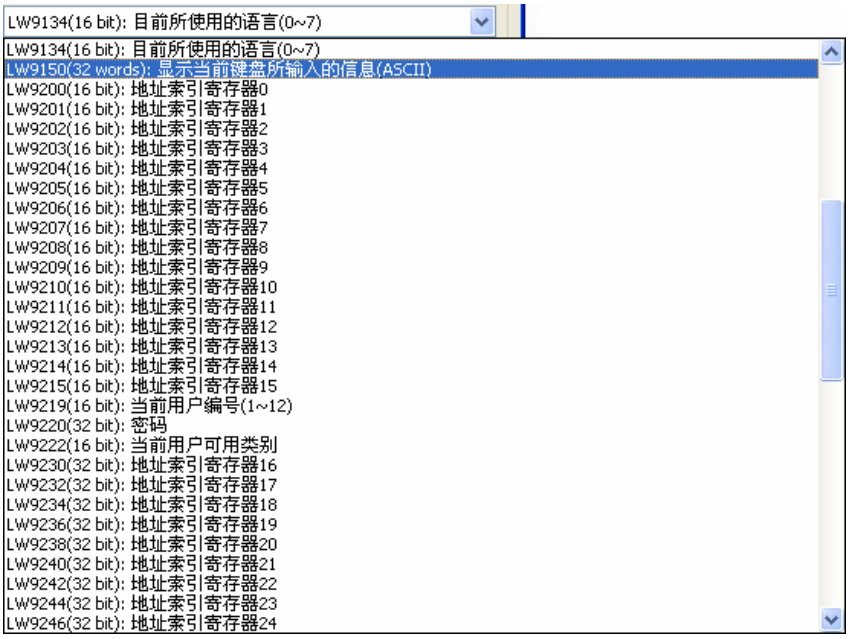
当“PLC 名称”项选择为“Local HMI”时，表示操作本地触摸屏中的地址。此时会有“系统寄存器”的选项存在。系统寄存器为系统保留作为特殊用途的地址，与一般 PLC 中的地址分配类型一样，分为 bit 型系统寄存器和 word 型系统寄存器。在勾选“系统寄存器”后，在“设备类型”后的列表中，选择需要的系统寄存器。此时，“设备类型”中将显示选中的系统寄存器的内容，“地址”里面会用灰色的方式显示该地址，且不可更改。如下图所示。



下图为 bit 型系统寄存器与 word 型系统寄存器的部分内容，详细内容可参考《地址标签库》的相关说明。



Bit 型系统寄存器



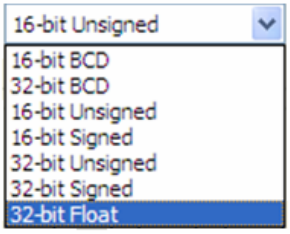
Word 型系统寄存器

[索引寄存器]

选择是否使用“索引寄存器”，可参考“索引寄存器”的说明。

10.3 数据格式选择

如果是 word 型“设备类型”，则会有数据格式的选项。InoTouch Editor 支持下列的各式数据格式，需正确选择数据格式，否则写入到 PLC 的数据或者从 PLC 读取到人机界面显示的数据就不一致。在使用系统寄存器时，也要特别注意数据格式，当数据格式不匹配时，执行的结果将会不正确。对于数据格式的详细说明，请参考“控件”章节的相关说明。



10.4 图库的使用

某些控件可以使用图库的图形，增加控件的视觉效果。图库的使用在控件属性页中的[图形属性]分页中设定，见下图。

[图形属性]设定页各项设定的说明如下：

[图库…]

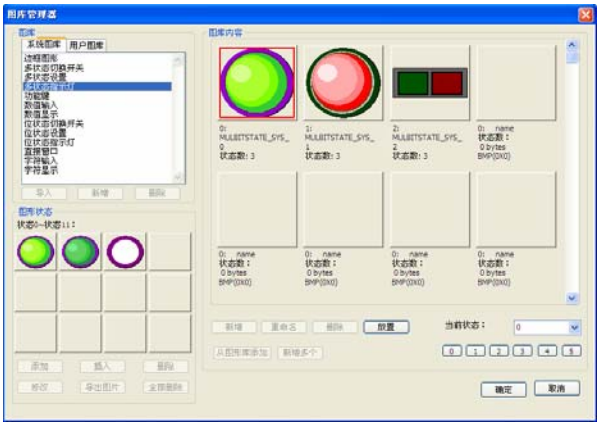
单击该按钮，将挑选 InoTouch Editor 软件的需要的图库。

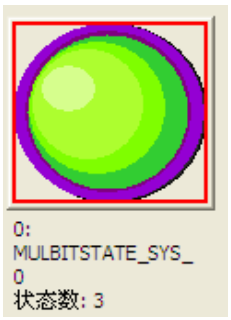
[使用图库]

选择图案是否使用图库中的图形。

10.4.1 如何使用系统图库

在按下[图库…]按钮后可以得到下面的“图库管理”对话框，由对话框中可看出被选中的向量图会使用红色的外框加以标示。



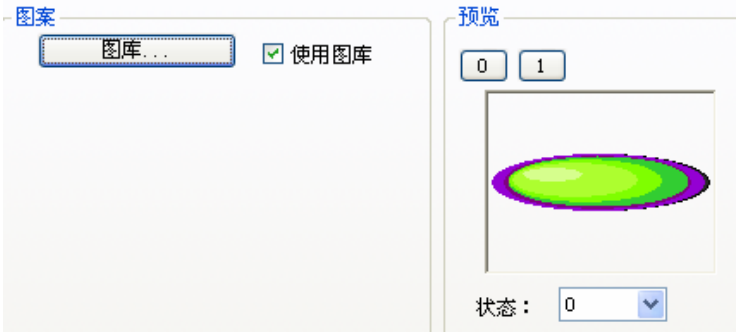


上图显示系统图库中某一样式的信息，这些信息的意义如下：

0: MULBITSTATE_SYS_0 表示此图形的名称与图库的编号

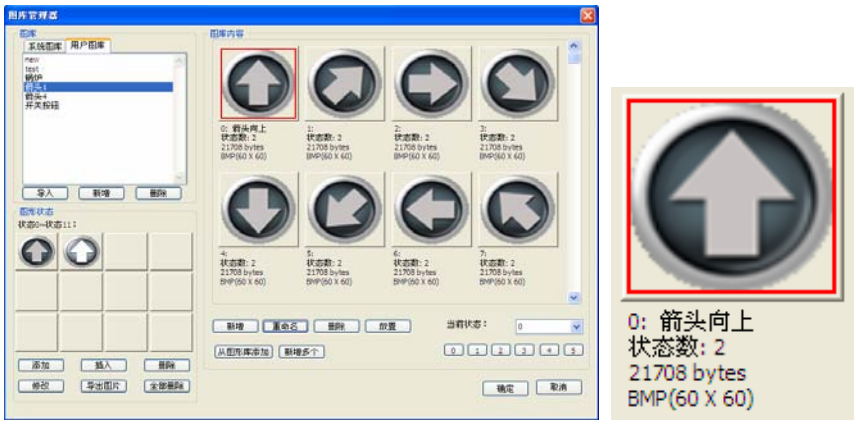
状态数 ：3　　表示此图形的总状态个数

“系统图库”各项目的说明可参考相关章节说明。在完成各项设定并按下确认键后，控件将使用目前所选择的图形，如下图。



10.4.2 如何使用用户图库

在按下[图库...]按钮后，选择“用户图库”可以得到下面的“用户图库”对话框，由对话框中可看出目前选择的图形会使用红色的外框加以标示。



上图显示图形库中选中图形的信息，这些信息的意义如下：

0:箭头向上　　表示此图形的名称与图库的编号

状态数 ：2　　表示图形的总状态个数

21708 bytes　　表示图形的大小

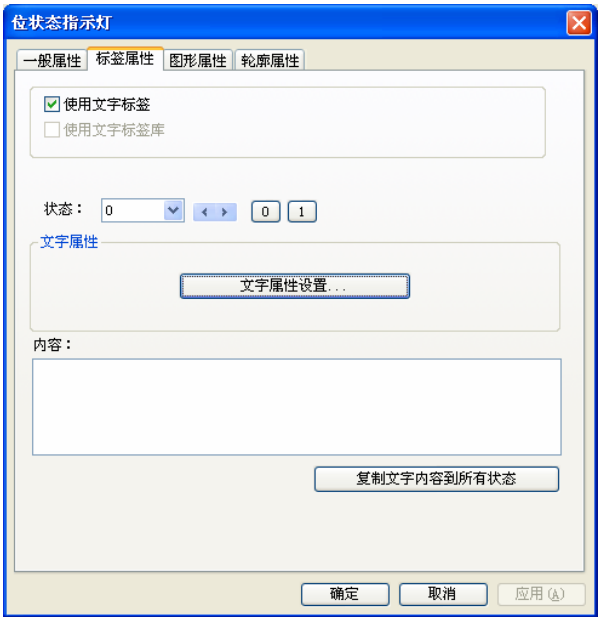
BMP (60 *60)表示图形的格式与原尺寸，BMP 表示图形使用 bitmap 格式，图形格式也可能为 JPG 或 GIF。60 * 60 表示图形的原尺寸长为 60 像素，高为 60 像素。

“用户图库对话框”各项目的说明参考相关章节说明。在完成各项设定并按下确认键后，控件将使用目前所选择的图形，如下图所示。

10.5 标签属性的设定

为了显示更好的视觉效果，控件除了可以使用图形外，还可以在控件中填写文字。不同的状态，可以使用不同的文字内容。

控件内文字的使用在控件属性页中的[标签属性]分页中设定，如下图。



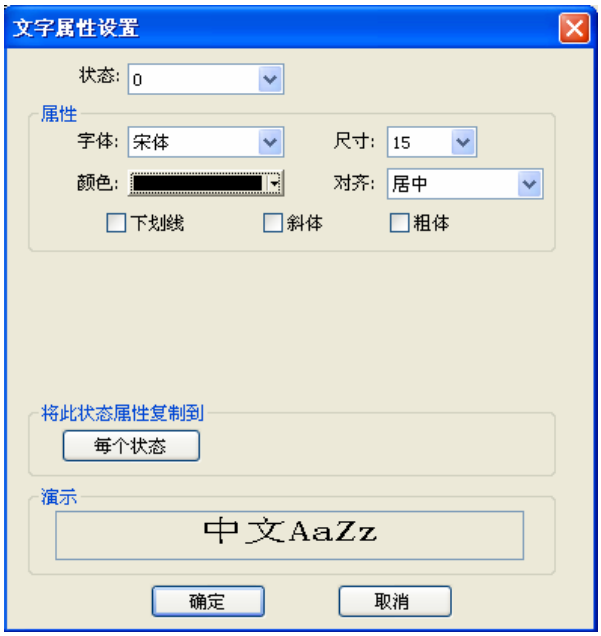
1）标签属性设定项

[使用文字标签]

勾选此选项控件才允许使用文字标签，即可以在控件上填写文字。

[文字属性设置...]

检视文字标签库的内容，有关这部分内容的说明，可参考《文字库与多语言显示》的说明。



[字体]

选择文字所使用的字型。InoTouch Editor 支持 Windows 的所有字体， 且字型均为矢量字型，如下图。

[颜色]

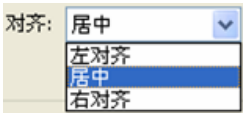
选择文字所使用的颜色。

[尺寸]

选择文字所使用的大小。

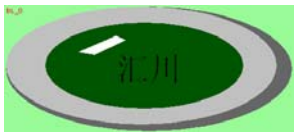
[对齐] 

选择文字的对齐方式，可选择的方式如下：

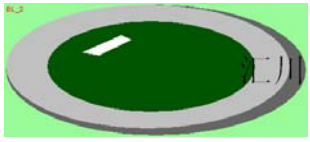


下图为选择“左对齐”的对齐方式。

下图为选择“居中对齐”的对齐方式。



下图为选择“右对齐”的对齐方式。

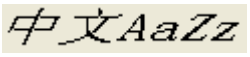


[下划线]

给文字加上底线。

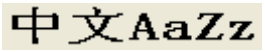
[斜体]

使用斜体字体。



[粗体]

把文字加粗。



[复制文字到所有的状态]

将目前的文字内容复制到其它所有的状态。

10.6 轮廓属性

如下图，控件的外型大小可以利用[轮廓]设定页加以调整。

1）位置设定

[图钉]

锁定设定，勾选此选项后将无法改变控件的位置与大小。

[X]、 [Y]的坐标表示控件在屏幕画面上的坐标位置。以左上角为坐标原点。

2）尺寸设定

[宽度]

设定控件的宽度，单位为像素。

[高度]

设定控件的高度，单位为像素。

11

控件的安全防护

第十一章　控件的安全防护

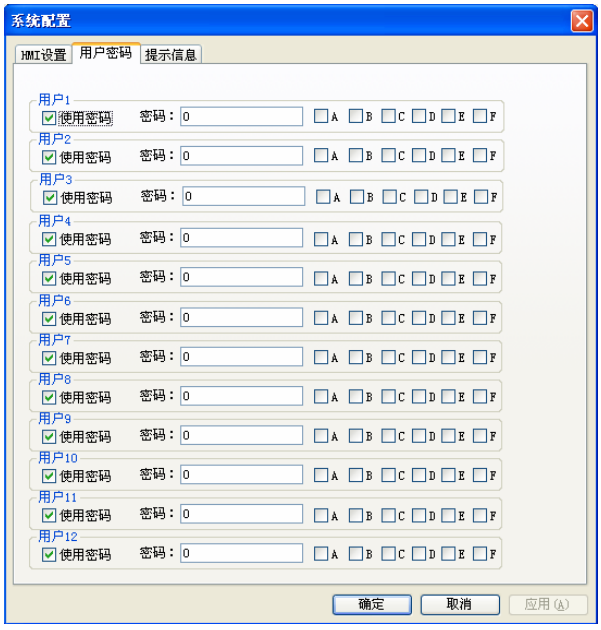
InoTouch Editor 软件提供的安全防护功能是对应控件来实现的，而非对应窗口，这个是首先要了解清楚的。InoTouch Editor 软件对控件提供了多种不同的安全防护，主要可分为：

11.1　用户密码与可操作控件类别设定

11.2　控件“安全属性”提供的安全防护

11.1　用户密码与可操作控件类别设定

在前面的说明中，有提到“系统参数”中的“用户密码”的设定。在此项设定中，用来设定用户的密码，并规划每个用户可操作的控件类别，在 InoTouch Editor 中，控件被划分为“无”与“A~F”等共 7 个类别。用户的密码必须是由 0~9 的数字所组成，InoTouch Editor 最多可规划 12 个用户，即用户 1 到用户 12。



画面程序运行时，用户在成功输入密码后，InoTouch Editor 会依照用户预先设定内容决定用户可以操作的控件类别。例如，当“用户 1”的规划内容如下时，此位用户只被允许操作控件类别属于“无”与 A、C、E 的控件。

正确的密码输入过程除了必须将密码输入到密码输入地址[LW9220](共 2 个 words, 32-bit)之外，用户也必须使用[LW9219] (共 1 个 word, 16-bit)指定目前的用户。[LW9219]中的数据需为 1 ~ 12，分别用来表示“用户 1”至“用户 12”。请在设定密码输入地址[LW9220]时，必须将其数据格式设置为 32-bit 整数，即可以为 32-bit unsigned 或者 32-bit signed 的格式，否则输入的密码将不会被识别。

当密码输入错误时，[LB9060]的状态将被设定为 ON 状态；当密码输入成功时，[LB9060]的

状态将自动被恢复为 OFF 状态。

用户 1 至用户 12 所有用户的密码可以利用读取系统寄存器[LW9500]至[LW9522]，共 24 words 的内容获得。

InoTouch Editor 也提供用户在线更改密码功能。当系统寄存器[LB9061]的状态由 OFF 转变为 ON 时，InoTouch Editor 会使用系统寄存器[LW9500]至[LW9522]内的数据，更新用户的密码，往后并使用这些新的密码。

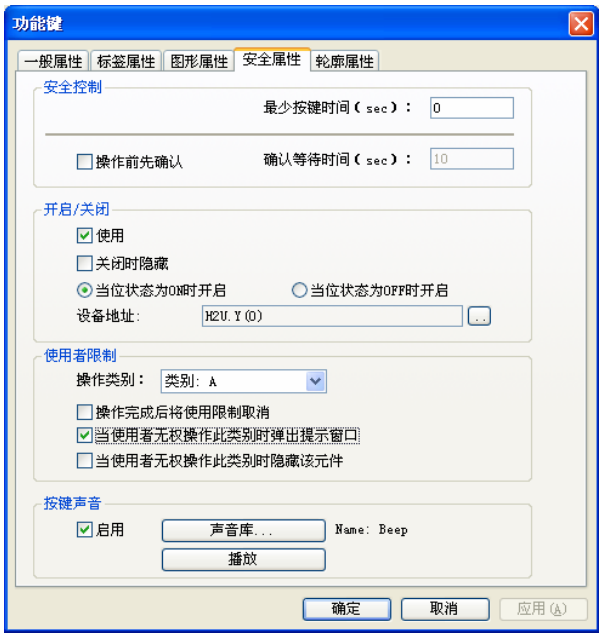
注意：此时用户可使用的控件类别并不会因密码的变更而改变。必须把系统寄存器[LB9061]设置为 ON 一次，新的密码才可以修改生效，否则修改无效。

当系统寄存器[LB9050] (用户登出)的状态由 OFF 变为 ON 时，可强迫目前的用户注销登录状态，此时系统将只允许该“用户”操作类别属于“无”的控件。

另外，[LW9222]记录目前的用户可以操作的控件类别，bit 0 为 1 表示目前的用户可操作类别属于“A”的控件；bit 1 为 1 表示目前的用户可操作类别属于“B”的控件，其余 bit 所表示的意义依此类推。

注意：不能使用数字全部为 0 的密码。

11.2 控件“安全属性”



上图为有关控件“安全”属性的内容，可分为几个部分：

11.2.1 安全控制

“安全控制”主要用来避免操作者在未知的情况下误按控件，目前提供两种保护方式。

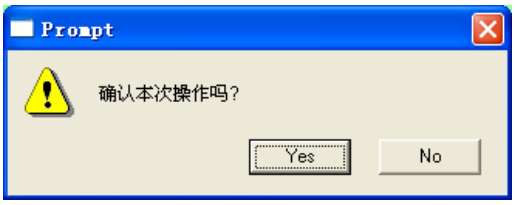
[最少按键时间]

此项的设定值表示，持续按压控件的时间不小于此项设定值才能成功操作控件。设置为 0 时，

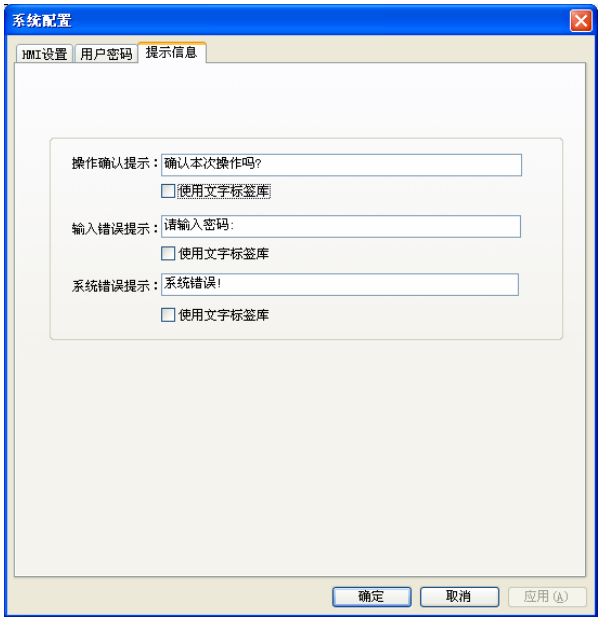
表示不使用此安全功能。此设定值的单位为秒（sec）。例如设定的数值为 3，表示按压该控件持续时间到 3 秒钟时，该控件的操作才有效，否则无效。这样就有效的防止了误操作的发生。

[操作前先确认]

若勾选了此项功能，在按下控件后将出现下图的对话框，用户可以依照实际需要，确认是否执行此项动作。超过[确认等待时间(sec)]所设定的时间后仍未决定是否执行此项动作，对话框会自动消失并且取消此项执行动作。



对话框中提示的文字(上图为“确认本次操作吗？”)被定义在[系统提示信息]中，用户可以利用[系统提示信息]对话框更改提示文字的内容。按下工作栏上的[HMI 系统配置]按钮后，会出现[系统配置]对话框，其中第三页第一项文字的内容被作为操作确认提示用途，第二项用于当用户密码输入错误时的提示，最后一项用于系统错误提示。



11.2.2 开启/关闭

当控件使用此项功能时，此控件是否允许被操作，将决定于特定地址(或称为“开启/关闭”地址)的状态。“开启/关闭”地址必须是 Bit 地址形式，地址的内容由下面的对话窗来决定。

举例来说，假使某一个“位状态设定”控件使用安全属性中的“开启/关闭”功能，并且它的“开启/关闭”地址为[LB0]，并选择[当位状态为 ON 时开启]，则必须在[LB0]状态为 ON 时，才允许操作此控件，否则操作无效。“开启/关闭” 提供下列的设定：

[开启/关闭]

勾选此选项则此控件将使用“开启/关闭”安全功能。

[关闭时隐藏]

若勾选此项，当控件使用“开启/关闭”功能且“开启/关闭”的控制地址状态不符合开启条件(ON 或 OFF)时，该控件将会被隐藏起来，画面上此时将无法看到该控件。

[当位状态为 ON 时开启]

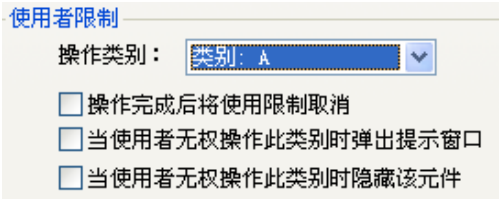
当勾选此项时，表示设定的控制地址状态为 ON 时，该控件才可以被操作，或者该控件的功能才有效。

[当位状态为 OFF 时开启]

当勾选此项时，表示设定的控制地址状态为 OFF 时，该控件才可以被操作，或者该控件的功能才有效。

11.2.3 使用者限制

此项功能即是对“系统参数/用户密码”中操作类别的作用的使用。在此设定控件的操作类别，设定操作类别后，该控件将只被允许操作此种类别的用户所操作。例如前面提到的在“系统参数/用户密码”的设定中，用户 1 的操作类别是 A、C、E。若在此处选择操作类别为“类别 A”，那么用户 1 在输入正确的密码后，将有权限操作该控件，否则无权限操作该控件。当“控件类别”选择“无”时，表示任意用户皆可操作此控件。此项功能也提供下列的设定：



[操作完成后将使用限制取消]

当用户目前的操作限制曾经符合此控件的操作条件后，将永远停止对此控件的操作做类别限制检查；也就是说即使目前的用户编号改变了，或者密码输入错了，也不会影响对此控件的操作，也即此时该控件的操作类别就是“无”。

[当使用者无权操作此类别时弹出提示窗口]

当使用者目前的操作身份无法符合此控件的操作条件时，按下此控件将出现警示对话框，如下图。

InoTouch Editor 使用 2005 号窗口作为操作身份权限不足时出现的警示对话框，使用者可以自行设计警示对话窗的内容。

[当使用者无权操作此类别时隐藏该元件]

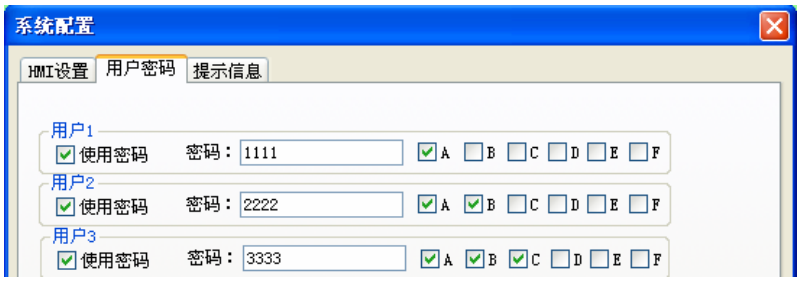
当使用者目前的操作身份无法符合此控件的操作类别时，隐藏该控件。即在使用者操作类别不符合该控件设定的操作类别时，该控件在屏幕上是不显示的。

11.2.4 按键声音

各个控件可以分别设定是否使用蜂鸣器。InoTouch Editor 也提供系统寄存器[LB9019]作为蜂鸣器的总开关，当[LB9019]的状态为 OFF 时，蜂鸣器才能被使用。重新开机时，InoTouch Editor 将使用前一次对蜂鸣器的设定状态。

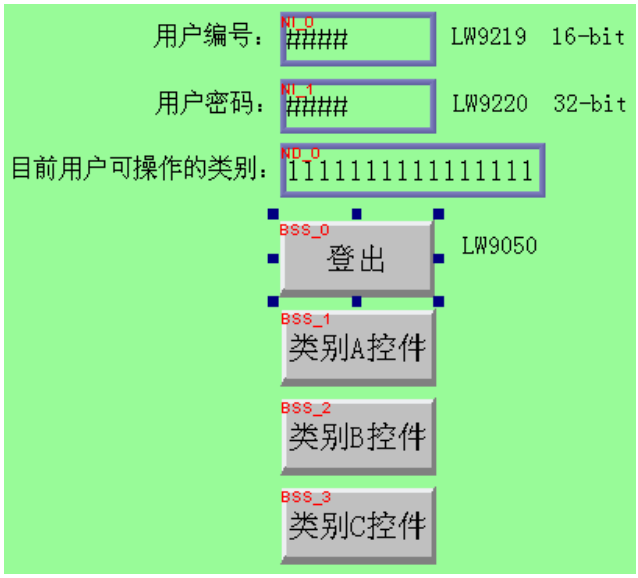
举例说明控件安全防护的使用：

步骤一：新建立一个工程档案，并在[系统配置]的[用户密码]设定页中启用三个用户，并且设定各个用户的密码与可操作的控件类别。如下图所示：



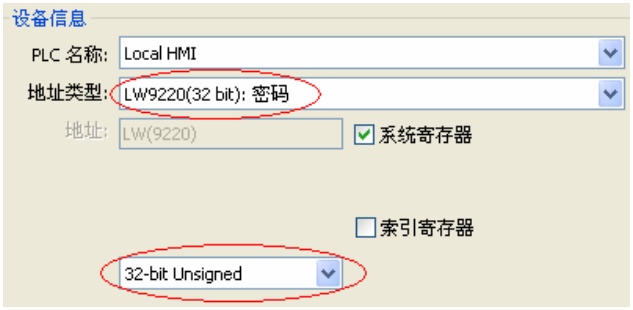
此时可发现“用户 1”可以操作类别 A 的控件，“用户 2” 可以操作类别 A 与 B 的控件，“用户 3”则可以操作类别 A、B 与 C 的控件 。

步骤二：可在“初始页面”设计如下图所示的控件：



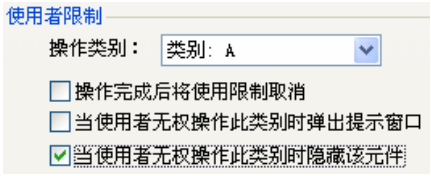
上图的[NI_0]与[NI_1]皆为数值输入控件，地址为[LW9219]与[LW9220]，分别用来输入用户编号与用户密码。其中系统寄存器[LW9219]被用来输入用户编号(1~12)，长度为 1 个 word，因此此控件必须选择 16-bit Unsigned 的资料格式，如下图：

系统保寄存器[LW9220]被用来输入用户密码，长度为 2 个 word，因此此控件必须选择 32-bit Unsigned 的数字格式，如下图：



[NO_0]为“数值显示”控件，地址为[LW9222]。用来显示目前用户可操作的类别。此控件必须选择 16-bit Binary 的数字格式。

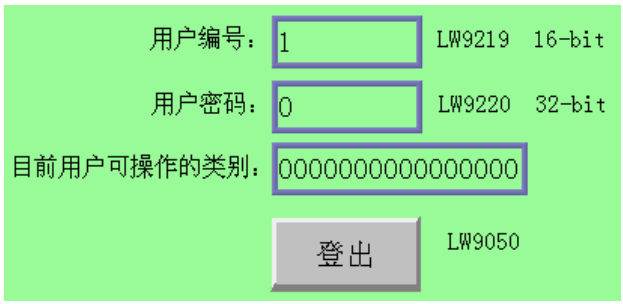
[BSS_1]~[BSS_3]为“位状态设置”控件，这三个控件刻意选择不同的操作类别，但都选择[当使用者无权操作此类别时隐藏该元件]。其中[BSS _1]的操作类别为“A”，[BSS_2]的操作类别为“B”，[BSS_3]的操作类别为“C”。下图为[BSS_1]的操作类别设定内容。



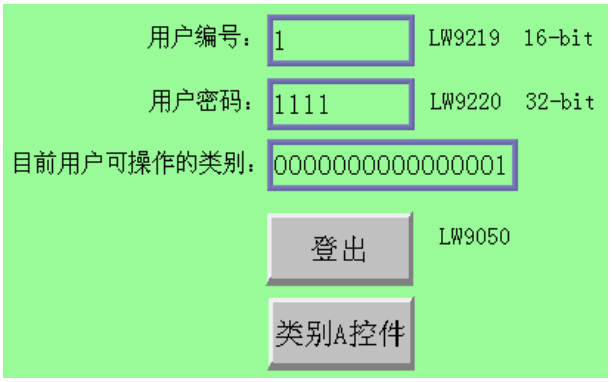
另外画面也设计一个“位状态设置”按键(BSS_0, LB9050)，作为用户登出的用途，参考下图。

步骤三：将设置好的页面保存、编译后即可执行离线仿真功能。

下图为离线仿真功能的起始画面，此时因尚未输入任何密码，所以[NO_0]控件显示“00000000000000”，表示目前的使用者仅只能使用类别为“无”的控件，且因[BSS_1]~[BSS_3]控件分别属于类别“A”～“C”并选择[当用户无权操作此类别时隐藏该元件]，所以[BSS_1]~[BSS_3]皆被系统所隐藏。

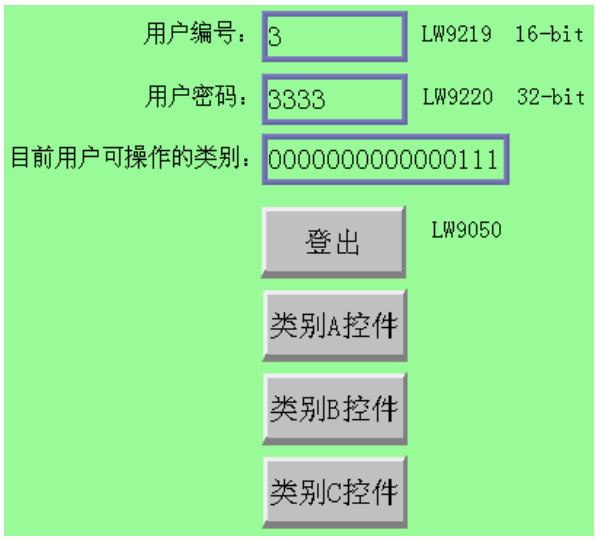


接着使用者可输入用户 1 的密码(1111)，输入完成后画面如下：



因原来规划“用户 1”允许使用类别属于 A 的控件，所以此时[BSS_1]控件将出现并允许使用者操作。此时也可发现[LW9222]的 bit 0 已变为 1，表示此时的用户允许使用类别属于 A 的控件。

接着使用者可输入用户 3 的密码(3333)，输入完成后画面如下：

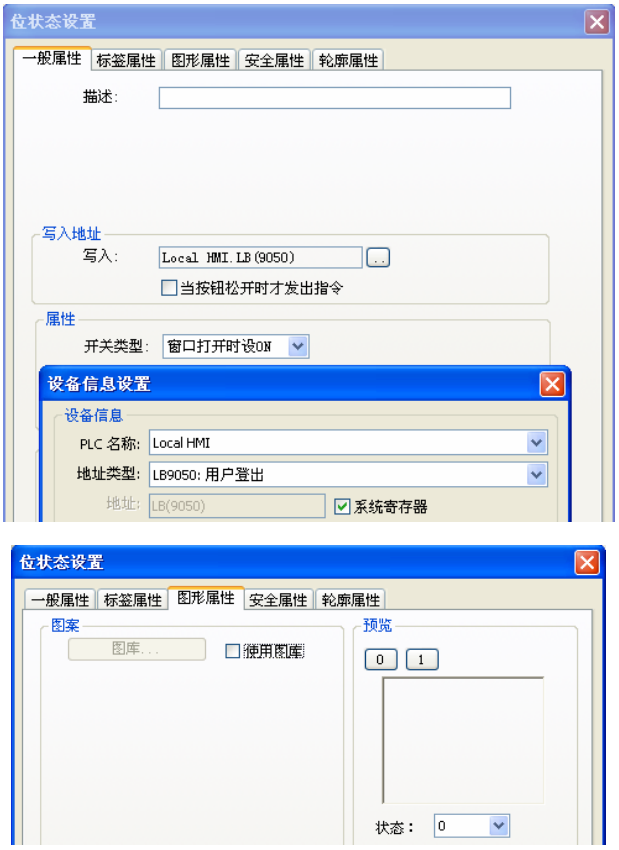


由上图可以发现，“用户 3”被规划为允许使用类别属于 A，B，C 的控件。此时[LW9222]的 bit

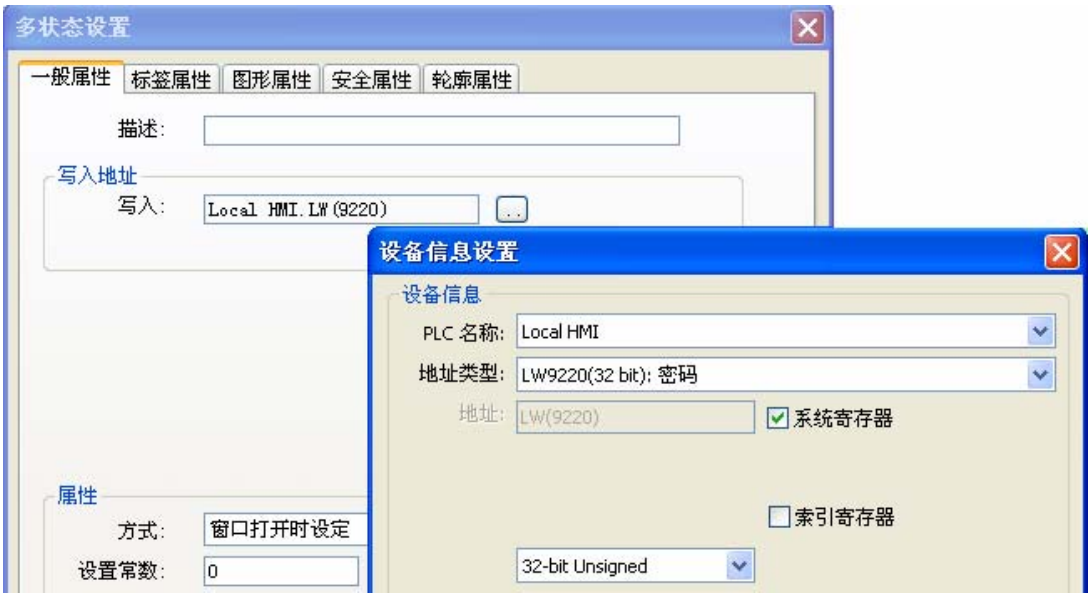
0~bit 3 皆变为 1，表示此时的用户的确被允许使用类别 A~C 的控件。

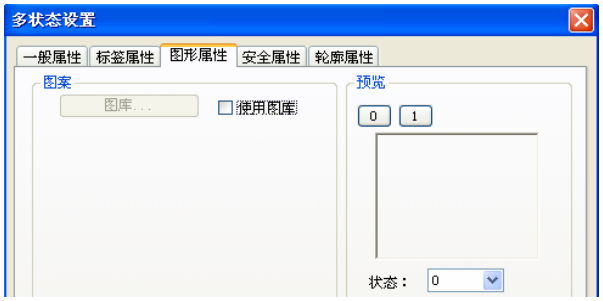
此时如按下[BSS_0]强迫用户注销，可以发现系统将回复到起始状态，此时操作者又只能操作类别为“无”的控件了。

注意：在正确输入用户名和用户密码以后，则可以操作相应类别的控件。但是往往操作者在使用时，操作完毕后，忘记登出，使得其他无权限的人也有可能操作到不该操作的控件。为此，在编辑工程画面时，建议在密码输入的窗口增加一个“位状态设置”控件，其写入地址填为“LB9050”，属性设置为“窗口打开时设 ON”，也即只要登陆到密码输入窗口，自动注销登录状态。设定的属性如下，同时在“图形”中不使用任何图形，即将该控件不显示在屏幕上。



另外，在用户密码输入错误时，系统也会自动的注销登录状态，为此，可以增加一个“多状态设置”控件，“写入地址”设置为“LW9220”，数据格式为“32-bit unsigned”，在“属性”中选择“窗口打开时设置”，在“常数”项填入“0”。同时也不使用任何图片，将原件隐藏起来，这样也可以在进入输入密码的窗口，系统会自动注销登录状态。设定的内容可以参考下图。


多状态设置
一般属性
标签属性
图形属性
安全属性
轮廓属性
描述:
写入地址
写入:
Local HMI.LW(9220)
属性
方式:
窗口打开时设定
设置常数:
0
设备信息设置
设备信息
PLC 名称:
Local HMI
地址类型:
LW9220(32 bit): 密码
地址:
LW(9220)
系统寄存器
索引寄存器
32-bit Unsigned


多状态设置
一般属性
标签属性
图形属性
安全属性
轮廓属性
图案
图库...
预览
0
1
状态:
0

12

索引寄存器

第十二章　索引寄存器

索引寄存器是 InoTouch Editor 软件提供的用于变址寻址的寄存器。有了索引寄存器后，用户可以在不改变控件地址内容的情况下，画面程序运行时，在人机界面上就可以在线修改控件的读取与写入地址。InoTouch Editor 软件提供了总共 32 个索引寄存器，分别为 16 个 16-bit 的索引寄存器，16 个 32-bit 的索引寄存器。

32 个索引寄存器的地址分别为：

INDEX 0　　[LW9200] (16-bit)

INDEX 1　　[LW9201] (16-bit)

INDEX 2　　[LW9202] (16-bit)

INDEX 3　　[LW9203] (16-bit)

……

INDEX 14　　[LW9214] (16-bit)

INDEX 15　　[LW9215] (16-bit)

INDEX 16　　[LW9230] (32-bit)

INDEX 17　　[LW9232] (32-bit)

……

INDEX 30　　[LW9258] (32-bit)

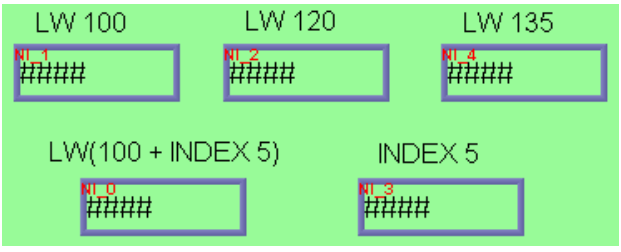
INDEX 31　　[LW9260] (32-bit)

INDEX0~INDEX31 为地址标签说明，后面的系统寄存器就是真正的索引寄存器的地址。其中 INDEX 0 ~ INDEX 15 为 16-bit 索引寄存器，INDEX 16 ~ INDEX 31 为 32-bit 索引寄存器。因此 INDEX 0 ~ INDEX15 可以寻址的范围最大为 65536 words，INDEX 16 ~ INDEX 31 可以寻址的范围为 4G words。

使用索引寄存器后，所使用设备类型的地址则由“设定的常量地址+所选择索引寄存器中的值”来决定。索引寄存器对工程画面系统参数中建立的所有设备列表都有效，且对 bit 格式与 word 格式的地址均有效。

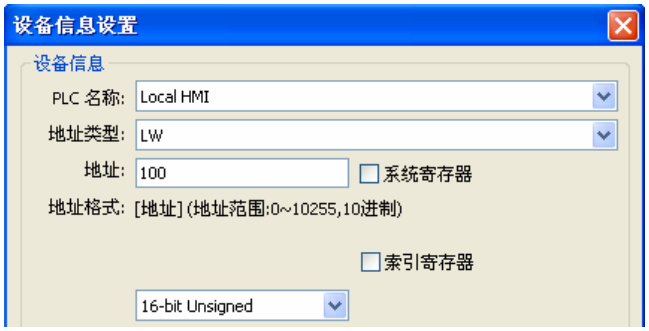
举例说明索引寄存器的使用方法：

步骤一：制作如下画面：

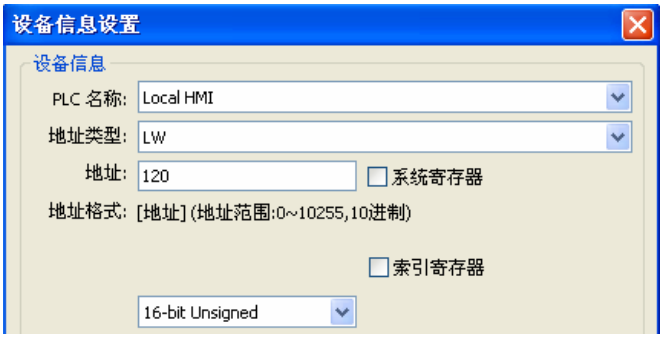


步骤二：设置控件属性

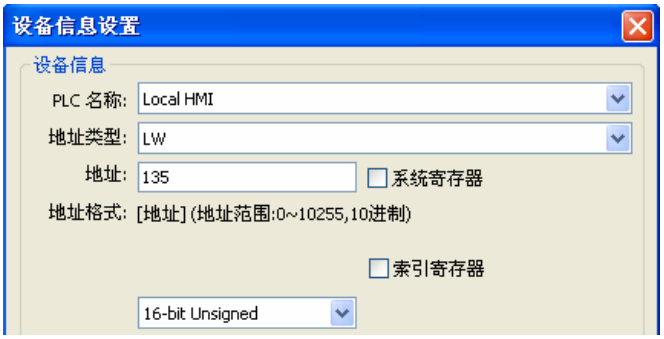
LW100 未勾选[索引寄存器]选项，此时的读取地址为 [LW100]。



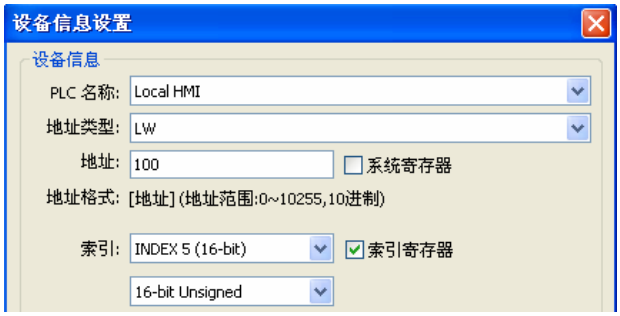
LW120 未勾选[索引寄存器]选项，此时的读取地址为 [LW120]。



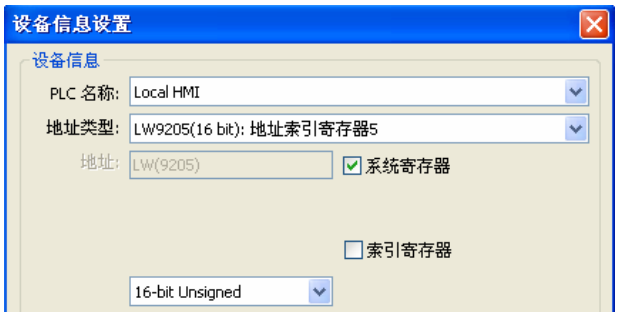
LW135 未勾选[索引寄存器]选项，此时的读取地址为 [LW135]。



LW(100 + INDEX 5)[索引寄存器]选项被勾选，且选择的索引寄存器为 INDEX5，此时的读取地址变为[LW(100 + INDEX 5)]，其中的 INDEX 5 表示索引寄存器 5 或 [LW9205]地址中的数据。例如[LW9205]地址中的数据为 20，则下图的读取地址变为[LW(100+20)]，也即读取地址为 LW120 的数据。

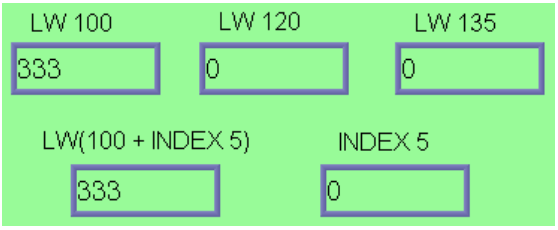


INDEX5 未勾选[索引寄存器]选项，此时的读取地址为 [LW9205]（或地址索引寄存器 5）。

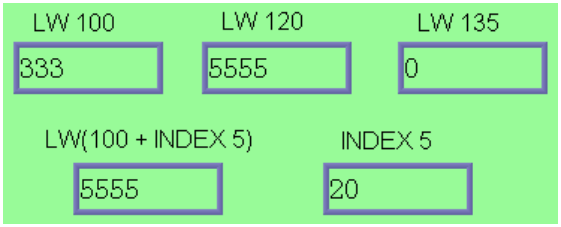


步骤三：保存、编译及离线仿真。

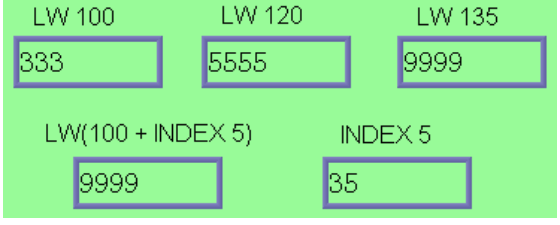
下图显示，此时 INDEX5 为 0，也就是[LW9205]地址中的数据为 0，则读取[LW100+INDEX3]将等同读取[LW100]的内容。



此时若将 INDEX5 的内容设定为 20，则读取 LW(100+INDEX5)将等同读取 LW120 的内容，如下图所示。



如上若将 INDEX5 的内容设定为 35，则读取 LW(100+INDEX5)将等同读取 LW135 的内容，如下图所示。



小结：经过上面的说明，了解到索引寄存器其实就是一个变址寻址的寄存器，通过索引寄存器就可以在不改变设备地址的情况下，只要通过改变索引寄存器中的数据，即可改变同一个控件读取或者写入不同的地址的数据。这样，就可以实现不同区域地址间数据的传送或者交换等功能。大家常用的“配方”传送和保存功能就是利用索引寄存器的这个特点来执行的。有关“配方”的说明，请参考相关章节。

注意：PLC 地址暂未支持索引寄存器。

13

控　件

第十三章　控件

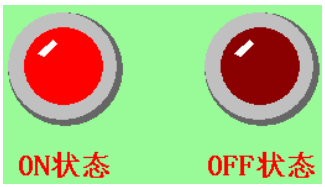
控件是为了实现某些需要的功能而设计的。一般来说，一个控件实现一个功能。支持控件叠加，叠加之后的控件都有效。有些控件必须搭配其他控件一起，才能够实现需要的功能，如下表所示。

控件	相关控件	搭配说明
数值输入控件	功能键	需由功能键创建键盘
文本输入	功能键	需由功能键创建键盘
间接窗口	窗口	显示的窗口必须存在
直接窗口	窗口	显示的窗口必须存在
事件/报警显示	事件/报警登录	事先需在”事件/报警登录”中登录信息
趋势图	资料采样	需先建立”资料采样”

下面分别说明控件的使用方式与相关的设定，未说明的设定请参考”控件一般属性设定”中的相关说明。

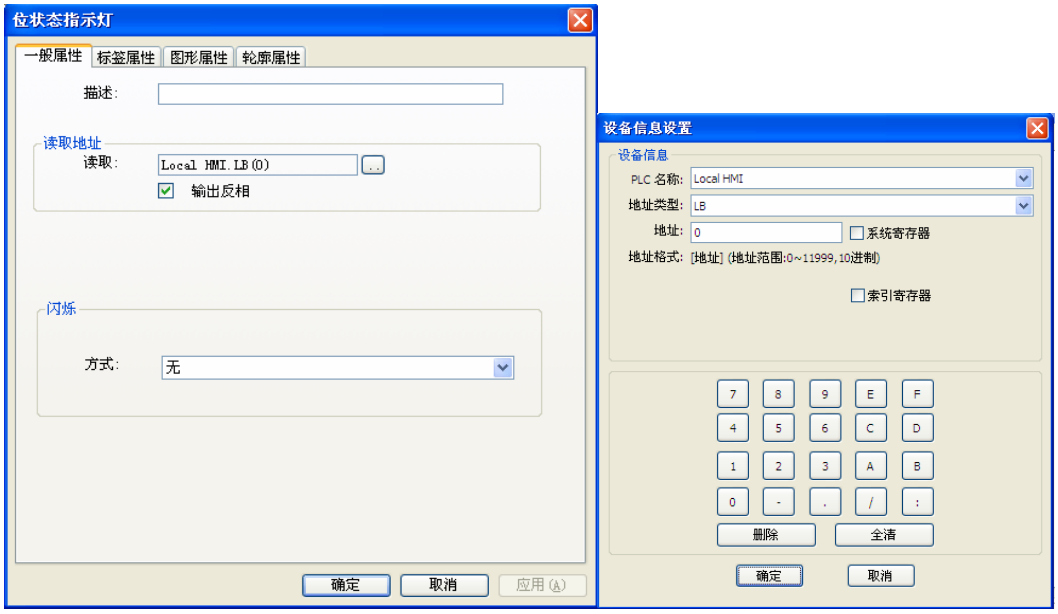
13.1 位状态指示灯控件 (bit lamp)

“位状态指示灯”用来显示 PLC 中 bit 型地址的状态。状态为 ON，则显示所使用图形的状态 1；状态为 OFF，则显示所使用图形的状态 0。



打开 InoTouch Editor 软件菜单的“控件/状态指示灯/位状态指示灯”，或者工具栏上的图标，在窗口中点击鼠标左键，就建立了“位状态指示灯”控件，如下图所示。

选择“位状态指示灯”双击或单击鼠标右键选择“属性”进行编辑，如下图：



[描述]

控件描述。一般写入注释性文字，方便阅读。

[读取地址]

设备信息的设置。

[PLC 名称]

选择要操作的 PLC 或者触摸屏。

[地址类型]

根据刚刚选定的“PLC 名称”，来选择读取的位地址。

[输出反相]

若勾选此项，可以将读取的状态作反相显示，例如读取的状态为 OFF，但控件在屏幕上会显示 ON 的图形。

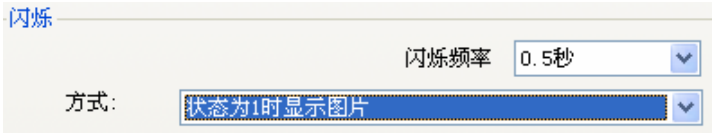
[闪烁]

设定控件的闪烁方式。

[模式]

闪烁模式	闪烁方式
无	不闪烁
状态为 0 时显示图片	状态为 OFF 时，使用图形 0 与图形 1 交互闪烁
状态为 1 时显示图片	状态为 ON 时，使用图形 0 与图形 1 交互闪烁
状态为 0 时闪烁	状态为 OFF 时，图形 0 出现与消失交互动作
状态为 1 时闪烁	状态为 ON 时，图形 1 出现与消失交互动作

选择闪烁效果时，[闪烁频率]用来选择闪烁频率的时间周期。

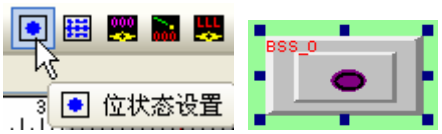


13.2 位状态设置控件 (set bit)

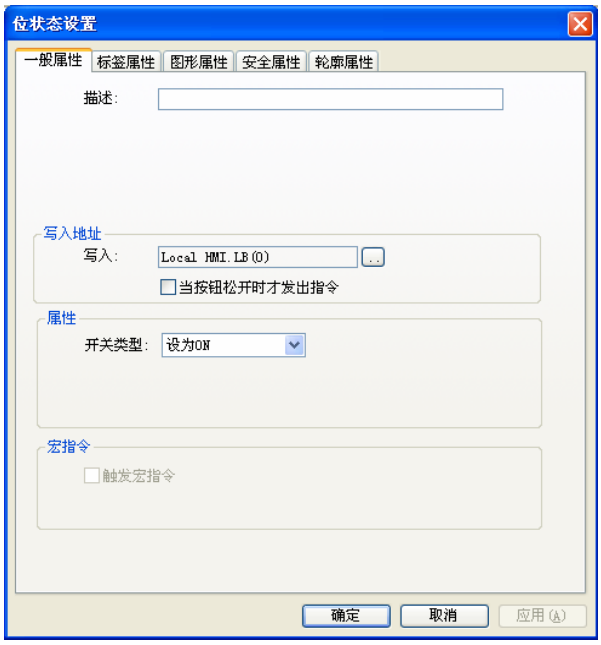
“位状态设定”控件提供手动操作与自动执行两种操作类型。使用“手动操作”模式，使用者可以利用“位状态设定”控件在窗口上定义一个触控区域，按压此区域可以将选定 PLC 中的位地址的状态设定为 ON 或 OFF。包括：设为 ON、设为 OFF、切换开关、复归型。

若使用“自动执行”模式，则在某些特定条件下会自动执行控件定义的动作，使用此种操作模式，在按压控件定义的碰触区域时，控件将不作任何反应。包括：周期切换开关、窗口打开时设 ON、窗口打开时设 OFF、窗口关闭时设 ON、窗口关闭时设 OFF、背光灯打开时设为 ON、背光灯打开时设为 OFF、背光灯关闭时设为 ON、背光灯关闭时设为 OFF。

打开 InoTouch Editor 软件菜单的“控件/状态设置/位状态设置”，或者工具栏上的图标，在窗口中点击鼠标左键，就建立了“位状态设置”控件，如下图所示。



选择“位状态设置”双击或单击鼠标右键选择“属性”进行编辑，如下图：



[写入地址]

定义要写入具体的哪个位地址

[当按钮松开时才发出指令]

使用此项设定表示在按下控件后，必须完全松开按压动作，控件定义的操作模式才会被执行。如未使用此项设定，只要一碰触此区域，将立刻执行控件的动作。但操作模式如果选择复归型(momentary)开关，将不支持此项功能。

属性

[开关类型]

选择控件的操作模式，可选择项目如下：

设为 ON	在触控控件定义的区域后，所指定位地址的状态将被设定为 ON。
设为 OFF	在触控控件定义的区域后，所指定位地址的状态将被设定为 OFF。

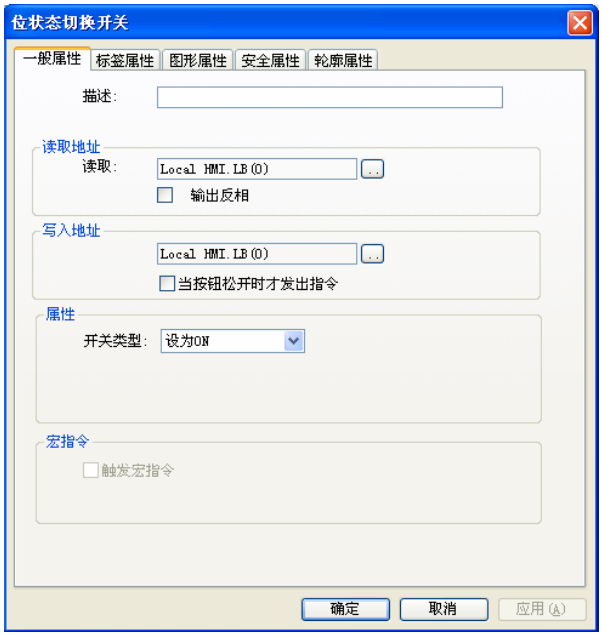
切换开关	切换型开关。每次触控控件定义的区域后，所指定位地址的状态将被反相。也就是状态由 OFF 变为 ON 或由 ON 变为 OFF，以此循环。
复归型	复归型开关。每次触控控件定义的区域时，所指定位地址的状态将先被设定为 ON，但离开碰触区域后，状态将被设定为 OFF。即定义为一个点动开关。
周期切换开关	周期性切换型开关。所指定位地址的状态将在 ON 与 OFF 间周期性切换，此模式不提供手动操作。可以使用下图显示的下拉式对话框中选择切换周期。 
窗口打开时设 ON	控件所在位置的窗口被打开时，所指定位地址的状态将自动被设定为 ON。
窗口打开时设 OFF	控件所在位置的窗口被打开时，所指定位地址的状态将自动被设定为 OFF。
窗口关闭时设 ON	控件所在位置的窗口被关闭时，所指定位地址的状态将自动被设定为 ON。
窗口关闭时设 OFF	控件所在位置的窗口被关闭时，所指定位地址的状态将自动被设定为 OFF。
背光灯打开时设为 ON	当背光灯打开时，所指定位地址的状态将自动被设定为 ON。
背光灯打开时设为 OFF	当背光灯打开时，所指定位地址的状态将自动被设定为 OFF。
背光灯关闭时设为 ON	当背光灯关闭时，所指定位地址的状态将自动被设定为 ON。
背光灯关闭时设为 OFF	当背光灯关闭时，所指定位地址的状态将自动被设定为 OFF。

13.3 位状态切换开关控件 (toggle switch)

“位状态切换开关”为“位状态指示灯”控件与“位状态设置”控件的组合。此控件除了可以用来显示指定位地址的状态外，也可以利用这个控件在窗口上定义一个触控区域，按压此区域可以设定所指定位地址的状态为 ON 或 OFF。

打开 InoTouch Editor 软件菜单的“控件/状态开关/位状态切换开关”，或者工具栏上的图标  ，在窗口中点击鼠标左键，就建立了“位状态切换开关”控件，如下图所示。

选择“位状态切换开关”双击或单击鼠标右键选择“属性”进行编辑，如下图：



[读取地址]

状态的读取地址。此处设定的位地址，当其从 PLC 中读取到状态有改变时，所使用的图形也即会改变显示状态，而不管是否有触控该控件。

[写入地址]

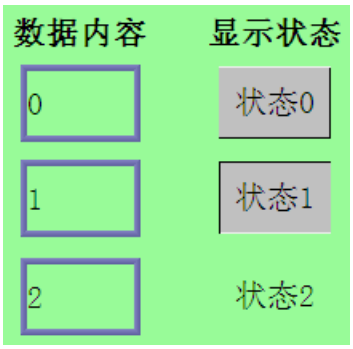
状态的写入地址，此位地址可以与“读取地址”所指定的位地址相同或不同。

[属性]

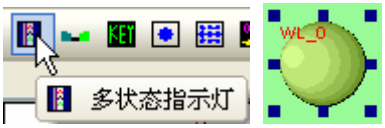
选择控件的操作模式，可选择项目包含“设为 ON”， “设为 OFF”，“切换开关”，“复归型”，可参考“位状态设置”控件的说明。

13.4 多状态指示灯控件 (word lamp)

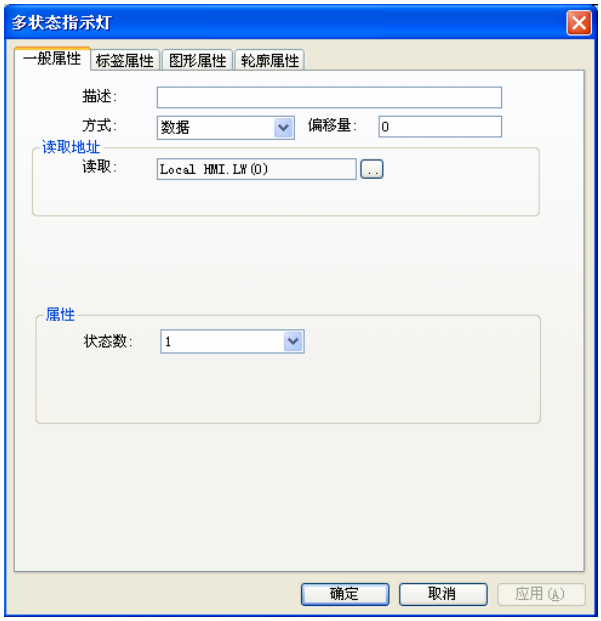
“多状态指示灯” 控件利用寄存器内的数据，显示相对应的状态与图形(InoTouch Editor 最多支持 256 种状态的显示)。



打开 InoTouch Editor 软件菜单的“控件/状态指示灯/多状态指示灯”，或者工具栏上的图标，在窗口中点击鼠标左键，就建立了“多状态指示灯”控件，如下图所示。



选择“多状态指示灯”双击或单击鼠标右键选择“属性”进行编辑，如下图：

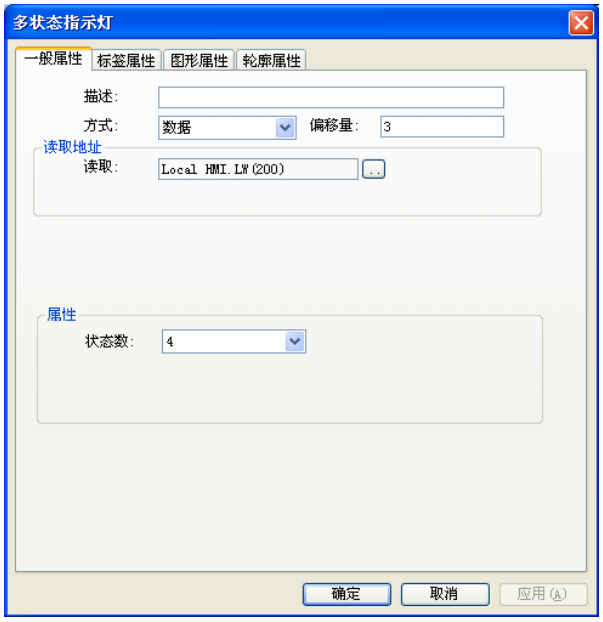


[方式]

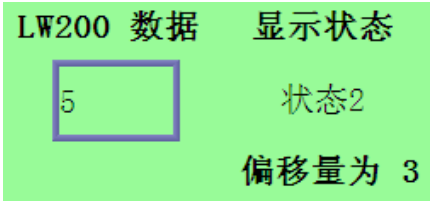
“多状态指示灯”控件提供下列三种选择模式：

a、“数据”显示模式

直接利用寄存器内的数据减去[偏移量]设定值的结果，做为控件目前的状态。例如下面增加一个新的“多状态指示灯”控件，控件设定内容如下图，注意此控件的[偏移量]为 3。



因此[LW200]内的数据如为 5，将显示状态 2(= 5 – 3)，参考下图。



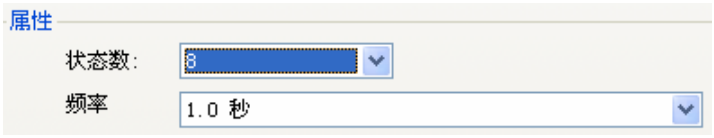
b、“LSB”显示模式

此模式首先会将寄存器内的数据先转换为 2 进制，接着使用不为 0 的最低位决定控件目前的状态。以地址[LW200]地址内的数据为例：

十进制	2 进制	显示的状态
0	0000	全部 bit 皆为 0，则显示状态 0
1	0001	不为 0 的最低位为 bit 0，此时显示状态 1
2	0010	不为 0 的最低位为 bit 1，此时显示状态 2
3	0011	不为 0 的最低位为 bit 0，此时显示状态 1
4	0100	不为 0 的最低位为 bit 2，此时显示状态 3
7	0111	不为 0 的最低位为 bit 0，此时显示状态 1
8	1000	不为 0 的最低位为 bit 3，此时显示状态 4

c、“周期转换状态”显示模式

控件的状态与寄存器无关，控件会使用固定频率依序变换状态。使用者可以利用[频率]设定状态改变频率。



[读取地址]

状态的读取地址。即要读取 PLC 中或者触摸屏中的哪个寄存器的状态

属性

[状态数]

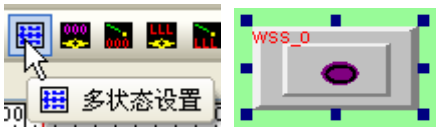
控件的状态数目，状态从 0 开始编号，因此能显示的最大状态为[状态号] 减 1。例如状态数目为 8，则显示的状态依序为 0， 1， 2，…， 7。当要求显示的状态超过[状态号] - 1 时，InoTouch Editor 会显示最后一个状态。

13.5 多状态设置控件 (set word)

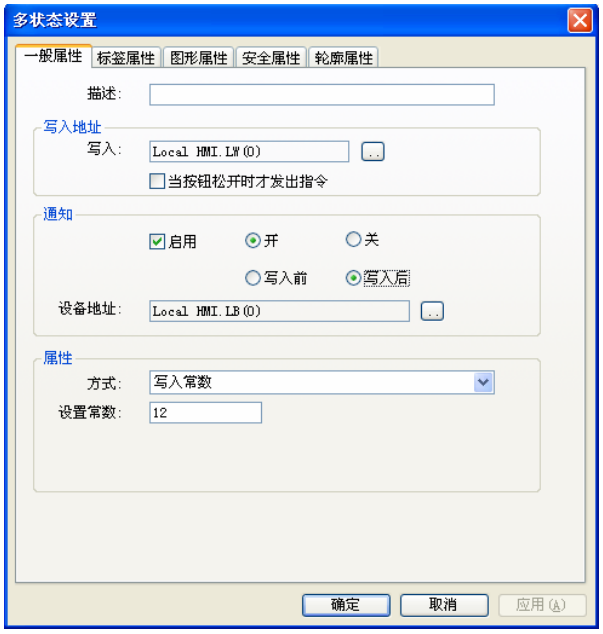
“多状态设置”控件提供“手动操作”与”自动执行“两种操作模式。使用“手动操作”模式，使用者可以利用“多状态设置”控件在窗口上定义一个触控区域，按压此区域可以设定所指定寄存器内的数据。包括：设为 ON、设为 OFF、切换开关、复归型。

若使用“自动执行”模式，则在某些特定条件下会自动执行控件定义的动作，使用此种操作模式，在按压控件定义的触控区域时，控件将不作任何反应。包括：周期切换开关、窗口打开时设 ON、窗口打开时设 OFF、窗口关闭时设 ON、窗口关闭时设 OFF、背光灯打开时设为 ON、背光灯打开时设为 OFF、背光灯关闭时设为 ON、背光灯关闭时设为 OFF。

打开 InoTouch Editor 软件菜单的“控件/状态设置/多状态设置”，或者工具栏上的图标  ，在窗口中点击鼠标左键，就建立了“多状态设置”控件，如下图所示。



选择“多状态设置”双击或单击鼠标右键选择“属性”进行编辑，如下图：



[写入地址]

数据的写入地址。设定具体操作 PLC 或者触摸屏中的数据寄存器。

[当按钮松开时才发出该指令]

使用此项设定表示在按压控件定义的触控区域后，必须完全离开此区域才会执行控件定义的动作。如未使用此项设定，则只要一接触到此区域，将立刻执行控件定义的动作。

[通知]

使用此项设定，则在使用“手动操作”模式时，在完成动作后可以连带设定此项目所指定位地址的状态，使用[ON]与[OFF]选择要设定的状态。

[启用]

选择是否开启此项功能。

[写入前]

在写入动作进行前先设定所指定位地址的状态。

[写入后]

在写入动作完成后才设定所指定位地址的状态。

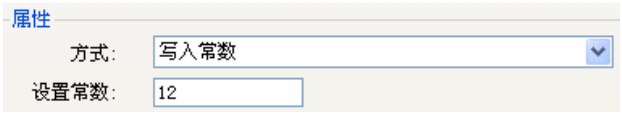
属性

[模式]

选择控件的动作模式，可以选择模式如下：

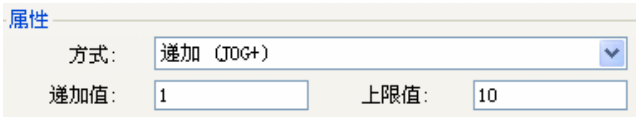
a、“写入常数”

设置常数功能。每按压一次控件，[设定常数]中的设定值将写至指定的寄存器中。常数的型态可为 16-bit BCD、32-bit BCD、…、32-bit float 等，数据格式在“写入地址”项目中决定。



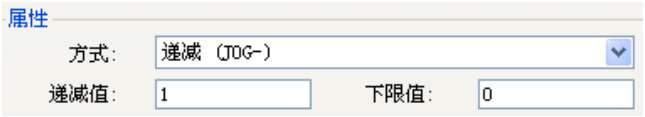
b、“递加 (JOG+)”

加值功能。每按压一次控件，所指定寄存器内的数据将加上[递加值]中设定的增量值，但增量的结果将不超过[上限值]中的设定值。



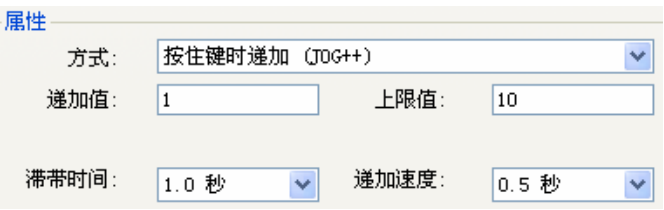
c、“递减 (JOG-)”

减值功能。每按压一次控件，所指定寄存器内的数据将减去[递减值]中设定的减量值，但是减值的结果不会低于[下限值]中的设定值。



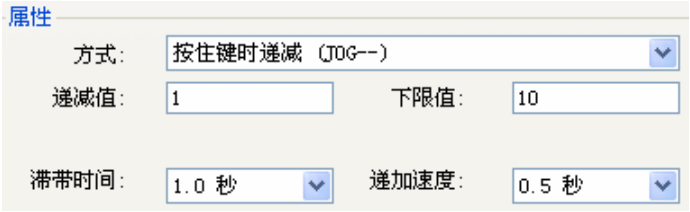
d、“按住键时递加 (JOG++)”

按住按钮时递加功能。若按压控件超过[迟滞时间]的设定时间，则所指定寄存器内的数据将以[递加速度]所设定的速度，每次增加[递加值]中设定的增量值，但增量的结果将不超过[上限值]中的设定值。



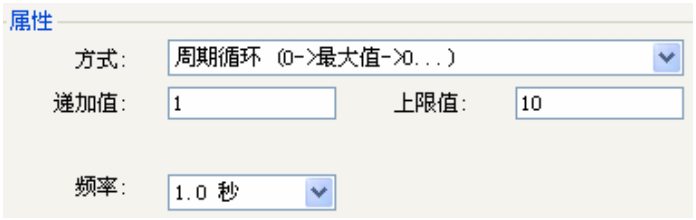
e、“按住键时递减 (JOG--)”

按住按钮时递减功能。若按压控件超过[迟滞时间]的设定时间，则所指定寄存器内的数据将以[递加速度]所设定的时间间隔，每次减少[递减值]中设定的减量值，但减值的结果不会低于[下限值]中的设定值。



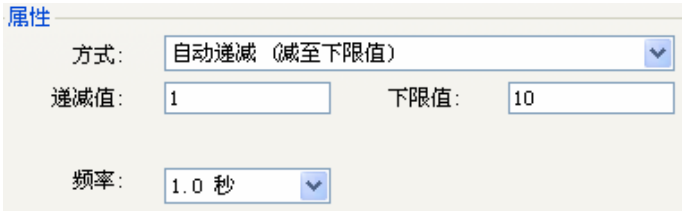
f、“周期循环 (0->最大值->0...)”

周期性递加功能。“多状态设定”控件会使用[频率]设定的周期与[递加值]中设定的增量值，自动增量所指定寄存器内的数据，但增量的结果将不超过[上限值]中的设定值。增加到最大值后，再又从零开始递增到最大值，如此往复循环。



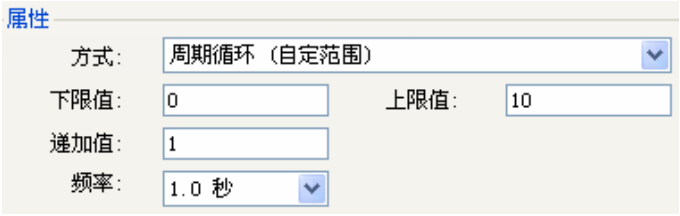
g、“自动递减 (减至下限值)”

周期性递减功能。“多状态设定”控件会使用[频率]设定的周期，自动将所指定寄存器内的数据减去[递减值]中设定的减量值，但减量的结果将不低于[下限值]中的设定值。



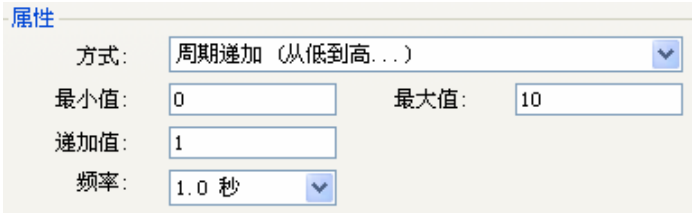
h、“周期循环 (自定范围)”

周期性循环功能。“多状态设定”控件会使用[频率]设定的周期，每次将所指定寄存器内的数据加上[递加值]中的设定值，直到寄存器内的数据等于[上限值]；接着使用相同的周期，将寄存器内的数据减去[递加值]中的设定值，直到寄存器内的数据等于[下限值]。如此周而复始，让数据一直保持动态变化。以下图为例，数据将作 0，1，2，...，9，10，9，8，7，...，1，0，1，2.....的周期性变化。



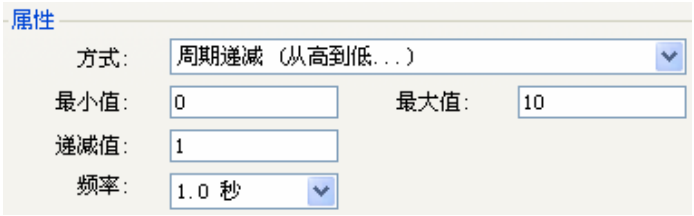
i、“周期递加”

步进功能。“多状态设定”控件会使用[频率]设定的周期，每次将所指定寄存器内的数据加上[递加值]中的设定值，直到寄存器内的数据等于[最大值]，接着会将寄存器内的数据复归为[最小值]，并重复先前的动作，让数据一直保持动态变化。以下图为例，数据将作 0，1，2，…，9，10，0，1，2，……的周期性变化。



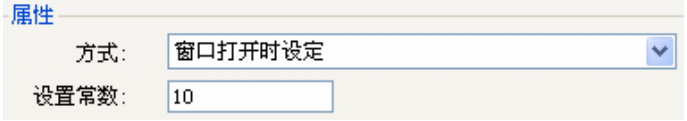
j、“周期递减”

步退功能。“多状态设定”控件会使用[频率]设定的周期，每次将所指定寄存器内的数据减去[递减值]中的设定值，直到寄存器内的数据等于[最小值]，接着会将寄存器内的数据复归为[最大值]，并重复先前的动作，让数据一直保持动态变化。以下图为例，数据将作 10，9，8，…，1，0，10，9，8，……的周期性变化。



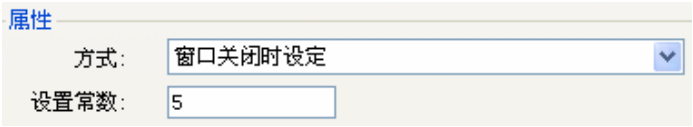
k、“窗口打开时设定”

打开控件所在位置的窗口时，会将[设置常数]中的设定值自动写至指定的寄存器中。



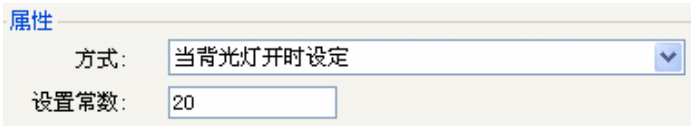
l、“窗口关闭时设定”

关闭控件所在位置的窗口时，会将[设置常数]中的设定值自动写至指定的寄存器中。



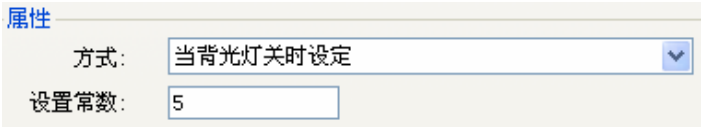
m、“当背光灯打开时设定”

当背光灯原处在关闭状态，若恢复为开启状态时，会将[设置常数]中的设定值自动写至指定的寄存器中。



n、“当背光灯关闭时设定”

当背光灯原处在开启状态，若关闭背光灯时，会将[设置常数]中的设定值自动写至指定的寄存器中。

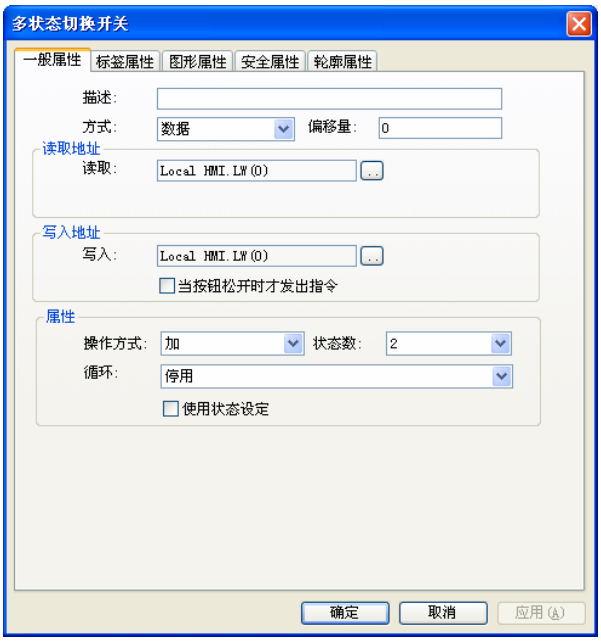


13.6 多状态切换开关控件 (multi-state switch)

“多状态切换开关” 控件为“多状态指示灯”控件与“多状态设定”控件的组合。此控件除了可以利用寄存器内的数据显示不同的状态外，也可以利用这个控件在窗口上定义一个触控区域，按压此区域可以设定所指定寄存器内的数据。

打开 InoTouch Editor 软件菜单的“控件/状态开关/多状态切换开关”，或者工具栏上的图标，在窗口中点击鼠标左键，就建立了“多状态切换开关”控件，如下图所示。

选择“多状态切换开关”双击或单击鼠标右键选择“属性”进行编辑，如下图：



[方式]

提供“数据”与“LSB”显示模式，可参考“多状态指示灯”控件的说明。

[偏移量]

使用在选择“数据”显示模式时，可参考“多状态指示灯”控件的说明。

[读取地址]

状态的读取地址。即设定读取 PLC 中哪个寄存器的数据状态。

[写入地址]

数据的写入地址，即要往 PLC 中具体的哪个寄存器中写入数据。可以与“读取地址”所定义的寄存器相同或不同。

[当松开按钮时才发出指令]

此项设定请参考“位状态设定”控件的说明。

[属性]

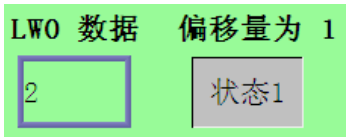
选择控件的操作方式。

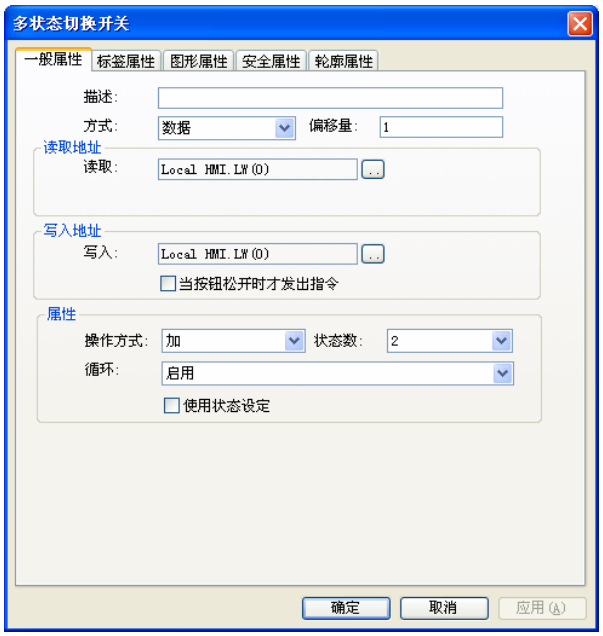
[操作模式]

可选择“加(JOG+)”或“减(JOG-)”。

当读写地址相同，并选择 “数据”显示模式时，则寄存器内数据的最小值将等于[偏移量]，此时的状态为状态 0；数据的最大值为([状态数] – 1) + [偏移量]，此时所显示的状态为[状态数] – 1。

可参考下图。



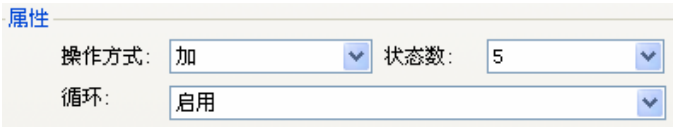


此图为上图结果中”多状态切换开关”控件的设定

a、“加(JOG+)”

每按下一次控件，将对“写入地址”所指定寄存器内的数据加上 1，当选择“数据”显示模式时，如果加值的结果大于等于[状态数] + [偏移量]的值时，如果[循环]选择“启用”，则寄存器内的数据会被复归为[偏移量]，并显示状态 0；否则寄存器内的数据将维持在([状态数] – 1) + [偏移量]，并显示状态([状态数] – 1)。也就是说，此时[偏移量]的设定值就是显示为“状态 0”。其他状态以此类推。

注意 ： 与“多状态指示灯”相同，“多状态切换开关”所显示的状态皆为寄存器内的数据减去[偏移量]。



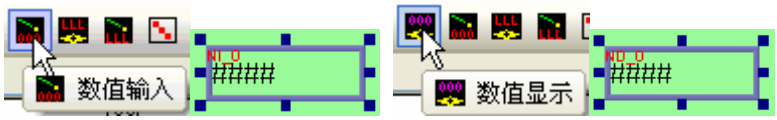
b、“减(JOG-)”

每按下一次控件，将对“写入地址”所指定寄存器内的数据减去 1，当选择“数据”显示模式时，如果减值的结果小于[偏移量]值时，且[循环]选择“启用”，则寄存器内的数据会被改变为([状态数] – 1) + [偏移量]，并显示状态([状态数] – 1)，否则寄存器内的数据将维持在[偏移量]，并显示状态 0。

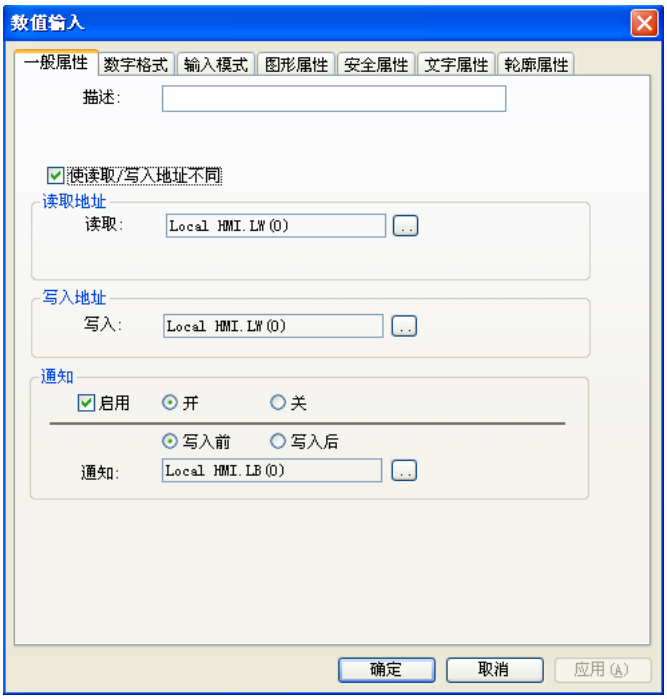
13.7 数值输入与数值显示控件 (numeric input and numeric display)

“数值输入”与“数值显示” 控件皆可以用来显示所指定寄存器内的数值，其中“数值输入”控件并可以使用键盘的输入值，更改寄存器内的数据。

打开 InoTouch Editor 软件菜单的“控件/ 数值/字符”选择“数值输入”或“数值显示”，或者工具栏上的图标  ，在窗口中点击鼠标左键，就建立了“数值输入”或“数值显示”控件，如下图所示。



选择“数值输入”或“数值显示” 双击或单击鼠标右键选择“属性”进行编辑，“数值输入控件属性对话框”与“数值显示控件属性对话框”的差别在于“数值输入”控件增加了“通知”与“键盘输入的设定”项目。下图为“数值输入”控件的[一般属性]设定页。如下图：



[读取地址]

数值的读取地址。读取该地址的数据。

[写入地址]

数值的写入地址。往该地址写入数据。

[通知]

在“数值输入”控件中使用此项设定，则在成功更改寄存器内的数值时(必须输入值在上下限定义的范围内，参考“数字格式”设定页的说明)，可以设定此项目所指定位地址的状态，使用“开”(ON)与“关”(OFF)选择要设定的状态。

[启用]

选择是否开启”通知”功能。

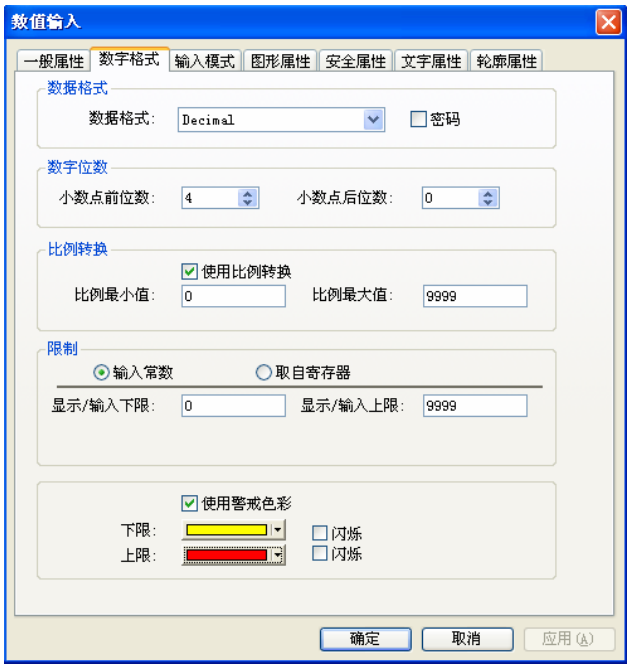
[写入前]

在寄存器中的数据被改变前就先设定所指定位地址的状态。

[写入后]

在寄存器中的数据被改变后才设定所指定位地址的状态。

下图为“数值输入”与“数值显示”控件皆包含的[数字格式]设定页，用来设定数值显示的方式。



[数据格式]

选择寄存器内数据的形式，可选择各数据类型如下图。

对以上数据格式的说明见下表。

资料格式	格式说明
Decimal	十进制数据类型
16-bit Binary	16 位二进制数据类型
32-bit Binary	32 位二进制数据类型

16-bit Hex	16 位 16 进制数据类型
32-bit Hex	32 位 16 进制数据类型

[密码]

勾选此项，则显示数值时将使用“*”号代替所有数字，并取消范围颜色警示功能。

[小数点前位数]

小数点前的显示位数。

[小数点后位数]

小数点后的显示位数。

[比例转换]

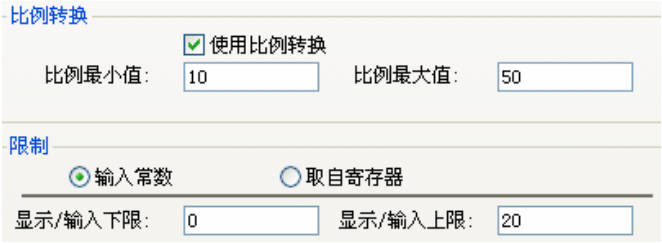
所显示的数据是利用寄存器中的原始数据经过公式换算后所获得。选择此项功能必须设定[比例最小值]，[比例最大值]与”限制”项目中的[输入下限]、[输入上限]。

假设原始数据使用 A 来表示，所显示的数据使用 B 来表示，则数据 B 可以使用下列的换算公式获得：

B = [输入下限] + (A - [比例最小值]) * 比例系数

其中比例系数 =([输入上限] - [输入下限]) /([比例最大值] - [比例最小值])

以下图的设定为例，当原始数据是 15 时，则经过换算得到的数值为 0 + (15 –10) * (20 - 0) / (50 – 10) = 2.5，控件上将显示 2（小数点后不显示）。



[限制]

用来设定输入数值上下限的来源，另外就是设定警示颜色与警示效果。

[输入常数]

选择输入数值的上下限分别来自“输入下限”(Input low)与“输入上限”(Input high)中的设定值。若输入值不在上下限定义的范围内，将无法更改该寄存器中的数值。

[取自寄存器]

选择输入数值的上下限来自所指定的寄存器。此时寄存器必须存在的资料长度与控件所显示的数据型态有关。举例来说，上图的上下限来自[LW100]，此时上下限的存放地址如下：

① 若显示的数据的型态为“Decimal”，则

[LW100]　　下限存放地址(十进制)

[LW100 + 2]　　上限存放地址(十进制)

② 若显示的数据的型态为“16-bit Binary”，则

[LW100]　　下限存放地址(16-bit Binary)

[LW100 + 1]　　上限存放地址(16-bit Binary)

③ 若显示的数据的型态为“32-bit Hex”，则

[LW100]　　下限存放地址(32-bit Hex)

[LW100 + 2]　　上限存放地址(32-bit Hex)

使用警示色彩

[下限]

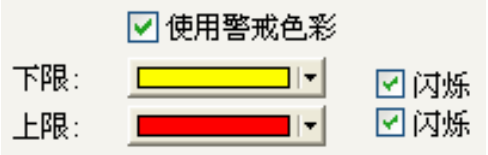
当寄存器内的数值小于输入下限设定值时，控件会使用此项设定的颜色来显示数值。

[上限]

当寄存器内的数值大于上限值时，控件会使用此项设定的颜色来显示数值。

[闪烁]

当寄存器内的数值小于下限值或大于上限值时，控件会使用闪烁的效果加以警示。



假如设定如上图所示，当寄存器读取的数值小于“下限”设定的数据时，该数值会显示为“黄色”，

并且在闪烁；当寄存器内的数值大于“上限”设定的数据时，该数值会显示为“红色”，且在闪烁。

下图为“数值输入”控件的[输入模式]设定页，用来设定数值输入时键盘输入的设定。

[输入次序]

当同一个画面上有多个数值输入控件的时候，目前的操作方法是触控需要修改数据的控件，键盘弹出，输入完毕后，键盘消失。接着再触控下一个需要修改数据的控件，键盘再弹出，输入完毕后，键盘再消失。如此循环。这样，在一次要修改多个数据寄存器的时候，这样的操作效率不高。InoTouch Editor 提供了“输入次序”的功能，使用该功能时，多个输入控件可以分次序分组别依次的输入，此时弹出的键盘在输入完一个数值输入控件并按下“Enter”后，不会消失，而是光标自动跳转到下一个数值输入控件。当数值输入控件都修改完成后，按下键盘上的“ESC”按键，键盘即消失。这样，可以大大的提高数值修改的效率。

[启用]

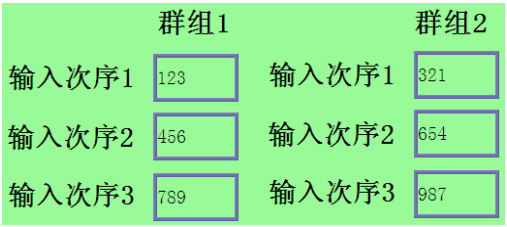
当勾选“启用”时，表示使用“输入次序”功能。

[输入次序]

设定修改数据时的输入次序。数值修改时以次序由小到大依次修改数据。

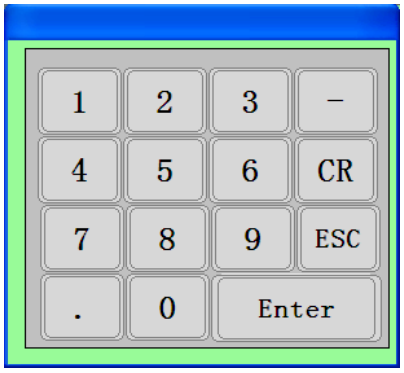
[群组]

设定该数值输入控件的群组编号。当画面上有多个数值输入控件，其输入次序也会按照不同的“群组”来依次输入。



[键盘]

使用“数值输入”控件时，允许使用者选择所使用的键盘型式，并设定键盘出现的起始位置。需选择存在键盘的窗口，执行时只要触控到“数值输入”控件，键盘会自动出现。更详细内容可参考《键盘的设计与使用》此章节。



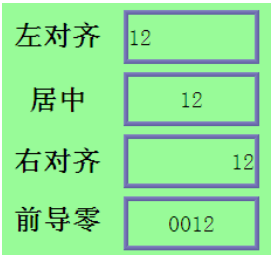
下图为“数值输入”与“数值显示”控件的[文字属性]设定页，用来设定数值显示时所使用的字体、字号与颜色，另外也包括数字对齐的方式。

[颜色]

当数值在上下限的范围内时，使用此项颜色显示。表示此时的数据是在正常范围内。

[对齐]

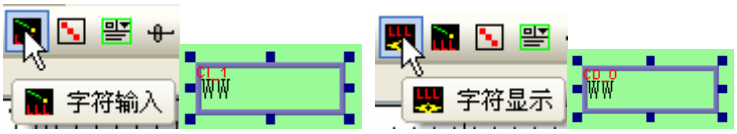
提供四种数字对齐方式：“左对齐”(left)、居中（ centered ）、“右对齐”(right)、 “前导零”(leading zero)，使用不同对齐方式的表现行为可参考下图。



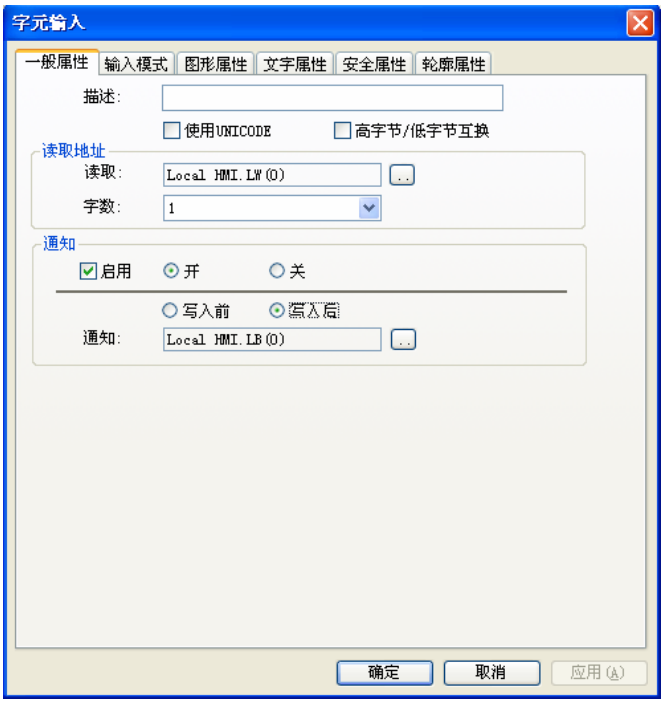
13.8 字符输入与字符显示控件 (ASCII input and ASCII display)

“字符输入”与“字符显示”控件使用 ASCII 编码的方式显示所指定寄存器中的数据，“字符输入”控件并可以用键盘的输入值，更改寄存器内的数据。

打开 InoTouch Editor 软件菜单的“控件/ 数值/字符”选择“字符输入”或“字符显示”，或者工具栏上的图标  ，在窗口中点击鼠标左键，就建立了“字符输入”或“字符显示” 控件，如下图所示。



选择“字符输入”或“字符显示” 双击或单击鼠标右键选择“属性”进行编辑。“字符输入”与“字符显示”控件属性对话窗的差别，在于“字符输入”控件增加“通知”与“键盘输入功能的设定”项目。下图为“字符输入”控件的[一般属性]设定页。如下图：



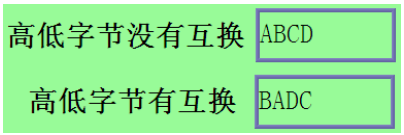
[使用 UNICODE]

勾选使用“UNICODE”后，则可以使用自己定义的“UNICODE”的键盘来输入自定义的字符。包

括汉字。在输入汉字时，必须此工程文件中要有这些汉字，否则无法显示出来。例如，现在使用自己定义的 UNICODE 的键盘输入“触摸屏”三个字，那么“触摸屏”这三个汉字必须要在当前的画面工程中有存在，不管是在哪个画面，无论是以何种方式存在，在此输入的三个字才会正常显示，否则无法显示出来。

[高字节/低字节互换]

勾选此项后，会将读取过来的字符高低字节颠倒来显示。如下图所示。

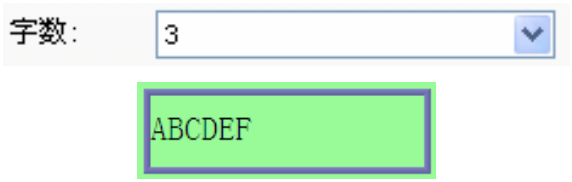


[读取地址]

字符的读取地址。同时也是往该地址修改字符数据。

[字数]

选择文字最多可显示的资料长度，单位为 word，可选择的最小值为 1。因每个 ASCII 字符长度为一个字节(byte)，所以每次最少会显示两个字符。以下图的设定为例，控件最多可以显示 3 * 2 = 6 个字符。



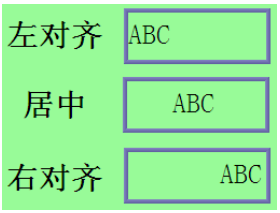
[通知]　[输入次序]　[键盘]

请参考“数值输入”的说明。

下图为“字符输入”与“字符显示”控件的[文字属性]设定页，用来设定字符显示时所使用的字体、字号与颜色，另外也包括文字对齐的方式。

[对齐]

提供三种文字对齐方式：“左对齐”、“居中”、“右对齐”，使用不同对齐方式的表现行为可参考下图。



注：直接窗口控件(direct window) /间接窗口控件 (indirect window)

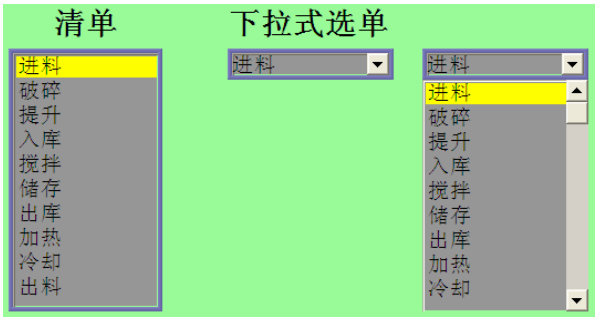
请参考《窗口》这一章的说明。

13.9 项目选单(Option List)

13.9.1 概要

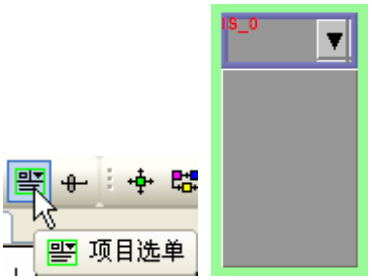
项目选单控件可以显示多样项目成一列表， 使用者可以藉此去检视并选择需要的项目。 一旦使用者选择了某一项目， 相对应的项目数值将被写入到设定的寄存器中。

项目选单有两种显示模式： 清单和下拉式选单。 清单可以完整显示所有的项目， 并把目前所选择的项目标示出来。 然而， 下拉式选单只显示目前所选择之项目。 但是当使用者点选下拉式选单时， 系统则会列出所有完整项目(类似于清单的显示法) 如下：



13.9.2 结构设定

打开 InoTouch Editor 软件菜单的“控件/项目选单”，或者工具栏上的图标，在窗口中点击鼠标左键，就建立了“项目选单”控件，如下图所示。



选择“项目选单”双击或单击鼠标右键选择“属性”进行编辑，如下图：

项目选单设定页

设　定	描　述
属性	[方式] 选择此对象显示方式：清单或下拉式选单。 [状态数] 设定此对象的状态数。每一个状态表示一个项目并会显示在列表上，此项目数值可被写入至[控制地址]。 [背景] 选择控件的背景颜色。 [选择] 设定当项目被选择时所标示的背景颜色。
控制地址	选择寄存器的[PLC 名称]，[地址类型]，[地址]去控制对象显示和系统状态数值写入。也即选择“控制地址”的来源和具体地址。
当按钮松开才发出指令	未勾选 当使用者碰触某项目时，系统将数值写入到[控制地址]。 勾选 当使用者碰触某项目并松开按钮时，系统将数值写入到[控制地址]。 注意：此选项只在清单模式下有作用。

状态设置页

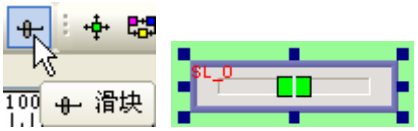
设　　定	描　　述
状态设置	此设置页显示所有状态/选项与文字和数值。如果要改变状态数，请从[项目选单设定页] ➔ [属性] ➔ [状态数]中修改。 [状态] 系统会列出目前所有使用的状态。每一个状态表示一个项目并且会显示在列表。此字段为只读。 [数值] 使用者可为每个项目设定数值，但须遵守以下两个规范： [读取] 如果系统侦测到[控制地址]的内容有任何改变，对象将会对照内容和其数值并选择第一个吻合的项目。如果没有项目吻合，将跳至错误状态并触发通知位 (如果有设定)。 [写入] 当使用者选择某项目，系统将数值写入至[控制地址]。 [文字] 使用者可为每个项目设定文字。项目选单对象将显示所有项目的文字在列表上供使用者检视和选择。 [错误状态] 如上图所示，当[状态数]设为 10 时，状态 10 即为错误状态。同样的，如果[状态数]设为 11，那状态 11 即为错误状态。 在错误状态发生时，清单模式将移除“选择”标示来表示没有任何项目被选择，而下拉式选单模式则会显示错误状态的文字。 错误状态的文字只能应用于下拉式选单模式，清单模式无法使用错误状态文字。

恢复默认值	将所有状态数值设为默认值，例如，设状态 0 为 0，状态 1 为 1…等等。
通知	[启用] 当错误发生时，系统将某个特定位设 ON/OFF。此位寄存器的通知可以使用于触发某个动作来修正错误。

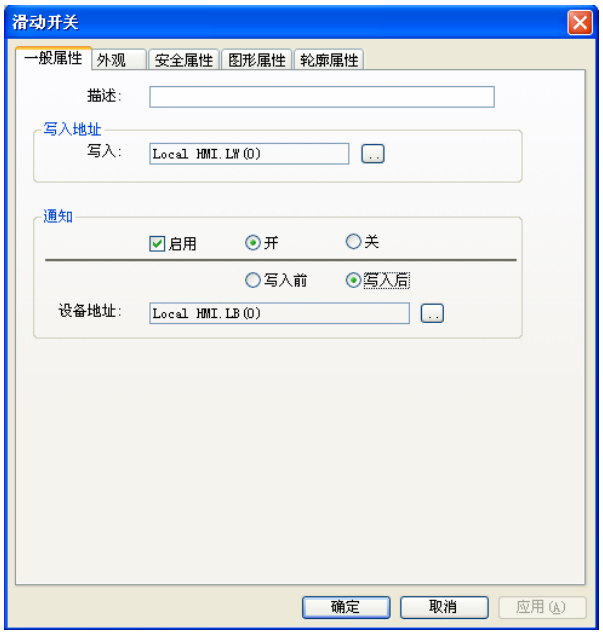
13.10 滑动开关控件 (slide object)

滑动开关控件是用来显示和线性修改指定寄存器内的数值。

打开 InoTouch Editor 软件菜单的“控件/滑动开关”，或者工具栏上的图标，在窗口中点击鼠标左键，就建立了“滑动开关”控件，如下图所示。



选择“滑动开关”双击或单击鼠标右键选择“属性”进行编辑，如下图：



[写入地址]

需要修改数据的地址。

[通知]

设定此项目所指定寄存器的状态，使用“开”(ON)与“关”(OFF)选择要设定的状态。

[启用]

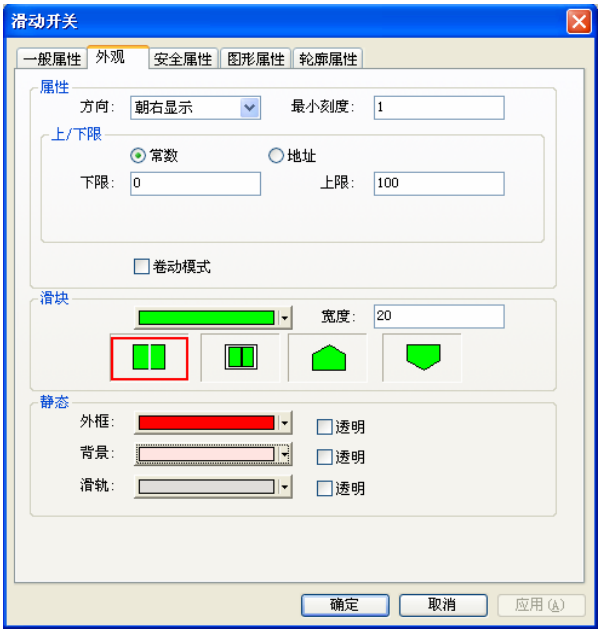
选择是否开启此项功能。

[写入前]

在寄存器中的数据被改变前就先设定所指定寄存器的状态。

[写入后]

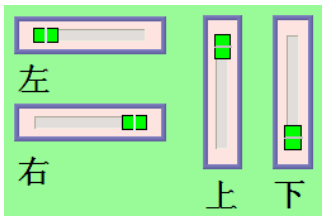
在寄存器中的数据被改变后才设定所指定寄存器的状态。



属性

[方向]

滑动开关控件可以四个方向来显示(朝右显示，朝上显示， 朝左显示，朝下显示)



[最小刻度]

依照所填入之最小刻度值来显示,例如 N 是最小刻度时，当

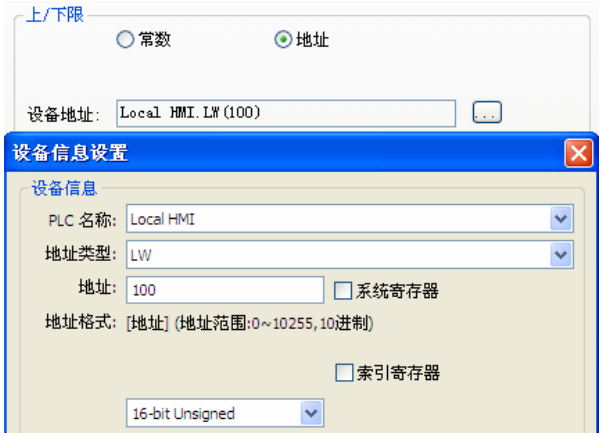
N=10，数值显示为每一次的显示都是依据 10 的刻度来变化,而且是任意的 10 的倍数

N=5， 数值显示为每一次的显示都是依据 5 的刻度来变化,而且是任意的 5 的倍数

N=1， 数值显示为每一次的显示都是依据 1 的刻度来变化,而且是任意的 1 的倍数

[下限]&[上限]

在此设定修改数据的上下限限制。如果勾选“地址”，则依据所设定的寄存器中的数据作为上下限的限制。假如此处的设定如下图所示：



则表明下限值由寄存器 LW100 中的数据决定，上限值由寄存器 LW101 中的数据决定。此处下限值寄存器地址和上限值寄存器地址的具体地址由数据格式来决定。

[卷动模式]

依据卷动模式内的设定值，会使滑动开关控件，依此设定值来递增或递减。

[滑块]

共有四种滑块样式可供选择，也可调整滑块宽度。

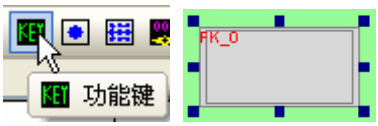
[外框][背景][滑轨]

可选择外框,背景,滑轨的颜色。

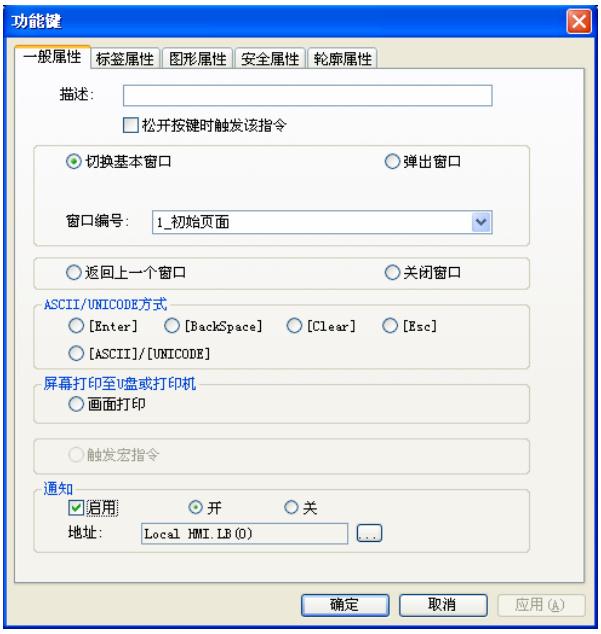
13.11 功能键控件 (function key)

“功能键”控件提供窗口切换、弹出窗口、关闭窗口等功能，也可用来设计键盘的按键。

打开 InoTouch Editor 软件菜单的“控件/功能键”，或者工具栏上的图标，在窗口中点击鼠标左键，就建立了“功能键”控件，如下图所示。



选择“功能键”双击或单击鼠标右键选择“属性”进行编辑，如下图：



“功能键”控件提供下列几种操作模式：

[松开按键时触发该指令]

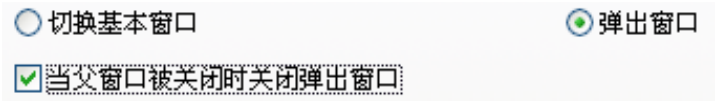
使用此性质表示必须在释放按压控件的动作后，选择的动作才会被执行。未选择此性质，则在触控控件后，将立刻执行选择的动作，例如切换窗口。

[切换基本窗口]

切换基本窗口。切换后将会关闭当前窗口，并显示目标窗口。

[弹出窗口]

弹出其它窗口。此时弹出的窗口必定在基本窗口的上面。使用此功能可以选择是否使用[当父窗口被关闭时关闭弹出窗口]，参考下图。选择此属性则弹出的窗口会在发生换页动作时自动消失，否则使用者必须自行在被弹出的窗口上设计[关闭窗口]功能键来关闭此窗口。



[窗口编号]

此项目用来选择在“切换基本窗口”、 “弹出窗口”时所使用的窗口。

[返回上一个窗口]

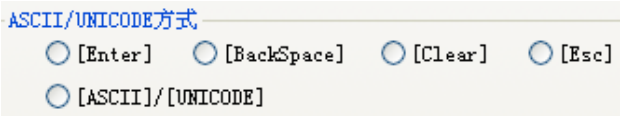
返回前一页基本窗口。例如当由“窗口 10”切换到“窗口 20”时，使用此功能可以再返回“窗口 10”。此功能只对基本窗口有效。

[关闭窗口]

关闭在基本窗口上的弹出的窗口，包括系统讯息窗口。

[ASCII 模式]

ASCII 模式被用来作为键盘的输入讯号，主要用在“数值输入”与“字符输入”控件需要使用键盘来输入数字或字符的场合。更详细的说明可参考“键盘的设计与使用”的章节。



[Enter]

与键盘的输入(Enter)动作相同。

[Backspace]

与键盘的后退删除(Backspace)动作相同。

[Clear]

清除目前对“数值输入”与“文符输入”控件已输入的资料。

[Esc]

与使用[关闭窗口]功能相同，可用来关闭弹出的键盘窗口。

[ASCII] / [UNICODE]

设定对“数值输入”与“字符输入”控件的输入字符，可选择 0，1，2，...数字键或 a，b，c，...等其它 ASCII 码。

通知

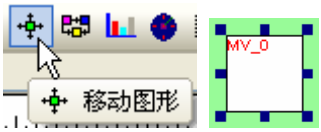
[启用]

使用此项设定，则在完成设定动作后可以连带设定此项目所指定位地址的状态，使用[开](ON)与[关](OFF)选择要设定的状态。

13.12 移动图形控件 (moving shape)

“移动图形”控件会利用寄存器内的数据，决定控件的状态与控件的移动距离。

打开 InoTouch Editor 软件菜单的“控件/移动图形”，或者工具栏上的图标，在窗口中点击鼠标左键，就建立了“移动图形”控件，如下图所示。



选择“移动图形”双击或单击鼠标右键选择“属性”进行编辑，如下图：

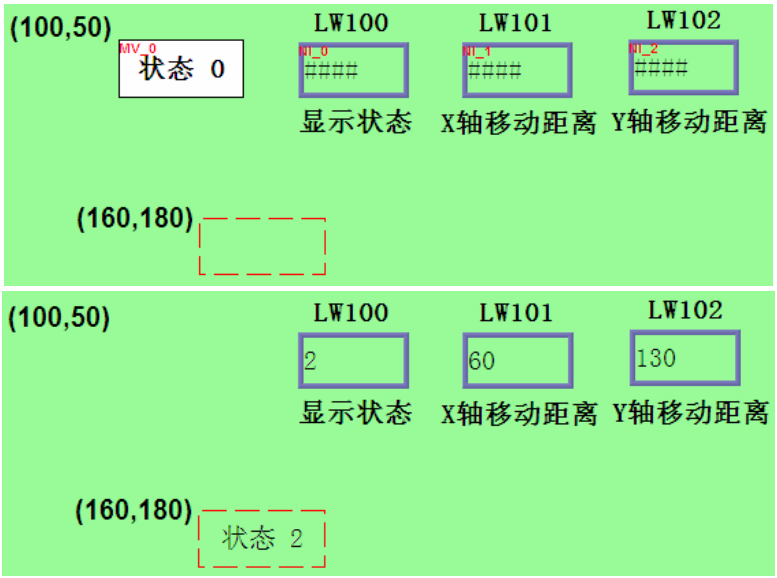
[读取地址]

控件状态与移动距离的读取地址。此时控件状态与移动距离的读取地址整理如下表。表中的 address 表示读取寄存器的地址值，例如读取寄存器为[LW100]时，address 等于 100。

变量型态	控件状态读取地址	X 轴方向移动距离读取地址	Y 轴方向移动距离读取地址
16-bit BCD	address	address + 1	address + 2
32-bit BCD	address	address + 2	address + 4
16-bit Unsigned	address	address + 1	address + 2
16-bit Signed	address	address + 1	address + 2
32-bit Unsigned	address	address + 2	address + 4
32-bit Signed	address	address + 2	address + 4

举例来说，若设定读取寄存器为[LW100]，且变量型态使用“16-bit Unsigned”，则[LW100]存放控件的状态，[LW101]存放 X 轴方向的移动距离，[LW102]存放 Y 轴方向的移动距离。

以下图为例，控件的地址为[LW100]且起始地址为(100， 50)，假使现在要移动控件至(160，180)且显示状态 2 的图形，则[LW100]需设定为 2，[LW101] = 160-100 = 60，[LW102] = 180-50 = 130。



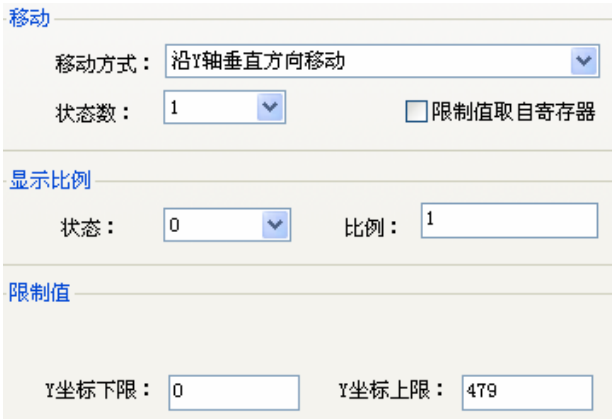
[移动方式]

a、沿 X 轴水平方向移动

只允许控件沿着 X 轴作水平方向的移动。移动范围由[X 轴坐标下限]与[X 轴坐标上限]来决定。

b、沿 Y 轴垂直方向移动

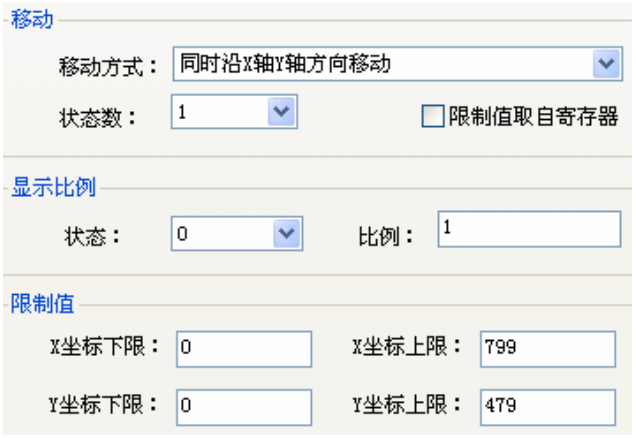
只允许控件沿着 Y 轴作垂直方向的移动。移动范围由[Y 轴坐标下限]与[Y 轴坐标上限]来决定。



c、可同时作 X 方向与 Y 方向的移动

允许控件沿着 X 轴与 Y 轴移动。移动范围由[X 轴坐标下限]、[X 轴坐标上限]与[Y 轴坐标下限]、

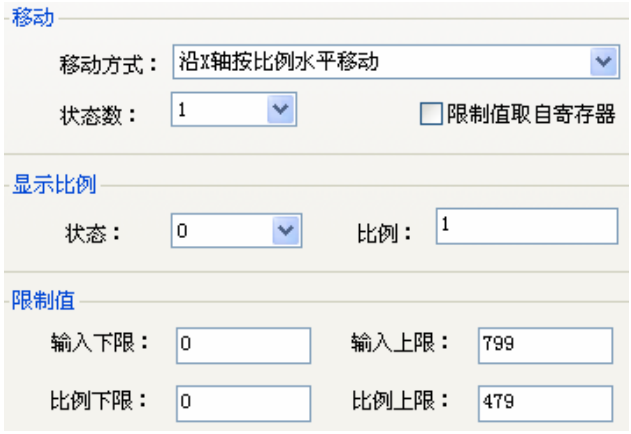
[Y 轴坐标上限]来决定。



d、沿 X 轴、按比例作水平方向的移动

只允许控件沿着 X 轴、按比例作水平方向的移动。假设寄存器中与 X 轴位移有关的数据为 data，则 X 轴的位移量可以使用下面的公式：

X 轴位移 = (data – [输入下限]) * ([比例上限–比例下限]) / ([输入上限] – [输入下限])



例如控件只允许作 200~500 大小的位移，但寄存器数据的大小范围为 300~1000，此时可以将[输入下限]设定为 300，[输入上限]设定为 1000，[比例下限]设定为 200，[比例上限]设定为 500，控件即会在要求的范围内移动。

e、沿 Y 轴、按比例作垂直方向的移动

只允许控件沿 Y 轴、按比例作垂直方向的移动，Y 轴位移量的换算公式与“沿 X 轴按比例作水平方向的移动”相同。

f、沿 X 轴、按反比例作水平方向的移动

此项功能与“沿 X 轴、按比例作水平方向的移动”相同，但移动方向相反。

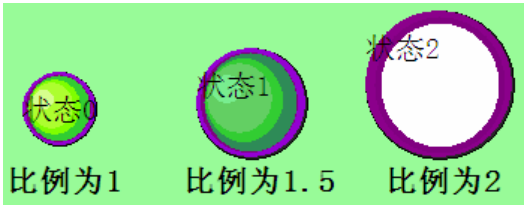
g、沿 Y 轴、按反比例作垂直方向的移动

此项功能与“沿 Y 轴、按比例作垂直方向的移动”相同，但移动方向相反。

[显示比例]

控件各个状态的图形在显示时，可以分开设定图形缩放比例，参考下图。“标签”中的文字不

会按照比例缩放。



[限制值地址]

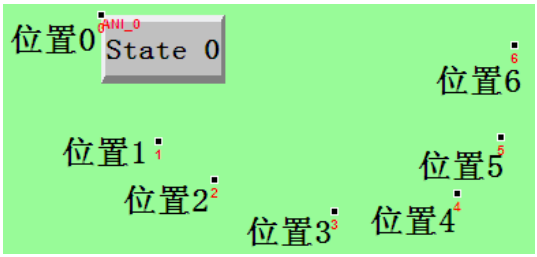
控件的显示区域除了可以直接设定[X 轴坐标下限]、[X 轴坐标上限]与[Y 轴坐标下限]、[Y 轴坐标上限]来决定外，也可以利用寄存器中的数据来决定。假设显示区域由 address 地址内的数据来决定，[X 轴坐标下限]、[X 轴坐标上限]与[Y 轴坐标下限]、[Y 轴坐标上限]的读取地址可参考下表。

变量型态	[X 轴坐标下限] 读取地址	[X 轴坐标上限] 读取地址	[Y 轴坐标下限] 读取地址	[Y 轴坐标上限] 读取地址
16-bit BCD	address	address + 1	address + 2	address + 3
32-bit BCD	address	address + 2	address + 4	address + 6
16-bit Unsigned	address	address + 1	address + 2	address + 3
16-bit Signed	address	address + 1	address + 2	address + 3
32-bit Unsigned	address	address + 2	address + 4	address + 6
32-bit Signed	address	address + 2	address + 4	address + 6

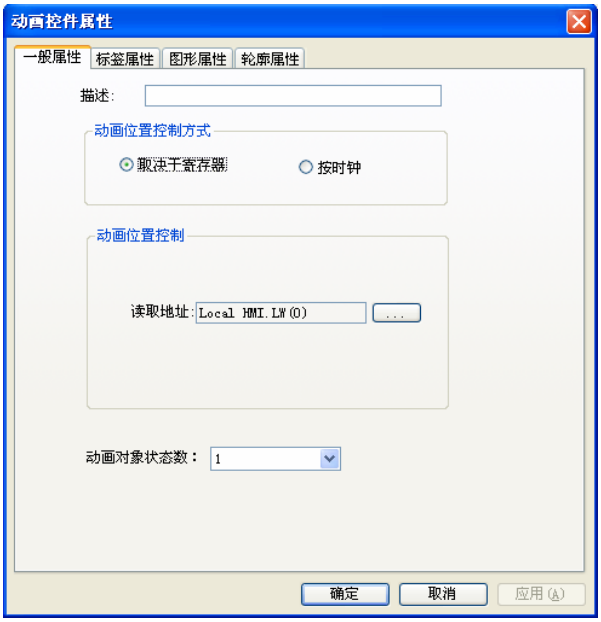
13.13 动画控件 (animation)

“动画”控件是事先定义了动画的移动轨迹，并利用更改寄存器内的数据，控制控件的状态与控件在该事先定义的移动轨迹上的位置。

要增加“动画”控件可以打开 InoTouch Editor 软件菜单的“控件/动画”，或者工具栏上的图标 后，在适当位置单击鼠标的左键，即可定义一个新的移动位置，将鼠标移动到另外一个位置并单击鼠标左键，则又定义了一个轨迹的位置。定义轨迹的第一个位置，其编号为 0，接下来为 1，以此类推。定义完成全部的移动位置后，按下鼠标的右键即可完成移动轨迹的规划并新增一个新的“动画”控件，参考下图。



要更改控件的属性，可以使用鼠标左键双击“动画”控件或单击鼠标右键选择“属性”进行编辑，下图为“动画”控件一般属性设定页。



[动画对象状态数]

设定控件的总状态数目。

[动画位置控制方式]

如果选择“取决于寄存器”，则控件的状态与位置由寄存器中的数据决定。

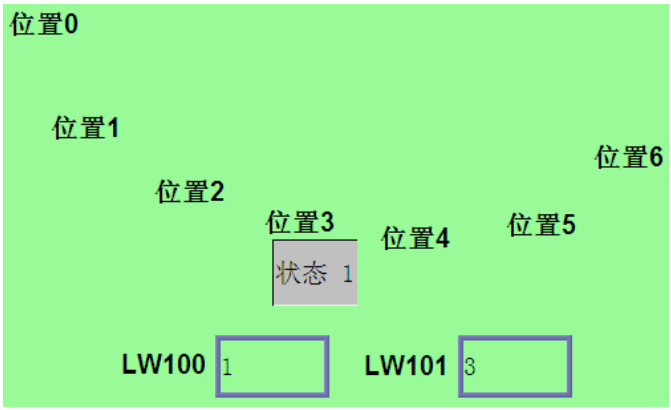
[读取地址]

如果控件的状态与位置由寄存器中的数据决定，必须正确设定控件状态与位置的读取地址。读取地址整理如下表。表中的 address 表示寄存器的地址值，例如寄存器为[LW100]时，address 等于 100。

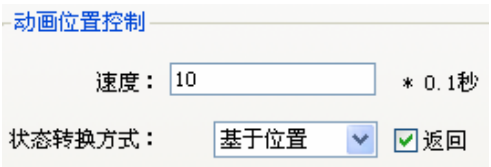
变量型态	控件状态读取地址	控件位置读取地址
16-bit BCD	address	address + 1
32-bit BCD	address	address + 2
16-bit Unsigned	address	address + 1

16-bit Signed	address	address + 1
32-bit Unsigned	address	address + 2
32-bit Signed	address	address + 2

举例：若寄存器为[LW100]，且变量型态使用“16-bit Unsigned”，则[LW100]存放控件的状态，[LW101]存放控件的显示位置。以下图为例，[LW100] = 1，[LW101] = 3，所以控件显示状态 1，并出现在位置 3。



若控件不选择“取决于寄存器”而选择“按时钟”的变化，则控件将自动改变状态与显示位置，“自动控制位置”项目用来设定状态与显示位置改变方式。



[速度]

位置改变速度，单位为 0.1 秒。例如设定为 10，则控件每隔 1 秒钟变换一个位置。

[返回]

假设控件有 4 个位置，分别为位置 0、位置 1、位置 2、位置 3。若未选择此项设定，当移动到最后一个位置(位置 3)后，将移动到初始位置位置 0，再重复原来位置改变方式，移动位置整理顺序如下：

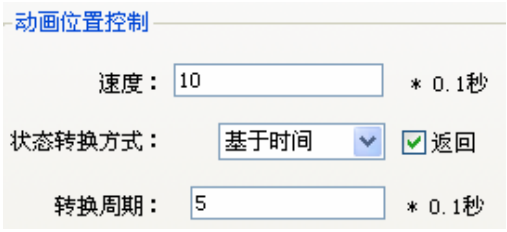
位置 0->位置 1->位置 2->位置 3-> 位置 0-> 位置 1-> 位置 2...

若选择此项设定，当移动到最后一个位置后，将使用反向的移动方式，移动到初始位置位置 0，再重复原来位置改变方式，移动位置整理顺序如下：

位置 0-> 位置 1->位置 2->位置 3-> 位置 2-> 位置 1-> 位置 0...

[状态转换]

状态改变的方式，可以选择“基于位置”与“基于时间”。选择“基于位置”表示位置改变，状态也随着改变。若选择“基于时间”，表示状态使用固定的频率自动变换，变换频率在[转换周期]中设定，参考下图。



下图的对话框用来设定“动画”控件的外型大小，也是利用鼠标双击“动画”控件即可出现。

[图形尺寸]

用来设定动画控件所显示图形的大小。

[控制点位置]

用来设定移动轨迹上各点的坐标位置。

关于“移动图形”和“动画”两个控件的区别

通过上面的说明，可以了解“移动图形”和“动画”的具体执行过程。

“移动图形”是利用连续的寄存器的地址的内容，来将画面上显示的物件从一个地方移动到另外一个地方，且其图形状态可以跟着变化。其移动的位置只能是沿着 X 轴、或者 Y 轴。或者同时沿着 X 轴和 Y 轴来移动，移动的距离完全有寄存器中的数值来决定。

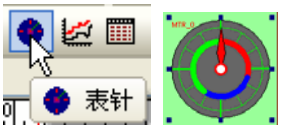
“动画”也是依据定义的连续的寄存器中的数据来实现图形物件的运动，移动过程中根据寄存

器的数据也可以改变图形的状态。而不同之处在于，“动画”的运动是根据事先定义好的位置来运动的，无法超越这个事先定义好的位置的范围，只能在这些事先定义好位置的地方出现。

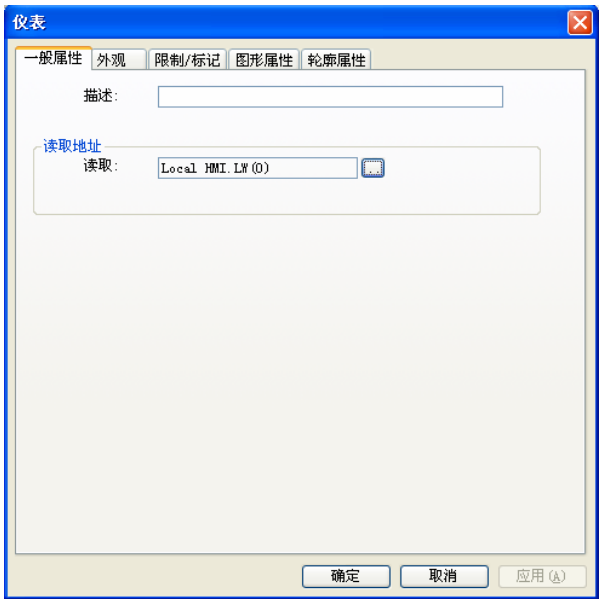
13.14 表针控件 (meter display)

“表针”控件会使用仪表图的方式，指示目前寄存器中的数据。

打开 InoTouch Editor 软件菜单的“控件/表针”，或者工具栏上的图标，在窗口中点击鼠标左键，就建立了“表针”控件，如下图所示。

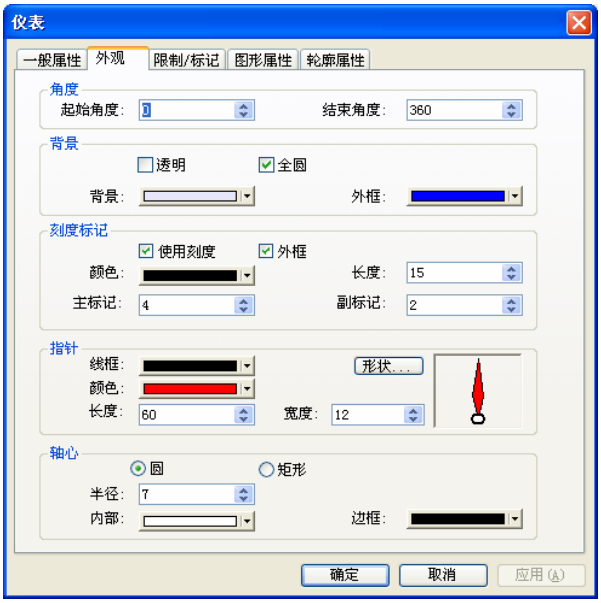


选择“表针”双击或单击鼠标右键选择“属性”进行编辑，如下图：

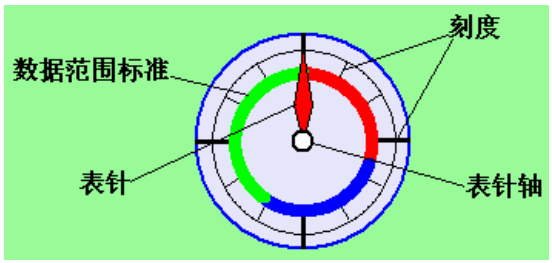


[读取地址]

用来设定表针控件数据的读取地址。

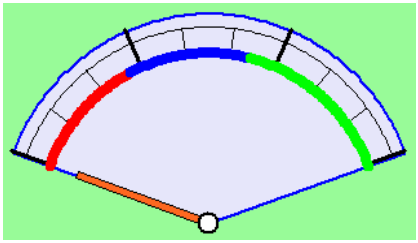


上图的设定对话窗用来设定“表针”控件的外观，各部分的名称可以参考下图。

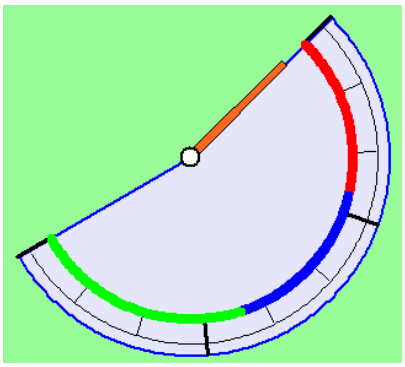


[角度]

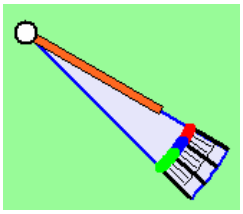
用来设定控件的起始角度与结束角度，角度可设定范围皆为 0~360 度。不同的设定值对控件外观的影响，可参考下面的几种不同的设定。



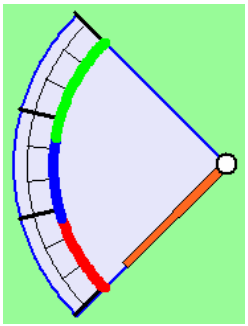
[起始角度] = 290，[结束角度] = 70



[起始角度] = 45，[结束角度] = 240



[起始角度] = 120，[结束角度] = 135



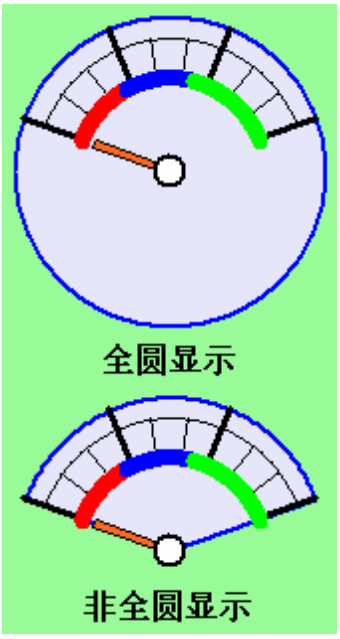
[起始角度] = 225，[结束角度] = 315

[背景]

设定控件的背景与圆周的颜色。

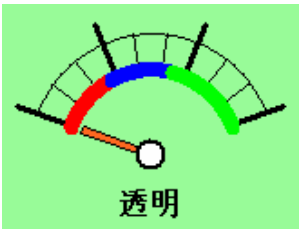
[全圆]

选择“全圆”，控件将显示全圆，否则只显示角度定义的范围，参考下图。



[透明]

选择“透明”，控件将不显示背景与圆周的颜色，参考下图。



[刻度标记]

设定控件的刻度数目与颜色。

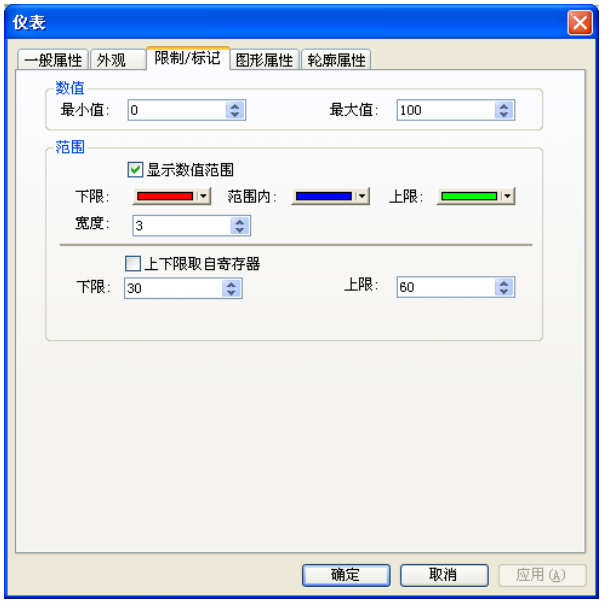
[指针]

设定控件指针的样式、长度、宽度与颜色。

[轴心]

设定控件指针轴心的样式、半径与颜色。

下图为上下限值设定对话框。



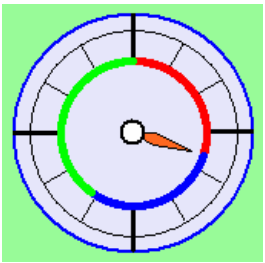
[数值]

设定控件所要显示的数值范围。“表针”控件会利用[最小值]与[最大值]的设定内容和由[读取地址]所读取的数值，换算指针的指示位置。举例来说，假使[最小值] = 0，[最大值] = 100，若此时读取的数据为 30，且[起始角度] = 0，[结束角度] = 360，则指针指示的角度为 (在[结束角度]大于[起始角度]的情形下)：

{(30 – [最小值])/([最大值] – [最小值])} * ([结束角度] - [起始角度]) =

{(30 – 0) / (100 - 0)} * (360 – 0) = 108

指针将指示在 108 度的位置，参考下图。



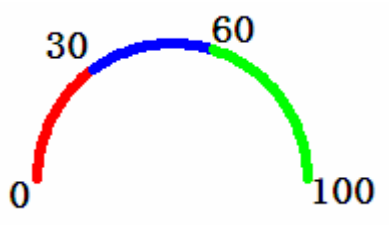
[范围]

设定上、下限值与上、下限标志的显示颜色与宽度。

[显示不同数值范围内的颜色]

选择是否显示上下限标志

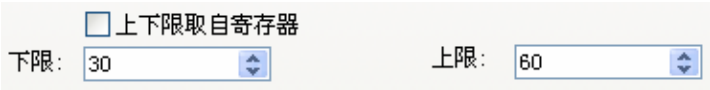
下图则为利用上面的设定值所显示的高低限标志。



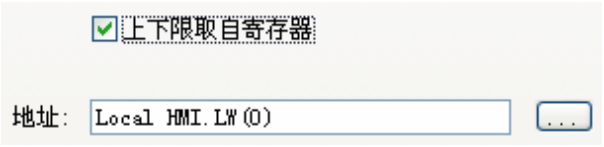
即数据小于 30 时，表针指示在红色区域里面，大于 30 小于 60 的范围，指针在蓝色区域里面，大于 60 小于 100 时，指针在绿色区域内。

[上下限取自寄存器]

未选择“上下限的值取自寄存器”，则高、低限值为固定值，来自直接设定的内容，参考下图。此时上限值为 60，下限值为 30。



若选择“上下限取自寄存器”，则上、下限值由寄存器中的数值来决定，参考下图。



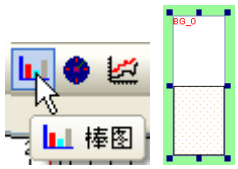
下表整理了上、下限的读取位置，其中“address” 表示寄存器的地址值，例如寄存器为[LW100]时，“address”等于 100。

变量型态	高限读取地址	低限读取地址
16-bit BCD	address	address + 1
32-bit BCD	address	address + 2
16-bit Unsigned	address	address + 1
16-bit Signed	address	address + 1
32-bit Unsigned	address	address + 2
32-bit Signed	address	address + 2

13.15 棒图控件 (bar graph)

“棒图”控件使用百分比例与棒图的方式，显示寄存器中的数据。

打开 InoTouch Editor 软件菜单的“控件/棒图”，或者工具栏上的图标，在窗口中点击鼠标左键，就建立了“棒图”控件，如下图所示。

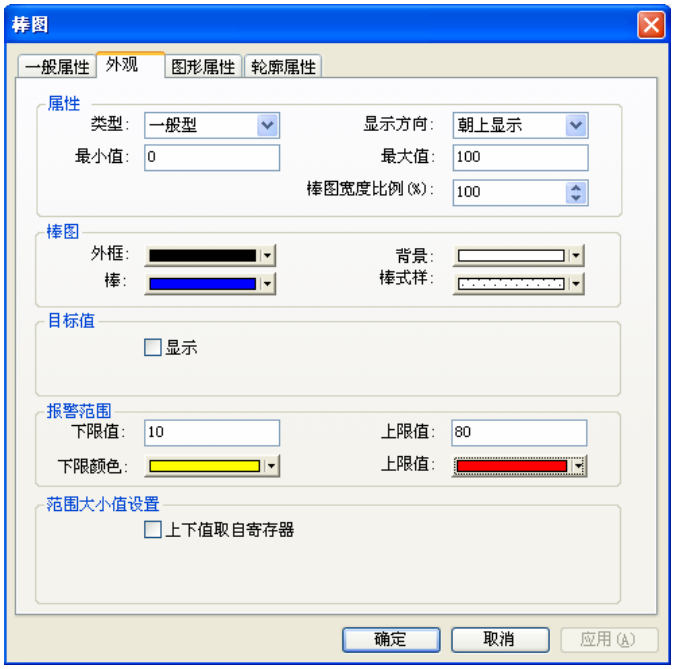


选择“棒图”双击或单击鼠标右键选择“属性”进行编辑，下图为“棒图”控件的一般属性设定页。

[读取地址]

数据的读取地址。即设定显示棒图状态的数据来源。

下图为“棒图”控件的外观设定页。

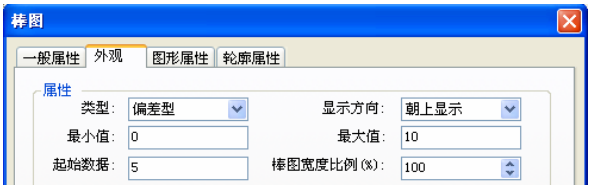


属性

[类型]

可以选择“一般型”与“偏差型”。当选择偏差型时，需设定原点位置，参考下图。则其原点的位

置在寄存器的数据=5。



[显示方向]

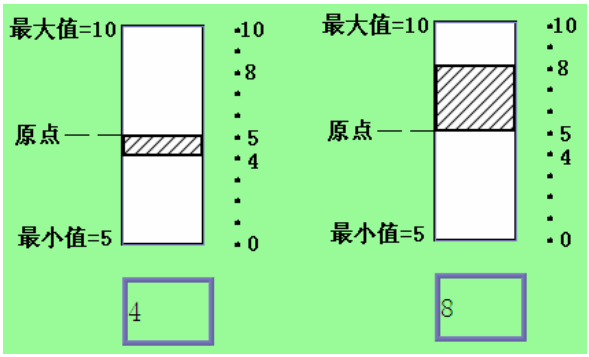
用来选择棒图的显示方向，可以选择“朝上显示”、“朝下显示”、“朝右显示”、“朝左显示”。

[最小值]、[最大值]

棒图填充的百分比可以利用下列的公式换算而得：

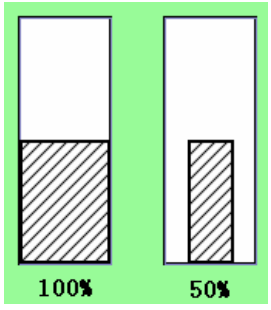
棒图填充区域百分比 = (寄存器数据 – [最小值]) / ([最大值] – [最小值]) *100%

但当选择偏差型时，若(寄存器数据– [原点位置])大于 0，则棒图将由[原点位置]的位置往上填充；若(寄存器数据– [原点位置])小于 0，则棒图将由[原点位置]的位置往下填充。下图显示在[原点位置]设定为 5，[最大值]为 10，[最小值]为 0 并使用不同数据时，棒图的填充情形。



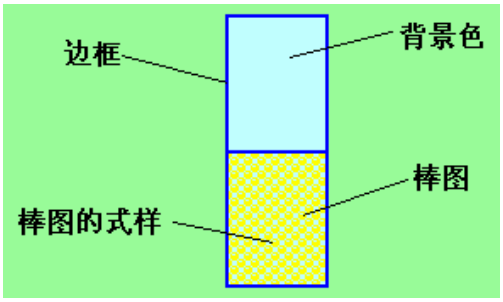
[棒图宽度比例(%)]

设定棒图的显示宽度与控件宽度间的百分比率，下图为使用两种不同设定值的显示情形。



棒图颜色/样式项目

用来指定棒图外框、背景颜色与填充区域的式样与颜色，参考下图。

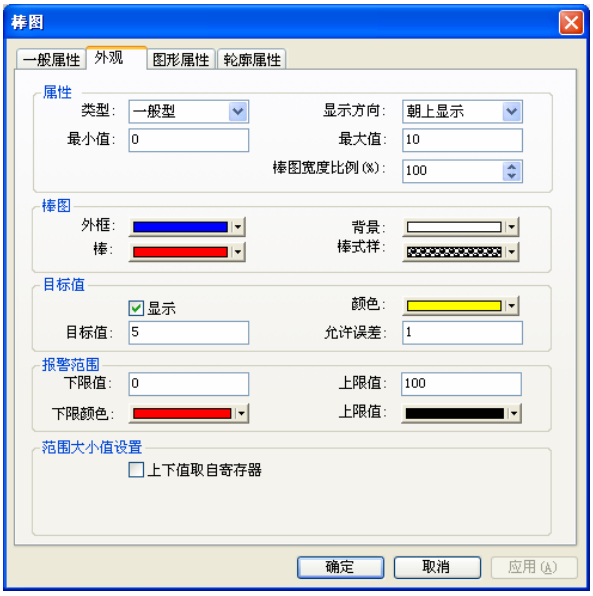


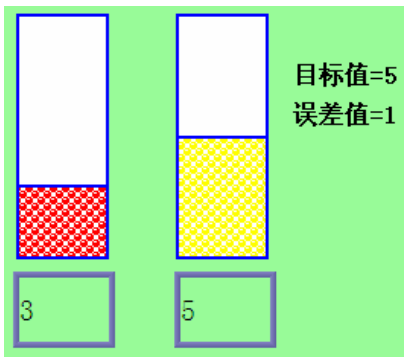
[目标值]

当寄存器内的数据符合下列的条件时，填充区域的颜色可以变更为此项目所定义的颜色。

[目标值] – [允许误差]＜＝寄存器内的数据＜＝[目标值] + [允许误差]

参考下图，此时[目标值] = 5，[误差值] = 1，则寄存器的值大于或等于 5-1=4，且小于或等于 5 + 1= 6，填充区域的部分将改变为“目标值颜色”。





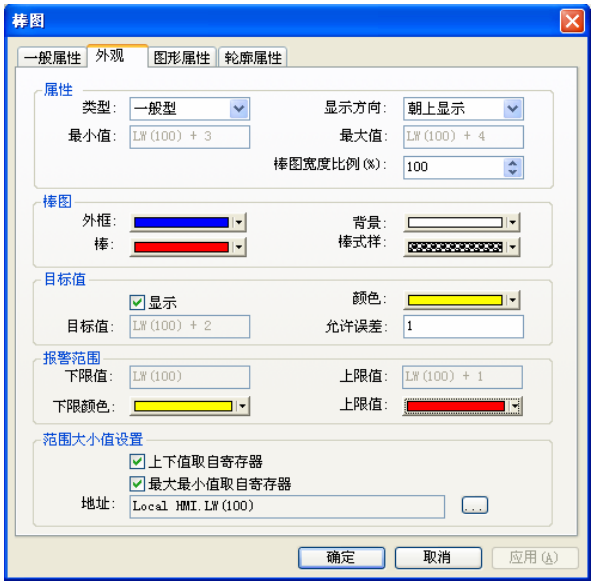
[报警范围]

当数据大于[上限值]时，填充区域的颜色可以变更为[上限颜色]所定义的颜色；若当数据小于

[下限值]时，填充区域的颜色可以变更为[下限颜色]所定义的颜色。

[范围大小值设置]

当选择[上下值取自寄存器]，“范围报警”中所使用的[下限值]、[上限值]与“目标值”中的[目标值]皆读取自指定的寄存器。若同时勾选了“最大/最小值取自寄存器”，则同样的最大值与最小值的设定值由设定的寄存器中的数据来决定。参考下图。



下表整理了上下限与目标值的读取位置，其中“address”表示寄存器的地址值，例如寄存器为[LW100]时，“address”等于 100。

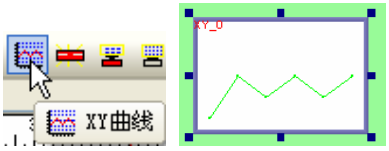
变量型态	[报警值下限] 读取地址	[报警值上限] 读取地址	[目标值] 读取地址	[最小值] 读取地址	[最大值] 读取地址
16-bit BCD	address	address + 1	address + 2	address + 3	address+4
32-bit BCD	address	address + 2	address + 4	address + 6	address+8
16-bit Unsigned	address	address + 1	address + 2	address + 3	address+4
16-bit Signed	address	address + 1	address + 2	address + 3	address+4
32-bit Unsigned	address	address + 2	address + 4	address + 6	address+8
32-bit Signed	address	address + 2	address + 4	address + 6	address+8

13.16 XY 曲线 (XY Plot)

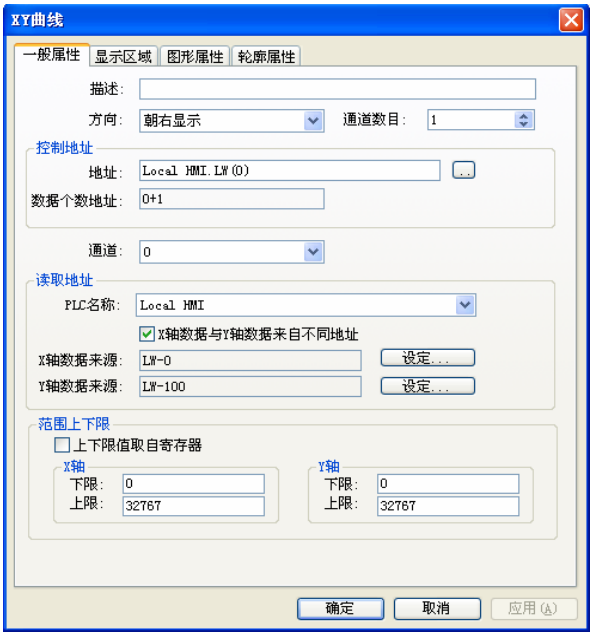
两组连续寄存器中的数据，寄存器中的数据分别为 X 轴和 Y 轴上的坐标，由这些坐标形成的点/形成的图形，就是 XY 曲线。XY 曲线可以是点或者是曲线的方式，还可以将形成的线往 X 轴或者 Y 轴投影，形成新的曲线。

XY 曲线可以同时显示最多 16 组的曲线，可让使用者由此方式来观察及比较各寄存器中的资料，负数亦可使用。

打开 InoTouch Editor 软件菜单的“控件/XY 曲线”，或者工具栏上的图标  ，在窗口中点击鼠标左键，就建立了“XY 曲线”控件，如下图所示。

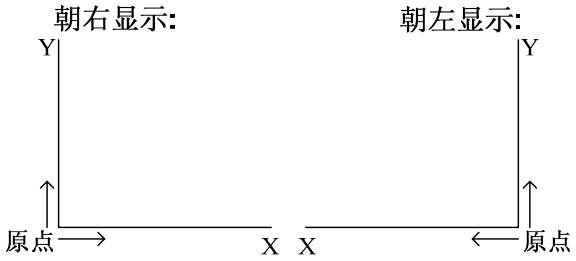


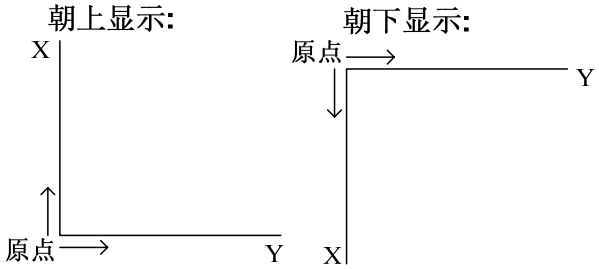
选择“XY 曲线”双击或单击鼠标右键选择“属性”进行编辑，如下图：



[一般属性]

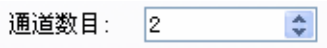
方向： 选择朝右显示，朝左显示，朝上显示或朝下显示。图示方向如下，均以 X 轴为正方向。





[通道数目]

设定[通道数目]。通道数目用来设定使用者欲观察 XY 曲线的数量。



例如上图显示通道数目被设定为 2，则使用者可以同时显示两条 XY 曲线图。

设定“控制地址”

[控制地址]

“控制地址” 被用来控制图形的显示及清除；假设“控制地址” 被设置为 LW10，则”数据个数地址”为 LW(10+1)，即 LW11。XY 曲线的显示与清除的过程与“数据群组”控件的曲线显示与清除的过程一样，即：

a．“控制地址”数据为 1 时，显示 XY 曲线

在[控制地址]写入“1” (将此地址的 bit 0 设定为 ON)；此时 InoTouch Editor 将以折线图画出目前寄存器的内容 (并保留先前图形)。InoTouch Editor 在完成前项动作后将对[控制地址]写入“0”。

b．“控制地址”数据为 2 时，清除目前显示的 XY 曲线

在[控制地址]写入“2” (将此地址的 bit 1 设定为 ON)；将清除先前显示的 XY 曲线。InoTouch Editor 在完成前项动作后将对[控制地址]写入”0”。

c．“控制地址”数据为 3 时，清除已显示的 XY 曲线并显示新的 XY 曲线

在[控制地址]写入“3” (将此地址的 bit 0 与 bit 1 皆设定为 ON)；此时 InoTouch Editor 会先将先前的 XY 曲线清除，再以目前地址内的数据形成新的 XY 曲线。InoTouch Editor 在完成前项动作后将于[控制地址]写入“0”。

d．“控制地址”数据为 4 时，清除先前所有的命令

在[控制地址]写入“4” (将此地址的 bit 0 与 bit 1 皆设定为 ON)；此时将清除先前所输入的所有尚未执行完的命令，停止执行，但已经画出的（部分）曲线不作清除。

在设定完控制地址后，InoTouch Editor 会自动设定数据个数地址。例如，如果 LW10=控制地址；此地址是用来控制曲线的显示及清除的；LW11 会自动被设成数据个数地址，也即点数地址；此地址是用来储存数据(点数)显示的数量。

此处说的点数，是指由 X 轴数据和 Y 轴数据形成的坐标点数。例如，点数为 3 时，则总共的

数据为 X 轴数据(X0，X1，X2)，Y 轴数据(Y0，Y1，Y2)，这样形成的点即为(X0，Y0)，(X1，Y1)，(X2，Y2)总共三个"点数"。也就是说要读 3 个 X 轴的数据，3 个 Y 轴的数据。以此类推。

[数据个数地址]

设定所指定 XY 曲线图要显示的数据个数；每个通道的点数小于 1024 点(1~1023) 。

[通道]

用来指定要设定哪条 XY 曲线的属性。

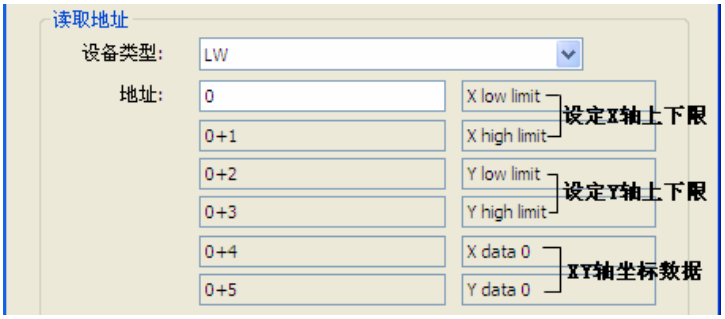
读取地址及范围上下限

[PLC 名称]

选择曲线图的数据来源。

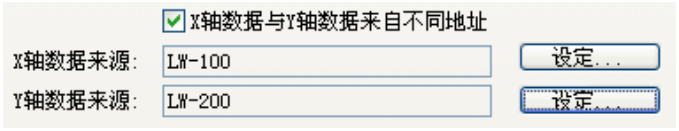
[读取地址]

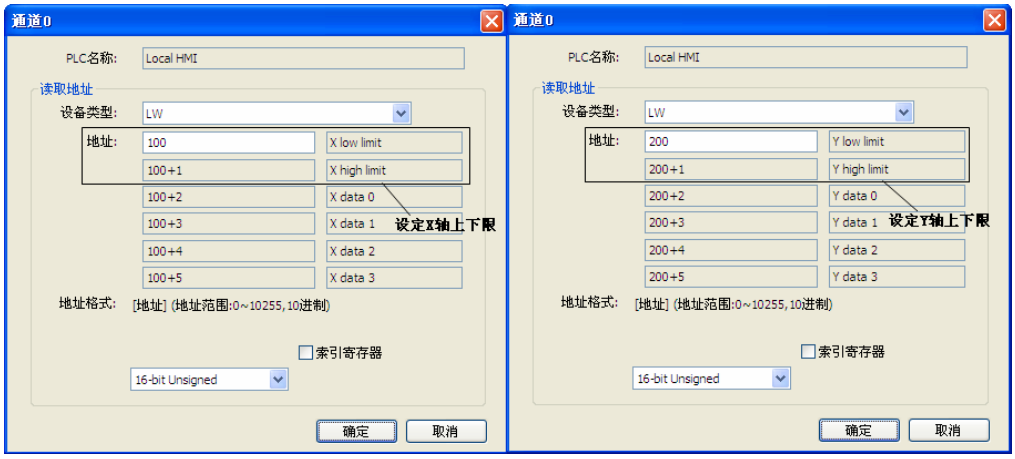
未勾选"X 轴数据与 Y 轴数据来自不同地址"，但勾选了"上下限值取自寄存器"时，按下设定键，会显示该通道数据来源的设定对话框，使用者可以设定设备类型及资料格式，对话框右半部是显示各数据地址的含义，包含 X 轴、Y 轴数据的最小值与最大值，以及 X 轴、Y 轴座标数据读取地址。如下图中右边的文字说明。



当勾选"X 轴数据与 Y 轴数据来自不同地址"，则需要各自设定 X 轴及 Y 轴的数据来源，并且

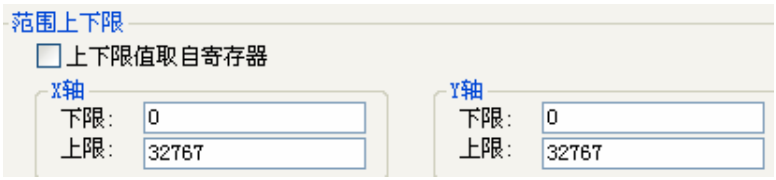
XY 的上下限则需各自分开设定。





[范围上下限]

关于设定 XY 上下限，当勾选“上下限值取自寄存器”时，X 轴和 Y 轴数据上下限值设定如上图所示。如果没有勾选的话，可自行分别设定上下限。

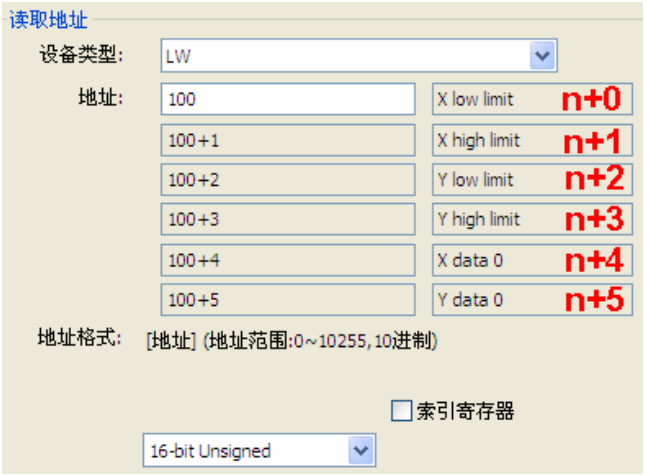


上下限是由 X 和 Y 轴的百分比来计算， 例如 X 或 Y 百分比=(X 或 Y 读取值-下限)/(上限-下限)

当未勾选“X 轴与 Y 轴数据来自不同地址”时，X 轴和 Y 轴数据分配是依据设定的 X 轴坐标地址和数据格式来分派。

举例：

1 word (16-bit signed， 16-bit unsigned):

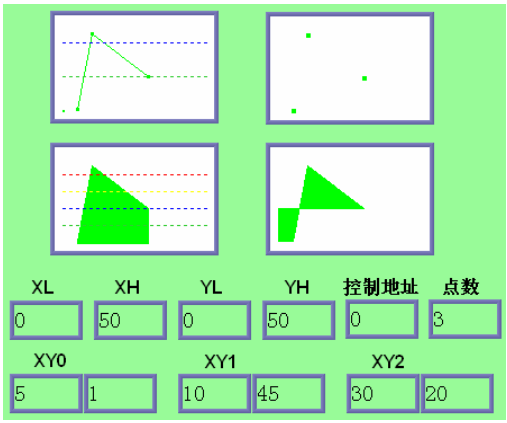


由上图可以看出，当数据格式为 1 个字，且勾选了“上下限值取自寄存器”时，则设定的地址表示 X 轴数据下限，接下来的数据依次为 X 轴数据上限、Y 轴数据下限、Y 轴数据上限、X 轴数据 0、Y 轴数据 0….，即当选择为 LW100 时，则 LW100 此时表示 X 轴最小值，LW101 表示 X 轴最大值，LW102 表示 Y 轴最小值，LW103 表示 Y 轴最大值，LW104 表示 X 轴数据 0，LW105 表示 Y 轴数据 0 等以此类推。

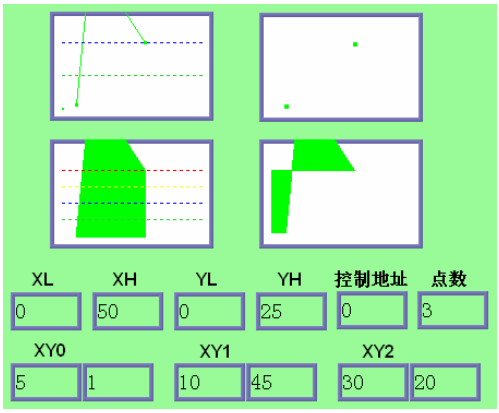
2 words (32-bit Float):

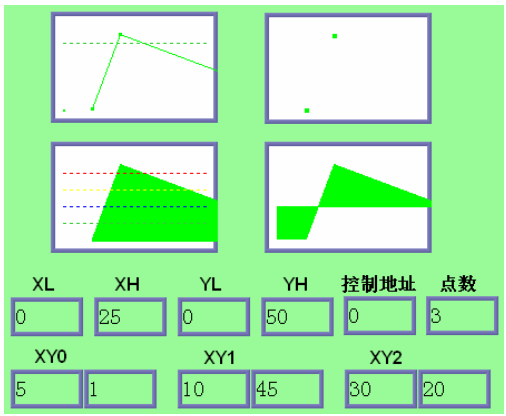
当设定数据格式为双字时，例如 32 位浮点数，则此时，LW100，LW101 表示 X 轴数据下限，LW102，LW103 表示 X 轴数据上限，LW104，LW105 表示 Y 轴数据下限，LW106，LW107 表示 Y 轴数据上限，LW108，LW109 表示 X 轴数据 0，LW110，LW111 表示 Y 轴数据 0 等依次类推。

当勾选使用“上下限值取自寄存器”时，可以依照改变 X 轴及 Y 轴的上下限来达到放大及缩小 XY 曲线的功能。如下图所示，原本显示的曲线如下，且 X 轴的最小值为 0，最大值为 50；同样的 Y 轴的最小值为 0，最大值为 50。

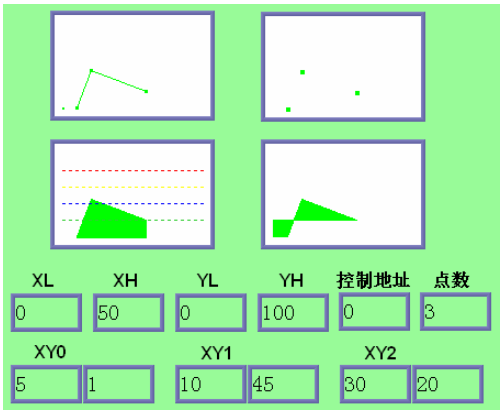


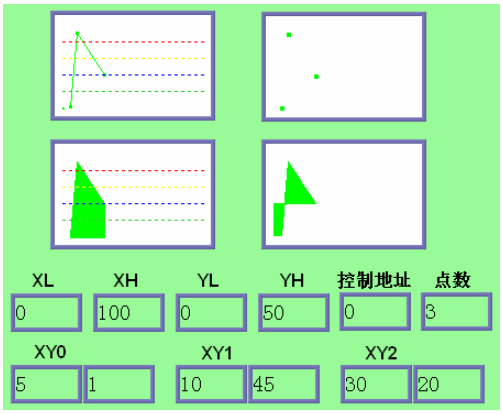
在 X 及 Y 轴设定显示范围。(XL=X 轴的下限，XH=X 轴的上限，YL=Y 轴的下限，YH=Y 轴的上限)





改变 Y 轴（X 轴）的上限可让使用者观察 Y 轴（X 轴）0~25 范围的资料，可达成放大的效果。如上图所示。将 Y 轴（X 轴）的范围改为 0~25 的范围，相当于将曲线垂直（水平）放大了 2 倍的效果。

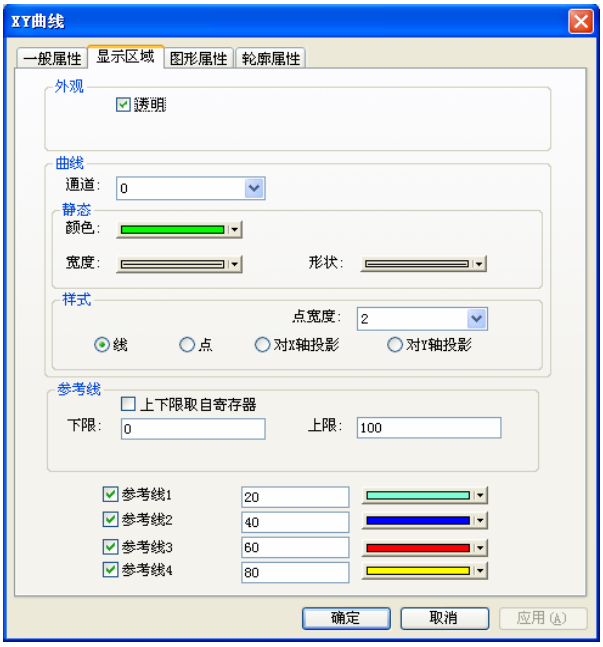




改变 Y 轴（X 轴）的上限，可达到缩小的效果。如上图所示。

[显示区域]

单击 XY 曲线对话框的“显示区域”分页，将会出现如下图。



[外观]

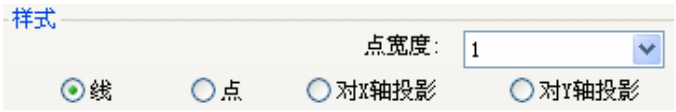
勾选外观为透明时，背景则无任何颜色；不勾选时则依照所选择的色彩来显示外框及背景。

[曲线]

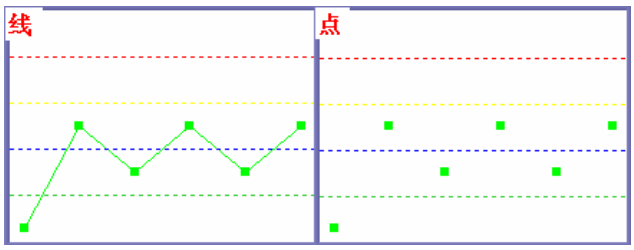
可在此设置各通道所要显示的曲线属性，线条颜色、宽度等。

[样式]

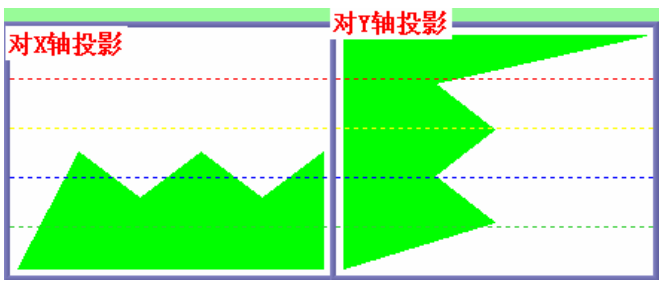
设定 XY 以线、点、对 X 轴投影或对 Y 轴投影显示。



线及点表示图如下：



X 轴投影及 Y 轴投影显示如下：



在使用 Y 轴投影的功能时，所形成的图形是以获取各点及 X 的原点加上，Y 的第一点及最后一点来绘制而成。

以下范例说明对 X 轴投影和对 Y 轴投影的过程：

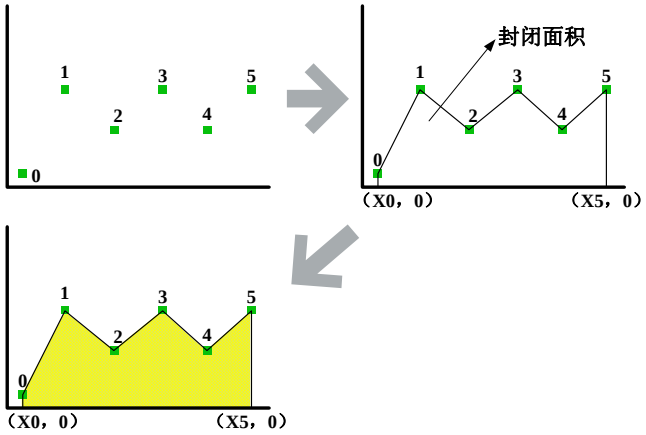
请参考以下例子 X 及 Y 轴投影说明， 假设共有 6 个点由 P0 到 P 5，在 X 轴投影的步骤如下：

a、系统自动在 X 轴计算二个投影点(X0，0) 和 (X 5，0)。

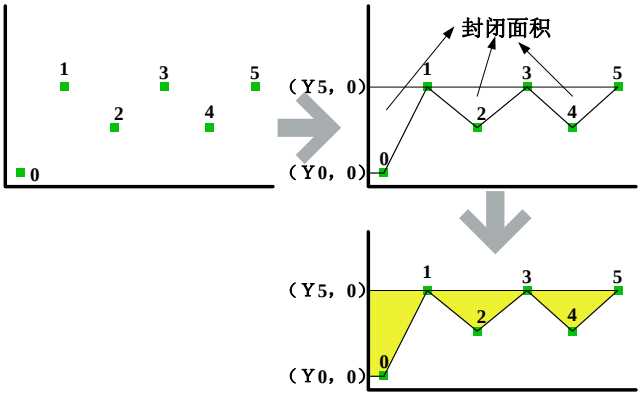
b、照顺序由(X0，0)， P0， P1… P 5， (X 5，0)和最后回到(X0，0)，将这些点连接起来。

c、填满所有封闭面积。

X 轴投影：

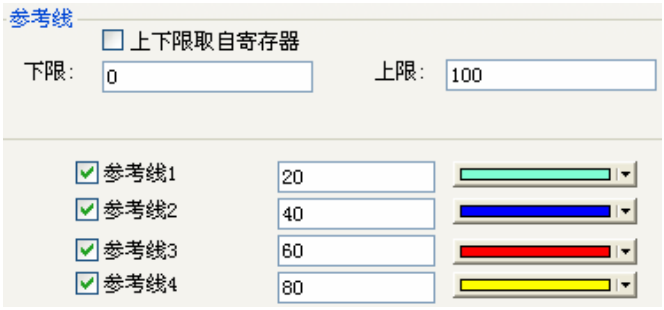


Y 轴投影亦同：

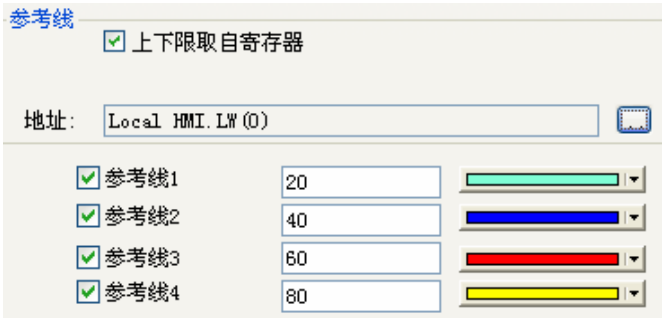


[参考线]

最多可画四条参考线在曲线图上，使用者可以自行选择线条的色彩及参考的数值，并且依据所设定数值来显示在画面上。



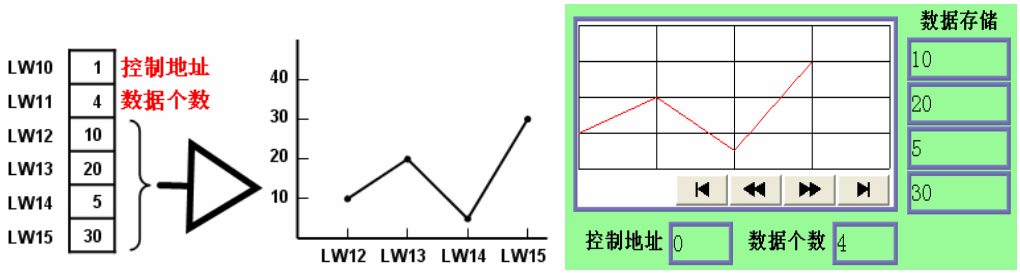
若勾选“上下限值取自寄存器”，则需设定一个参考线的读取地址。



13.17 数据群组显示(data block)

数据群组是指一组连续地址中的数据，例如 LW12、LW13、LW14、LW15 等。数据群组显示控件可同时显示多个数据群组的内容，例如同时显示 LW12~LW15 与 RW12~RW15 两个数据群组，使用者可由此方式来观察及比较各寄存器中的资料。

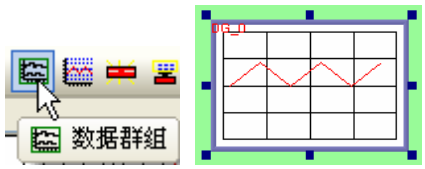
也就是说，数据群组是用来显示一组或者多组连续的寄存器的数据形成的曲线。利用这个特性，在实际应用中可以实际比较一下预想的数据和实际得到的数据，使用两条曲线来对比即可。例如，在锅炉温度控制中，事先设置一组预期的数据，在实际锅炉运行时，获得的数据跟这个预期的数据来比较，使用两条群组显示的曲线来比较，即可了解到实际的差异在哪里，以便更好的控制锅炉的运行。下图为使用数据群组显示控件显示单一数据群组 LW12~LW15 中的数据。下面说明数据群组的具体使用。



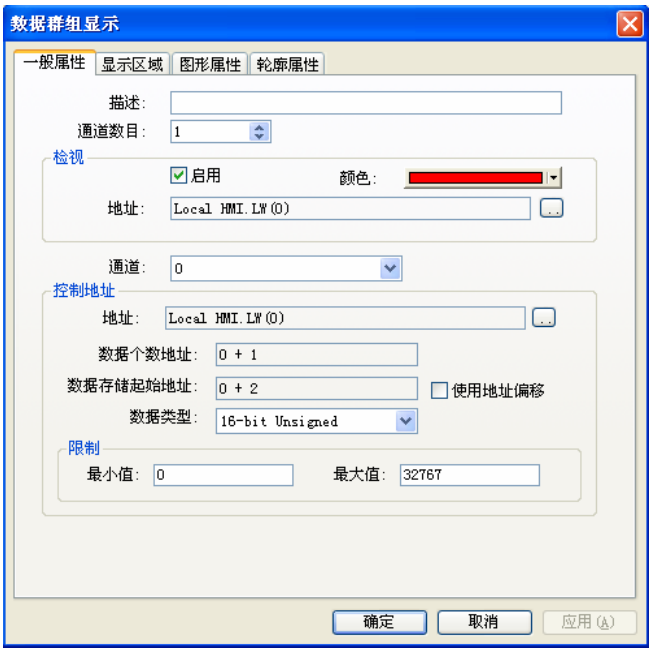
实际执行结果

13.17.1 新增控件及属性设置

打开 InoTouch Editor 软件菜单的“控件/数据群组”，或者工具栏上的图标，在窗口中点击鼠标左键，就建立了“数据群组”控件，如下图所示。

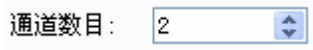


选择“数据群组”双击或单击鼠标右键选择“属性”进行编辑，如下图：



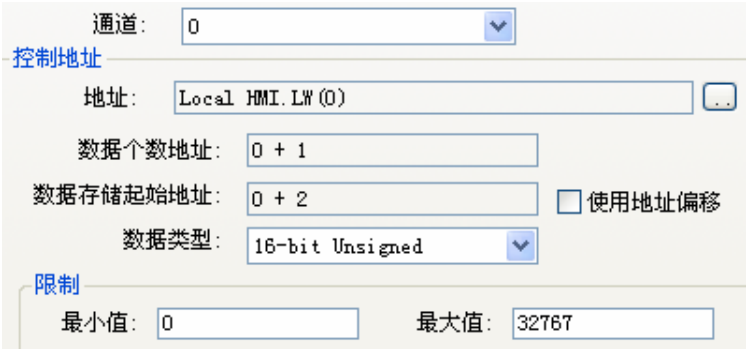
一般属性

设定[通道数目]。通道数目用来设定使用者欲观察数据群组的组数，也即显示多少条曲线，最多可同时显示 12 条曲线。



例如上图显示数据群组被设定为 2，则使用者可以同时观察两个不同地址类型中的内容。

分别设定各数据群组的读取地址、格式及图形之高、低限。



[通道]

用来指定要设定哪一个通道的控制地址等。

[控制地址]

“控制地址”被用来控制曲线的显示及清除；

0 = 无动作 (系统默认值)

1 = 描绘曲线

2 = 清除曲线

3 = 清除并重绘曲线

在执行完以上指令后，系统会将控制地址的数值设为 0。

当完成”控制地址”的设定时，InoTouch Editor 将自动计算产生“数据个数地址”与“数据储存起始地址”，地址差距皆为 1。当不使用“地址偏移”功能时，假使“控制地址”被设置为 LW10，则“数据个数地址” 与 “数据储存起始地址”将分别为 LW(10+1)与 LW(10+2)，也就是 LW11 与 LW12。

[数据个数地址]

数据个数地址来源是由“控制地址的地址+1”

设定所指定数据群组要显示的数据个数，也即多少个点。

[数据储存起始地址]

若“使用地址偏移”没有被勾选，则数据储存起始地址的来源是由“控制地址的地址+2”

实际的数据读取起始地址。曲线就是由该地址开始的连续的多个寄存器中的数据所形成。

[使用地址偏移]

若勾选“使用地址偏移”，则[数据储存起始地址]是[使用地址偏移]+[控制地址]的地址，即 [控制地址]+2。

[数据类型]

设定数据格式。例如 LW12 为数据资料的起始地址，当数据格式设定为 16 bit Unsigned 时，LW12 为 Data 1、LW13 为 Data 2，依此类推。

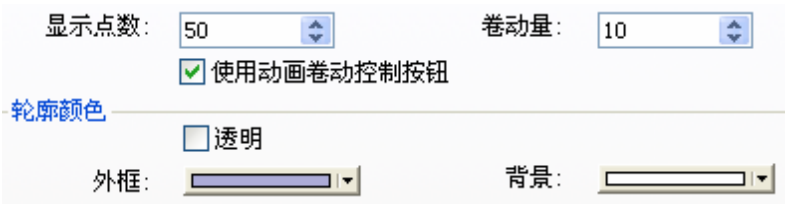
但当数据格式设定为 32 bit Unsigned 时，LW12 为 Data 1、LW14 为 Data 2，依此类推。

[限制]

用来设定所显示图形之高、低限。即曲线能够显示的最小值与最大值。

显示区域

设定图形一页所能显示最大数据个数（点数）、左右移动点数及框线、背景颜色。

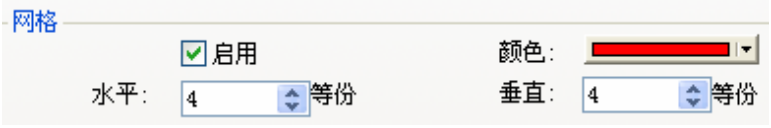


动画卷动控制按钮功能与“趋势图”按钮功能一致，请参考“趋势图”的相关说明。

设定群组显示控件的轮廓的外框颜色和背景颜色。若勾选“透明”，则外框颜色和背景颜色无需设定，且不使用“图形”。

设定群组显示控件网格

即是将群组显示控件水平方向和垂直方向分别等分为等份的网格以及颜色，如下图所示。如果不需要网格效果，则不勾选“启用”。



设定各数据群组图形的线条颜色、类型及宽度。

设定好以上各项属性后，就可以根据设定，将读取过来的数据以曲线的方式显示在画面上。

13.17.2 数据群组控件的工作过程

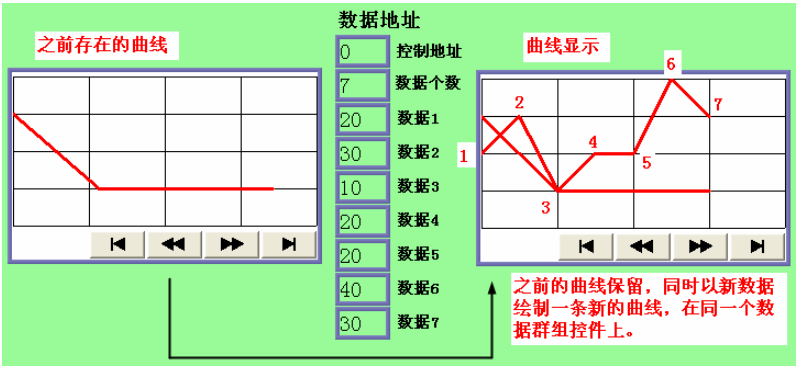
1、如何显示数据群组的内容

a、在[数据个数地址]写入欲显示的数据笔数。

b、在[数据储存起始地址]依序填入数据内容。

c、在[控制地址]写入“1” (将此地址的 bit 0 设定为 ON)；此时 InoTouch Editor 将以折线图画出目前寄存器的内容 (并保留之前图形)。

d、InoTouch Editor 在完成前项动作后将对[控制地址]写入“0”。



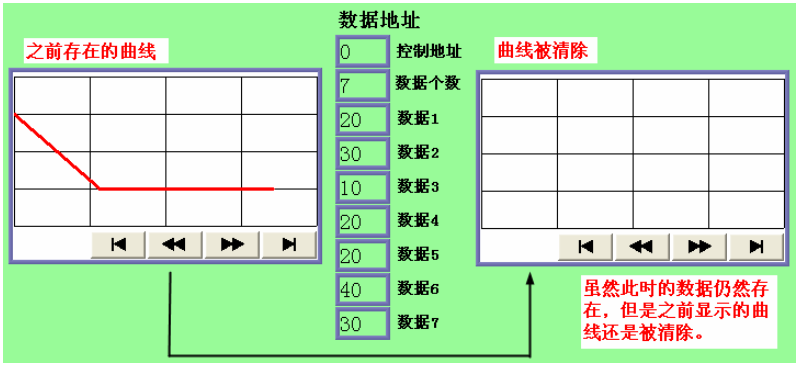
注意：在上述动作 c 和 d 之间，请勿更改[控制地址]、[数据个数地址]及[数据储存起始地址]

内容，否则可能产生非预期结果。

2、如何清除已显示的图形

a、在[控制地址]写入“2” (或将此地址的 bit 1 设定为 ON)；将清除先前所画之线图。

b、InoTouch Editor 在完成前项动作后将于[控制地址]写入“0”。



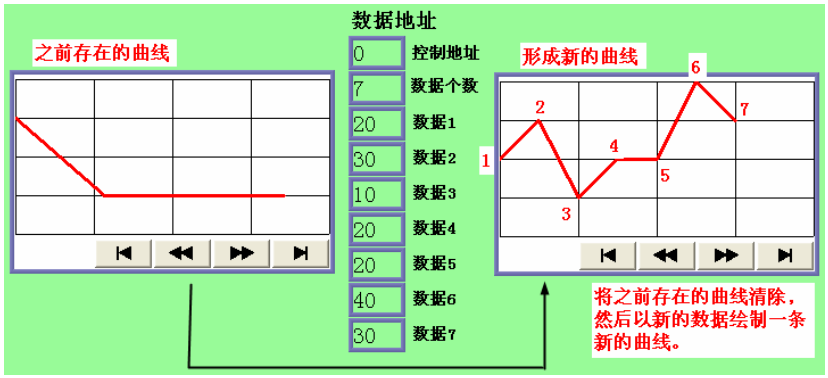
3、清除已显示的图形并显示以新数据形成的曲线

a、在[数据个数地址]写入欲显示的数据笔数。

b、在[数据储存起始地址]依序填入数据内容。

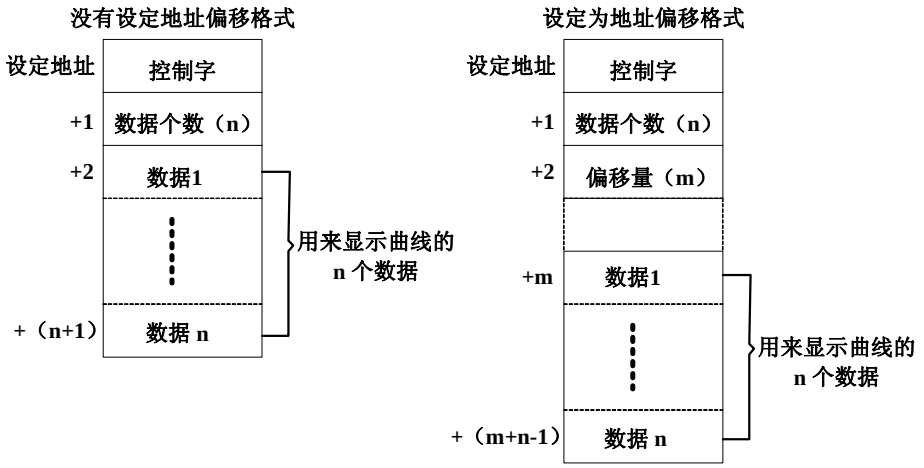
c、在[控制地址]写入“3” (将此地址的 bit 0 与 bit 1 皆设定为 ON)；此时 InoTouch Editor 会先将先前的曲线清除，再以目前地址里面新的数据形成一条新的曲线。

d、InoTouch Editor 在完成前项动作后将于[控制地址]写入“0”。



4、地址偏移模式

若勾选[地址偏移]模式，则原本[数据储存起始地址]将变成[数据储存偏移地址]，请参考下图。



左图为未勾选[地址偏移]模式，在此种模式下[数据储存起始地址]即为[控制地址(Designated address)] + 2。

但在地址偏移模式下，原来的[数据储存起始地址]变为[数据储存偏移地址]，用来存放数据储存的地址偏移值，假设其值为 m，则可推得[数据储存起始地址]为[控制地址] + m。

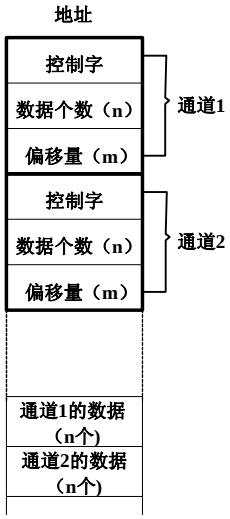
注：[控制地址]、[数据个数地址]及[数据储存偏移地址]固定为 16 位无符号数据格式，在控件属性对话框所选择的数据类型是针对需要显示曲线的数据类型。

- 当指定的寄存器地址为数据 32 位时，只有较低的 16 位产生作用，请将较高的 16 位内容设为 0。

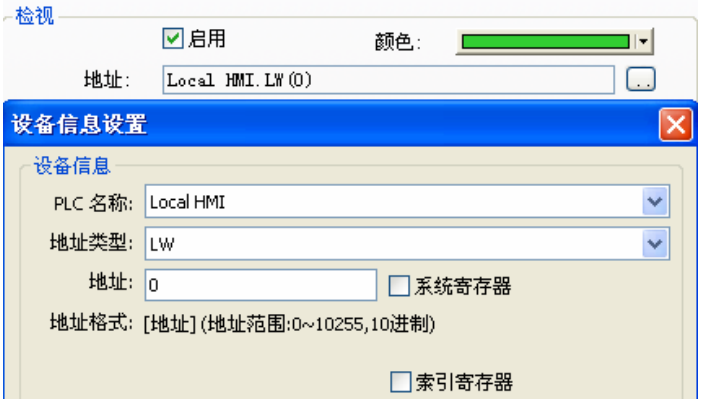
32 bit数据格式

	31 16	15 0
+0	0	控制字
+1	0	数据个数
+2	0	偏移量

- 控件在建立后将持续地读取[控制地址]、[数据个数地址]及[数据储存偏移地址]内容，但只有在[控制地址]的 bit 0 为 ON 时才去读取偏移量地址里面的数据。
- 当指定了两个以上的通道、且各通道使用同类型寄存器时，建议使用地址偏移模式。请参考下图：将两个通道的[控制地址]、[数据个数地址]及[数据储存偏移地址]设定为连续的地址，系统即可在一次通讯周期中将其全部读回，可有效提升画曲线效率。



5、数据检视功能(Watch)

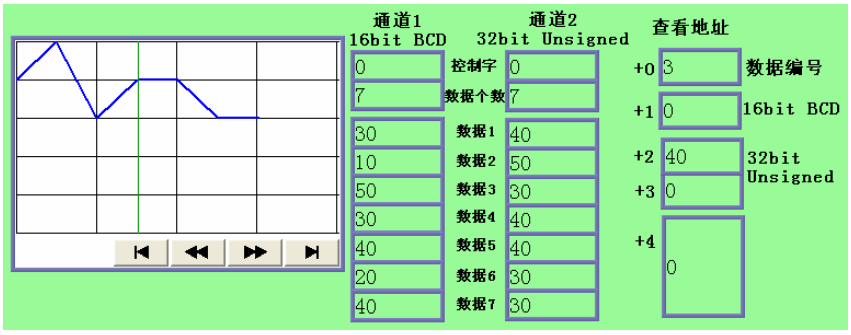


数据检视功能，也即数据查看功能。工作方式与前面讲的“趋势图”中的“查看”功能是一样的。使用者除了可以利用图形比较各数据群组外，亦可使用[数据检视]功能，查看各描绘点的数据。开启此功能时，使用者只需触控画面上生成的曲线，InoTouch Editor 将依序将目前所查看的“数据编号”与各通道的数据依序写入指定的地址中，再由数值显示等控件读出实际内容。所写入各通道数据的数据格式则依照原来各通道定义的数据格式。

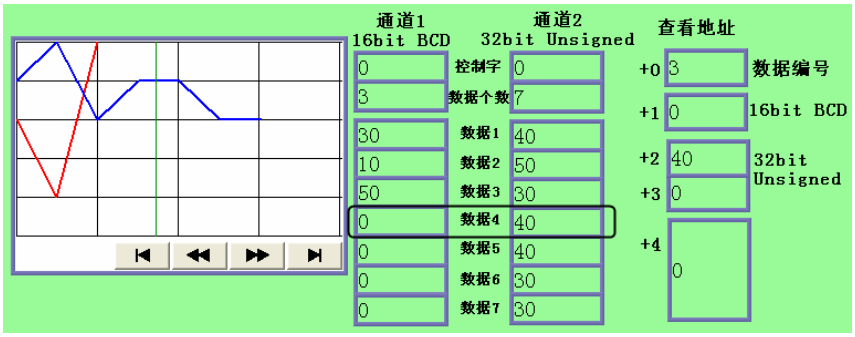
举例说明：以两组数据群组显示为例，通道 1(数据群组 1)为 16 bit BCD 格式，通道 2(数据群组 2)为 32 bit Unsigned 格式；当查看数据 4 时，控件会依序将数据编号(数据编号从 0 开始，也就是检视数据 4 时，数据编号的值将为 3)及两组数据群组的数据 4 内容送至指定地址中，其中所写入的通道 1 数据使用 16 bit BCD；所写入的通道 2 数据使用 32 bit Unsigned。

注：当使用 32 bit Unsigned 时，控制地址和数据个数的设置还是 16 bit Unsigned，写入的数据地址是 32 bit Unsigned（控制地址+2，+4，+6 等，依次类推）。

- [数据编号]：为从 0 开始的 16 位无符号整数；当指定的寄存器为 32 位时，只有较低的 16 位产生作用。
- 一个通道可由将[控制地址]设定为 1 来显示出不同时间点的数据内容，但检视时所输出的内容为各通道最后一次显示时的值，先前显示时的值无法被检视(查看)。
- 如图，若通道 1 在检视前被清除(或尚未显示)，其查看的数据以 0 代替。



- 如图，若通道 1 仅有 3 个数据，当检视数据 4 时(数据个数不足)，其查看的数据以 0 代替。



[限制]

1）通道数量上限为 12，即一个数据群组控件最多可以显示 12 条曲线。

2）曲线刷新上限个数为 32；当到达上限后，则不再接受显示命令。

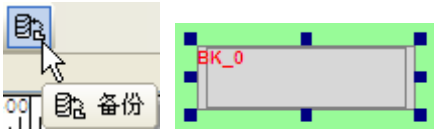
3）每一个通道最多可以显示 1024 点。

13.18 备份控件 (backup)

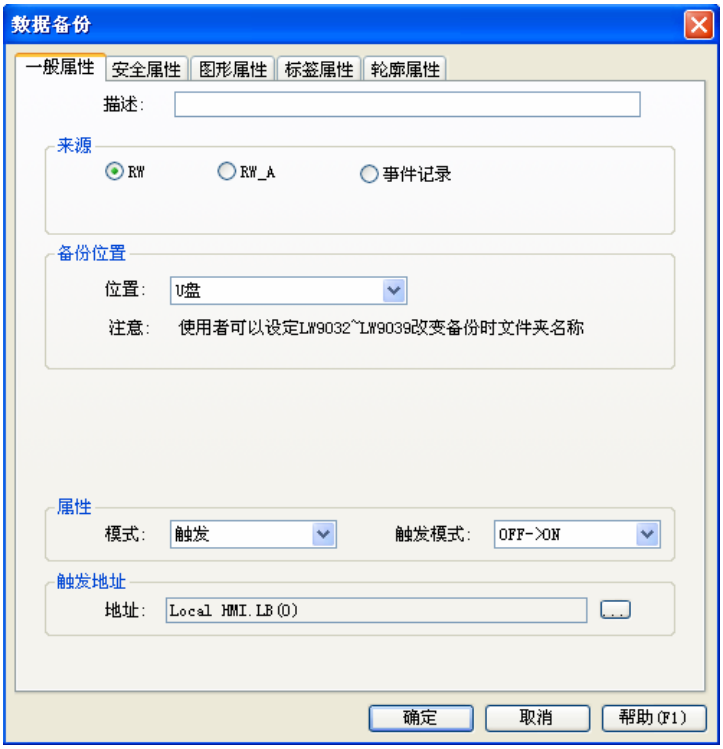
利用备份控件可以将配方资料、事件记录或指定的资料取样纪录复制到指定的装置 (U 盘)，并可以指定备份的时间范围。例如事件记录原来储存在 U 盘，这些数据即可移置到计算机做进一步的分析或者打印等。同时，还可以将保存在人机界面或者 U 盘、SD 卡等里面的资料备份到该计算机上保存起来。

当备份动作进行中时，系统保留位[LB9039]的状态将为 ON 的状态。下面说明“备份”控件的使用。

打开 InoTouch Editor 软件菜单的“控件/备份”，或者工具栏上的图标，在窗口中点击鼠标左键，就建立了“备份”控件。



选择“备份”双击或单击鼠标右键选择“属性”进行编辑，如下图：



[RW]、[RW_A]、[事件记录]

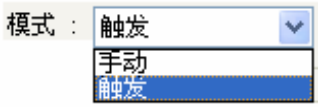
这些选项用来选择要复制的资料来源。

[备份位置]

档案复制位置可以选择[U 盘] ，当资料来源与复制位置相同时，控件将不执行任何复制的动作。

[属性]

选择控件的执行方式，可以选择“手动”与“触发”两种模式，参考下图

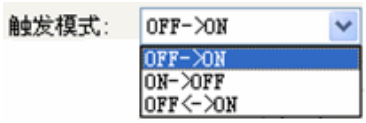


[手动]

使用者只需按压该控件，即可执行资料复制动作。

[触发]

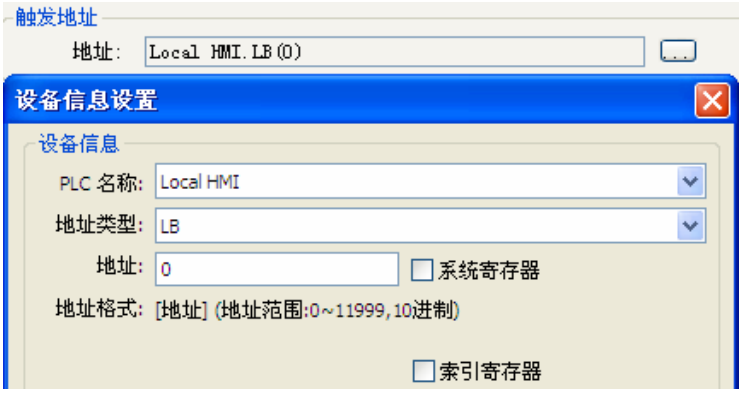
当指定的寄存器状态改变符合触发条件时，控件将执行资料复制动作。触发条件包含下列几种方式：



[OFF->ON]	当指定寄存器的状态由 OFF 变为 ON，将执行资料复制动作。
[ON->OFF]	当指定寄存器的状态由 ON 变为 OFF，将执行资料复制动作。
[OFF <-> ON]	当指定寄存器的状态改变，将执行资料复制动作。

[触发地址]

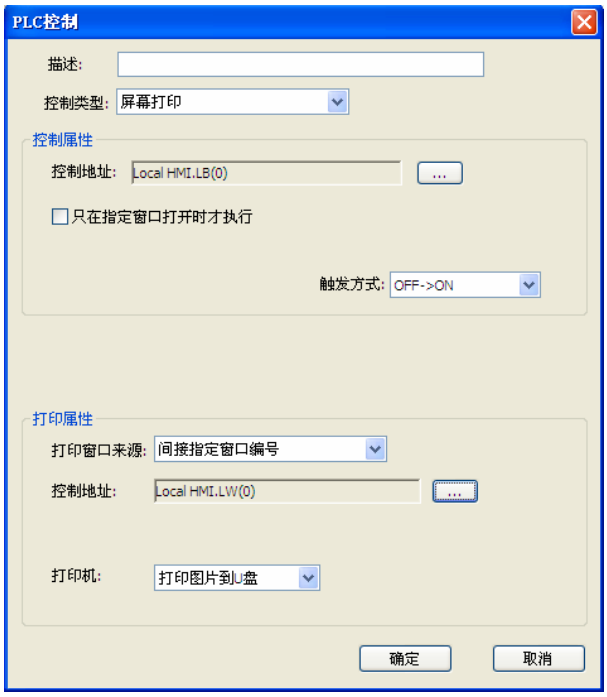
当使用“触发”模式时，触发地址被用来指定控件将使用哪一个寄存器来触发资料复制动作。



13.19 PLC 控制控件(PLC Control)

“PLC 控制”控件是用来定义 PLC 中的某个位的状态或者寄存器的数据内容等条件，当条件满足时，执行在该控件里面定义的特定动作。例如切换画面等动作。

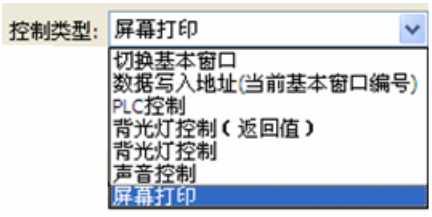
单击 InoTouch Editor 软件左侧项目管理下“PLC 控制对象表”，就会打开“PLC 控制”对话框，如下图所示。



属性

[控制类型]

选择 PLC 控制控件的控制类型，可选择项目如下图。



对以上功能设定，分别说明如下。

a、“切换基本窗口”

切换基本窗口的功能。当[触发地址]中的数据改变，且改变后的数据为一个有效的窗口编号时，将关闭目前的窗口并切换至[触发地址]中数据所指定的窗口，并将此时切换后的窗口编号写至指定的地址中(此写入地址请参考下文的说明)。例如假使目前的窗口编号为 10，且控件的设定内容如下图：

当 LW0 中的数据由其它数据改变为 11 时，InoTouch Editor 除了会将基本窗口切换到窗口 11 之外，也会将 LW1 中的数据更改为 11。

当切换窗口成功时，此切换后的窗口编号的写入地址与[触发地址]中设定的读取地址、变量型态皆有关系，下表整理了欲切换的目的窗口编号读取地址，与切换后的窗口编号的写入地址。其中“address”表示寄存器的地址值，例如寄存器为[LW100]时，“address”等于 100。

变量型态	目的窗口编号读取地址(触发地址)	切换后窗口编号的写入地址
16-bit BCD	address	address + 1
32-bit BCD	address	address + 2
16-bit Unsigned	address	address + 1
16-bit Signed	address	address + 1
32-bit Unsigned	address	address + 2
32-bit Signed	address	address + 2

但当系统保留位[LB9017]的状态被设定为 ON 时，切换后的窗口编号将不再写至特定的地址中。

若选择[换页后地址数据归零]，则在切换窗口成功后会将触发地址中的数据归零。

当背光灯为关闭状态时，若选择[换页后背光灯打开]，则在切换窗口成功后会自动开启背光灯。在将“背光节能时间”设定为非 0 的数据，也即使用了背光关闭的功能时，建议勾选此项。

b、“数据写入地址 (当前基本窗口编号)”

当切换基本窗口时，会将基本窗口的编号写至[触发地址]指定的地址中。

c、“PLC 控制”

此项功能提供使用者可以利用寄存器中的数据控制 PLC 与 HMI 之间的数据传输，数据传输方向包含四种类型，参考下表的内容：

数据传输类型	数据传输方向
1	PLC 寄存器中的数据->HMI 上的 RW 寄存器
2	PLC 寄存器中的数据->HMI 上的 LW 寄存器
3	HMI 上的 RW 配方资料->PLC 上的寄存器
4	HMI 上的 LW 寄存器->PLC 上的寄存器

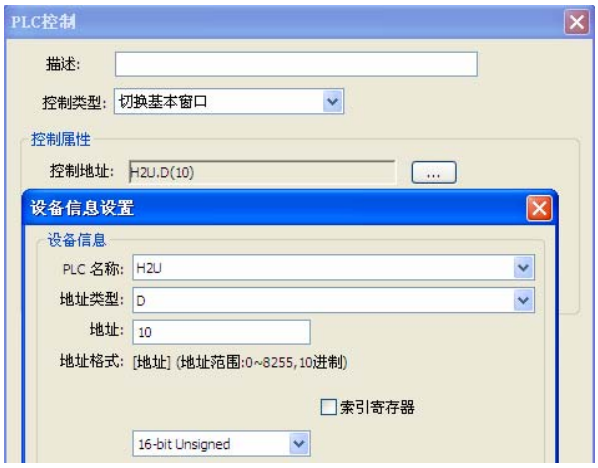
使用此项功能时，InoTouch Editor 将利用[触发地址]中所设定的地址开始，以连续四个寄存器中的数据，决定数据传输类型、数据传送个数、数据来源地址与数据传送目的地址等。下表表示各寄存器中数据所表示的意义。其中[触发地址]用来指示 PLC 寄存器的位置，例如[触发地址] = D100，即表示使用 D100~ D103 共四个寄存器中的数据来决定数据传输的内容。

地　址	用　途	说　明
[触发地址]	存放数据传输类型，并决定数据传输的方向。	这个寄存器被用来决定数据传输类型，如上所述，共有四种类型。 当寄存器被写入新的数据时，InoTouch Editor 即执行相应的传输，传输完成后会将寄存器中的数据复归为 0。
[触发地址] + 1	存放欲传输数据的个数，单位为 word。	
[触发地址] + 2	存放传输过程中数据来源的地址偏移量。	传输的数据来源的起始地址为: [触发地址] + 4 + 地址偏移量 以汇川 PLC 为例，如果此时设定的[触发地址]为 D100，而在寄存器[触发地址] + 2 也就是 D102 中的资料为“5”，则传输的数据来源的起始地址为 D109， 其中 109 = (100＋4)＋5。
[触发地址] + 3	存放传输过程中配方资料寄存器(RW)或者本地数据寄存器(LW)的起始地址。	以汇川 PLC 为例，如果此时设定的[触发地址]为 D100，而在寄存器[触发地址] + 3 也就是 D103 中的资料为“100”，则传输过程中操作的目的地址是以 RW100 或 LW100 为起始地址的连续寄存器。

举例说明如下：

假使现在需要使用“PLC 控制”的功能，将汇川 PLC 中从 D100 起始的 16 words 的数据，传输到 InoTouch Editor 配方内存 RW200 开始的地址中，实现的方法如下：

首先，假设用 D10 起始的四个数据寄存器来控制传输，则应该先建立一个 PLC 控制控件，选择类型为“PLC 控制”，读取地址设定为 D10。如下图所示。



接下来，应该确定操作数据的大小和地址的偏移量，将 D11 设定为 16，表示传输数据的大小为 16 words；将 D12 设定为 86，表示数据的来源地址为 D100 (100 = 10+4+86)；将 D13 设定值为 200，表示目标地址为 RW200。

最后，依照数据传输的方向，设定传输类型。应该将 D10 设定为 1，表示将传输 PLC 寄存器中的数据到 InoTouch Editor 上的 RW 寄存器中。如果设定 D10 值为 3，则传输方向相反。

其它两种传输方式具有类似的设定方法，差别只在 InoTouch Editor 的数据存储器变成了本地资料寄存器 LW。

d “背光灯控制 (返回值)”

当[触发地址]的状态由 OFF 变为 ON 时，InoTouch Editor 将关闭背光灯，此时也会将[触发地址]的状态设定为 OFF。背光灯关闭时，使用者只需触控屏幕的任意位置，背光灯即会再度打开。

e “背光灯控制”

当[触发地址]的状态由 OFF 变为 ON 时，InoTouch Editor 将关闭背光灯，但因不具备“返回值”(write back)功能，此时并不会将[触发地址]的状态设定为 OFF。

f “声音控制”

当[触发地址]的状态改变符合触发条件时，“PLC 控制” 控件将播放预先指定的声音文件。[触发方式]可以选择：

OFF->ON	状态由 OFF 变为 ON
ON->OFF	状态由 ON 变为 OFF
OFF<->ON	只需状态改变

注意：目前只有带以太网的人机界面才有音频输出口。

g “屏幕打印”

当[触发地址]的状态改变符合触发条件时，“PLC 控制”控件将打印指定的画面。触发方式可以选择：

OFF->ON	状态由 OFF 变为 ON
ON->OFF	状态由 ON 变为 OFF
OFF<->ON	只需状态改变

可以选择要打印的画面，共有三种指定方式，参考下图。

[打印当前基本窗口]

“PLC 控制”控件将打印目前人机界面显示的窗口画面。

注意：当前只能支持保存图片到 U 盘的方式，暂未支持打印机直接打印

13.20 排程 (Schedule)

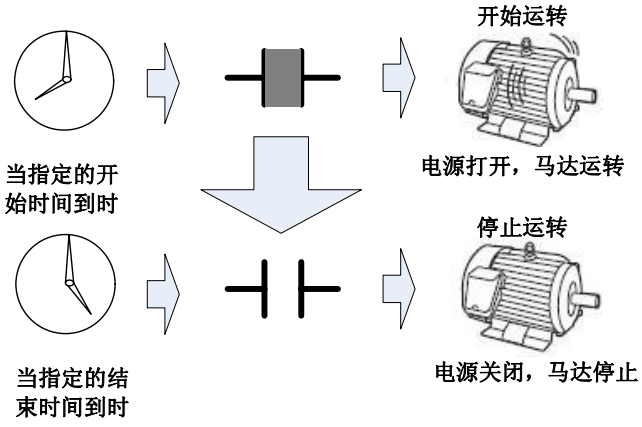
排程的功能是在指定的时间到达时，执行指定的动作。例如，指定星期一的早上 8：00 整开机，星期一的下午 17：00 整关机等。这个就是一个简单的”排程”功能。排程的功能是在特定的时间超过指定的 bit 或者 word 寄存器。可以让指定的 bit 状态改变或者数据寄存器里面写入指定的数据。排程一般包含以下各项内容，下面以案例的方式来说明”排程”的使用方法。

1）设定选单（或者说是某一个工作要求）；2）在特定时间打开/关闭马达；

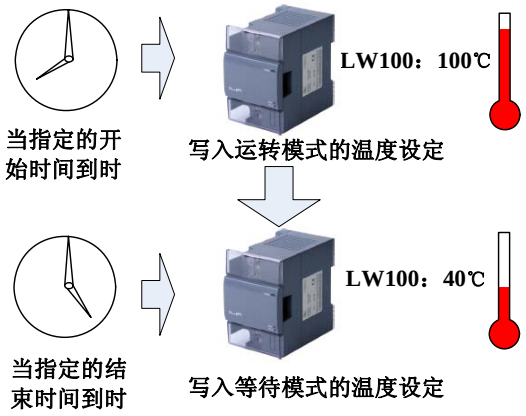
3）在特定时间改变温度；4）时间设定与排程动作过程说明；5）限制

13.20.1 设定选单

在指定时间打开/关闭马达

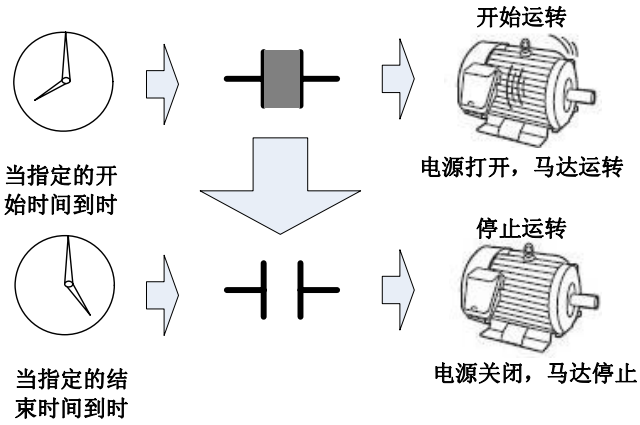


在指定时间改变温度



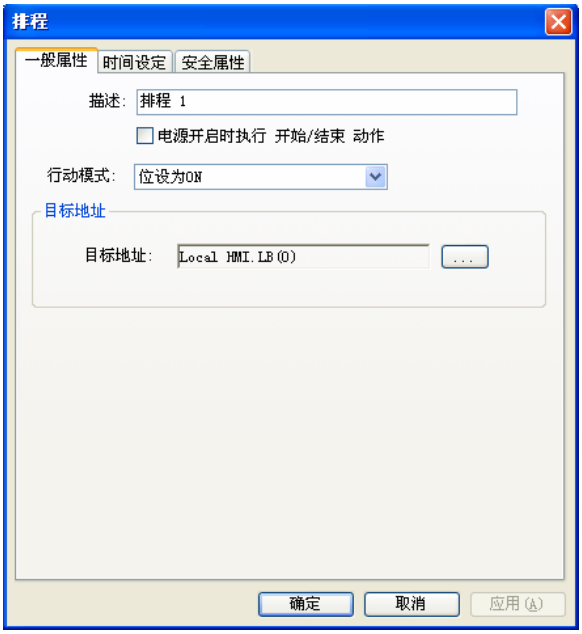
13.20.2 在特定时间打开/关闭马达

马达(假设地址：LB100)从星期一一直运转到星期五，时间由每天上午 8 点到下午 5 点。在此介绍的设定程序为在起始时间(早上 8 点)将地址 LB100 设为 ON，在结束时间(下午 5 点) 将地址 LB100 设为 OFF。



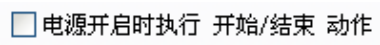
了解了上述的工作要求，那么就使用“排程”控件来设定这个工作过程。

步骤一：单击 InoTouch Editor 软件左侧项目管理下“排程对象表”，就会打开“排程”对话框，如下图所示。

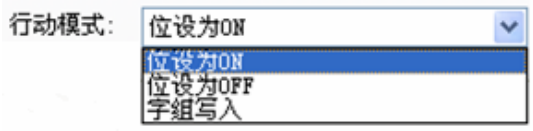


步骤二：设定[一般属性]页：

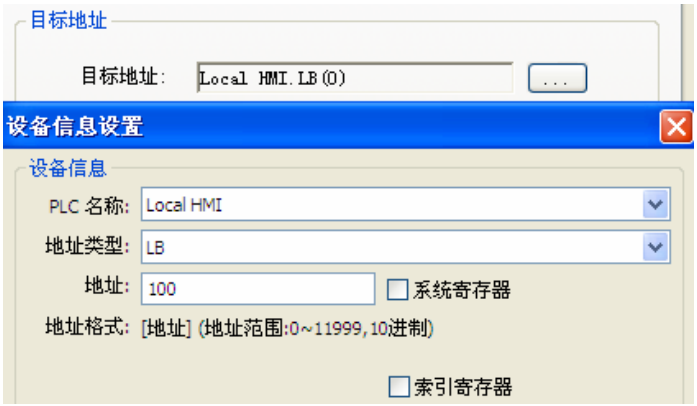
1）决定是否勾选[电源开启时执行开始/结束动作]。



2）选定[行动模式]为[位设为 ON]。

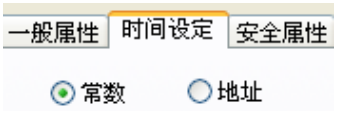


3）设定[目标地址](例如：LB100)。



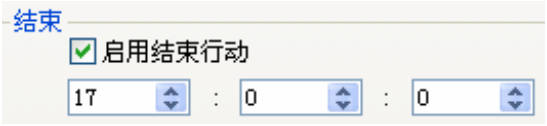
步骤三：设定[时间设定]页：

1）选择[时间设定]页签，接着选择[常数]。



2）设定[起始时间及日期]。将时间设为 8 点 0 分 0 秒，接着勾选星期一到星期五，取消[设定为单一日期]勾选盒。

3）设定[结束时间及日期]。勾选[启用结束行动]勾选盒，将结束时间设定为 17 点 0 分 0 秒。

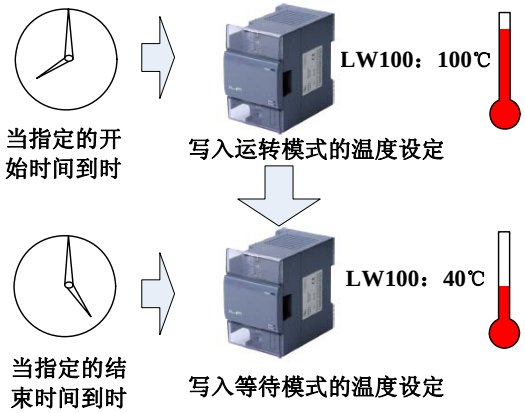


4）按下[确定]键。

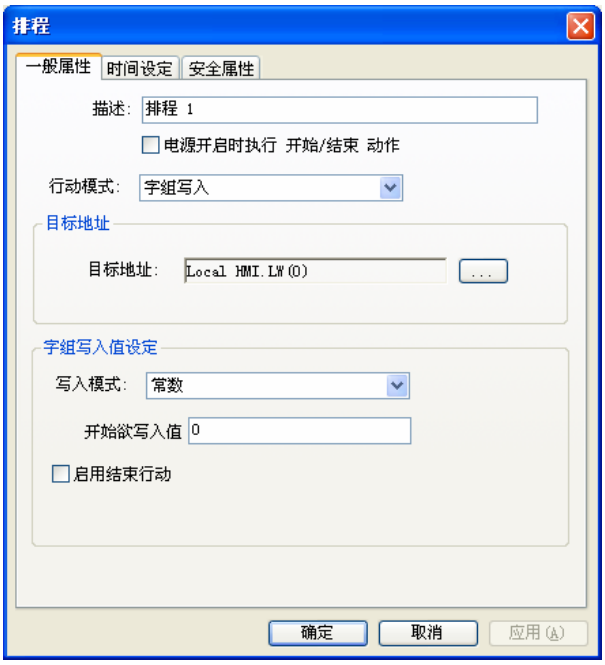
通过以上步骤，就假定 LB100 为控制马达启动/停止的控制位，经过上述设定后，“排程”的工作过程就是，每个星期一到星期五的早上 8：00 整，LB100 状态被设置为 ON，马达开始运转，到了下午的 17：00 整，LB100 状态被设定为 OFF，马达停止运转，如此每周循环。

13.20.3 特定时间改变温度

从星期一到星期五，在起始时间 8 点把温度设定值 100 度写入寄存器地址 LW100，此时系统进入运转模式。在结束时间 17 点把温度设定值 40 度写入寄存器地址 LW100，此时系统进入等待模式。

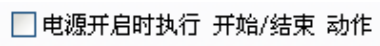


步骤一：单击 InoTouch Editor 软件左侧项目管理下“排程”，就会打开“排程”对话框，如下图所示。



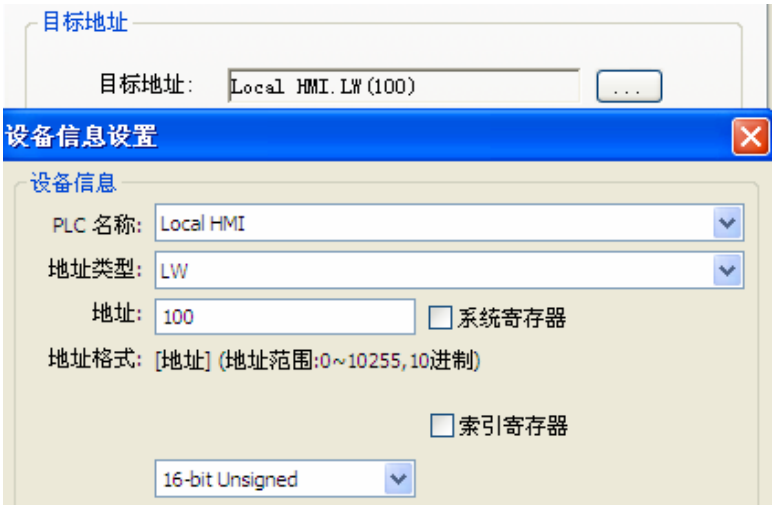
步骤二：设定[一般属性]页：

1）决定是否勾选[电源开启时执行开始/结束动作]。



2）选定[行动模式]为[字组写入]。

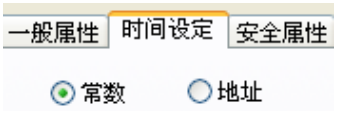
3）设定[行动地址](例如：LW100)。



4）选择[常数]，设定[开始欲写入数值]为 100。

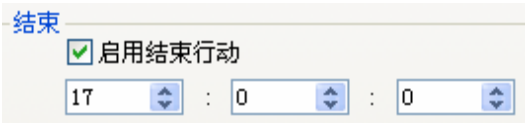
步骤三：设定[时间设定]页：

1）选择[时间设定]页，接着选择[常数]。

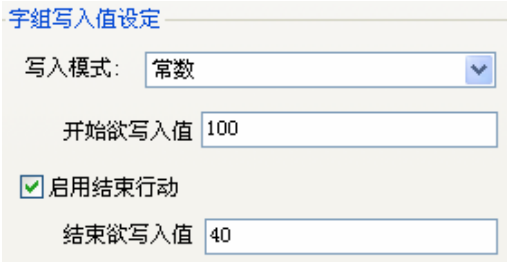


2）设定[起始时间及日期]。将时间设为 8 点 0 分 0 秒，接着勾选星期一到星期五。取消[设定为单一日期]勾选盒。

3）设定[结束时间及日期]。勾选[启用结束行动]勾选盒，将结束时间设定为 17 点 0 分 0 秒。



4）回到[一般属性]页，设定[结束欲写入数值]为 40。

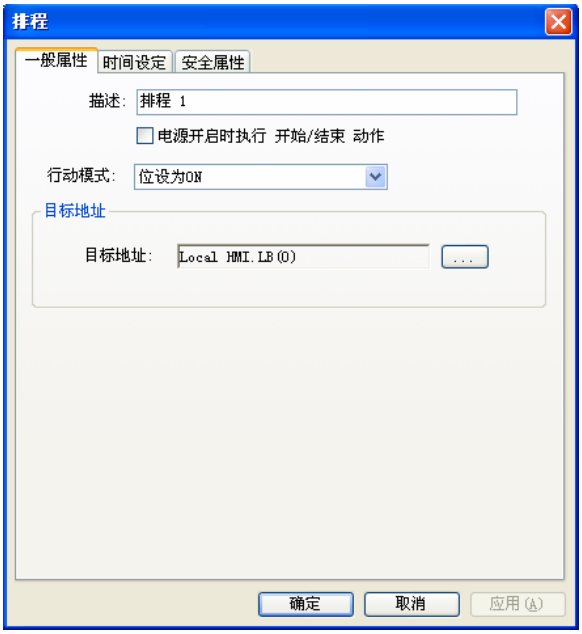


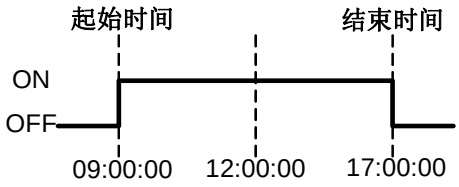
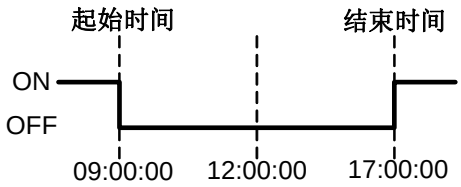
5）按下[确定]键。

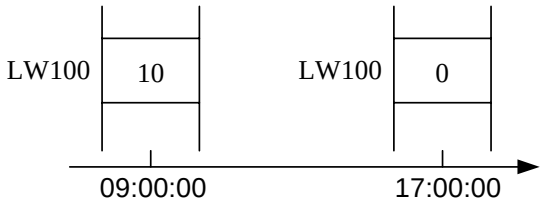
通过以上设定，就完成了一个“排程”功能，即每个星期一到星期五的早上 8：00 整，将 100 这个数据写入到 LW100 这个寄存器中，然后在下午 17：00 整，又将 40 这个数据写入到 LW100 这个寄存器中。

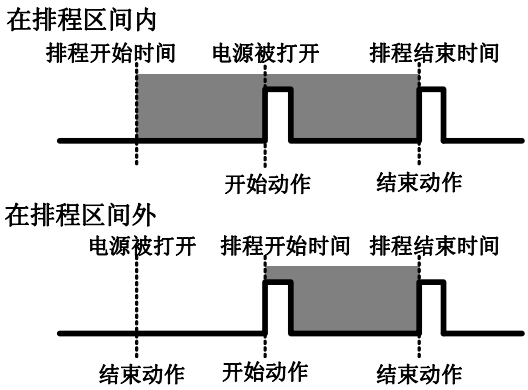
13.20.4 时间设定与排程动作过程说明

1、行动模式设定：



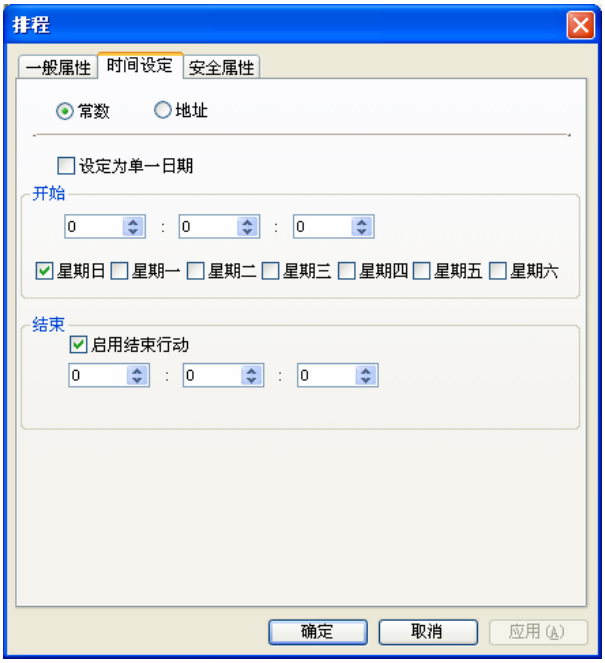
设定	动作描述
位为 ON	在起始时，把指定位设为 ON。在结束时，设为 OFF。 例如：起始时间：09:00:00 结束时间：17:00:00 
位为 OFF	在起始时，把指定位设为 OFF。在结束时，设为 ON。 例如：起始时间：09:00:00 结束时间：17:00:00 
字组写入	在起始时，把指定值[开始欲写入数值]写入指定数据字寄存器地址，在结束时，写入[结束欲写入数值]。 例如：寄存器地址：LW100 起始时间：09:00:00　　结束时间：17:00:00 开始欲写入数值：10　　结束欲写入数值：0

设定	动作描述
	
行动地址	用来控制排程的指定地址

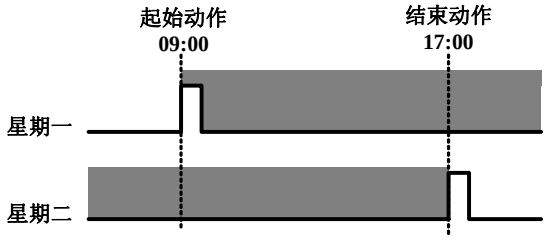
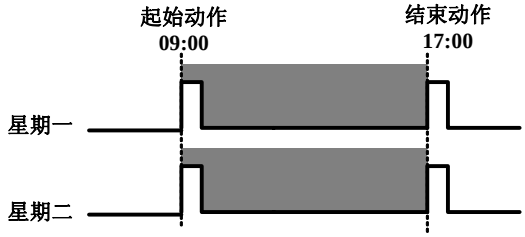
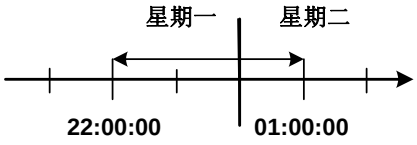
设　定	动作描述
电源开启时执行开始/结束动作	当电源打开时，执行已设定的行动。 ● 启用时 假如 HMI 的电源在排程区间内被打开，则开始行动会被执行。假如 HMI 的电源在排程区间外被打开，则结束行动会被执行。  ● 停用时 假如电源打开时晚于排程开始时间，则开始动作不会自动执行。然而结束动作会自动执行。 当然，假如结束动作未被设定，将无法正确地判定排程区间，因此动作将不会被执行。
位设为 ON/位设为 OFF/字组写入	● 位设为 ON 将[0]写入指定位。 ● 位设为 OFF 将[1]写入指定位。 ● 字组写入 将指定数值写入指定的数据寄存器地址中。
起始欲写入数值	当指定窗口被打开时，执行排程动作。 ● 选[常数]时

	排程起始时，欲写入的数值。 ● 选[地址]时 排程起始时，欲写入数值的地址。
结束欲写入数值	当指定窗口被打开时，执行排程动作。 ● 选[常数]时 排程结束时，欲写入的数值。 ● 选[地址]时 排程结束时，欲写入数值的地址。 **注意：**你必须在[时间设定]页签中勾选[启用结束行动]才能使用这个选项。

2、时间设定(当使用者选择[常数])。

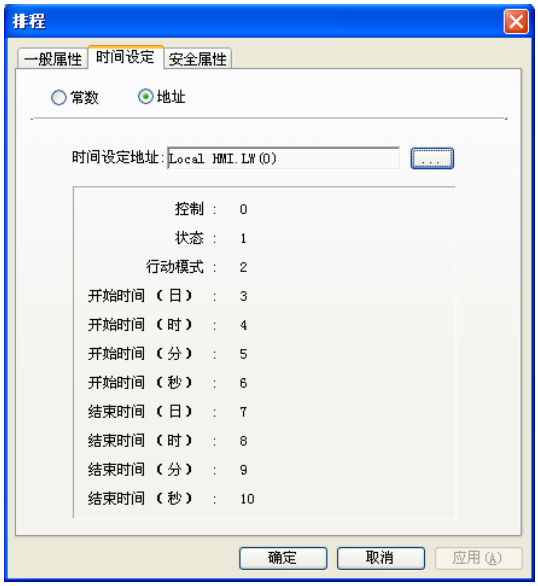


设定	动作描述
常数/地址	选择设定起始时间和结束时间的方法。 ● 常数 指定一个固定的时间和日期。 ● 地址 特定地址储存时间和日期的信息。
设定为单一日期	● 启用时 假如使用者欲设定一个范围为 2 天以上的排程，可以勾选这个选项。但只能设定单一的起始时间和单一的结束时间。

设定	动作描述
	 **注意：** ➢ 必须要输入起始时间和结束时间。 ◆ 不能在起始时间和结束时间字段里输入一模一样的时间和日期。 ● 停用时 排程时间必须被限定在一天之内(起始时间和结束时间必须在 24 小时内)。可于排程内选择多个起始和结束日期，也可在每天的相同时间执行特定动作。 当使用者想指定结束时间，请勾选[启用结束行动]。  **注意：** ➢ 不能在起始时间和结束时间字段里输入一模一样的时间和日期。 ➢ 此种时间排程只适用于一天之内的排程，因此如果所键入的结束时间早于起始时间，则结束动作将会等到下一天才会执行。 **例如：** **起始日期：星期一** **起始时间：22:00:00** **结束时间：01:00:00** 
起始	选择起始的时间和日期。 当[设定为单一日期]停用时，你可以指定一天以上的日期。
结束	当[启用结束行动]启用时，才能指定结束时间。 日期只能在[设定为单一日期]启用时才能被设定。

3、时间设定(当使用者选择[地址]，即通过设定寄存器的数据来设定时间，如下图)

若“地址”被勾选，系统经由设备类型的地址设定开始/结束时间及日期。并且，使用者能改变及设定排程于运转期间。

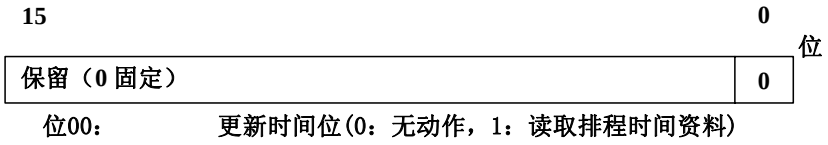


以下资料长度以 16-bit 为例。(当指定之寄存器为 32 位时，只有较低的 16 位产生作用)

设　定	描　　述
时间设定地址	指定一个位于任一设备/PLC 的内存区块起始地址，这个区块用来储存时间设定的所有信息。
控制	要求读取出资料，包括[模式]、[起始时间]和[结束时间]。 ➢ 控制(时间设定地址 + 0)
状态	在[控制]时间资料读取完成之后或所键入的时间资料有错时会改变一个位为 ON。 ➢ 状态(时间设定地址 + 1)
行动模式	指定[启用结束行动]和[设定为单一日期]。 ➢ 时间设定(当[常数]被选择时) ➢ 模式(时间设定地址 + 2)
行动模式	
开始时间(日)	指定开始日期。 ➢ 开始日期(开始日期：时间设定地址 + 3)
开始时间(时)	指定开始时间。 ➢ 开始时间(开始时间：时间设定地址 + 4 到 + 6)
开始时间(分)	
开始时间(秒)	
结束时间(日)	指定结束日期。 ➢ 结束日期(结束日期：时间设定地址 + 7)
结束时间(时)	指定结束时间。 ➢ 结束时间(结束时间：时间设定地址 + 8 到 + 10)
结束时间(分)	
结束时间(秒)	

a. 控制(时间设定地址 + 0)

当[更新时间位]被侦测为 ON(0->1)时，则读出[模式]、[开始时间]和[结束时间]。

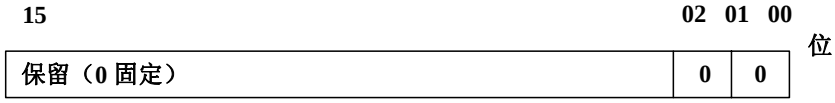


由上可以看出，将"控制地址"的 bit0 设置为 ON 时，更新排程设定的时间。

注意：
- 从时间设定地址的[模式](地址 + 2)到[结束时间(秒)] (地址 + 10)里的资料将不会定期地被读出来。当 HMI 中对应时间设定的资料改变时，请务必把[控制]中的[时间取得要求位]设为 ON (0->1)。

b. 状态(时间设定地址 + 1)

在[控制]中的时间资料读取完成之后，HMI 将会把[时间取得完成位]设为 ON(0->1)。同样地，若输入的时间资料不正确，[错误通知位]将会同时被设为 ON(0->1)。

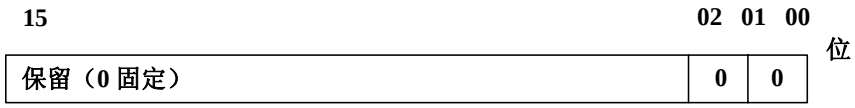


位00：时间取得完成位(0：不是时间取得动作正在执行，就是还没完成；1：时间资料取得完毕)
位元01：错误通知位(0：时间资料正被正确资料所取代；1：时间资料中包含错误)

注意：
- 一旦[时间取得完成位]的触发被设备/PLC 所识别，请务必把[控制]中的[时间取得要求位]设为 OFF(1->0)。一旦这个位被设为 OFF(1->0)，则请将[状态]中的[时间取得完成位]及[错误通知位]同时设为 OFF(1->0)。

c. 模式(时间设定地址 + 2)

启用或停用[结束时间行动设定]和[单一日期指定模式]。不管[结束时间行动设定]的状态是如何，这个间接指定的时间资料([时间设定地址]中的 11 个字组地址)都会被读出。



位00：结束时间行动设定(0：停用；1：启用)
位元01：单一日期指定模式(0：停用；1：启用)

注意：
- 假如在[结束时间行动设定]输入 0(停用)，则结束时间资料将被读出但忽略掉。
- 假如在[单一日期指定模式]输入 1(启用)，请确认你有输入开始及结束时间信息。假如有 2 个以上的开始/结束日期位被同时设为 ON，则会产生错误。

d. 开始/结束日期(开始日期：时间设定地址 + 3；结束日期：时间设定地址 + 7)

指定一个日期，用来触发开始/结束行动。

15 ... 07	06	05	04	03	02	01	00	位
保留（0 固定）	Sat	Fri	Thu	Wen	Tue	Mon	Sun	

位 00： 星期日(0：无；1：指定)

位 01： 星期一(0：无；1：指定)

位 02： 星期二(0：无；1：指定)

位 03： 星期三(0：无；1：指定)

位 04： 星期四(0：无；1：指定)

位 05： 星期五(0：无；1：指定)

位 06： 星期六(0：无；1：指定)

e. 开始/结束时间(开始时间：时间设定地址 + 4 到 + 6；结束时间：时间设定地址 + 8 到 + 10)

时：0 - 23；分：0 - 59；秒：0 - 59。

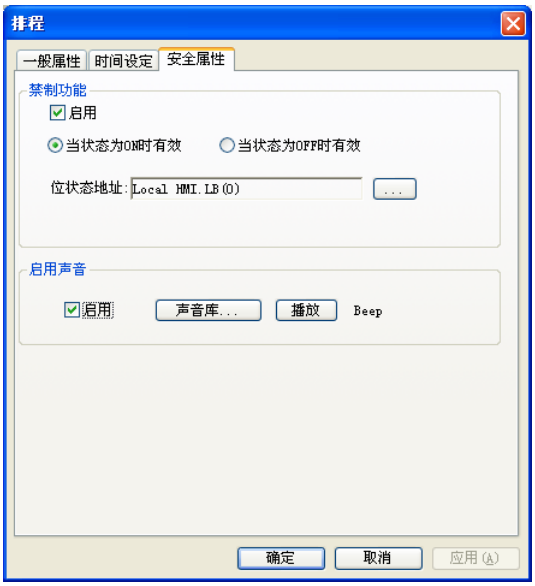
假如你所指定的值超出上面的范围，将会产生错误。

注意：

- 使用者所输入的时间资料应为二进制格式，系统不接受 BCD 格式的时间资料。
- 结束时间取决于[模式](地址 + 2)设定。同样地，[结束时间行动设定] (位 00)有效与否取决于[单一日期指定模式] (位 01)的使用。

单一日期指定模式	使用	不使用	
结束时间行动设定	使用	使用	不使用

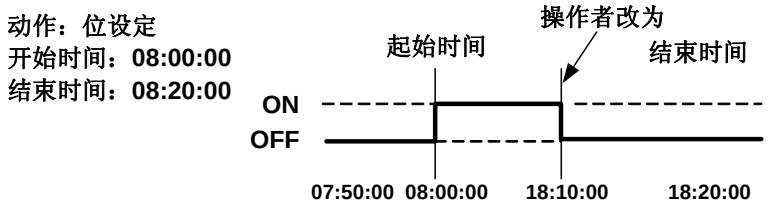
4、 禁制



设 定	描 述
禁制位	InoTouch Editor 在执行“开始”前会先读取位的状态。若位的状态是 ON，则排程不会被执行。
声音	假如使用者有设定声音对象，则不论是执行开始时间行动或结束时间行动皆会播放选定的声音。

13.20.5 限制

- 每个工程最多可注册 32 个[排程]对象。
- 时间排程的特性为一次动作。当开始时间到达时，特定的设备地址只会被写入一次，这个写入的动作将不会重复。



- [开始/结束欲写入数值]和[禁制位]只会在排程开始时读取一次。因此不允许定时读取，且也许会有少许数据通讯延迟导致起始时间延迟。
- 当使用者改变 HMI 的系统时间，系统将会重新确认排程中起始与结束时间的范围。假如编辑的对象位于新范围中，则开始行动会被执行。
- 当相同的起始和结束时间出现在多个排程中，他们将依编号由小到大顺序被处理。
- 当[时间设定]指定为[地址]，系统将会定期去读取[控制]地址，时间长短视系统忙碌程度而定。当[控制]地址的位 00([时间取得要求位])被设为 ON，在[状态]地址和前面资料被读出之前，可能会产生一段时间延迟。同样地，多个[排程]对象的位 00([时间取得要求位])同时被设为 ON，则行动之前可能会产生延迟。
- 当[时间设定]指定为[地址]且你指定开始时间和结束时间超过时间合法的范围，则设定的时间可能不会正确地运作。而且，不能使用 BCD 当成输入值。
- 当[时间设定]指定为[地址]，请注意[控制]里的[时间取得要求位]是否有被设定。
- 目前[排程]功能的执行是以“周”为循环单位执行的，还无法指定任意日期来执行特定的动作。

14

资料取样、趋势图与历史数据显示

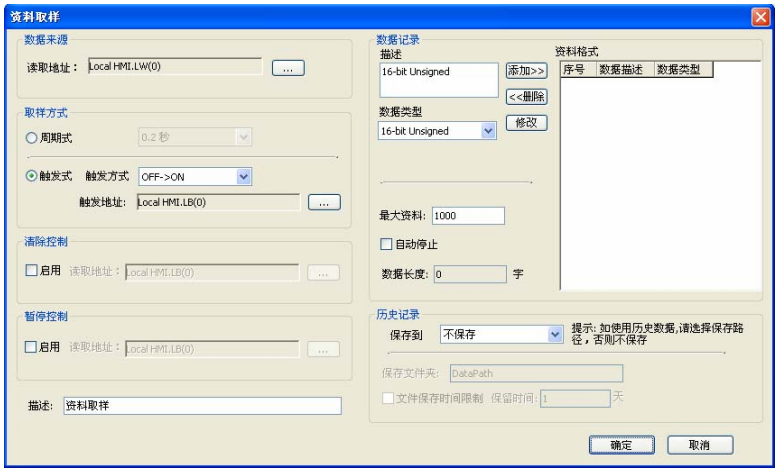
第十四章　资料取样、趋势图与历史数据显示

实际应用中，有时候需要用到实时监控到某一个参数的变化过程，以及查询其之前的变化过程。例如检测锅炉出水温度、压力的变化过程等。InoTouch Editor 提供的趋势图控件就是来实现这个功能的。趋势图是使用连续的线段描绘出显控参数的变化过程，并能够查询历史的变化过程。而趋势图中的数据是从哪里来的呢？趋势图所绘出的参数变化过程，它是使用“资料取样”控件采样到的数据而绘制成的曲线。为此，先了解一下“资料取样”控件。

14.1 资料取样

资料取样控件可以周期性的读取 PLC 中的数据寄存器的数值，并且将读取到的数据以文件的方式保存在指定的位置，可以使人机界面本身的内存中，或者是 SD 卡、U 盘，甚至是一台计算机中。这样采样到的资料，就可以被“趋势图”控件等用来显示成曲线。

a、单击 InoTouch Editor 软件左侧项目管理下“采样数据表”，就会打开“资料取样”对话框，如下图所示。



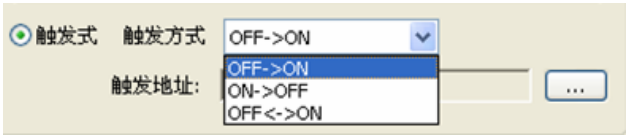
[数据来源]

定义资料取样读取的目标 PLC 的寄存器地址。多一次采样多个资料时，此时设定的是起始寄存器的地址。

[取样方式]

周期式：最小采样周期为 0.1 秒，最大采样周期为 120 分钟。设定好之后，即以这个频率去读取 PLC 的数据；

触发式：当设定的条件满足时，才进行资料采样。可以设定为 PLC 的某一个位，当该位的状态改变时，执行资料取样。触发式采样的条件有三种，即：



“OFF->ON” ：当指定地址的状态由 OFF 变为 ON 时，将触发一次取样动作。

“ON->OFF” ：当指定地址的状态由 ON 变为 OFF 时，将触发一次取样动作。

“ON<->OFF” ：当指定地址的状态改变时，将触发一次取样动作。

[清除控制]

当指定地址的状态被设定为 ON 时，将清除已取样获得的资料，取样资料的数目也会被归零，但不影响已存入到外部设备中的取样资料文件。其它设定项目请参考“控件一般属性的设定”。

[暂停控制]

当指定地址的状态被设定为 ON 时，将暂停取样动作，直到指定地址的状态被恢复为 OFF。其它设定项目请参考“控件一般属性的设定”。

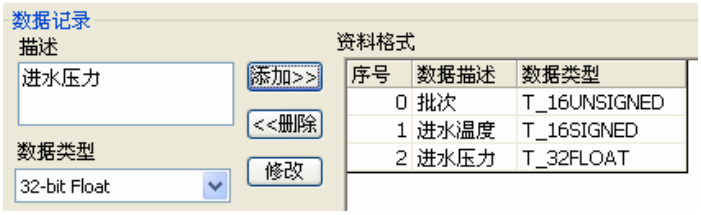
[数据记录]

最大记录

一个资料采样一天中的最大取样记录条数。最大为 86400 条记录。也即按 1 秒钟采样一次来计算。

资料格式

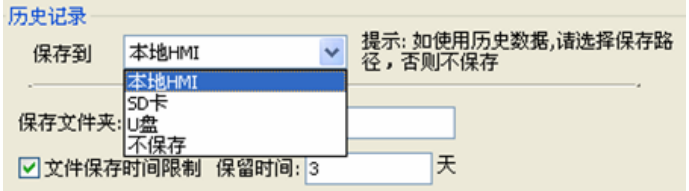
建立的一个资料取样所包含的数据格式。一个资料取样可能包含超过一项以上的数据，InoTouch Editor 提供的资料取样动作可以同时采样不同格式的数据。按下[数据类型]，使用者可以自行定义一个取样资料的内容。以下图的例子来说，使用者共定义了三种格式的数据，分别为“批次”(16-bit Unsigned)、“进水温度”(16-bit Signed)与“进水压力“(32-bit Float)，长度为总共为 4 words。 也就是说每次的取样动作，InoTouch Editor 会自指定的地址每次取样长度为 4 words 的数据，作为建立的这个取样资料的内容。



其中，名称“批次”、“进水温度”、“进水压力”等名称，是在 “描述”里面定义的，如上图所示。这些名称在之后将采样的文件转换为 CSV 或 Excel 表格时，就会显示在列表上。

[历史记录]

用来指定取样资料的保存位置，但在计算机上使用离线仿真功能时，文件记录一律储存在与 InoTouch Editor.exe 相同目录下的“datalog”文件夹中。



[保存到 HMI]

将取样资料储存在 IT5000 人机界面里面。

[保存到 SD 卡]

将取样资料储存到 SD 卡中。

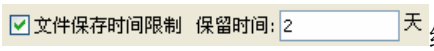
[保存到 U 盘]

将取样资料储存到 U 盘 中。

[文件名]

设定保存的资料取样的文件名称。此文件名称请不要使用汉字，尽量使用 ASCII 或者数值的方式来命名。

当选 [文件保存时间限制]此项设定，此项设定值用来决定资料取样记录档案被保留的时间。以下图为例，此时设定保留时间为 2 天，也就是说系统将仅保留昨天与前天的资料取样记录档案，超过这个时间范围的档案将自动被删除，以避免储存空间被耗尽。由于人机界面储藏空间有限，一般如果选择将资料取样资料保存到人机界面中，建议设定此项参数。

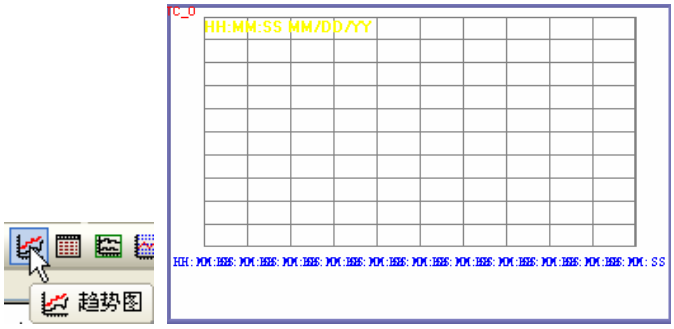


经过以上设定，就建立了需要的资料取样。

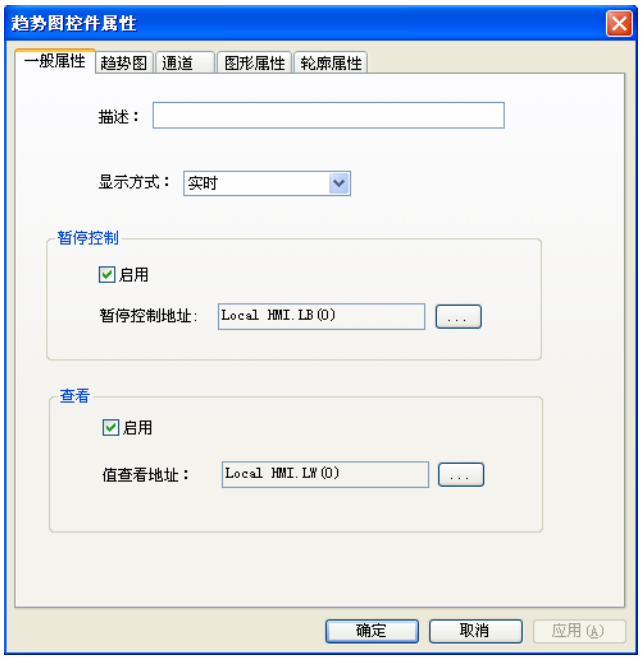
14.2 趋势图

建立好了资料取样后，此时就可以使用趋势图控件了。

单击 InoTouch Editor 软件菜单“控件/趋势图”或者工具栏上的图标  ，在窗口中点击鼠标左键，就建立了“趋势图”控件，如下图所示。



选择“趋势图”双击或单击鼠标右键选择“属性”进行编辑，如下图所示：



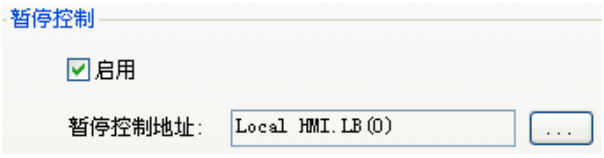
[显示方式]

选择数据来源的形式，可以选择“实时”或“历史” 两种方式。

a. 实时

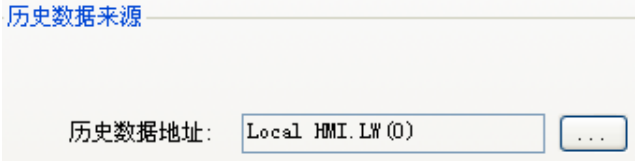
可显示来自“资料取样”控件从开机后到目前的取样资料，如需显示之前的资料，需选择“历史模式”，从历史资料中读取。

可以利用“暂停控制”功能暂停控件画面更新的动作，但仅只暂停画面刷新，并不会暂停“资料取样”控件的取样动作。下图为“暂停控制”的设定画面，将“暂停控制”中指定地址的状态设定为 ON，画面将暂停刷新。



b. 历史

历史记录来自“资料取样”控件使用日期来分类并储存的取样资料，当只有“资料取样”里面有设置历史记录保存时，趋势图才会有“历史”模式。使用“历史”模式可以利用 [资料取样控件索引]选定要显示的历史记录，并利用“历史控制”选择不同日期的历史记录。下图为“历史控制”的设定画面。



InoTouch Editor 会将取样资料的历史记录档案依时间先后排序，日期最新的档案记录为 0(一

般是今日已存盘的取样资料)，日期次新的档案为记录 1，其余记录依此类推。对于每天都开机运转的设备，当设置了资料采样保存时，则可以认为当天已经存盘的取样资料编号为 0，昨天保存的取样资料编号为 1 等依此类推。

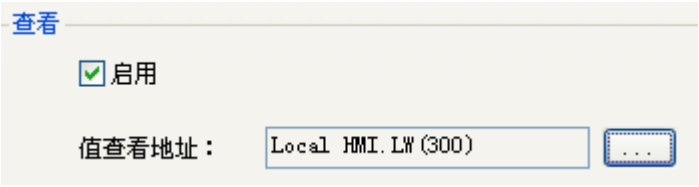
在“历史控制”中所指定寄存器中的数据如果为 0，“趋势图”控件将显示记录 0 的数据； 寄存器中的数据如果为 1，将显示记录 1 的数据，也就是说寄存器中的数据如果为 n，将显示记录 n 的数据。

举例说明“历史控制”的使用方式，上图的寄存器为[LW200]，假使目前的“资料取样”控件已储存的取样数据文件依时间先后分别为 pressure_20110315.dtl、pressure_20110317.dtl、pressure_20110319.dtl、pressure_20110322.dtl，共 4 个档案，并且今日时间为 2000/03/22，则依照[LW200]中的数据内容，“趋势图”所显示的取样数据文件整理如下：

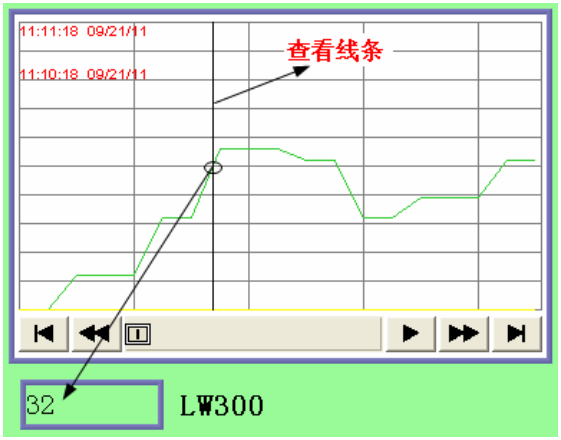
[LW200]中的数据	所显示历史资料的来源档案
0	pressure_20110322.dtl
1	pressure_20110319.dtl
2	pressure_20110317.dtl
3	pressure_20110315.dtl

也就是说[LW200]中的数据愈小，所观察到的为与今日时间愈接近的历史记录；另一种情形是，当[LW200]中的数据并无相对应的取样数据文件时，InoTouch Editor 将显示最后一个历史记录，例如[LW200]的值为 4 时，InoTouch Editor 仍显示 pressure_20110315.dtl 的取样资料数据。

[查看]



使用“查看”的功能可以让使用者触控“趋势图”控件时产生一条垂直的查看线条，并可以将该线条与趋势图交叉所在位置的取样数据输出到指定的地址，以下图为例，将查看线条所在位置的取样数据写至[LW300]中。



“查看”功能也可以输出多条取样曲线的取样数据，InoTouch Editor 会依照“资料取样”控件中所定义的取样资料数据格式，依序将标记所在位置的取样数据，从“查看”功能所定义的起始位置依序写入。例如“资料取样”控件的每个取样资料包含四个数据，依序为“16-bit unsigned”、“32-bit unsigned”、“32-bit float”与“16-bit Signed”，假设此时[LW300]为“查看”功能所定义的寄存器，则查看线所标记的取样数据的输出位置如下。

[LW300]　Line 0 : 16-bit Unsigned　(储存位置为 1 个 words)

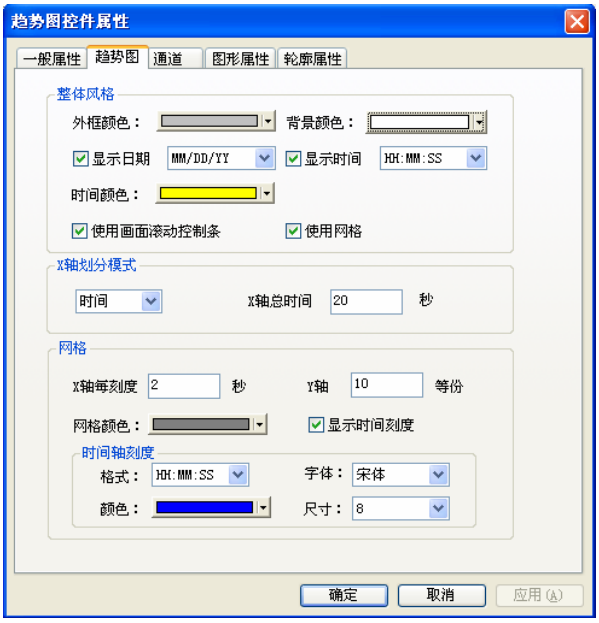
[LW301]　Line 1 : 32-bit Unsigned　(储存位置为 2 个 words)

[LW303]　Line 2 : 32-bit Unsigned　(储存位置为 2 个 words)

[LW305]　Line 3 : 16-bit Signed　(储存位置为 1 个 words)

所以，如果“查看”地址的数据格式与在建立“资料取样”时的资料格式不一致时，将无法查看到准确的数据。

下图为“趋势图”控件的“趋势曲线”设定页。



[外框]

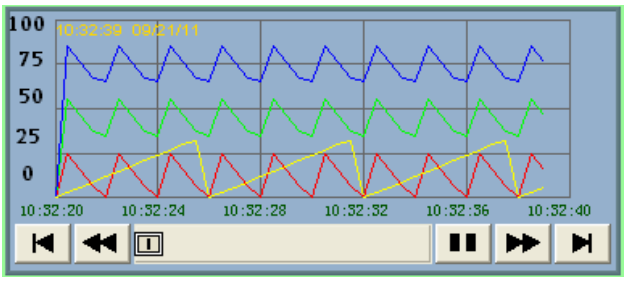
设定趋势图控件的外框颜色。

[背景]

设定趋势图控件的背景颜色。

[使用画面滚动控制条]

系统一般默认是勾选此项功能的。勾选此项后，趋势图控件显示为如下图所示式样。



其中各按钮的功能描述如下：

 按下后画面将显示最初的取样资料，并关闭画面自动卷动功能。

 按一下画面将显示 1 个垂直间隔前的取样资料。

 显示此图形表示目前已关闭画面自动卷动功能，按下后将重新开启此项功能。

 按一下画面将显示 1 个垂直间隔后的取样资料。

 按下后画面将显示目前最新的取样资料。

 显示此图形表示目前画面自动卷动功能已被开启，按下后将关闭此项功能。

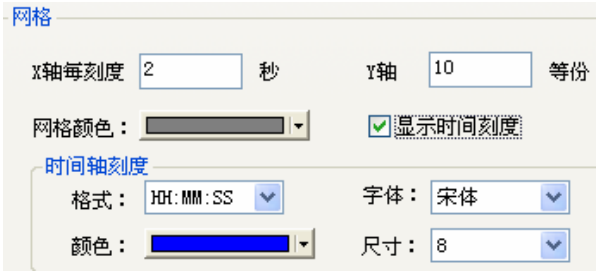
[使用网格]

选择是否使用网格线。

X 轴划分模式

当选择[像素]来设定取样点的描绘距离时，则此时用来选择每两个垂直网格线间将包含几个取样点。

若选择[时间]来设定控件宽度所显示资料的时间范围，则此时用来选择每两个垂直网格线间所显示资料的时间范围。



网格项目

设定趋势图线条范围内网格线的数目与颜色。

[X 轴每刻度]

可以用来设定控件宽度所显示资料的时间范围，则此时用来选择每两个垂直网格线间所显示资料的时间范围。

[Y 轴]

用来设定取样点的描绘距离，则此时用来等分 Y 轴上的取样点。

[显示时间刻度]

当选择“显示时间刻度”可以在图上看到时间轴刻度。

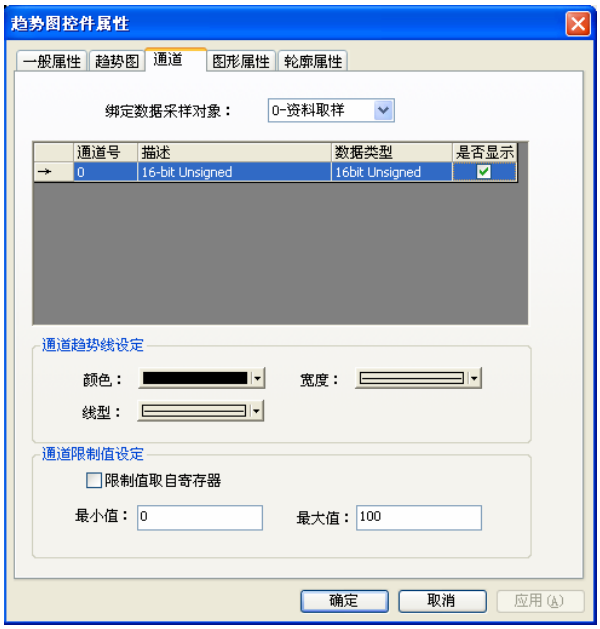
[时间轴刻度]

最新的取样资料所获得的时间会被标示在控件的左上角，此项目用来设定时间的显示格式与颜色。

通道

[通道趋势线设定]

设定各条曲线的样式与颜色，与曲线所能描绘数据的上下限值。如下图所示。



[通道限制值设定]

[最小值]、[最大值]

[最小值]与 [最大值]用来设定各曲线所描绘的取样数据的最小值与最大值。也就是说如果存在某一曲线所描绘的取样数据最小值为 50，最大值为 100，则[最小值]与[最大值]需设定为[50]与[100]，如此所有的取样数据才会完全被描绘在控件中。

[限制值取自寄存器]

若勾选“限制值取自寄存器”，则表明曲线的最小值与最大值范围有设定的寄存器来决定。此最大值与最小值的数据寄存器的数据格式，必须与该曲线在建立“资料取样”时的数据格式一致，否则设定结果不正确。假设取自寄存器的地址为“address”，那么取样资料设置数据格式不同时，则最大值与最小值的地址关系列表如下：

变量型态	最小值读取地址	最大值读取地址
16-bit BCD	address	address + 1
32-bit BCD	address	address + 2
16-bit Unsigned	address	address + 1
16-bit Signed	address	address + 1
32-bit Unsigned	address	address + 2
32-bit Signed	address	address + 2
32-bit Float	address	address+2

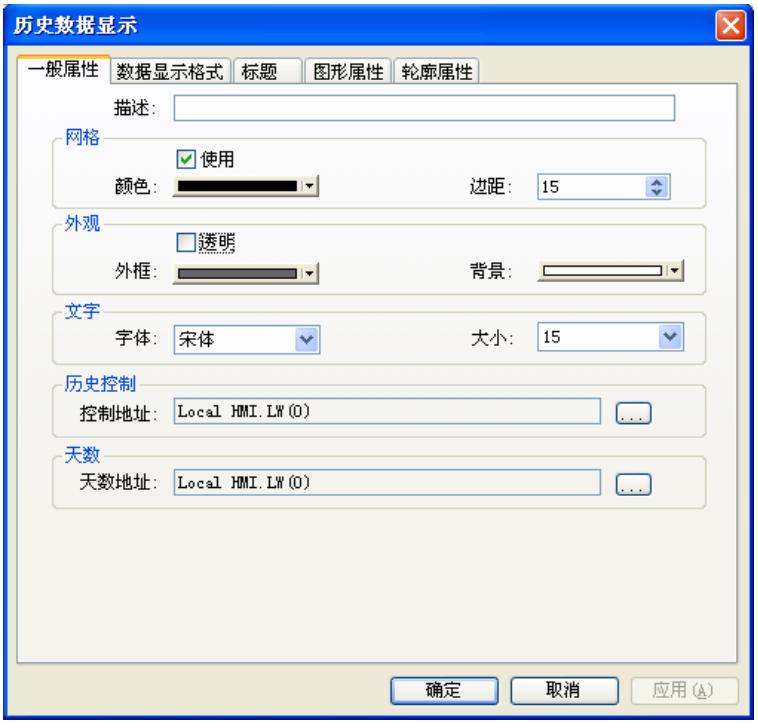
14.3 历史数据显示

趋势图是将“资料取样”采集的数据，以连续线段的方式显示出来的。而“历史数据显示”控件是以表格的方式来显示保存在 HMI、SD 卡或 U 盘等设备上的历史取样资料的数据。

注意：使用历史数据显示控件前，也必须先建立“资料取样”控件，且要选择保存历史记录数据；当历史资料取样数据保存到 SD 卡或 U 盘时，需要接入到人机界面中，否则历史数据显示表格中数据均显示为零。

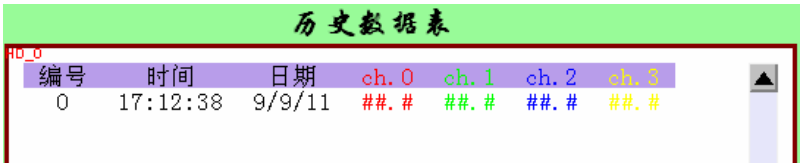
打开 InoTouch Editor 软件菜单的“控件/历史数据表”，或者工具栏上的图标  ，在窗口中点击鼠标左键，就建立了“历史数据”控件，如下图所示。

选择“历史数据”双击或单击鼠标右键选择“属性”进行编辑，如下图所示：



[网格线]

选择控件是否使用网格线区分每个字段，下图为不使用网格线的情形。



[颜色]

网格线所使用的颜色

[边距]

此项设定值用来调整各字段间的距离，下图为使用不同[边距]设定时的显示情形。

外观

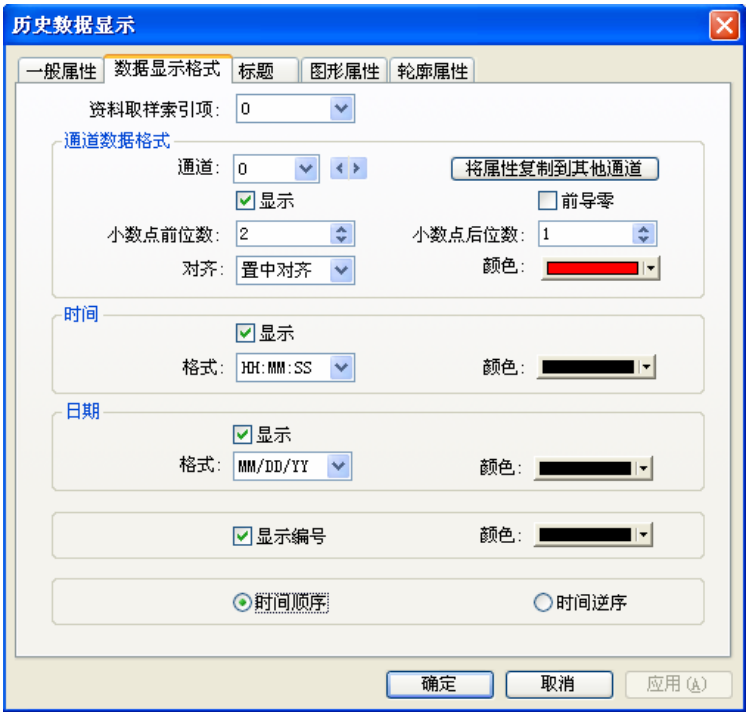
设定控件的外框与背景颜色，若勾择**[透明]**表示不使用外框与背景颜色，此时控件的外观如下图所示(此时控件也未使用图形与向量图)。此时的底色完全取决于画面窗口的底色颜色。

[文字]

设定表格标题的文字字体和字体大小。

[历史控制]

InoTouch Editor 会将取样资料的历史记录档案依时间先后排序，日期最新的档案为记录 0(一般是今日已存盘的取样资料)，日期次新的档案为纪录 1，其余记录依此类推。“历史控制地址”则用来指定要显示哪一个记录的数据地址。



[资料取样索引项]

选择哪个“资料取样”控件作为所需的数据来源，可参考“资料取样”控件的说明。

[通道数据格式]

上图的对话框用来设定“资料取样”的历史数据显示格式，由上图可以发现目前使用的“资料取样”控件执行一次取样的动作将读取 4 个数据(通道 0～通道 3)，由上图也可以发现各数据的数值格式(例如通道 0 为 16 bit Unsigned)，这些是事先定义在“资料取样”控件中。

时间、日期与编号

用来选择是否显示资料的取样时间与日期，并决定时间与日期的显示格式。同时设置这些文字的颜色。

[按时间顺序]

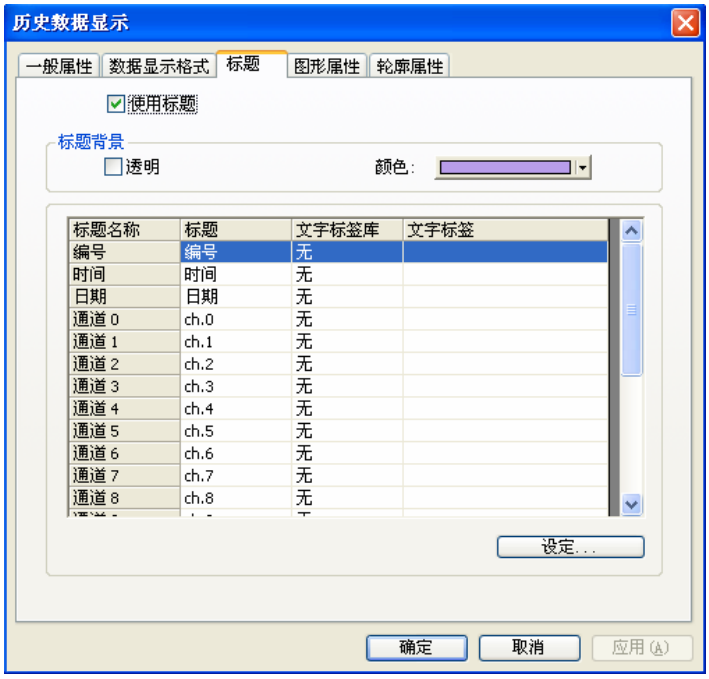
选择[按时间顺序]表示将先显示取样时间较早的资料。

[按时间逆序]

选择[按时间逆序]表示将先显示取样时间较晚的资料。

[标题]

单击“历史数据显示”控件属性的分页“标题”，则显示如下对话框。在此用来设定历史数据显示控件所使用的标题。



[使用标题]

选择是否使用标题。

[透明]

勾选[透明]表示不使用标题文字的背景色。

[背景颜色]

设定标题文字的背景色，不勾选“透明”时有效。

选择 “需要修改的一栏”在文字标签库处双击鼠标，则可以修改这些标题的文字内容。

小结：本章主要介绍了如何建立“资料取样”，如何利用“资料取样”采集的数据来显示趋势图和使用报表的方式来显示资料取样的历史数据。也就是说，“资料取样”采集的数据，可以在人机界面上以“趋势图”和“历史数据显示”两个控件使用两种不同的方式来显示出来。在使用这些控件时，特别是“趋势图”和“历史数据显示”两个控件，其数据的相关寄存器的数据格式，一定要与“资料取样”建立时定义的数据格式要一致。一个“趋势图”控件最多显示 20 条曲线，一个“历史数据显示”控件一次最多显示 20 个通道的数据。

15

事件登录、事件显示与报警显示、报警条

第十五章　事件登录、事件显示与报警显示、报警条

“事件登录”用来定义事件的内容与触发这些事件的条件，InoTouch Editor 并可以将已被触发的事件(这时事件又被称为报警)与这些事件的处理过程储存到指定的位置，所储存文件的名称一律使用 EL_yyyymmdd.evt 格式，其中 yyyymmdd 为文件建立的时间，由系统自行加入。例如事件记录文件名称为 EL_20110315.evt，即表示此档案记录 2011 年 3 月 15 日所发生的事件。若在实际做工程画面时，若要实时的显示机器运行中的报警信息或者显示曾经发生过的报警信息等内容，需要先了解 InoTouch Editor 提供的[事件登录]控件。

InoTouch Editor 并提供下列系统寄存器来管理这些事件记录文件：

[LB 9021]　清除目前的事件记录

[LB 9022]　删除最旧的事件记录文件

[LB 9023]　删除全部事件记录文件

[LB 9024]　更新事件记录统计信息

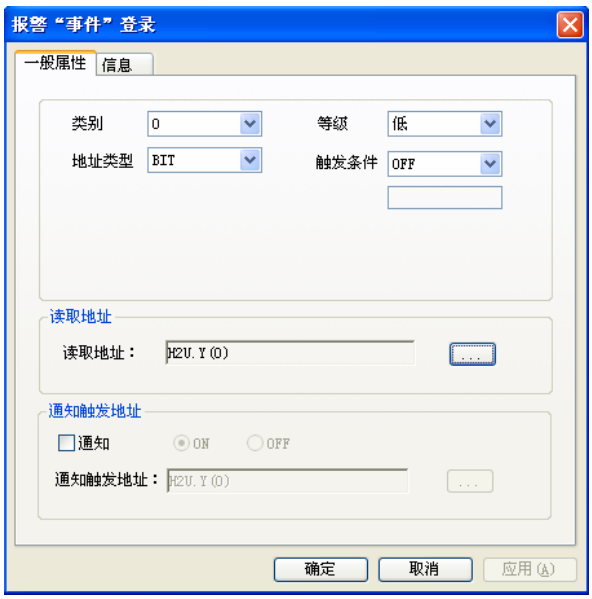
[LW 9060]　目前存在的事件记录文件的数目

[LW 9061]　全部事件记录文件的大小

15.1 事件登录管理

在使用[事件显示]、[报警显示]、[报警条]等控件可以显示事先定义好的事件/报警内容，将这些控件放置在屏幕画面上，当条件满足时，相应的事件/报警文字信息就会显示出来。在使用这些控件之前，需要事先在[事件登录]这个控件里面，根据报警显示的条件定义各种报警需要显示的文字内容。

单击 InoTouch Editor 软件左侧项目管理下“事件注册表”，就会打开“事件注册表”对话框，如下图所示。



[类别]

事件的类别。事件的类别选择范围是 0~255，在建立事件内容时设定。事件的类别在事件建立好之后是不可更改的。若需要更改除非将该事件删除，重新建立一个需要的类别，文字内容相同的事件信息。

[等级]

事件的等级，依照重要程度可选择为“低”、“中”、“高”、“紧急”。当已发生事件的数目等于系统允许的最大值数目时(预设为 1000 条，如需增加请参考“系统参数”的[一般属性]中的说明)，重要程度较低的事件将从事件记录中被剔除，并加入新发生的事件。

[地址类型]

事件地址类型可以选择为“Bit”或“Word”模式。

[读取地址]

系统利用读取此地址所获得的数据，来检查事件是否满足触发条件。其余设定请参考“控件一般属性设定”章节的相关说明。

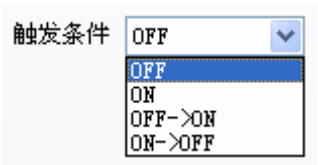
[通知触发地址]

当事件被触发时，将利用此项位地址送出特定的讯号。选择[设 ON]将对此特定位地址送出 ON 讯号；选择[设 OFF]则对此特定地址送出 OFF 讯号。

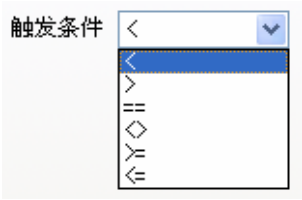
[触发条件]

当事件的[地址类型]触发条件选择“Bit”时，触发条件可选择“ON”、“OFF”、“OFF->ON”、“ON->OFF” 等四种，参考下图。

ON	当[读取地址]的状态为 ON 时，事件被触发，并产生一个事件记录
OFF	当[读取地址]的状态为 OFF 时，事件被触发，并产生一个事件记录
OFF->ON	当[读取地址]的状态由 OFF 变为 ON 的瞬间，事件被触发，并产生一个事件记录
ON->OFF	当[读取地址]的状态有 ON 变为 OFF 的瞬间，事件被触发，并产生一个事件记录

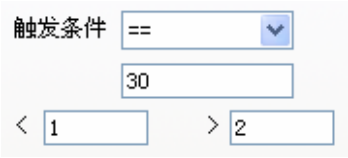


当事件的[地址类型]触发条件选择“Word”时，触发条件可选择项目如下。



此时系统会使用从[读取地址]读取的数据与触发条件相比较，判断事件是否被触发。比较特别的是如果触发条件选择“==”或“<>”，触发条件必须设定[小于]与[大于]这两项属性，参考下图，其中[小于]用在事件触发条件，[大于]用在系统恢复正常时的条件。举下面两个例子来说明：

设定实例 1:



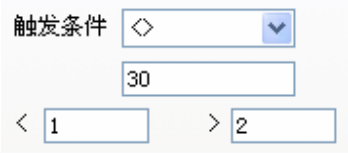
上面的设定内容表示当[读取地址]中的数据大于等于 29(= 30 -1)且小于等于 31(=30 + 1)时，事件将被触发。也就是事件被触发的的条件为：

29 <= [读取地址]中的数据 <= 31

事件被触发后，当 [读取地址]中的数据大于 32(=30 +2)或小于 28(= 30 -2)时，系统将恢复为正常状态。也就是系统恢复为正常状态的条件为：

[读取地址]中的数据 < 28 或[读取地址]中的数据 > 32

设定实例 2:



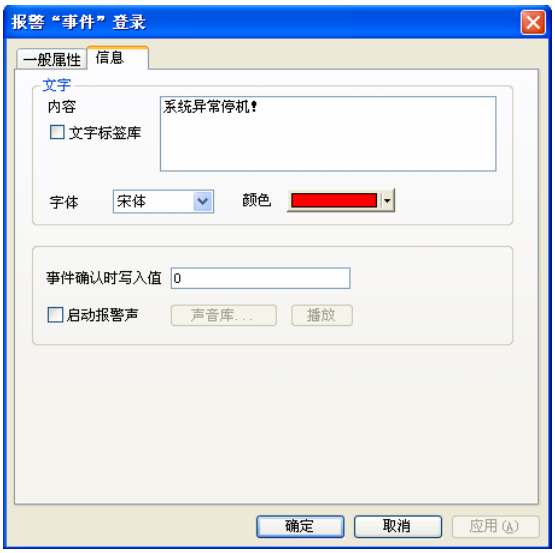
上面的设定内容表示当[读取地址]中的数据小于 29(30 - 1)或大于 31(= 30 +1)时，事件将被触发。也就是事件被触发的的条件为：

[读取地址]中的数据 < 29 或[读取地址]中的数据 > 31

事件被触发后，当 [读取地址]中的数据大于等于 28(= 30 -2)且小于等于 32(=30 +2)时，系统将恢复为正常状态。也就是系统恢复为正常状态的条件为：

28 <= [读取地址]中的数据 <= 32

另一个分页为 [信息]设定页，参考下图。



文字

[内容]

事件记录在[报警条]、[报警信息显示]与[事件显示]控件中显示的内容。内容也可以使用多行文字，其余设定请参考“控件一般属性设定”章节的相关内容。

可以在显示内容中包含事件被触发当时本机触摸屏中 LW 地址中的数据，使用格式为

%#d

此种格式使用**%**作为起始符号，#用来指定 LW 的地址，使用 **d** 作为结束符号。例如显示内容被设定为“温度太高= %20d”，表示在事件被触发时，将显示当时 LW20 中的数据。也就是说事件被触发时，如果 LW20 中的数据为 100，则在“事件显示”组件中的显示内容将为“温度太高 = 100”。

也可以在显示内容中包含事件被触发当时，特定地址类型中的数据，此特定地址类型与事件登陆的[读取地址]需为相同地址类型，例如[读取地址]选择 MW 地址类型，则此种方法也仅能显示 MW 地址中的数据。使用格式为

$#d

此种格式使用**$**作为起始符号，#用来指定地址，使用 **d** 作为结束符号。例如显示内容被设定为“温度太高 = $15d”，且[读取地址]使用 MW 地址类型，则表示在事件被触发时，将显示 MW15 的数据。也就是说事件被触发时，如果 MW15 中的数据为 42，则在”事件显示”组件中的显示内容将为“温度太高 = 42”。

注意：以上事件内容中显示的数值是在事件发生时的显示的数据。当事件显示的条件一直存在，也即该事件或者警告信息还没有恢复正常时，这个数据可能还会一直在升高，但此时的数据不会显示在“事件内容”中。只有当事件恢复正常，触发事件的条件再次出现时，此时的数据才会在事件内容中显示。也就是说，事件内容中显示的数据，是在事件发生时的数据。

[字体][颜色]

每一个事件可以单独设定字体与颜色，在[报警信息显示]与[事件显示]控件中所显示事件的字型与颜色来自这些设定值。

[事件确认时写入值]

事件在事件显示控件中的显示记录被触控时，对特定位置的输出数据值，请参考下面有关“事件显示控件”的说明。

[报警声]

事件被触发时，可选择使用音效警示。可点选声音库选择警示的声音，并使用[播放]按钮确认选择的声音。

注意：只有硬件配置含音频输出口的，才可以选择启动报警声；没有音频输出的，请不要选择声音效果。

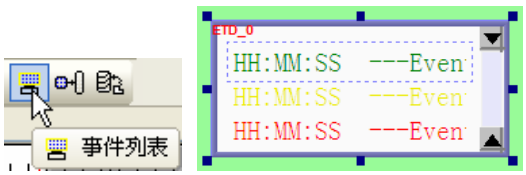
完成上述的各项设定后，即可增加一条新的事件登录信息，如需要再增加，请在空白处点击鼠标右键“新增事件项”，也可以在你想插入的地方“插入事件项”。

完成建立“事件登录”信息后，就可以使用[事件显示]、[报警显示]和[报警条]控件来显示刚刚登陆的各种报警信息。

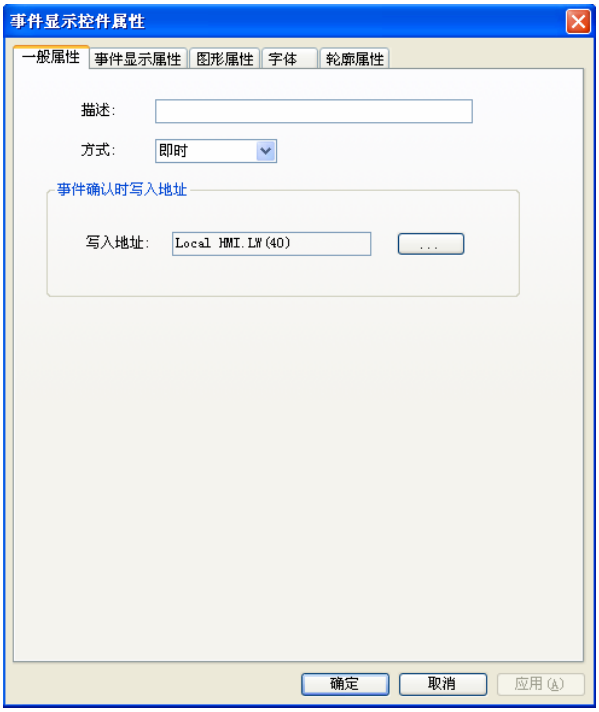
15.2 事件显示

“事件显示”控件可以用来显示已被定义在“事件登录”中，且曾经满足触发条件的事件，“事件显示”控件将利用事件被触发的时间先后，依序显示这些事件。“事件显示”控件也可显示事件被触发、确认与恢复正常状态(也就是事件信息恢复)的时间。

打开 InoTouch Editor 软件菜单的“控件/事件列表”，或者工具栏上的图标  ，在窗口中点击鼠标左键，就建立了“事件列表”控件，如下图所示。



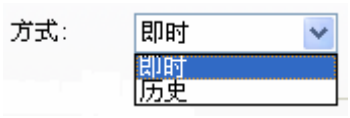
选择“事件列表”双击或单击鼠标右键选择“属性”进行编辑，如下图：



［方式］

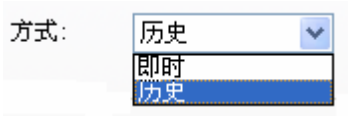
选择事件来源的形式，可以选择“即时”或“历史”。

a. 即时

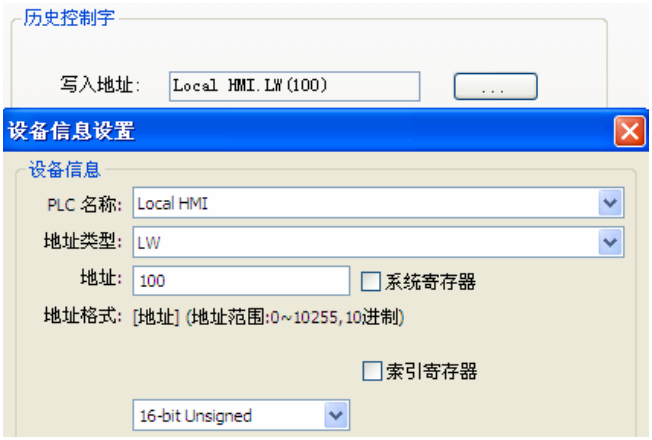


可显示“事件登录”中的事件从开机后到目前被触发的事件，如需显示其他日期的资料，需选择“历史”模式，从历史资料中读取。

b. 历史



选择此模式则“事件显示”控件会显示所储存的事件历史记录，故事先需要选择在“事件登录”控件里面要选择保存历史事件记录文件。InoTouch Editor 会使用日期分类所储存的事件历史记录，使用者可以利用“历史控制”选择要显示的记录。下图为“历史控制字”的设定画面。



InoTouch Editor 会将事件历史记录依时间先后排序，日期最新的档案为记录 0(一般是今日已存盘的事件记录)，日期次新的档案为记录 1，其余记录依此类推。

在“历史控制”中所定义寄存器中的数据如果为 0，“事件显示”控件将显示记录 0 中的数据；寄存器中的数据如果为 1，将显示记录 1 中的数据，也就是说寄存器中的数据如果为 n，将显示记录 n 中的数据。

举一个简单的例子说明“历史控制”的使用方式，上图的寄存器为[LW100]，假使目前的已储存的事件历史记录档案依时间先后分别为 EL_20110320.evt、EL_20110323.evt、EL_20110327.evt、EL_20110403.evt，并且今日时间为 2011/4/3,则依[LW100]中的数据，”事件显示”所显示的事件历史记录档案整理如下：

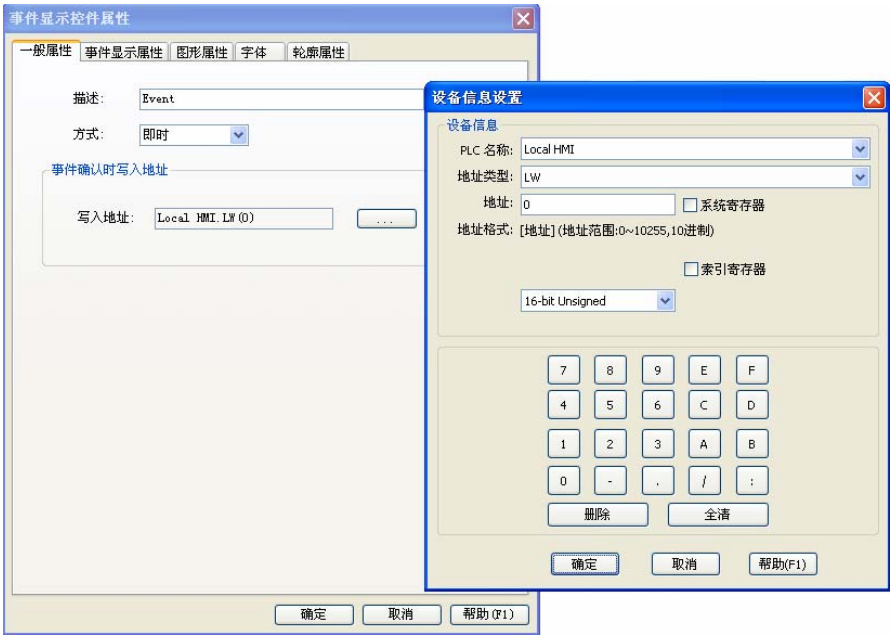
[LW100]中的数据	所显示的事件历史记录档案
0	EL_20110403.evt
1	EL_20110327. evt
2	EL_20110323. evt
3	EL_20110320. evt

也就是说[LW100]中的数据愈小，所观察到的为与今日时间愈接近的历史记录。

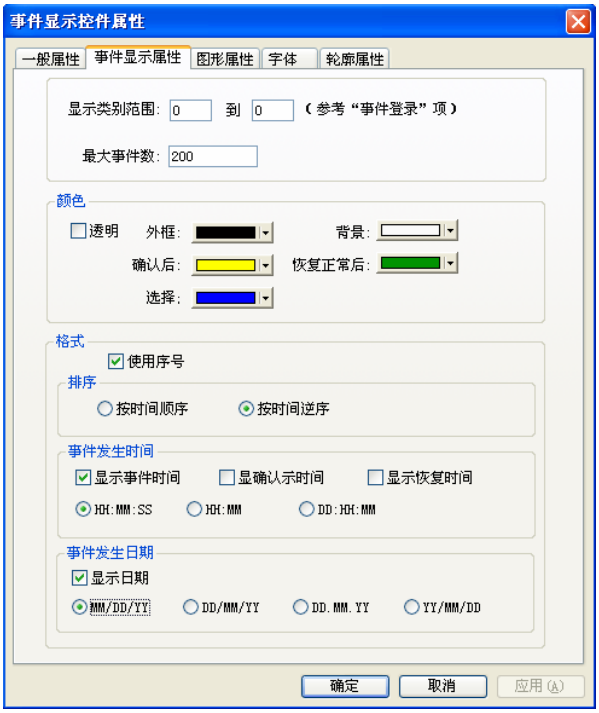
另一种情形是，当[LW100]中的数据并无相对应的取样数据文件时，InoTouch Editor 将显示最后一个历史记录，例如[LW100]的值为 4 时，InoTouch Editor 仍显示 EL_20110320. evt 这个历史记录。

[写入地址]

当“方式”选择为“即时”时，则会显示“写入地址”。当即时发生的事件被确认时，会将此事件预先设定的输出值，写至“写入地址”所设定的寄存器中。此项输出值在“事件登录”中登记每一条事件时设定，如下图所示。



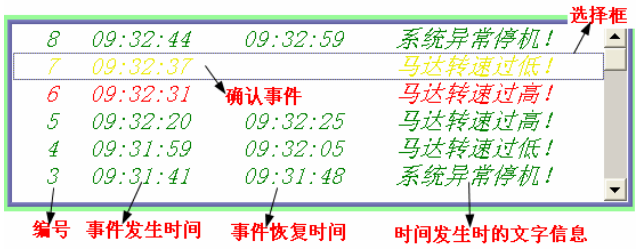
单击[事件显示]控件的“事件显示”分页，显示的内容如下图所示。



[显示的类别范围]

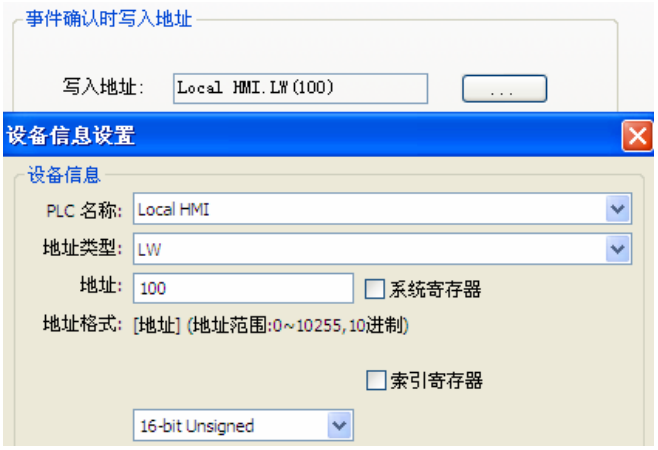
事件的“类别”需满足此项设定范围才会被显示(事件的“类别”在“事件登录”中设定)。例如当“事件显示”控件的“类别”此时被设定为 2~4，则仅有“类别”等于 2 或 3 或 4 的事件，才会被显示在“事件显示”控件中。可以参考“事件登录”说明中有关“类别”的解释。

[确认方式]



选择“确认”的动作方式，可以选择“单击”。此处所谓“确认”的动作是指使用者对于已发生并显示在“事件显示”控件上的事件，可以在该事件上“单击”该事件，此时 InoTouch Editor 除了会将该事件的显示颜色转变为“确认”的颜色之外，也会将此事件预先设定的输出值，写至[输出地址]中所设定的地址上。

以下图为例，当写入地址为[LW100]，且事件确认时的写入值为 31，则当使用者使用“确认”的动作时，[LW100]中的数据将被设定为 31，利用此项功能搭配“间接窗口”控件，可以让不同的事件弹出不同的窗口，这些窗口通常被用来说明事件的内容或者发生该故障时该如何处理等等。



[最大事件数]

控件所能显示事件的最大数目。当控件所显示的事件已等于所设定的最大数目时，新发生的事件将取代安全等级较低的事件。默认为 1000 条。

[颜色]

设定事件在各种状态下的显示颜色。

[确认后]

事件被确认后，事件内容的文字显示时所使用的颜色。

[恢复正常后]

事件恢复正常后，事件文字内容的显示颜色。

[选择]

显示的事件内容被选择时，该事件内容会被矩形包围起来，选择此时矩形的显示颜色。

[使用序号]

选择是否在所显示的事件前加上序号，较早发生的事件使用较低的序号。

[排序]

设定事件显示的顺序。

[按时间顺序]

最近发生的事件被排列在后(或在下)。

[按时间逆序]

最近发生的事件被排列在前(或在上)。

时间

[事件发生时间]

选择显示事件发生的时间。

[用户确认时间]

选择显示事件被“确认”的时间。

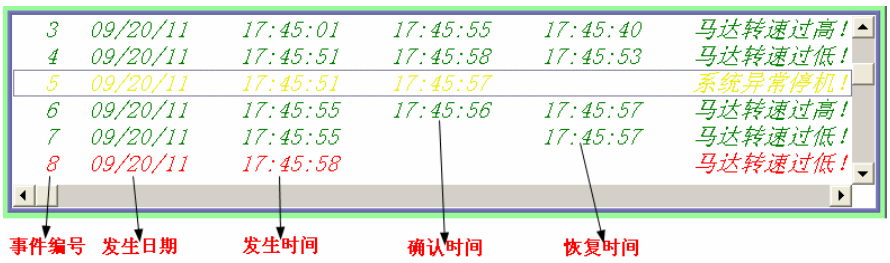
[恢复时间]

选择显示事件恢复正常的时间和显示格式。

日期

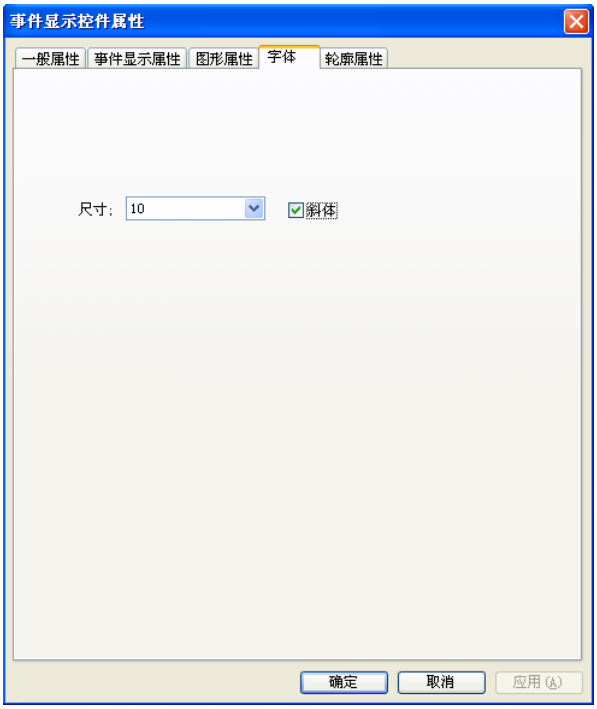
[事件发生日期]

选择显示事件发生的日期和显示的格式。



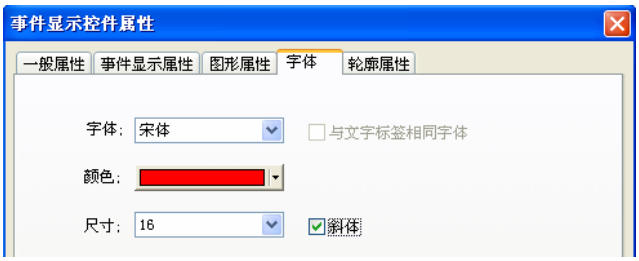
字体

当“事件显示”控件的模式设置为“即时”时，再单击“事件显示”控件的“字体”分页，将会显示如下对话框。



此时，只能设定“事件显示”控件中字体的大小和是否使用“斜体”效果。故障发生时，文字的字体，使用在“事件登录”时设定的字体。

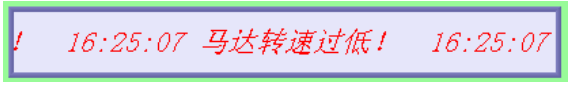
若在“事件显示”控件中的模式设置为“历史”，再单击“字体”分页，将会出现如下对话框。



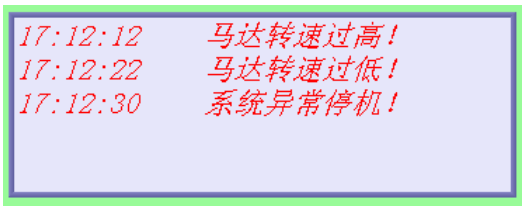
此时的字体可以由使用者自己来选择需要的字体，文字大小以及是否使用“斜体”效果等。在使用“多语言文字”显示的情况下，还可以勾选“使用与文字标签相同的字体”，即使用与在建立“文字标签”时，设定的字体一样。

15.3 报警显示与报警条

“报警显示”与“报警条”控件可以用来显示已被定义在“事件登录”中，且系统目前状态满足触发条件的事件，此时这些事件也被称为报警。“报警条”与“报警显示”控件将利用事件被触发的时间先后，依序显示这些报警，其中“报警条”控件将使用单行文字以配以“走马灯”的效果显示所有报警的内容；“报警显示”控件则使用多行文字，各行文字显示单一报警的内容。下图显示不同控件对警示的表示方式。

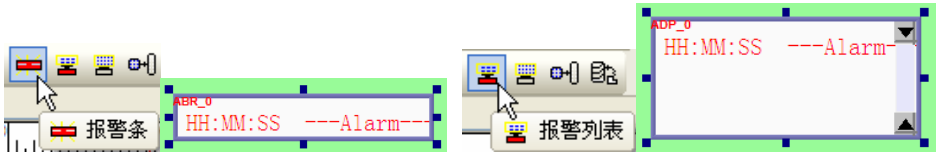


“报警条”控件

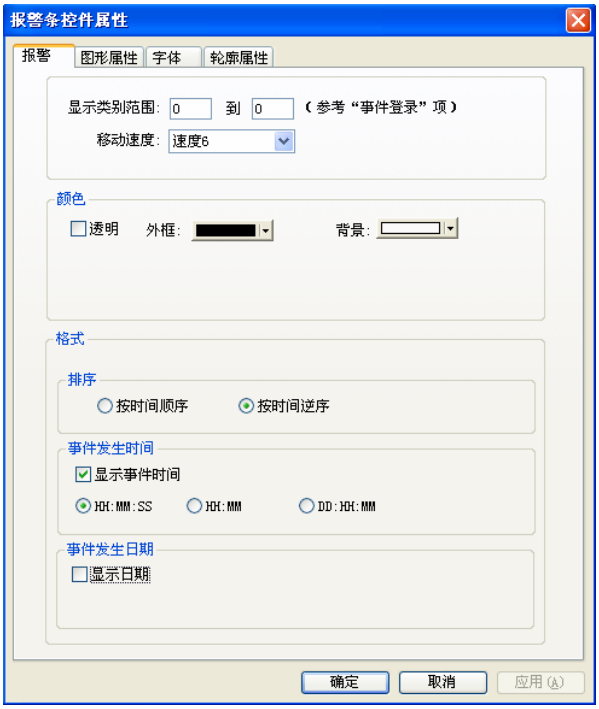


“报警显示”控件

打开 InoTouch Editor 软件菜单的“控件/报警条或报警列表”，或者工具栏上的图标 或 ，在窗口中点击鼠标左键，就建立了“报警条或报警列表”控件，如下图所示。



选择“报警条或报警列表”双击或单击鼠标右键选择“属性”进行编辑，如下图：



[显示的类别范围]

被触发事件的“类别”需符合此处设定的显示范围才会被显示(事件的“类别”在“事件登录”中设定)。例如当“报警条”控件的“类别”此时被设定为 2~4，则仅有“类别”为 2 或 3 或 4 的事件，才会被显示在“报警条”控件中。可以参考“事件登录”说明中关于“类别”的说明。

[移动速度]

“报警条”控件中所显示文字的移动速度。提供“速度 1~速度 10”总共 10 段速度选择，速度 1 移动速度最慢，速度 10 移动速度最快。

[颜色]

设定报警条控件的背景颜色和内部颜色。

[透明]

若勾选此选项，则报警条背景为透明，且不使用任何图片与颜色。

[内部]与[背景]

分别设定内部与背景的颜色

[排序]

设定报警显示的顺序，可以选择“按时间顺序”或“按时间逆序”。

[按时间顺序]

最新发生的报警被排列在后(或在下)。

[按时间逆序]

最新发生的报警被排列在前(或在上)。

时间

[事件发生时间]

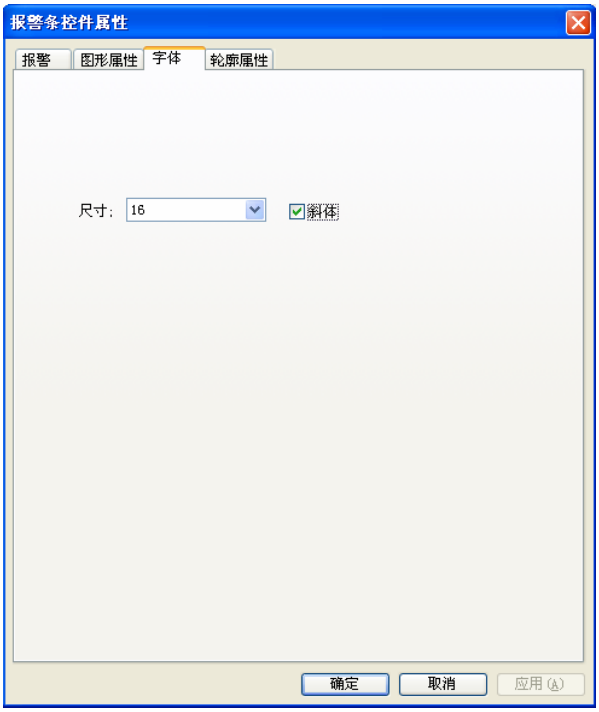
选择控件是否显示报警发生的时间。

日期

[事件发生日期]

选择控件是否显示警示发生的日期。

可以使用“字体”设定对话框设定控件所使用文字的尺寸与是否使用斜体效果，参考下图。各个警示所使用的字型与颜色在“事件登录”中设定。



小结：本章主要介绍了“事件登录”控件，以及利用“事件登录”控件为来源，分别使用“事件显示”、“报警显示”和“报警条”这三个不同的控件，来显示事件/报警信息的。在使用“事件登录”登记需要的事件/报警信息时，需事先要确定各事件/报警信息的类别、触发条件和显示的文字信息以及字体等。同样的，如果需要显示的文字信息为中文等，必须将字体选择为中文字体，例如“宋体”、“楷体”等。否则在事件/报警发生时，文字不能正常被显示出来。

“事件显示”、“报警显示”和“报警条”的区别：

“事件显示”是一个完整的从事件条件满足被触发显示出来，到事件恢复正常，这整个过程都被记录在“事件显示”中。它可以显示“事件”发生的日期、操作人员确认的时间以及“事件”恢复的时间。且在“事件”恢复正常时，“事件”的内容也不会消失，只是文字颜色发生改变而已。而且“事件显示”还可以使用“历史”的方式来查询其他日期发生的“事件记录”。

“报警显示”是在触发条件满足时，显示相应的“报警”信息，也可以显示“报警”发生的具体日期和时间。但是当“报警”恢复时，这些显示的“报警”信息，包括日期和时间又会消失掉。也就是说“报警显示”只能够显示当前发生的“报警”。

“报警条”同“报警显示”一样，只能显示当前发生的“报警”，只能显示单行文字，并使用“走马灯”的效果，让“报警”信息文字滚动显示。“报警”恢复时，文字消失。此控件一般用来作为广告效应的文字介绍。

16

数据和配方资料传送

第十六章　数据和配方资料传送

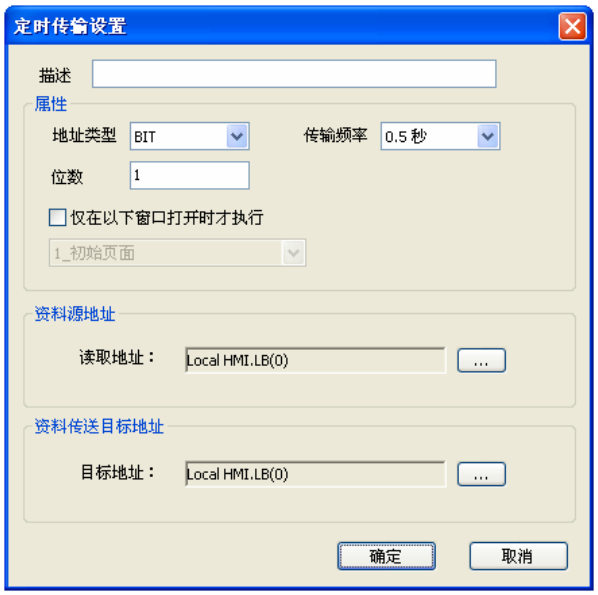
InoTouch Editor 软件提供了两种数据传输的方式，一种是“定时式资料传输”，一种是“触发式资料传输”。“定时式资料传输”是基于时间间隔的连续不断的传送数据的方式。时间间隔从 0.5 秒开始，传输间隔时间是以 0.1 秒来增长的，最大时间间隔为 25.5 秒。“定时式资料传输”可以传送 bit 型资料，也可以传送 word 型资料。每次传送的数据最大为 16 个 word 或者 256 个 bit。传送 bit 型数据时，只能将连接的 PLC 或者控制器的 bit 传送给本机触摸屏的 bit 地址中，例如 LB，LW-bit 等 bit 型地址中。

“触发式资料传输”是在该控件被“触发”时，会将指定的一块连续寄存器的数据传输到另外一个指定位置的连续的寄存器中，传输的资料格式均为 word 型。利用这个特点，可以作为“配方”数据传送，或者 InoTouch Editor 系列人机界面连接的不同 PLC 或者控制器间的数据交换。

在日常的生产过程中，同一个机器在生产不同的产品时，就会遇到要设定不同的参数。例如，钢板切割机，在切割 A 型钢板时，长度为 1000mm，宽度为 1000mm；切割 B 型钢板时，长度为 2000mm，宽度为 1500mm 等等。刚刚的两组参数，就可以把它们看着是两组“配方”数据。由于对“配方”的要求都是要能够在人机界面掉电的情况下，数据还是能够保存，故一般使用 InoTouch Editor 提供的可以掉电保存的寄存器 RW 和 RW_A 两种寄存器来实现配方的保存和传送。

16.1 建立定时式资料传输

单击 InoTouch Editor 软件左侧项目管理下方“定时式数据传送表”，即可打开“定时式资料传输”对话框，见下图。



[描述]

对建立的“定时式资料传输”物件功能的说明，可以不填。为了方便的看懂程序可以写一些注释性的文字内容。

[属性]

可以设定传输资料的格式是 bit 还是 word 型。资料传输的间隔时间和每次传送多少个 bit 或者多少个 word 型数据。当勾选“仅在以下窗口打开时才执行”选项，并制定一个已经建立的窗口时，表明刚刚定义的资料只在指定的窗口打开时才执行，否则这个资料传送的动作将不会被执行。由于“定时式资料传输”是连续的基于时间间隔不断的在传输资料，所以在一定程度上会占用通讯频宽。当不需要每时每刻要执行资料交换时，选择“仅在以下窗口打开时才执行”这个选项后，会加快通讯速度。

[资料源地址]

定义资料传输的来源，并根据之前选择的资料类型来选择资料来源的“PLC 名称”和“地址类型”、“设备地址”等。

[目标地址]

定义接收这些资料的 PLC 名称和 bit 型地址或者 word 型地址。

单击“确定”按钮后，就会显示刚刚建立的“定时数据传送表”。双击刚刚建立的“定时数据传送表”，可以重新编辑列表中选定的“定时传输设置”。

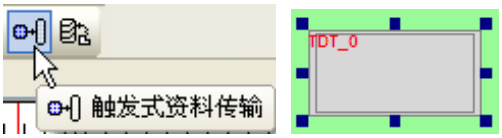
单击“关闭”即关闭打开的“定时式资料传输”对话框。这样，在工程画面运行时，会自动的去执行“资料传输”的动作，而不需要在工程画面窗口中建立这些控件。

16.2 使用触发式资料传输/建立配方资料传输

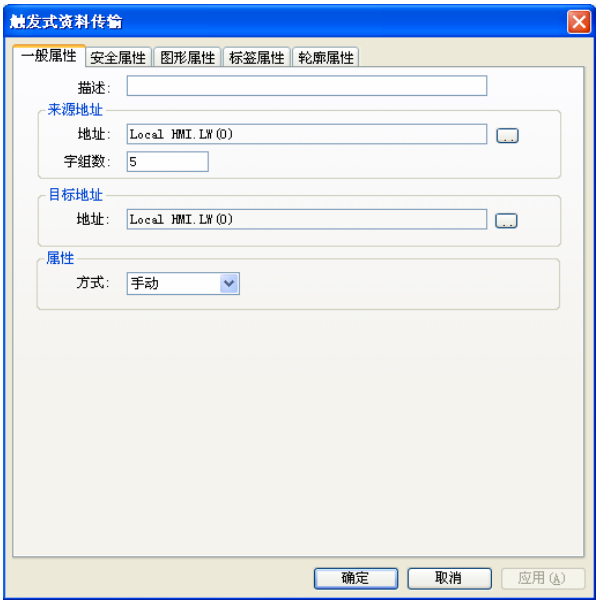
触发式资料传输可以实现本机触摸屏的寄存器数据与连接的控制器的某一个数据区域之间的资料交换，也可以实现人机界面连接的控制器之间的数据交换。但通常，使用这个控件的功能，来实现“配方”的传输。

RW 与 RW_A 地址上的配方资料大小皆为 64K words，使用者可以利用 U 盘、SD 卡或以太网等方式更新配方资料，并利用这些资料更新 PLC 上的数据。使用者也可以利用 U 盘、SD 卡或以太网等方式上传配方数据至指定位置；此外，使用者也可以将 PLC 上的数据保存在配方资料中。下面来说明如何使用“触发式资料传输”控件来制作“配方”传输。

打开 InoTouch Editor 软件菜单的“控件/触发式资料传输”，或者工具栏上的图标  ，在窗口中点击鼠标左键，就建立了“触发式资料传输”控件，如下图所示。



选择“触发式资料传输”双击或单击鼠标右键选择“属性”进行编辑，如下图：



[描述]

说明这个配方传送的功能。为了方便读懂程序，可以使用注释性文字说明。

[来源地址]

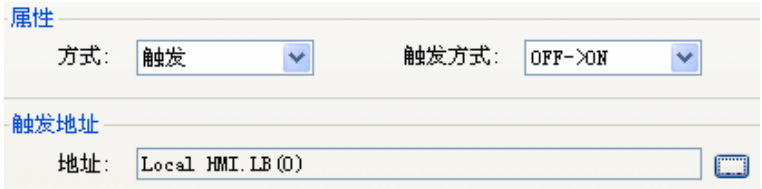
设置配方数据的来源地址和配方传送的个数。如果是将保存在人机界面 RW 地址中的配方数据传输给 PLC，那么在此选定“PLC 名称”为“Local HMI”，“设备类型”为“RW”。如果传送的数据不止一组配方，则勾选使用“索引寄存器”，通过改变“索引寄存器”的内容，来实现多组配方的传输。有关“索引寄存器”的详细使用方法，可以参考《索引寄存器》这章的说明。

[目标地址]

设置接收“配方”数据的地址。一般设置为 PLC 中某一块连续的数据寄存器。设置方法同第 3 项的设置，只是地址选择为设备列表中的某一个 PLC。

[属性]

在此设定配方传输的方式。设置为“手动”时，表示当触控屏幕上的该控件时执行配方数据传输；当传输方式设置为“触发”时，需要设置出“触发”的条件，和执行传输的 bit。如下图所示。



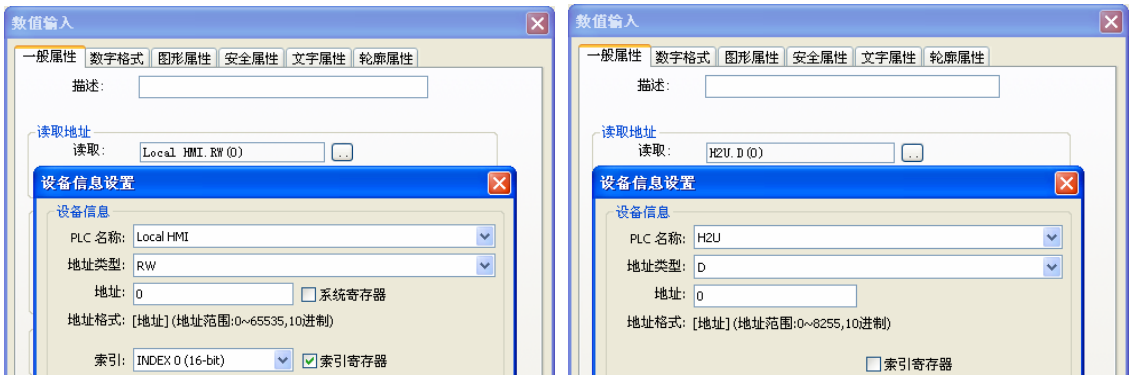
由上图的设置可以看出，当 Local HMI 中的 LB0 状态由 OFF 转为 ON 时执行“配方”数据的传输。

根据需要设定“安全”、“图形”、“标签”、 “轮廓”各分页中的内容。

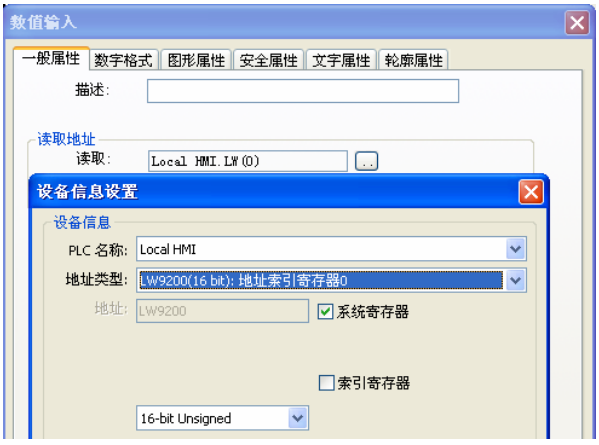
下面以制作一个小的“配方传输”程序为例，说明“配方”传输的功能。

假设连接的 PLC 为汇川 H2U PLC，那么先点击 “添加设备”，如下图所示。

在画面上建立 10 个数据输入控件，设备类型是 Local HMI 的 RW0 开始，并且勾选“索引寄存器”为 INDEX 0。并建立 10 个数值输入控件，设备类型设置为“汇川 H2U”PLC 中的 D0 开始，不勾选“索引寄存器”。如下面两幅图所示。

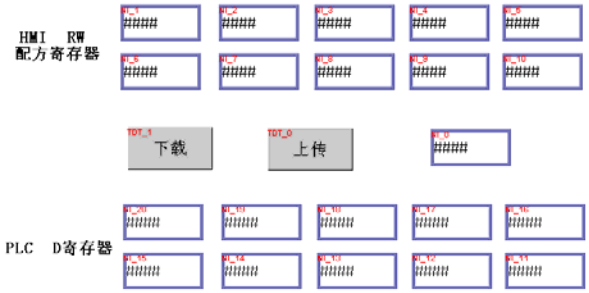


建立一个数值输入控件，选择为 Local HMI 的系统寄存器 INDEX0，如下图所示。

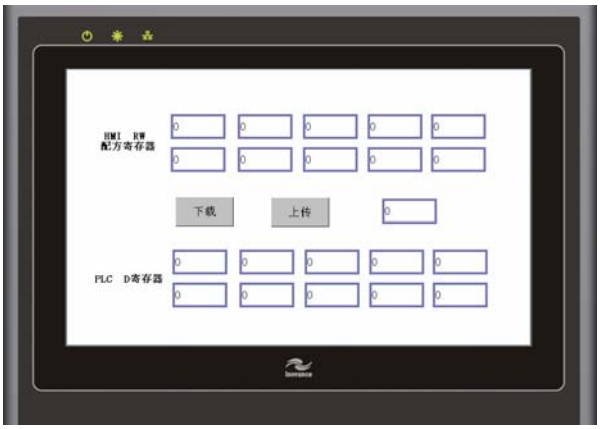


再建立两个“触发式资料传输”控件，一个“来源地址”选择为 Local HMI 的 RW0 开始的连续的 10 个寄存器，且勾选“索引寄存器”为“INDEX0”，“目标地址”为“汇川 H2U” PLC 的 D0 寄存器，目标地址不勾选“索引寄存器”，在该控件的“标签”中设置文字为“下载”。 再建立另外的一个“触发式资料传输”控件，与刚刚设定的控件的“来源地址”和“目标地址”的设定颠倒一下，并在“标签”中设置文字内容为“上传”。两个控件的属性均设置为“手动”。建立后的图形见下图所示。

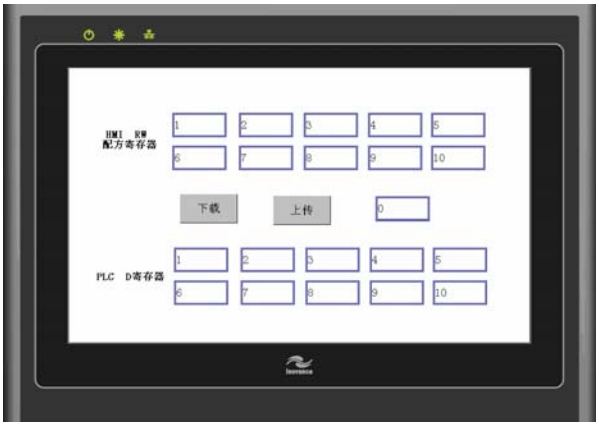
在相应的位置填上文字说明。建立后的程序如下图所示。



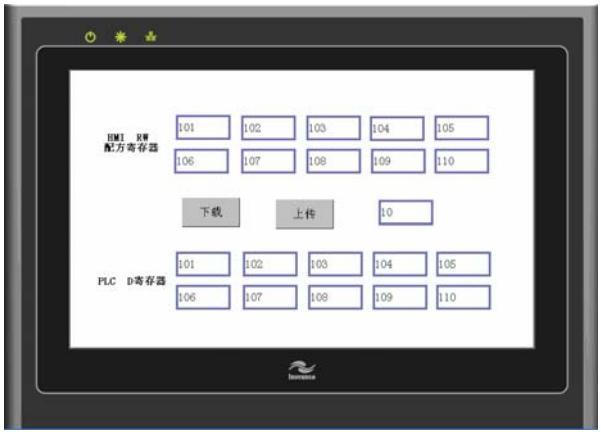
将刚刚建立的程序保存、编译并执行离线仿真后，其效果见下图所示。



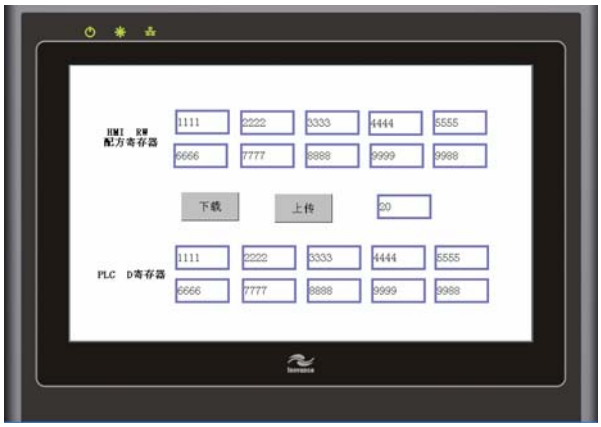
由于此时没有输入任何数据，故数据均为 0。给 RW 配方寄存器分别输入 1~10 共 10 个数据，再按一下“下载”按钮，就会发现，屏幕上边的 RW 寄存器中的数据全部下载到了屏幕下方的 PLC D 寄存器中。如下图所示。



此时的 INDEX0 寄存器的值还是 0，故如果要下一组 RW 连续的 10 个寄存器的数据，即 RW10~RW19 的数据传输到 D0 开始的 10 个寄存器中时，只要将 INDEX0 寄存器中输入 10，然后单击“下载”即可实现。如下图所示。



如果修改 PLC D 寄存器的数据，然后单击上传，就会把寄存器中配方数据传回到 HMI RW 寄存器中来保存。如下图所示，此时 INDEX0=20，表明把 PLC 中新的配方数据保存到 RW20 开始的连续的 10 个寄存器中。



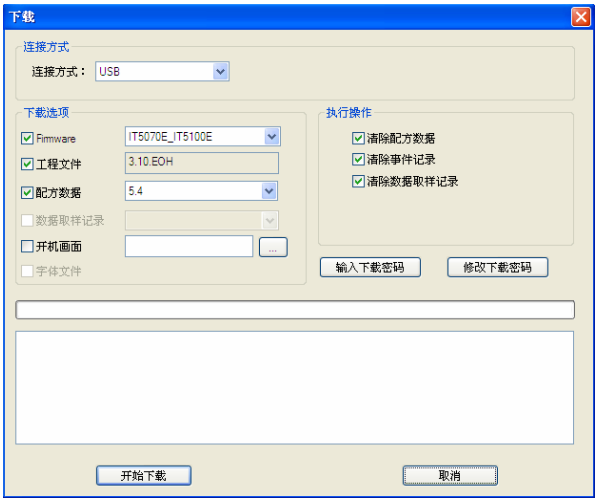
可以灵活的使用系统提供的“索引寄存器”，结合“触发式资料传输”控件，可以很方便的实现“配方数据”的传输和保存。

16.3 InoTouch Editor 人机界面上配方资料更新与保存

16.3.1 使用以太网或者 USB 线更新配方资料

下载时勾选[配方数据]后选择要下载的文件来源。下载成功后自动重新启动 InoTouch Editor 人机界面，将会更新 RW 全部数据。故使用此方法时，请先将修改之前的配方资料上传备份。

如果[清除配方资料]选项被勾选，在进行任何下载动作前，InoTouch Editor 会先将[RW]上的数据内容全部清除为 0。



16.3.2 配方资料强迫储存

为了增加人机界面上 flash 的使用寿命，InoTouch Editor 以每隔 1 分钟的时间间隔将配方资料保存在机器上，为了避免配方资料在两次储存动作间因关机而造成资料的流失，InoTouch Editor 提供[LB9029]让使用者可以自行进行配方资料的储存动作，只需对[LB9029]送出 ON 的讯号，InoTouch Editor 即会执行一次配方资料储存动作。另外如对[LB9028]送出 ON 的讯号，InoTouch Editor 会将所有的配方数据复归为 0。

17

键盘的设计与使用

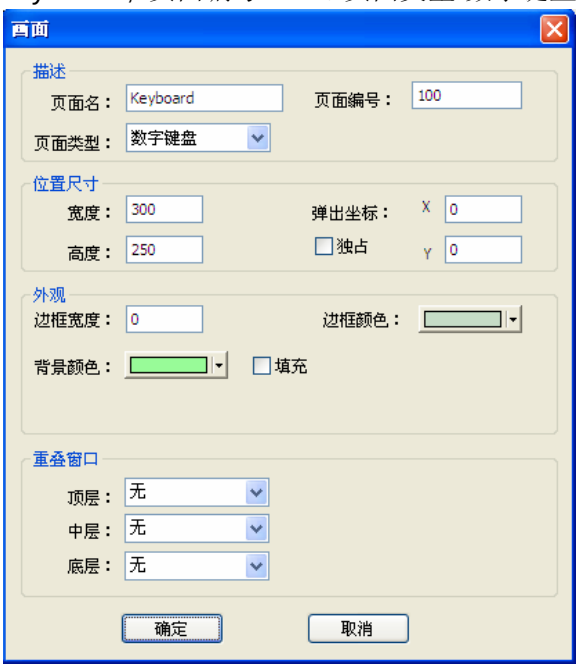
第十七章　键盘的设计与使用

“数值输入”与“字符输入”均需使用键盘做为输入工具，除了可以使用调用键盘的方式,另外还能调用没有移动窗口控制条及直接将键盘设计在屏幕上固定的位置，还可以制作输入汉字输入键盘等应用。数字键盘以及字符键盘均是使用“功能键”控件来制作的，下文说明键盘的设计过程及用法。

17.1 调用自制的键盘

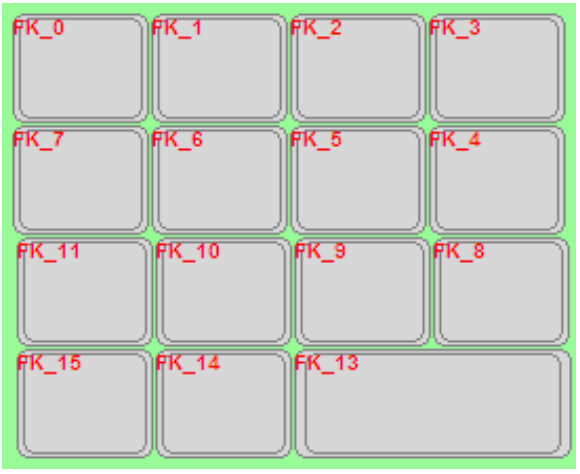
步骤一：新增页面

先建立要作为键盘的窗口，在 InoTouch Editor 软件左侧项目管理空白处点击鼠标右键，选择“添加页面”，设置页面名“Keyboard”，页面编号“100”，页面类型“数字键盘”，宽度“300”，高度“250”。

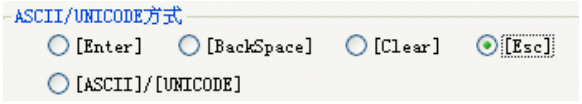


步骤二：使用功能键设置键盘

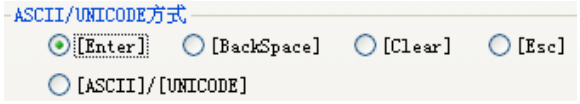
选择“确定”之后，在上面安排各式“功能键”控件，当按下“功能键”控件时将触发各种输入讯号。



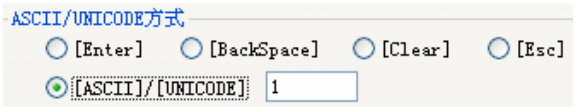
“窗口 100”上的“功能键”控件安排如上图，此时的“功能键”控件均须选择[ASCII 模式]，其中 FK_8 用来触发“取消” (ESC)讯号，FK_8 的部分设定内容如下图。



FK_13 用来触发“输入”(Enter)讯号，FK_13 的部分设定内容如下图。

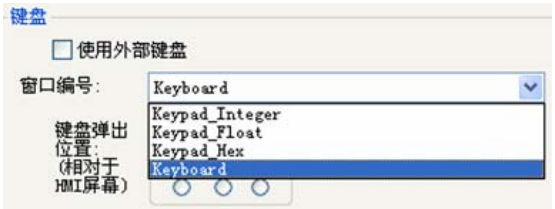


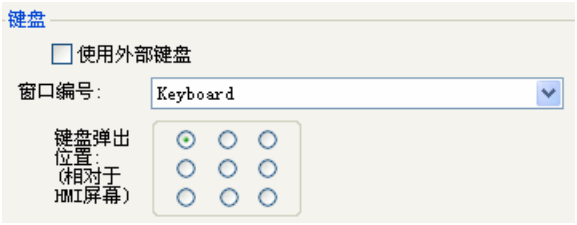
其它大部分“功能键”控件是用来触发数值或文字输入讯号，例如 FK_0 是用来触发数值“1”的输入讯号，FK_0 的部分设定内容如下图。



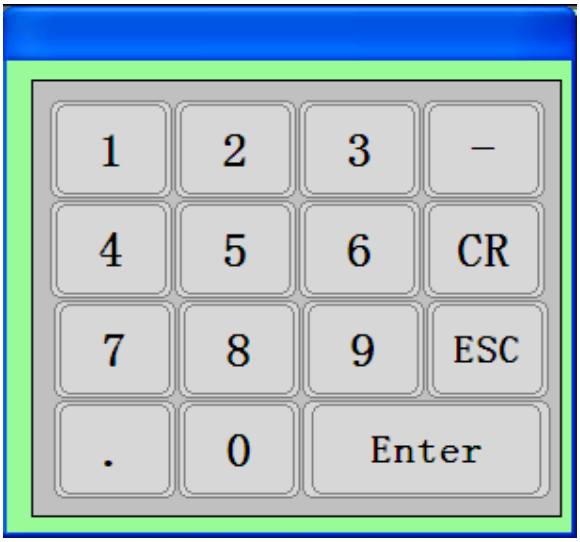
最后为“功能键”控件挑选适合的图形，被置于所有控件的最下层，作为背景图案，如下图。

在完成上述的所有步骤后，当使用者使用“数值输入”或“字符输入”控件的设定页时，即可发现在[键盘]设定项目的[窗口编号]中，增加了“Keyboard” 的选项，如下图所示。下图的[键盘弹出位置]被用来选择键盘的出现位置，InoTouch Editor 将屏幕分为 9 个区域，键盘的左上角将出现在所挑选区域的左上角位置。





在选择“Keyboard”后，当使用者按下“数值输入”或“字符输入”控件时，InoTouch Editor 画面中选定“窗口 100”作为输入键盘的控件，将自动弹出“窗口 100” ，按下“窗口 100”上的 “功能键”控件将等同键盘输入讯号，如下图所示。



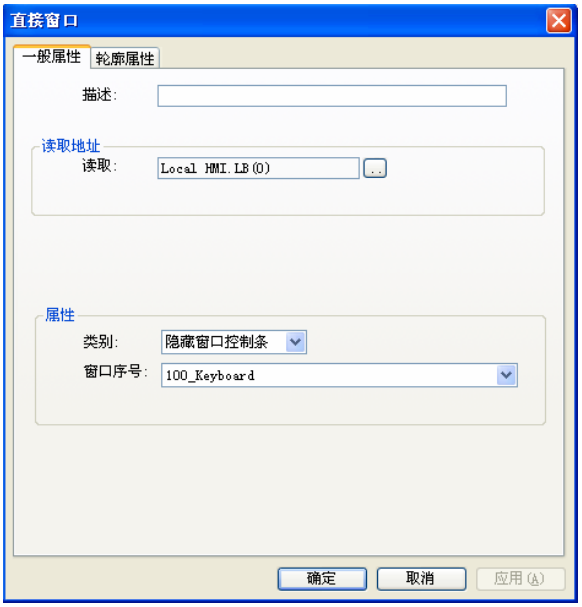
若不想要显示出键盘的窗口控制条，可在屏幕上建立一个直接窗口来使用，请参考以下方式。

17.2 使用直接窗口的方式来调用键盘

步骤一：

新增一个直接窗口，设定读取地址来激活直接窗口。

在属性内选择隐藏窗口控制条及键盘所在窗口序号。

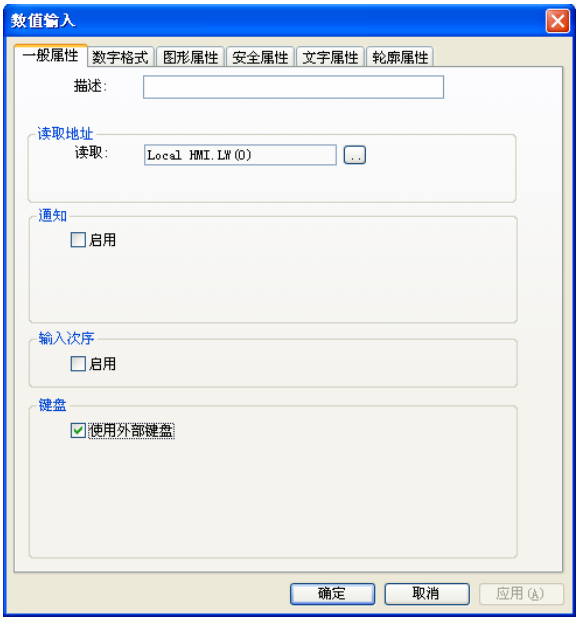


步骤二：

在设定完直接窗口的一般属性后，再次开启设定页,将“轮廓”设定等同键盘大小的尺寸

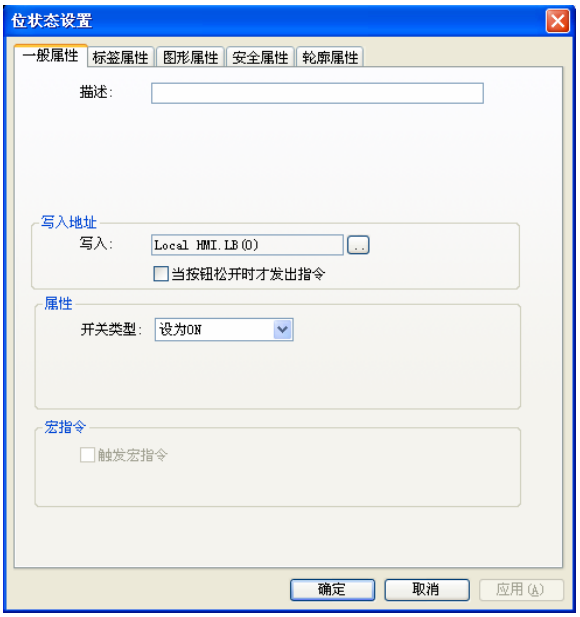
步骤三：

新增数值输入控件，在一般属性内勾选“使用外部键盘”。

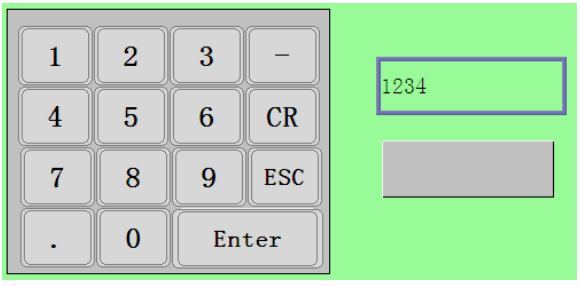


步骤四：

设定一个位状态控件，设为 ON，用于键盘的显示。



将工程保存、编译、离线仿真，如下图所示。



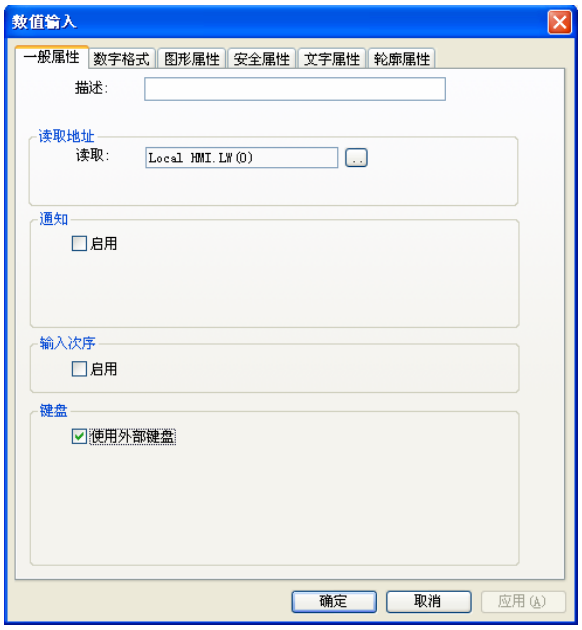
使用直接窗口调用的键盘，键盘所在位置是固定的，无法移动或取消键盘。

17.3 将键盘固定在需要输入的画面上

另外还可以设定将功能键固定在屏幕上而不是采用弹出方式或是使用直接窗口来固定键盘所在位置，采用此方式则无法移动或取消键盘。

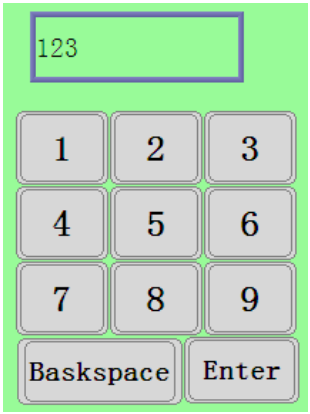
步骤一：

新增数值输入控件，在“键盘”项，勾选“使用外部键盘”。



步骤二：

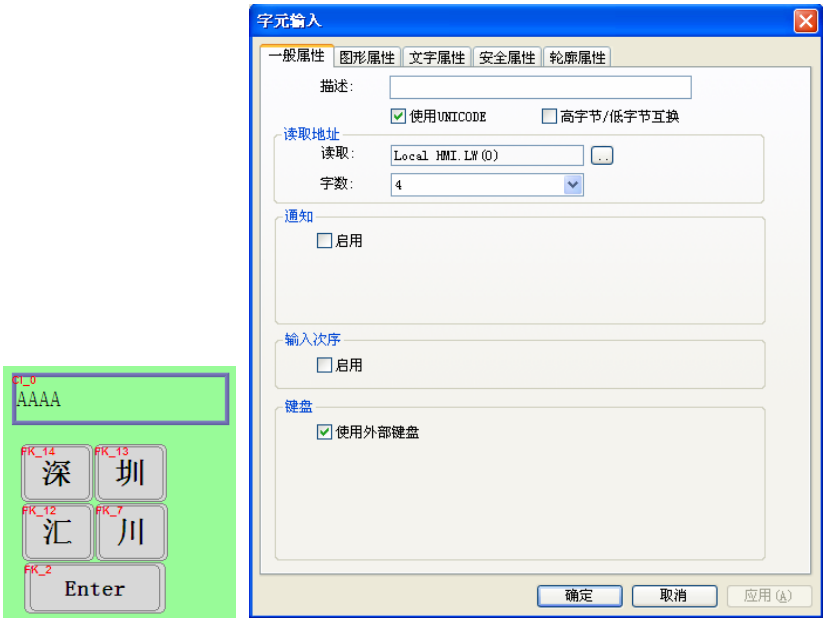
使用功能键将键盘按键设计好后放置于屏幕上即可使用，如下图所示。



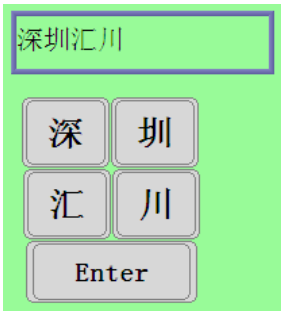
17.4 制作汉字键盘输入汉字

制作汉字键盘与制作数字键盘一样，也是使用功能键来制作的。如下图所示。

经过上面的步骤，制作了“深”“圳”“汇”“川”这四个汉字输入功能键，再制作一个“Enter”输入功能键，即做好了一个简单的汉字键盘。在放置一个“字符输入”控件在画面上，字数选为 4，并勾选“使用 UNICODE”，最后做好的画面如下图所示。



执行离线仿真功能，就可以输入这几个汉字在画面上，如下图所示。



小结：数字键盘和字符键盘均是通过使用“功能键”控件制作，并组合在一起形成的。并可以将自制的键盘群组为“群组图片”添加到“群组图库”中，以便于后续的调用。如果不使用系统预设的键盘，可以将自制的键盘，作为新增的系统键盘。如果在做“数值输入”或者“字符输入”时，不想使用系统键盘，则需在这些控件属性中勾选“使用外部键盘”，这样就可以使用其他方式的键盘，包括外接的 USB 键盘。

18

系统保留寄存器地址和作用

第十八章　系统保留寄存器地址和作用

系统保留寄存器地址分类如下：

一般状态与控制/数值输入状态/配方资料/工作按钮与快选窗口；

事件纪录/资料取样纪录/密码与操作等级/HMI 时间；

HMI 硬件/与远程 HMI 的联机状态/与 PLC 的联机状态；

与本机连接的远程的机器/MODBUS Server 站号/COM 通讯参数更改；

文件管理/PLC&远程 HMI 的 IP 地址设定/远程打印服务器设定；

地址索引功能/本机 HMI 内存地址范围；

18.1 一般状态与控制

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LB-900n	n = 0~9 HMI 激活时，这些位的状态将被初始化为 ON	读/写	读/写	读/写	是
LB-9017	状态为 ON 时，将关闭[PLC 控制] [切换基本窗口]的返回功能	读/写	读/写	读/写	是
LB-9018	鼠标光标控制,隐藏时状态为 ON/显示时状态为 OFF	读/写	读/写	读/写	是
LB-9019	取消/开启声音输出功能	读/写	读/写	读/写	是
LB-9059	作为起始页面的更改寄存器　（见下面备注）	读/写	读/写	读/写	是
LB-9020	显示(设置为 ON)/隐藏(设置为 OFF)系统任务栏	读/写	读/写	读/写	否
LB-9070	Toshiba T/C　写控制位	读/写	读/写	读/写	否
LW-9025	CPU 使用率	读	读	读	否
LW-9050	目前机器上所显示的基本窗口之编号	读	读	读	是

LW-9100~ LW-9115	机器所使用的工程文件名称	读	读	读	否
LW-9116~ LW-9117	工程文件大小(单位 byte)	读	读	读	否
LW-9118~ LW-9119	工程文件大小(单位 K byte)	读	读	读	否
LW-9120~ LW-9121	工程文件所使用的编译器版本号	读	读	读	否
LW-9122	工程文件编译时间(年)	读	读	读	否
LW-9123	工程文件编译时间(月)	读	读	读	否
LW-9124	工程文件编译时间(日)	读	读	读	否
LW-9125	机器所使用的以太网 IP0 地址 (实际地址为 IP0.IP1.IP2.IP3)	读	读	读	否
LW-9126	机器所使用的以太网 IP1 地址	读	读	读	否
LW-9127	机器所使用的以太网 IP2 地址	读	读	读	否
LW-9128	机器所使用的以太网 IP3 地址	读	读	读	否
LW-9129	机器所使用的以太网 gateway 0 (实际地址为 gateway 0. gateway 1. gateway 2. gateway 3)	读	读	读	否
LW-9130	机器所使用的 Ethernet gateway 1	读	读	读	否
LW-9131	机器所使用的 Ethernet gateway 2	读	读	读	否
LW-9132	机器所使用的 Ethernet gateway 3	读	读	读	否
LW-9133	机器所使用的 Ethernet 通讯端口	读	读	读	否
LW-9134	目前所使用的语言(0~7)	读/写	读/写	读/写	是

备注：LW-9059：即作为起始页面的更改寄存器，向该寄存器写入有效的页面编号值，在下一次启动时将会使用新的起始页面，改写该寄存器将会变更EOH文件里的起始页面编号，反编译出来后将会是更改后的值。如果写入值为系统页面或无效页面编号，将视为无效操作。

进入系统设置的条件修改为开机时如果一直按住屏幕不放，在起始页面出现后将会弹出系统设置登录页面，不再使用原来的按住屏幕不放就出现系统设置框。

18.2 数值输入状态

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用

LW-9002~ LW-9003	目前使用中的数值输入组件允许输入的最大数值 数据型态为 32-bit (float)	读	读	读	是
LW-9004~ LW-9005	目前使用中的数值输入组件允许输入的最小数值 数据型态为 32-bit (float)	读	读	读	是
LW-9150~ LW-9181	键盘的输入资料，使用 ASCII 储存，储存的资料长度为 32 words	读	读	读	是
LW-9540	保留作为键盘的 Caps Lock 用途	读	读	读	是

18.3 配方资料

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LB-9010	当下载配方资料时状态为 ON	读	读	读	否
LB-9011	当上传配方资料时状态为 ON	读	读	读	否
LB-9012	当下载/上传配方资料时状态为 ON	读	读	读	否
LB-9028	对此地址送出 ON 的讯号时，全部配方数据将被设定为 0	写	写	写	是
LB-9029	InoTouch 系列 HMI 会每隔 1 分钟将配方数据(RW)存放至机器的内部存储器；对此地址送出 ON 的讯号时，可强迫 InoTouch 系列 HMI 储存配方数据至机器上	写	写	写	是

18.4 工作按钮与快选窗口

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LB-9013	对此地址送出 ON 的讯号时，隐藏快选窗口； 对此地址送出 OFF 的讯号时，显示快选窗口	写	写	写	否
LB-9014	对此地址送出 ON 的讯号时，隐藏工作按钮； 对此地址送出 OFF 的讯号时，显示工作按钮	写	写	写	否
LB-9015	对此地址送出 ON 的讯号时，隐藏快选窗口/工作按钮； 对此地址送出 OFF 的讯号时，显示快选窗口/工作按钮	写	写	写	否

18.5 事件纪录

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LB-9021	对此地址送出 ON 的讯号时，将删除机器上当前系统内存中保存的全部事件纪录	写	写	写	是
LB-9022	对此地址送出 ON 的讯号时，将删除机器上最旧一笔的事件纪录文件(此功能只针对机器上的事件纪录)	写	写	写	是
LB-9023	对此地址送出 ON 的讯号时，将删除机器上全部事件纪录文件(此功能只针对机器上的事件纪录)	写	写	写	是
LB-9024	对此地址送出 ON 的讯号时，将重新计算机器上事件纪录全部文件的大小(此功能只针对机器上的事件纪录有效)	写	写	写	是
LB-9042	确认全部事件,设定为 ON	写	写	写	是
LB-9043	存在未经确认的事件(当状态为 ON)	读	读	读	是
LW-9060	机器上事件纪录文件的个数	读	读	读	是
LW-9061	机器上事件纪录全部文件的大小(32-bit Unsigned)	读	读	读	是

18.6 资料取样纪录

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LB-9025	对此地址送出 ON 的讯号时，将删除机器上最旧一笔的资料取样记录文件(此功能只针对机器上的资料取样纪录)	写	写	写	是
LB-9026	对此地址送出 ON 的讯号时，将删除机器上全部资料取样记录文件，(此功能只针对机器上的资料取样纪录)	写	写	写	是
LB-9027	对此地址送出 ON 的讯号时，将重新计算器器上资料取样纪录全部文件的大小(此功能只针对机器上的资料取样纪录)	写	写	写	否
LW-9063	机器上事件取样纪录文件的个数	写	写	写	否
LW-9064	机器上资料取样纪录全部文件的大小(32-bit Unsigned)	写	写	写	否

18.7 密码与操作等级

地址	说明	读写控制	宏指令	远程HMI控制	汇川是否使用
LB-9050	对此地址送出 ON 的讯号，将强迫使用者注销系统 此时用户仅能操作类别属于“无”的控件	写	写	写	是
LB-9060	当密码输入错误时，状态将被设定为 ON	读	读	读	是
LB-9061	对此地址送出 ON 的讯号时，HMI 将使用[LW9500]到[LW9535]中的内容，更新密码	写	写	写	是
LW-9219	此地址的内容用来判断在[LW9220]中输入的数据，是用户 1 至用户 12 中的何者所输入的密码	读	读	读	是
LW-9220~ LW-9221	密码输入地址(32-bit)	读/写	读/写	读/写	是
LW-9222	指示目前使用者可以操作的组件类别 bit 0 为 1 表示此时的用户可以操作类别为 A 的元件 bit 1 为 1 表示此时的用户可以操作类别为 B 的元件 bit 2 为 1 表示此时的用户可以操作类别为 C 的元件 bit 3 为 1 表示此时的用户可以操作类别为 D 的元件 bit 4 为 1 表示此时的用户可以操作类别为 E 的元件 bit 5 为 1 表示此时的用户可以操作类别为 F 的元件	读	读	读	是
LW-9500~ LW-9501	用户 1 密码	读/写	读/写	读/写	是
LW-9502~ LW-9503	用户 2 密码	读/写	读/写	读/写	是
LW-9504~ LW-9505	用户 3 密码	读/写	读/写	读/写	是
LW-9506~ LW-9507	用户 4 密码	读/写	读/写	读/写	是
LW-9508~	用户 5 密码	读/写	读/写	读/写	是

LW-9509					
LW-9510~ LW-9511	用户 6 密码	读/写	读/写	读/写	是
LW-9512~ LW-9513	用户 7 密码	读/写	读/写	读/写	是
LW-9514~ LW-9515	用户 8 密码	读/写	读/写	读/写	是
LW-9516~ LW-9517	用户 9 密码	读/写	读/写	读/写	是
LW-9518~ LW-9519	用户 10 密码	读/写	读/写	读/写	是
LW-9520~ LW-9521	用户 11 密码	读/写	读/写	读/写	是
LW-9522~ LW-9523	用户 12 密码	读/写	读/写	读/写	是

18.8 HMI 时间

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LW-9010	本地时间(秒，BCD)	读/写	读/写	读/写	是
LW-9011	本地时间(分，BCD)	读/写	读/写	读/写	是
LW-9012	本地时间(时，BCD)	读/写	读/写	读/写	是
LW-9013	本地时间(日，BCD)	读/写	读/写	读/写	是
LW-9014	本地时间(月，BCD)	读/写	读/写	读/写	是
LW-9015	本地时间(年，BCD)	读/写	读/写	读/写	是
LW-9016	本地时间(星期，BCD)	读	读	读	是
LW-9017	本地时间(秒，BIN)	读/写	读/写	读/写	是
LW-9018	本地时间(分，BIN)	读/写	读/写	读/写	是
LW-9019	本地时间(时，BIN)	读/写	读/写	读/写	是
LW-9020	本地时间(日，BIN)	读/写	读/写	读/写	是
LW-9021	本地时间(月，BIN)	读/写	读/写	读/写	是
LW-9022	本地时间(年，BIN)	读/写	读/写	读/写	是
LW-9023	本地时间(星期，BIN)	读	读	读	是
LW-9030~ LW-9031	系统时间(单位为 0.1 秒)，从开机时开始计数	读	读	读	否

18.9 HMI 硬件

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LB-9019	状态为 ON 时，关闭蜂鸣器，状态为 OFF 时，开启蜂鸣器	读/写	读/写	读/写	是
LB-9040	对此地址送出 ON 的讯号时，提高背光灯亮度	写	写	写	是
LB-9041	对此地址送出 ON 的讯号时，降低背光灯亮度	写	写	写	是
LW-9040	背光灯亮度索引值，数值范围从 0 ~31 初次使用机器时，可将背光灯调整为最亮或最暗，则索引值将被设定为 0 或 31，以作为未来调整的基准值	写	写	写	是
LW-9070	HMI 可用空间警示下限	读/写	读/写	读/写	是
LW-9071	系统保留的可用空间	读/写	读/写	读/写	是
LW-9072	HMI 目前的可用空间	读/写	读/写	读/写	否

18.10 与远程 HMI 的联机状态

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LB-910n	n = 0~31 这些暂存器用来指示与远程 HMIn 间的联机状态 状态为 ON 表示目前联机正常；状态为 OFF 表示目前与远程 HMIn 间处在断线状态，此时可以将此状态重设为 ON，系统将尝试与远程 HMIn 再联机一次	读/写	读/写	读/写	是

18.11 与 PLC 的联机状态

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LB-9150	状态为 ON 时，若与连接在 COM 1 的 PLC 断线，系统将自动联机 / 状态为 OFF 时，忽略与此 PLC 的断线状态	读/写	读/写	读/写	是
LB-9151	状态为 ON 时，若与连接在 COM 2 的 PLC 断线，系统将自动联机 / 状态为 OFF 时，忽略与此 PLC 的断线状态	读/写	读/写	读/写	是
LB-9152	状态为 ON 时，若与连接在 COM 3 的 PLC 断线，系统将自动联机 / 状态为 OFF 时，忽略与此 PLC 的断线状态	读/写	读/写	读/写	是

LB-9153~ LB-9184	状态为 ON 时，若与使用以太网接口的 PLCn 断线，系统将自动联机，n = 0~31 状态为 OFF 时，忽略与此 PLC 的断线状态	读/写	读/写	读/写	是
LB-9200~ LB-9455	这些寄存器用来指示与连接在 COM 1 的 PLC 间的联机状态 LB9200 指示与站号为 0 的 PLC 的联机状态，LB9201 指示与站号为 1 的 PLC 的联机状态，依此类推 状态为 ON 表示目前联机正常 状态为 OFF 表示目前与 PLC 为断线状态，此时可以将此状态重设为 ON，系统将尝试与 PLC 再联机一次	读/写	读/写	读/写	是
LB-9500~ LB-9755	这些寄存器用来指示与连接在 COM 2 的 PLC 间的联机状态 LB9500 指示与站号为 0 的 PLC 的联机状态，LB9501 指示与站号为 1 的 PLC 的联机状态，依此类推 状态为 ON 表示目前联机正常 状态为 OFF 表示目前与 PLC 为断线状态，此时可以将此状态重设为 ON，系统将尝试与 PLC 再联机一次	读/写	读/写	读/写	是
LB-9800~ LB-10055	这些寄存器用来指示与连接在 COM 3 的 PLC 间的联机状态 LB9800 指示与站号为 0 的 PLC 的联机状态，LB9801 指示与站号为 1 的 PLC 的联机状态，依此类推 状态为 ON 表示目前联机正常 状态为 OFF 表示目前与 PLC 为断线状态，此时可以将此状态重设为 ON，系统将尝试与 PLC 再联机一次	读/写	读/写	读/写	是
LB-10100~ LB-10131	这些寄存器用来指示与使用以太网接口的 PLC 间的联机状态 状态为 OFF 表示目前与 PLC 为断线状态，此时可以将此状态重设为 ON，系统将尝试与 PLC 再联机一次	读/写	读/写	读/写	是
LW-930n	本地装置所使用的驱动程序编号	读	读	读	否
LW-935n	发送给本地装置且尚未处理的命令数目	读	读	读	否

LW-940n	与本地装置联机时，最新的联机错误内容	读	读	读	否

18.12 与本机连接的远程的机器

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LB-9016	当发现远程装置连接到本机时设置为 ON	读/写	读/写	读/写	否
LW-9006	连接本机的(server)远程装置(client)的数目	读	读	读	否

18.13 MODBUS Server 站号

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LW-9541	当 HMI 作为 MODBUS server 所使用的站号(COM 1)	读/写	读/写	读/写	是
LW-9542	当 HMI 作为 MODBUS server 所使用的站号(COM 2)	读/写	读/写	读/写	是
LW-9543	当 HMI 作为 MODBUS server 所使用的站号(COM 3)	读/写	读/写	读/写	是
LW-9544	当 HMI 作为 MODBUS server 所使用的站号(以太网)	读/写	读/写	读/写	是

18.14 COM 通讯参数更改

下面地址存放各 COM 口所使用的通讯参数，更改这些参数后，重新开机后将使用这些新的参数。

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LB-9030	当 LB9030 由 OFF 变为 ON, 会使用 LW9050~LW9054 的内容立即更改 COM 1 的通讯参数	读/写	读/写	读/写	是
LW-9550	COM 1 通讯模式 0: RS232 2W 1: RS232 4W 2： RS485 2W 3： RS422 4W	读/写	读/写	读/写	是
LW-9551	COM 1 波特率 1~7 分别表示 4800、9600、19200、38400、57600、115200、187500	读/写	读/写	读/写	是

LW-9552	COM 1 数据位 7： 7 位 8： 8 位	读/写	读/写	读/写	是
LW-9553	COM 1 校验 0： 无校验 1： 奇校验 2： 偶校验	读/写	读/写	读/写	是
LW-9554	COM 1 停止位 0： 1 位 1： 2 位	读/写	读/写	读/写	是
LB-9031	当 LB9031 由 OFF 变为 ON, 会使用 LW9056~LW9059 的内容立即更改 COM 2 的通讯参数	读/写	读/写	读/写	是
LW-9556	COM 2 波特率 1~7 分别表示 4800、9600、19200、38400、57600、115200、187500	读/写	读/写	读/写	是
LW-9557	COM 2 数据位 7： 7 位 8： 8 位	读/写	读/写	读/写	是
LW-9558	COM 2 校验 0： 无校验 1： 奇校验 2： 偶校验	读/写	读/写	读/写	是
LW-9559	COM 2 停止位 0： 1 位 1： 2 位	读/写	读/写	读/写	是
LB-9032	当 LB9032 由 OFF 变为 ON, 会使用 LW9060~LW9064 的内容立即更改 COM 3 的通讯参数	读/写	读/写	读/写	是
LW-9560	COM 3 通讯模式 0：RS232 2W 1-RS232 4W 2-RS485 2w 3-RS422 4W	读/写	读/写	读/写	是
LW-9561	COM 3 波特率 1~7 分别表示 4800，9600，19200，38400，57600，115200，187500	读/写	读/写	读/写	否

LW-9562	COM 3 数据位 7： 7 位 8： 8 位	读/写	读/写	读/写	否
LW-9563	COM 3 校验 0： 无校验 1： 奇校验 2： 偶校验	读/写	读/写	读/写	否
LW-9564	COM 3 停止位 0： 1 位 1： 2 位	读/写	读/写	读/写	否

18.15 文件管理

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LB-9034	强迫储存事件纪录与取样数据到 HMI (设定为 ON)	写	写	写	否
LB-9035	HMI 可用空间不足警示 (ON 状态)	读	读	读	是
LB-9036	SD 卡可用空间不足警示 (ON 状态)	读	读	读	是
LB-9037	U 盘可用空间不足警示 (ON 状态)	读	读	读	是
LB-9038	USB 2 可用空间不足警示 (ON 状态)	读	读	读	否
LB-9039	文件备份动作状态(备份中状态为 ON)	读	读	读	是
LW-9074	CF 目前的可用空间	读	读	读	否
LW-9076	USB1 目前的可用空间	读	读	读	否
LW-9078	USB2 目前的可用空间	读	读	读	否

18.16 PLC & 远程 HMI 的 IP address 设定

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LW-9600~ LW-9629	PLC 的 IP 地址及连接端口设定 (IP0:IP1:IP2:IP3)	读/写	读/写	读/写	否
LW-9800~ LW-9839	远程 HMI 的 IP 地址及连接端口设定 (IP0:IP1:IP2:IP3)	读/写	读/写	读/写	否

18.17 远程打印服务器设定

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LW-9770~	远程打印服务器的 IP 设定	读/写	读/写	读/写	否

LW-9773	(IP0:IP1:IP2:IP3)				
LW-9774	登入打印服务器所需的使用者名称	读/写	读/写	读/写	否
LW-9780	登入打印服务器所需的密码	读/写	读/写	读/写	否

18.18 地址索引功能

地址	说明	读写控制	宏指令	远程 HMI 控制	汇川是否使用
LW-9200~LW-9260	地址索引寄存器	读/写	读/写	读/写	是

18.19 本机 HMI 内存地址范围

18.19.1 位地址

内存	设备类型	地址范围	地址格式
本机位地址寄存器	LB	0~11999	AAAAA
本机字地址寄存器取位寄存器	LW_BIT	0~11999	AAAAABB AAAAA:字地址 BB: 位地址 (00~15) 举例: 56712 字地址 = 567 位地址 = 12 即表示 LW567 寄存器的 bit12 这个位。
保持寄存器(RW)的位地址	RW_Bit	0~65535	AAAAAB AAAAA:字地址 B:位地址 (0~f) 举例: 567a 字地址 = 567 位地址 = a
保持寄存器(RW_A)的位地址	RW_A_Bit	0~65535	AAAAAB AAAAA:字地址 B:位地址 (0~f) 举例: 567a 字地址 = 567 位地址= a

18.19.2 字地址

内存	设备类型	地址范围	地址格式
本机字地址	LW	0~11999	AAAAA

			AAAAA:字地址
本机保持寄存器 RW	RW	0~65535	AAAAA AAAAA:字地址
本机保持寄存器 RW_A	RW_A	0~65535	AAAAA AAAAA:地址

19

以太网络通讯与多台 InoTouch 系列 HMI 互联

第十九章　以太网络通讯与多台 InoTouch 系列 HMI 互联

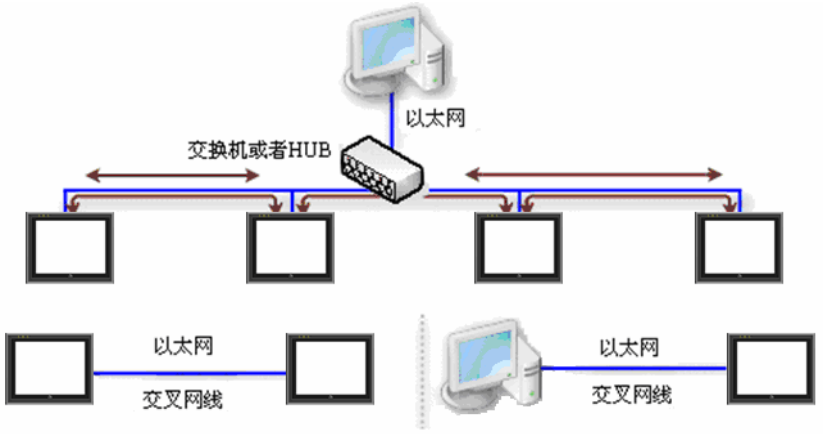
以太网通讯是 InoTouch 系列人机界面的一个特点，通过以太网连接，可以实现多台 InoTouch 系列人机界面的互联，计算机与 InoTouch 系列人机界面的连接和一台人机界面控制另外一台人机界面所连接的 PLC 或者控制器等。即连接方式为以下三种：

a、HMI 与 MHI 间的通讯

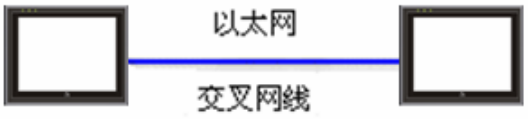
b、计算机与 HMI 间的通讯

c、HMI 控制连接在其它 HMI 上的 PLC

一般来说，以太网络连接的方式有两种：可以使用 RJ45 直连线(straight through cable)，搭配集线器(hub)使用；另一种是使用 RJ45 交叉线(crossover cable)，不需使用集线器，但只限使用在一对一联机的情况下(HMI 对 HMI，或计算机对 HMI)。下面说明各项联机方式的设定与操作。



19.1 HMI 与 HMI 间的通讯



不同的人机界面间通过以太网络可以互相读写对方的数据，利用系统保留寄存器(LB 与 LW)，一台人机界面可以控制另一台人机界面的行为表现。一台 HMI 最多可以同时处理来自 32 个不同 HMI 的访问要求。

以两台 HMI 的通讯为例 (HMI A 与 HMI B) 为例，假使 HMI A 欲使用“位状态设定”控件控制 HMI B 的[LB123]地址的内容，则使用在 HMI A 上的工程文件(afs)设定步骤如下：

步骤一：

设定各台 HMI 的 IP(详情请参考相关章节)，假设 HMI A 与 HMI B 的 IP 已分别设定为“192.168.60.201”与“192.168.60.202” 。

步骤二：

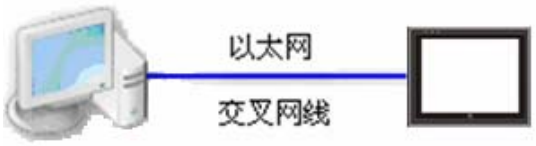
在 InoTouch Editor 的左侧[项目管理]中的[通讯连接]远端设备的“Ethernet”处，点击鼠标右键，选择“添加设备”，增加一台远程 HMI (HMI B)。下图为 HMI B 的设定内容。

步骤三：

在“位状态设定”控件设定页的[PLC 名称]中，选择“HMI B”，此时 HMI A 即可操作 HMI B 的 LB 地址之内容。

同样的，在 HMI B 中也可以建立一个远程的 HMI，将 IP 地址设置为 HMI A 的 IP 地址和端口号，这样，在 HMI B 上，也可以控制 HMI A 上的地址和数据了。

19.2 计算机与 HMI 间的通讯



利用 InoTouch Editor 的在线仿真功能，计算机可以通过以太网络获取 HMI 上的数据，并可以将这些数据保存在计算机上。

计算机也可以利用控制 HMI 上的系统保留寄存器(LB 与 LW)，直接控制 HMI。相对的，HMI 也可以直接控制计算机的行为表现，例如要求计算机储存 HMI 或 PLC 上的数据。

一台计算机可以控制不限数目的 HMI。

假使计算机欲通讯的对象为与两台 HMI (HMI A 与 HMI B)，则计算机端所使用 afs 文件的设定步骤如下：

步骤一：

设定各 HMI 的 IP(详情请参考相关章节)，假使 HMI A 与 HMI B 的 IP 已分别设定为“192.168.60.201”与“192.168.60.202” 。

步骤二：

在 InoTouch Editor 的左侧[项目管理]中的[通讯连接]远端设备的“Ethernet”处，点击鼠标右键，选择“添加设备”，增加 HMI A 与 HMI B。

步骤三：

选择正确的[PLC 名称]。在控件的设定页中，正确选择要操作的装置，例如要控制 HMI A 上的 LB，则[PLC 名称]需选择“HMI A” ，如下图。

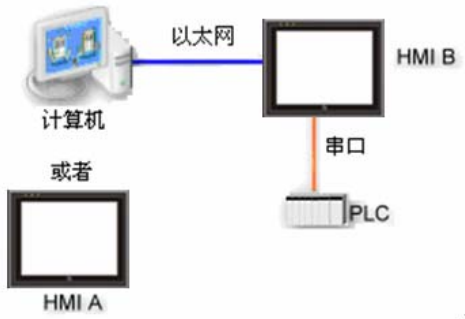
步骤四：

在计算机端利用这个 afs 文件，使用 InoTouch Editor 执行在线仿真功能(离线或在线模式皆可)，即可在计算机端操作 HMI A 与 HMI B 上的所有数据。

程序设定好之后，也可以制作一个快捷方式在计算机桌面上，或者放置到计算机的启动项里面，这样可以快速的执行这个程序。

HMI 也允许操作计算机上的数据，此时只需将计算机视为另一台 HMI 即可，也就是必须在 HMI 使用的工程文件中新增一台 HMI，并将此 HMI 的 IP 指向计算机即可。设定方法是一样，在此就不做详述，使用者可以自行比照该方法来测试。

19.3 控制连接在其它 HMI 上的 PLC



透过以太网络，计算机与 HMI 可以操作连接在其它 HMI 上的 PLC；举例来说，假设现有一台汇川 PLC 连接到 HMI B 的 COM 1 口，当计算机或 HMI A 欲读取此台 PLC 上的数据，则计算机端或 HMI A 上所使用工程文件设定步骤如下：

步骤一：

设定 HMI B 的 IP，假设 HMI B 的 IP 已设定为“192.168.60.202”。

步骤二：

在 InoTouch Editor 的左侧[项目管理]中的[通讯连接]远端设备的“Ethernet”处，点击鼠标右键，选择“添加设备”，增加一台 PLC 的定义，并正确设定通讯参数。

由上图可以发现，此时新增的 PLC 的所在位置为“远端”。因此台远程 PLC 是连接在 HMI B 上，所以设定的 IP 为 HMI B 的 IP (192.168.60.202)。

步骤三：

假设要使用“位状态设定” 控件控制 HMI B 上的汇川 PLC，则只需将控件设定页中的[PLC 名称]挑选为“PLC on HMI B”，即可在计算机使用在线仿真的方式，控制连接在远程 HMI B 上的 PLC 了。

同样的，如果在 HMI A 上连接有一台或者多台 PLC，如果 HMI B 要访问 HMI A 上的这些 PLC，均是通过此种方法来连接，在此就不多做详述，使用者可以比照此设定来制作程序。

小结：以上的说明内容，了解了 InoTouch Editor 软件通过以太网实现 HMI 与 HMI 之间、计算机与 HMI 之间，HMI 或者计算机与另外一台 HMI 连接的 PLC 之间等的连接方法。由此可以看出，通过以太网，轻松的实现了“一机多屏”的连接方式。在一台 HMI 或者一台计算机访问另外一台 HMI 所连接的 PLC 或者控制器时，该 HMI 或者计算机与另外一台 HMI 之间互为“远端”，故必须建立“远端”PLC，且 IP 地址和端口号为“远端”HMI 的 IP 地址和端口号。计算机在此种连接方式下执行的在线仿真功能，是没有时间限制的。

注：汇川所有 HMI 出厂默认的 IP 地址为：192.168.60.201；如需通讯，请修改另一台 HMI 的 IP 地址。

上面的说明内容均是以局域网来说明的。其实，通过 Internet，也同样的可以访问“远端”HMI 上所连接的 PLC 或者控制器。此时，需要在“远端”HMI 的当地路由器中作 IP 地址映射。假设远端 HMI B 的 IP 地址设定为“192.168.60.202”，端口号为 10086，那么在该 HMI B 所连接的路由器中需要设定，将“192.168.60.202”这个 IP 地址映射到 10086 这个端口上。

此时，假设 HMI B 接入外部 Internet 的 IP 地址(此 IP 地址由网络运营商提供)为 12.34.56.78，那么在 HMI A 的设备列表中，建立的“远端”HMI 的 IP 地址则需设置为“12.34.56.78”，端口号还是设定为 10086。

这样，HMI A 就是访问远端的 IP 地址 12.34.56.78，和端口号 10086。由于访问的是外网 IP，且设置了端口号为 10086，这样当地的路由器就会自动的去找 10086 这个端口号对应的 IP 地址的设备，这样就找到了 IP 地址为 192.168.60.202 这个设备，即 HMI B。

此种方法在 HMI A 访问 HMI B 上连接的设备时，与此设定相同，就不再累述。

20

如何将 InoTouch 系列 HMI 设定成 MODBUS 从站

第二十章　如何将 InoTouch 系列 HMI 设定成 MODBUS 从站

InoTouch 系列人机界面使用 MODBUS 协议跟 PLC 或者控制器连接的时候，一般都是作为主站。实际上 InoTouch 系列人机界面也是可以设置为 MODBUS 从站的，这样，其他做 MODBUS 主站的设备，就可以使用 MODBUS 协议读写 InoTouch 人机界面上的数据。

上图显示 InoTouch 被设定成 MODBUS 从站(又称为 MODBUS Server)，InoTouch 人机界面、计算机或其它设备只需使用 MODBUS 协议，使用 RS232/485 串口，即可读写 InoTouch 上的数据。下面将说明如何将 InoTouch 设定成 MODBUS 从站，并说明读写 InoTouch 数据的方式。

20.1 增加设定一个 MODBUS Server 设备

要将 InoTouch 设定为 MODBUS 设备，首先需在 InoTouch 使用的 AFS 程序的设备列表中增加一个新的设备，此时 PLC 种类需挑选“MODBUS Server”，PLC 接口可以挑选 RS232、RS485 2W、RS485 4W 等方式。

当接口挑选使用 RS232 或 RS485 时，需选择使用的连接端口(COM 1 ~COM 3)，并设定正确的通讯参数，参考下图，此时 MODBUS Server 的站号设定为 1。

在按下“确定”键后，此时即完成了 MODBUS 设备的设定，在完成 afs 档案的编译并将获得的 EOH 档案下载到 HMI 后，即可透过 MODBUS 协议即可读写 InoTouch 上的数据。

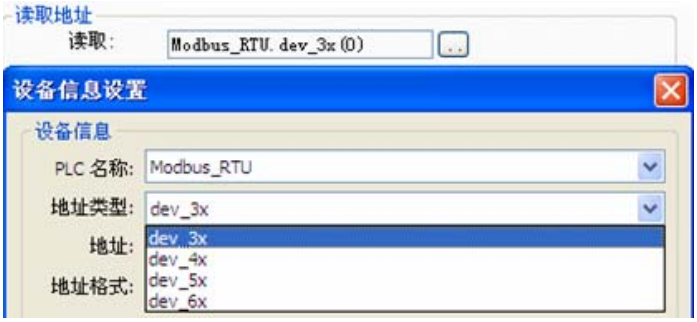
20.2 如何读写一个 MODBUS Server 设备

一台 InoTouch (又称为主站)透过 MODBUS 协议可以读写另一台已设定为 MODBUS 从站的 InoTouch (又称为 server 端)。

首先在主站所使用的 afs 档案的设备清单中，需增加一个新的设备，若主站要使用 RS232/485 串口，则 PLC 种类需挑选“MODBUS RTU”，并正确设定各项通讯参数。

完成各项设定并按下“确定”键后，即可在设备列表中发现一个新的设备：“MODBUS_RTU”。

打开各个对象的设定页，在 PLC 名称中选择“MODBUS_RTU”后，即可发现可以设定 MODBUS 设备的各项读写地址。



此时因被读写的从站(server 端)为 HMI，所以实际读写的位置的对应关系如下：

读写 0x/1x(0~12000)　　对应到　读写 LB(0~11999)

读写 3x/4x/5x(0~10000)　　对应到　读写 LW(0~9999)

读写 3x/4x/5x(10000~65535)　对应到 读写 RW(0~55535)

20.3 如何在线更改 MODBUS Server 的站号

InoTouch Editor 提供下列系统保留寄存器，让使用者可以在线更改 MODBUS Server 所使用的站号。

[LW9541]　MODBUS server 站号 (COM 1)

[LW9542]　MODBUS server 站号 (COM 2)

[LW9543]　MODBUS server 站号 (COM 3)

[LW9544]　MODBUS server 站号 (以太网)

修改这些参数的数据，即可修改 MODBUS Server 设备的站号。

20.4 关于 MODBUS 各地址的说明

InoTouch Editor 软件中 MODBUS 协议中设备类型为 0x，1x，3x，4x，5x，6x，还有 4x_bit，3x_bit 等，下面分别说明这些设备类型在 MODBUS 协议中支持哪些功能码。

0x： 是一个可读可写的设备类型，相当于操作 PLC 的输出点。该设备类型读位状态的时候，发出的功能码为 01H，写位状态的时候发出的功能码为 05H。

1x：是一个只读的设备类型，相当于读 PLC 的输入点。读位状态的时候发出的功能码为 02H。

3x： 是一个只读的设备类型，相当于读 PLC 的模拟量。读数据的时候，发出的功能码为 04H。

4x：是一个可读可写的设备类型，相当于操作 PLC 的数据寄存器。当读数据的时候，发出的功能码是 03H，当写数据的时候发出的功能码是 10H。

5x： 该设备类型与 4x 的设备类型属性是一样的。即发出读写的功能码完全一样。不同之处在于，当为双字时，例如 32_bit unsigned 格式的数据，使用 5x 和 4x 两种设备类型分别读取数据时，高字和低字的位置是颠倒的。例如，使用 4x 设备类型读到的数据是 0x1234，那么使用 5x 设备类型读取的数据是 0x3412。

6x： 是一个可读可写的设备类型，读数据的时候发出的功能码也是 03H，与 4x 不同之处在于写数据的时候，发出的功能码为 06H，即写单个寄存器的数据。

3x_bit： 该设备类型支持的功能码与 3x 设备类型完全一致，不同之处是，3x 是读数据，而 3x_bit 是读数据中的某一个 bit 的状态。

4x_bit： 该设备类型支持的功能码与 4x 设备类型完全一致，不同之处是，4x 是读数据，而 4x_bit 是读数据中的某一个 bit 的状态。

21

宏指令说明

第二十一章 宏指令说明

宏指令提供了应用程序之外附加的你所需要的功能。在 InoTouch 系列人机界面运行时，宏指令可以自动的执行这些命令。它可以担负执行例如复杂的运算、字符串处理，和使用者与工程之间的交流等功能。本章主要介绍如何使用和编程方法等功能。

21.1 怎样建立和执行宏指令

21.1.1 怎样建立一个宏指令

按照以下步骤可以建立一个宏指令。

步骤一：

打开InoTouch Editor 软件菜单的“工具/宏指令…”，如下图所示。

[新增]

新增一个宏指令，弹出一个对话框，编写宏指令的名称，点击“确认”之后，新的宏指令编辑区，就建好了。

[删除]

删除选中的宏指令，但是正在使用的宏指令不能被删除。

[添加函数]

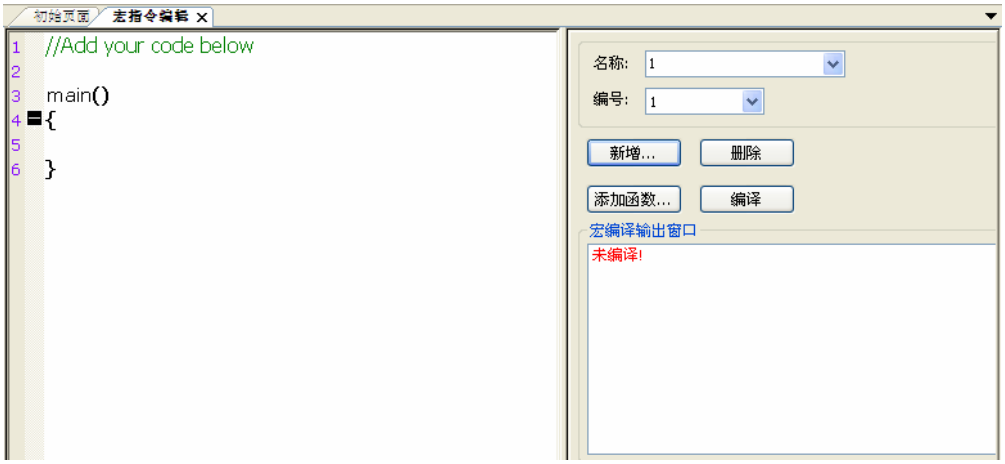
点击“添加函数”，可以选择您需要的函数，设定必要的参数。

[编译]

宏指令编写完成之后，可以点击“编译”，检查语法是否正确。编译成功的则“宏编译输出窗口”显示“编译完成”，未编译的会显示“未编译”，存在语法错误时双击错误列表可定位出错行。

步骤二：

按下“新增”按钮，新增一个宏指令编辑区。每一个宏指令都有一个唯一的编号，定义在“编号”这个位置。在“宏指令名称”中也必须输入宏指令的名称，否则无法新增宏指令。

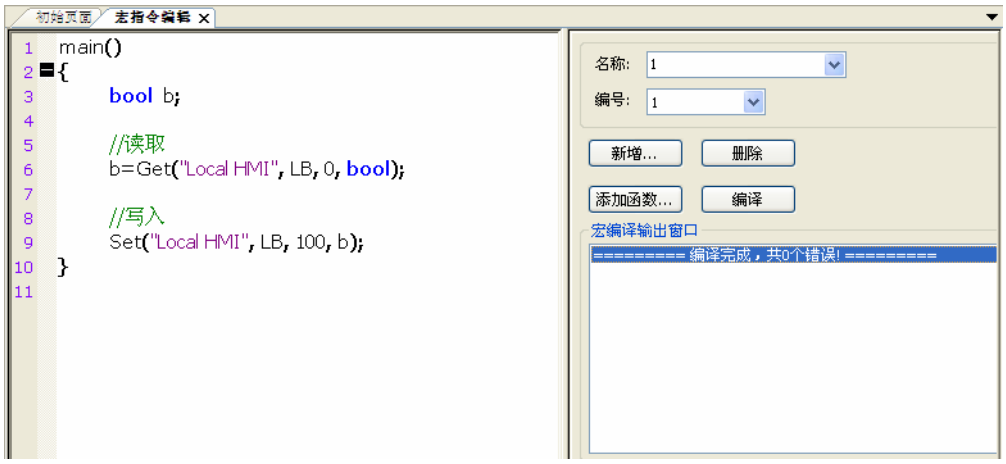


步骤三：

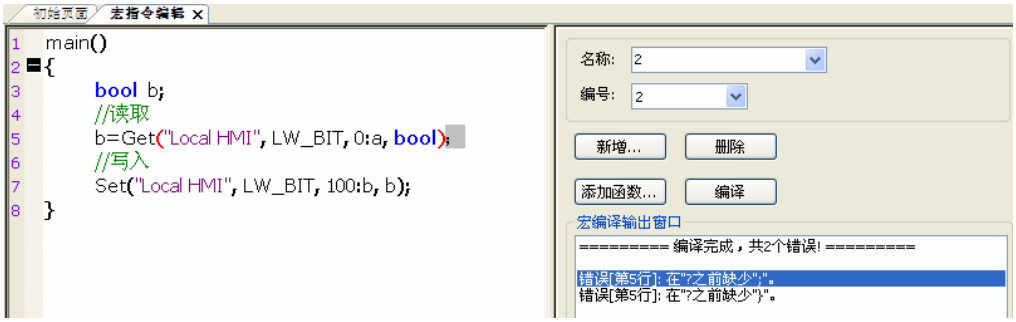
编写您的宏指令程序。如果有需要的话，可以使用函数，例如 Sqrt(x)或者 Sin(x)等函数。单击“添加函数...”弹出一个函数列表对话框，选择需要的函数，并设定必要的参数。

步骤四：

编写完成一个新建的宏指令程序后，单击“编译”按键，对该宏指令进行编译工作。编译完成，没有错误，这样一个宏指令程序就编写完成了。有错误时，双击错误列表，可定位出错误行。



提示：请特别注意标点符号，一定使用英文模式下的标点，否则会出错，如下图所示：



21.1.2 执行宏指令

执行宏指令有多种不同的方法，下面分别说明：

宏指令程序设置点：

控件名称	描述
PLC控制	可以在“PLC控制”项目中按如下方式使用宏指令： 1） 打开“PLC控制对象表”，选择“新增PLC控制项”，选择“控制类型”为“执行宏指令”。 2） 在“控制属性”选择需要执行的宏指令名称；选择某个位地址作为宏指令触发控制位并设定触发条件。在条件满足时，该宏指令将会被执行。 3） 用户还可以选择“只在指定窗口打开时才执行”。
位状态设置	可以在“位状态设置”控件中按如下方式使用宏指令： 1） 在“一般属性”页“触发宏指令”中，勾“触发宏指令”。 2） 选择要执行的宏指令。当触发条件满足时，选择的宏指令就会被执行一次。

位状态切换	可以在“位状态切换”控件中按如下方式使用宏指令： 1） 在“一般属性”页“触发宏指令”中，勾“触发宏指令”。 2） 选择要执行的宏指令。当触发条件满足时，选择的宏指令就会被执行一次。
功能键	可以在“功能键”控件中按如下方式使用宏指令： 1）在“一般属性”页“触发宏指令”中，勾“触发宏指令”。 2）选择要执行的宏指令。每按一下这个功能键时，选中的宏指令就会被执行一次。

组合运用：

方法：运用各种控件之间的关联，导通位状态的设置和PLC控制信息来达到触发宏指令的功能。

常见的组合有：

1） 运用数值输入控件+PLC控制。（使用数值输入控件的通知地址）

2） 运用报警+PLC控制。（事件注册表也有通知触发地址）

启动宏指令　当窗口打开时触发宏指令：

1）使用位状态设置控件-》开关类型为：当窗口打开时为ON。

2）使用位状态设置控件-》开关类型为：当窗口打开时为OFF。

退出宏指令　当窗口关闭时触发宏指令：

1）使用位状态设置控件-》开关类型为：当窗口关闭时为ON。

2）使用位状态设置控件-》开关类型为：当窗口关闭时为OFF。

循环宏指令　按照设定的时间循环运行

1）使用位状态设置控件-》开关类型为：周期切换开关，可设置周期时间。

2）使用一般触发方式，并在宏指令中，用while 或者for命令，配合delay运行宏指令。

事件宏指令　当触发条件满足时触发宏指令

1）使用事件注册表内的报警信息，当事件满足条件时通知触发地址配合PLC控制触发，常见的事件触发为 ON , OFF , ON->OFF , OFF->ON , == , > , < , >= , <= , <>。

2）数值输入控件写入后触发宏指令，数值输入控件-》通知位触发配合PLC控制实现。

21.2 函数功能

InoTouch Editor 软件宏指令中提供了一些函数，用来从 PLC 获取数据和传输数据到 PLC、数据处理和数学运算等。

21.2.1 数学运算函数

1）开平方根运算

函数名	Sqrt
函数签名	Sqrt(x);
功能描述	开平方根，数据来源必须为一个正数。

用法举例	main()
	{
	float x,result;
	result = Sqrt(15);
	x = 9.0;
	result = Sqrt(x); // 执行后 result = 3.0
	}

2）正弦运算

函数名	Sin
函数签名	Sin(x);
功能描述	三角函数的正弦计算。

用法举例	main()
	{
	float x, result;
	result = Sin(90); // 执行后 result = 1.0
	x = 30;
	result = Sin(x); // 执行后 result = 0.5
	}

3）余弦运算

函数名	Cos
函数签名	Cos(x);
功能描述	三角函数的余弦计算。

用法举例	main()
	{
	float x, result;
	result = Cos(90); // 执行后 result = 0

	x = 60;
	result = Cos(x);　//　执行后 result = 0.5
	}

4）正切运算

函数名	Tan
函数签名	Tan (x);
功能描述	三角函数的正切计算。

用法举例	main()
	{
	float　x, result;
	result = Tan(45);　//　执行后 result = 1.0
	x = 60;
	result = Tan(x);　//　执行后 result = 1.732
	}

5）余切运算

函数名	Cot
函数签名	Cot(x);
功能描述	三角函数的余切计算。

用法举例	main()
	{
	float　x, result;
	result = Cot(45);　//　执行后 result = 1.0
	x = 60;
	result = Cot(x);　//　执行后 result = 0.5774
	}

6）正割运算

函数名	Sec
函数签名	Sec(x);
功能描述	反三角函数中反正割计算。

用法举例	main()
	{
	float　x, result;
	result = Sec(45);　//　执行后 result = 1.414
	x = 60;

	result = Sec(x);　//　执行后 result = 2.0 }

7）余割运算

函数名	Csc
函数签名	Csc(x);
功能描述	反三角函数中反余割计算。

用法举例	main()
	{
	float　x, result;
	result = Csc(45);　//　执行后 result = 1.414
	x = 30;
	result = Csc(x);　//　执行后 result = 2.0
	}

8）反正弦运算

函数名	Asin
函数签名	Asin(x);
功能描述	反三角函数中反正弦计算。

用法举例	main()
	{
	float　x, result;
	result = Asin(0.8660);　//　执行后 result = 60
	x = 0.5;
	result = Asin(x);　//　执行后 result = 30
	}

9）反余弦运算

函数名	Acos
函数签名	Acos(x);
功能描述	反三角函数中反余弦计算。

用法举例	main()
	{
	float　x, result;
	result = Acos(0.8660);　//　执行后 result = 30
	x = 0.5;
	result = Acos(x);　//　执行后 result = 60
	}

10）反正切运算

函数名	Atan
函数签名	Atan(x);
功能描述	反三角函数中反正切计算。

用法举例	main()
	{
	float x, result;
	result = Atan(1); // 执行后 result = 45
	x = 1.732;
	result = Atan(x); // 执行后 result = 60
	}

11）产生随机数值

函数名	Rand
函数签名	Rand();
功能描述	产生一个随机数。

用法举例	main()
	{
	int x;
	x = Rand(); // 执行后 x 获得一个随机整数
	}

21.2.2 数据转换函数

1）Bin2Bcd

函数名	Bin2Bcd
函数签名	Bin2Bcd(x);
功能描述	将 BIN 格式的数据(x)转换为 BCD 格式的数据(result)。

用法举例	main()
	{
	int x, result;
	result = Bin2Bcd(1234); // 执行后 result = 0x1234
	x = 123456789;
	result = Bin2Bcd(x); // 执行后 result = 0x123456789
	}

2）Bcd2Bin

函数名	Bcd2Bin
函数签名	Bcd2Bin(x);
功能描述	将 BCD 格式的数据(x)转换为 Bin 格式的数据(result)。

用法举例	main()
	{
	int x, result;
	result = Bcd2Bin(0x1234); // 执行后 result = 1234
	x = 0x123456789;
	result = Bcd2Bin(x); // 执行后 result = 123456789
	}

3）Dec2Ascii

函数名	Dec2Ascii
函数签名	Dec2Ascii(x， result[ ]， index， len);
功能描述	将十进制的数据(x)转换为 ASCII 格式的数据，并存放在一个 char 类型的一维数组(result)中，由 index 指定存放的起始下标，len 指定可存放的最大长度。

用法举例	main()
	{
	int x=5678;
	char result[4];
	Dec2Ascii(x, result, 0, 4);
	// 执行后:
	// result[0] = ‘5’
	// result[1] = ‘6’
	// result[2] = ‘7’
	// result[3] = ‘8’
	}

4）Hex2Ascii

函数名	Hex2Ascii
函数签名	Hex2Ascii (x， result[ ]， index， len);
功能描述	将十六进制的数据(x)转换为 ASCII 格式的数据，并存放在一个 char 类型的一维数组(result)中，由 index 指定存放的起始下标，len 指定可存放的最大长度。
用法举例	main()

	{
	int x=0x5678;
	char result[4];
	Hex2Ascii (x, result, 0, 4);
	// 执行后:
	// result[0] = ‘5’
	// result[1] = ‘6’
	// result[2] = ‘7’
	// result[3] = ‘8’
	}

5）Ascii2Dec

函数名	Ascii2Dec
函数签名	Ascii2Dec(array[ ]， index， len);
功能描述	将字符型 ASCII 数据转换为十进制格式的数据。字符型 ASCII 数据存放在一个 char 类型的一维数组中，由 index 指定操作的起始下标，len 指定操作的长度。

用法举例	main()
	{
	int result;
	char array[4]={ '5', '6', '7', '8'};
	result=Ascii2Dec(array, 0, 4);
	// 执行后 result = 5678
	}

6）Ascii2Hex

函数名	Ascii2Hex
函数签名	Ascii2Hex(array [ ]， index， len);
功能描述	将字符型 ASCII 数据转换为十六进制格式的数据。字符型 ASCII 数据存放在一个 char 类型的一维数组中，由 index 指定操作的起始下标，len 指定操作的长度。

用法举例	main()
	{
	int result;
	char array[4]={ '5', '6', '7', '8'};
	result= Ascii2Hex(array, 0, 4);
	// 执行后 result = 0x5678
	}

21.2.3 数据操作函数

1）Fill

函数名	Fill
函数签名	void Fill(val， result[ ]， index， len);
功能描述	将值 val 放置到一维数组 result 由下标 index 开始的连续 len 个位置中。其中，type 可以是任意算术类型（除 void）。

用法举例	main()
	{
	bool b = false;
	bool ba[4];
	char c = 'M';
	char ca[4];
	short s = 9;
	short sa[4];
	int i = 9;
	int ia[4];
	float f = 3.14;
	float fa[4];
	double d = 3.14;
	double da[4];
	Fill(true, ba, 0, 4);
	// 执行后: ba[0] = true, ba[1] = true
	// ba[2] = true, ba[3] = true
	Fill(b, ba, 1, 2);
	// 执行后: ba[1] = false, ba[2] = false
	Fill ('a', ca, 0, 4);
	// 执行后: ca[0] = ‘a’, ca[1] = ‘a’
	// ca[2] = ‘a’, ca[3] = ‘a’
	Fill(c, ca, 1, 2);
	// 执行后: ca[1] = ‘M’, ca[2] = ‘M’
	Fill (6, sa, 0, 4);
	// 执行后: sa[0] = 6, sa[1] =6
	// sa[2] = 6, sa[3] =6
	Fill(s, sa, 1, 2);
	// 执行后: sa[1] = 9, sa[2] = 9
	Fill(6, ia, 0, 4);

	// 执行后: ia[0] = 6, ia [1] = 6
	// ia [2] = 6, ia [3] = 6
	Fill(i, ia, 1, 2);
	// 执行后: ia [1] = 9, ia [2] = 9
	Fill(6.18, fa, 0, 4);
	// 执行后: fa[0] = 6.18, fa[1] = 6.18
	// fa[2] = 6.18, fa[3] = 6.18
	Fill(f, fa, 1, 2);
	// 执行后: fa[1] = 3.14, fa[2] = 3.14
	Fill(6.18, da, 0, 4);
	// 执行后: da[0] = 6.18, da[1] = 6.18
	// da[2] = 6.18, da[3] = 6.18
	Fill(d, da, 1, 2);
	// 执行后: da[1] = 3.14, da[2] = 3.14
	}

2）SwapByte

函数名	SwapByte
函数签名	SwapByte (x);
功能描述	将一个 16 位字的高低字节颠倒。

用法举例	main()
	{
	short x, result;
	result = SwapByte(0x1234); // 执行后 result = 0x3412
	x = 0x123;
	result = SwapByte(x); // 执行后 result = 0x2301
	}

3）SwapWord

函数名	SwapWord
函数签名	SwapWord(x);
功能描述	将一个 32 位双整型数据的高位字和低位字颠倒。

用法举例	main()
	{
	int x, result;
	result = SwapWord(0x12345678);
	//执行后 result = 0x56781234

	x = 0x12345;
	result = SwapWord(x);
	// 执行后 result = 0x23450001
	}

4）LoByte

函数名	LoByte
函数签名	LoByte(x);
功能描述	获取一个 16 位数据的低字节。

用法举例	main()
	{
	short x, result;
	result = LoByte(0x1234); // 执行后 result = 0x34
	x = 0x123;
	result = LoByte(x); // 执行后 result = 0x23
	}

5）HiByte

函数名	HiByte
函数签名	HiByte(x);
功能描述	获取一个 16 位数据的高字节。

用法举例	main()
	{
	short x, result;
	result = HiByte(0x1234); // 执行后 result = 0x12
	x = 0x123;
	result = HiByte(x); // 执行后 result = 0x01
	}

6）LoWord

函数名	LoWord
函数签名	LoWord (x);
功能描述	获取一个 32 位数据的低位字。

用法举例	main()
	{
	int x, result;
	result = LoWord(0x12345678); //执行后 result = 0x5678

	x = 0x12345;
	result = LoWord(x);　//　执行后 result = 0x2345
	}

7）HiWord

函数名	HiWord
函数签名	HiWord(x);
功能描述	获取一个 32 位数据的高位字。

用法举例	main()
	{
	int　x, result;
	result = HiWord(0x12345678);　//执行后 result = 0x1234
	x = 0x12345;
	result = HiWord(x);　//　执行后 result = 0x0001
	}

21.2.4　位状态转换函数

1）GetBit

函数名	GetBit
函数签名	GetBit(x，offset);
功能描述	获取一个 32 位数据的指定位的状态，offset=0 表示最低位。

用法举例	main()
	{
	　　bool　result;
	int　x, offset;
	result = GetBit(9, 3);　//执行后 result = true
	x = 4, offset=2;
	result = GetBit(x, offset);　//执行后 result = true
	}

2）SetBitOn

函数名	SetBitOn
函数签名	SetBitOn(x，　offset);
功能描述	将一个 32 位数据的指定位设为 1，offset=0 表示最低位。
用法举例	main() {

	int result;
	int x, offset;
	result = SetBitOn (1, 3); //执行后 result = 9
	x = 0, offset=2;
	result = SetBitOn (x, offset); // 执行后 result = 4
	}

3）SetBitOff

函数名	SetBitOff
函数签名	SetBitOff(x，offset);
功能描述	将一个 32 位数据的指定位设为 0，offset=0 表示最低位。

用法举例	main()
	{
	int result;
	int x, offset;
	result = SetBitOff (9, 3); //执行后 result = 1
	x = 4, offset=2;
	result = SetBitOff (x, offset); // 执行后 result = 0
	}

4）ReverseBit

函数名	ReverseBit
函数签名	ReverseBit(x，offset);
功能描述	将一个 32 位数据的指定位反转，offset=0 表示最低位。

用法举例	main()
	{
	int result;
	int x, offset;
	result = ReverseBit(4, 1); //执行后 result = 6
	x = 6, offset=1;
	result = ReverseBit(x, offset); // 执行后 result = 4
	}

21.2.5 通讯有关的函数

1）Delay

函数名	Delay
函数签名	Delay(ms);

功能描述	让宏指令暂停执行至少是指定的这个时间（毫秒）。

用法举例	main()
	{
	int ms=500;
	Delay(100); // delay 100 ms
	Delay(ms); // delay 500 ms
	}

2）AddSum

函数名	AddSum
函数签名	AddSum (array [ ]， index， len);
功能描述	计算累加校验和(checksum)。待校验数据存放在一个 char 类型的一维数组中，由 index 指定校验的起始下标，len 指定校验的长度。

用法举例	main()
	{
	int checksum;
	char array[5]={ 0x01, 0x02, 0x03, 0x04, 0x05};
	checksum = AddSum (array, 0, 5);
	// 执行后 checksum = 0x0f
	}

3）XorSum

函数名	XorSum
函数签名	XorSum(array [ ]， index， len);
功能描述	计算异或校验和(checksum)。待校验数据存放在一个 char 类型的一维数组中，由 index 指定校验的起始下标，len 指定校验的长度。

用法举例	main()
	{
	int checksum;
	char array[5]={ 0x01, 0x02, 0x03, 0x04, 0x05};
	checksum = XorSum(array, 0, 5);
	// 执行后 checksum = 0x01
	}

4）CrcSum

函数名	CrcSum
函数签名	CrcSum (array [ ]， index， len);
功能描述	计算 16-bits CRC 校验和(checksum)。待校验数据存放在一个 char 类型的一维数组中，由 index 指定校验的起始下标，len 指定校验的长度。

用法举例	main()
	{
	short checksum;
	char array[5]={ 0x01, 0x02, 0x03, 0x04, 0x05};
	checksum = CrcSum (array, 0, 5);
	// 执行后 checksum = 0xbb2a
	}

5）Get

函数名	Get
函数签名	Get(device，addt，addr，type);
功能描述	从指定设备，指定地址类型，指定地址读取一个指定类型的数据。其中，type-name 可以是任意算术类型名（除 void），用于指明数据的存储格式。

用法举例	main()
	{
	bool b;
	char c;
	short s;
	int i;
	float f;
	double d;
	// 从设备"H2U "读取 M0
	b=Get("H2U", M, 0, bool);
	// 从设备"H2U "读取 D0 的低字节
	c=Get("H2U ", D, 0, char);
	// 从设备"H2U "读取 D0，并解释为 short
	s=Get("H2U ", D, 0, short);
	// 从设备"H2U "读取 D0~D1，并解释为 int
	i=Get("H2U ",D, 0, int);
	// 从设备"H2U "读取 D0~D1，并解释为 float

	f=Get("H2U ", D, 0, float);
	// 从设备"H2U "读取 D0~D3, 并解释为 double
	d=Get("H2U ", D, 0, double);
	}

6）GetBlock

函数名	GetBlock
函数签名	GetBlock (device，addt，addr，target[ ]， index， len);
功能描述	从指定设备，指定地址类型，指定地址读取连续 len 个指定类型的数据；数据存放到一维数组由 target[index]到 target[index+len-1]的区域。其中，type 可以是任意算术类型（除 void）。

用法举例	main()
	{
	bool b[3];
	char c[3];
	short s[3];
	int i[3];
	float f[3];
	double d[3];
	// 从设备"H2U "的 M0~M2 读取数据到数组 b
	GetBlock ("H2U", M, 0, b, 0, 3);
	// 从设备"H2U "的 D0~D1(低字节)读取数据到数组 c
	GetBlock ("H2U", D, 0, c, 0, 3);
	// 从设备"H2U "的 D0~D2 读取数据到数组 s，解释为 short 格式
	GetBlock ("H2U",D, 0, s, 0, 3);
	// 从设备"H2U "的 D0~D5 读取数据到数组 i，解释为 int 格式
	GetBlock ("H2U", D, 0, i, 0, 3);
	// 从设备"H2U "的 D0~D5 读取数据到数组 f，解释为 float 格式
	GetBlock ("H2U", D, 0, f, 0, 3);
	// 从设备"H2U "的 D0~D8 读取数据到数组 d，解释为 double 格式
	GetBlock ("H2U", D, 0, d, 0, 3);
	}

7）Set

函数名	Set
函数签名	Set(device，addt，addr， val);
功能描述	向指定设备，指定地址类型，指定地址写入一个指定类型的数

	据。其中，type 可以是任意算术类型（除 void）。

用法举例	main()
	{
	bool b1=true, b2=false;
	char c= 'c';
	short s=999;
	int i=999999;
	float f=3.14;
	double d=3.1415926;
	// 向设备"H2U "的 M0 写入 b1 的值
	Set("H2U", M, 0, b1);
	// 向设备"H2U "的 M[0:a]写入 b2 的值
	Set ("H2U", M, 0:a, b2);
	// 向设备"H2U "的 D 低字节写入 c 的值
	Set("H2U", D, 0, c);
	// 向设备"H2U "的 D 写入 s，保存为 short 格式
	Set("H2U", D, 0, s);
	// 向设备"H2U "的 D0~D1 写入 i，保存为 int 格式
	Set("H2U",D, 0, i);
	// 向设备"H2U "的 D0~D1 写入 f，保存为 float 格式
	Set("H2U", D, 0, f);
	// 向设备"H2U "的 D0~D3 写入 d，保存为 double 格式
	Set("H2U", D, 0, d);
	}

8）SetBlock

函数名	SetBlock
函数签名	SetBlock (device，addt，addr，source[ ]， index， len);
功能描述	向指定设备，指定地址类型，指定地址写入连续 len 个指定类型的数据；待写数据存放在一维数组由 source[index]到 source[index+len-1]的区域。其中，type 可以是任意算术类型（除 void）。
用法举例	main() { bool b1[3]={true, true, false}; char c[3]={ 'a', 'b', 'c'}; short s[3]={10, 20, 30};

	int i[3]={1000, 2000, 3000};
	float f[3]={0.1, 0.2, 0.3};
	double d[3]={0.01, 0.02, 0.03};
	// 向设备"H2U "的 M0~M2 写入数组 b1
	SetBlock ("H2U", M, 0, b1, 0, 3);
	// 向设备"H2U "的 M[0:a~c]写入数组 b1
	SetBlock ("H2U", M, 0:a, b1, 0, 3);
	// 向设备"H2U "的 D0~D1(低字节)写入数组 c
	SetBlock ("H2U", D, 0, c, 0, 3);
	// 向设备"H2U "的 D0~D2 写入数组 s, 保存为 short 格式
	SetBlock ("H2U", LW, 0, s, 0, 3);
	// 向设备"H2U "的 D0~D5 写入数组 i, 保存为 int 格式
	SetBlock ("H2U", D, 0, i, 0, 3);
	// 向设备"H2U "的 D0~D5 写入数组 f, 保存为 float 格式
	SetBlock ("H2U", D, 0, f, 0, 3);
	// 向设备"H2U "的 D0~D8 写入数组 d, 保存为 double 格式
	SetBlock ("H2U", D, 0, d, 0, 3);
	}

21.2.6 宏程序间通信函数

AsynTriMacro

函数名	AsynTriMacro
函数签名	AsynTriMacro(id);
功能描述	异步方式触发以 id 指定的宏指令。不必等待指定宏指令执行完毕，继续执行该宏指令的剩余指令。id 若为当前宏指令，调用无效。

用法举例	main()
	{
	short s=0;
	AsynTriMacro(3); //异步触发 id=3 的宏指令
	// 运行环境确保不必等待 id=3 的宏指令执行完毕，马上执行下面语句
	s++;
	}

21.3 宏指令的语法

21.3.1 语言保留字

保留字是指不允许用户以任何形式作为名字来声明的标识符。具体包括关键字和其它非关键字。

1）关键字（20 个）

bool	break	case	char	continue
default	do	double	else	false
float	for	if	int	return
short	switch	true	void	while

2）非关键字

主函数名“main”。

标识符

在语言中，标识符是作为用户声明的变量名，数组名，函数名等形式出现的。宏语言接受的标识符是除任意保留字以外的满足如下条件的字符序列：

以字母或下划线开头，后跟任意数量的字母,下划线或数字。

举例：

Id，S300，_m_func，...

像 main，for，while 这些保留字是不作为用户使用的标识符的。

21.3.2 常量/变量/数组

1）常量

常量是一个可以被各式语句直接使用的固定的数据。有如下格式：

常量类型	使用说明	举例
十进制（dec）	不以 0 开头的任意数[0~9]序列	10，168，5012，…
八进制（otc）	以 0 开头的任意数[0~7]序列	03，000127，010101，…
十六进制（hex）	以 0x 或 0X 开头的任意数（[0~9]&[a~f]&[A~F]）序列	0x1，0x0a0，0X1AdF，…
字符型(ASCII 编码)	字符必须使用单引号	'a','A','6','9'， ...
浮点数	a. 十进制 b. 科学计数法（e 或 E 表示“乘 10”的意思，后面是指数）	0.，0.123，3.1415926，... 0e0，1E-3，3.14159E0，...
布尔型	0，1	false , true

注意：为不丢失精度，整数常量默认为 int 类型；浮点数常量默认为 double 类型。

2）变量

变量是一个代表着各种资料的名称。在宏指令中，这些资料可以随着宏指令语句执行的结果改变而改变。

类型	含义	描述	数值范围
bool	布尔型	1 bit (一个位)	[0，1]
char	字符型	8 bits (一个字节)	[-128，127]
short	短整型	16 bits (一个字)	[-32768，32767]
int	双整型	32 bits (双字)	[-2147483648，2147483647]
float	单精度浮点型	32 bits (双字)	保证 6 位有效数字
double	双精度浮点型	64 bits（四字）	保证 10 位有效数字
void	特殊类型,作为无返回值函数的返回值类型	无	无

变量遵循“先声明后使用”规则。在宏指令中，所有的变量都必须在语句使用前都被声明完成。声明变量时，先定义变量的类型，后面再跟着变量名称。

举例：用户可以按以下任意方式声明变量：

a. 简单声明：　　short　s;

b. 简单声明并初始化：　　short　s=100;

c. 系列声明：　　short　s1, s2, s3;

d. 系列声明并初始化：　　short　s1=1, s2=20, s3=300;

e. 系列声明并部分初始化：　　short　s1, s2=1024, s3;

3）数组

数组的使用也遵循“先声明后使用”规则。数组的访问需使用操作符“[ ]”，并且规定数组的起始为 0，举例如下：

① 用户可以按以下任意方式声明数组：

a. 一维声明：　　short　s[3];

b. 一维声明并初始化：　　short　s[3]={1，2，3};

② 按如下方式初始化：

```
short    array[3]={1,2,3};        //声明并初始化一维数组
…
array[0]=20;        //将第一个元素的值置为 20
```

21.3.3 操作符

1）算术操作符

操作符	功能	用法
+	unary plus（一元正号）	+ expr
-	unary minus（一元负号）	- expr
*	multiplication（乘法）	expr * expr
/	division（除法）	expr / expr
%	remainder（求余）	expr % expr
+	addition（加法）	expr + expr
-	subtraction（减法）	expr - expr

2）位操作符

操作符	功能	用法
~	bitwise NOT（位求反）	~expr
<<	left shift（左移）	expr1 << expr2
>>	right shift（右移）	expr1 >> expr2
&	bitwise AND（位与）	expr1 & expr2
^	bitwise XOR（位异或）	expr1 ^ expr2
|	bitwise OR（位或）	expr1 | expr2

3）关系操作符

操作符	功能	用法
<	less than（小于）	expr < expr
<=	less than or equal（小于等于）	expr <= expr
>	greater than（大于）	expr > expr
>=	greater than or equal（大于等于）	expr >= expr
==	equality（相等）	expr == expr
!=	inequality（不等）	expr != expr

4）逻辑操作符

操作符	功能	用法
!	logical NOT（逻辑非）	!expr
&&	logical AND（逻辑与）	expr && expr
||	logical OR（逻辑或）	expr || expr

5）赋值&复合赋值操作符

操作符	功能	用法
=	assignment（赋值操作）	lvalue = expr
*=	assignment multiplication（赋值乘法）	lvalue *= expr
/=	assignment division（赋值除法）	lvalue /= expr
%=	assignment remainder（赋值求余）	lvalue %= expr
+=	assignment addition（赋值加法）	lvalue += expr
-=	assignment subtraction（赋值减法）	lvalue -= expr
<<=	assignment left shift（赋值左移）	lvalue <<= expr
>>=	assignment right shift（赋值右移）	lvalue >>= expr
&=	assignment bitwise AND（赋值位与）	lvalue &= expr
^=	assignment bitwise XOR（赋值位异或）	lvalue ^= expr
|=	assignment bitwise OR（赋值位或）	lvalue |= expr

6）自增&自减操作符

操作符	功能	用法
++	postfix increment（后自增操作）	lvalue++
--	postfix decrement（后自减操作）	lvalue--
++	prefix increment（前自增操作）	++ lvalue
--	prefix decrement（前自减操作）	-- lvalue

7）条件操作符

操作符	功能	用法
? :	conditional（条件操作）	expr ? expr : expr

8）逗号操作符

操作符	功能	用法
,	comma（逗号）	expr , expr

9）下标操作符

操作符	功能	用法
[ ]	subscript（下标）	variable [expr]

10）函数调用操作符

操作符	功能	用法
( )	function call（函数调用）	name(expr_list)

21.3.4 语句

1）条件语句：if

a. 不带 else 分支的 if 语句

语法形式：

```
if(a>b )                    //判断 a 是否大于 b
        b=a;                //如果 if 条件成立，则把 a 值赋给 b
```

b．带 else 分支的 if 语句

语法形式：

```
if(a>b )                    //判断 a 是否大于 b
        b=a;                //如果 if 条件成立，则把 a 值赋给 b
 else                       //条件不成立
        a=b;                //如果 if 条件不成立，则把 b 值赋给 a
```

c．悬垂 else 的语义说明

```
if( a>b )                   //判断 a>b
{
    if(a>c )                //在 a>b 条件下，再判断 a>c
        b=a;                //如果 a>b  且 a>c，则把 a 值赋给 b
    else                    //如果 a>b  但 a!>c
        c=a;                //如果 a>b  且 a!>c，则把 a 值赋给 c
}
```

即：将 else 匹配给最后出现的尚未匹配的 if 子句。可以通过引入块语句，改变这种默认的匹配规则：

```
if( a>b )                   //判断 a>b 条件
{
    if( a>c )               //判断  a>c
        c=a;                //如果  a>b  且 a>c  则把 a 值赋给 c
}
else                        //如果 a!>b
{
        b=a;                //如果 a!>b,把 a 值赋给 b
}
```

2）开关语句：switch

语法形式：

```
switch( a )                 //选择变量 a
```

```
{
    case   1:                          //判断 a==1
        b=1；                          //如果 a==1  则把 1 赋给 b
        break;                         //结束

    case   2:                          //判断 a==2
        b=2；                          //如果 a==2  则把 2 赋给 b
        break;                         //结束
    …
    case   n:                          //判断 a==n
        b=n；                          //如果 a==n  则把 n 赋给 b
        break;                         //结束

    default:                           //a 不与上面等式一样
        b=10；                         //把 10 赋给 b
        break;                         //结束
}
```

3）循环语句：while / do-while / for

a. while 语法形式：

```
while ( a==1 )                 //判断是否 a==1
    b=1；                      //条件成立把 1 值赋给 b
```

b. do-while 语法形式：

```
    do
    b=1                        //开始执行把 1 值赋给 b
while ( a==1 )；                //判断 a 是否==1，是则继续执行 do
```

while 语句与 do-while 语句主要有两点不同：

① do-while 语句保证循环体至少执行一次。

② 与 while 语句不同，do-while 语句总是以分号“; ”结束。

c. for 语法形式：

```
for(i==0,i<10,i++)      //开始把 0 值赋给 i 变量，判断 i 是否小于 10，为真则执行下面指
                          令，执行完后把 i 自加 1
    a=a++;            //上面 i<10 成立 a 自加 1
```

4）跳转语句：break

语法形式：

```
break ；
```

break 语句用于结束最近的 while、do-while、for 或 switch 语句，并将程序的执行权传递给紧接在被终止语句之后的语句。

break 语句有如下方面的特点：

①只能出现在循环或 switch 结构中，或者出现在嵌套于循环或 switch 结构中的语句里。

②只终止直接包含 break 的 switch 或循环语句，而外层的 switch 或者循环不受影响。

③程序的执行权转交给被终止的 switch 或循环语句之后的第一条语句。

5）函数

a．声明和定义

语法形式：

```
Type    Name( parameter_list )
{
    statement_list
}
```

函数由函数名 Name 及一组类型明确的操作数 parameter_list 唯一地表示，结构上，则由函数头和函数体组成。函数的操作数，称为函数的“形参”，在圆括号中声明，形参之间以逗号分隔。函数执行的运算，在一个称为“函数体”的块语句中定义。每个函数都要指定一个明确的返回类型 Type。

b．函数调用

语法形式：

```
Name(argument _list )；
```

使用函数调用操作符“( )”实现函数调用。正如其它操作符一样，调用操作符需要操作数并产生一个结果。调用操作符的操作数是函数名 Name 和一组（有可能为空）由逗号分隔的实参 argument _list。函数调用的结果类型就是函数的返回值类型，该运算的结果本身就是函数返回值。

函数调用做了两件事情：用对应的实参初始化函数的形参，并将控制权转移给被调用函数。主调函数的执行被挂起，被调用函数开始执行。函数的运行以形参的（隐式）定义和初始化开始。

函数调用要求实参个数必须与形参个数相同。并且，按从左到右的顺序，每一个实参自动类型转换成形参类型。实参可以是任意表达式，包括函数调用，但不可以是返回类型为 void 的函数调用。

函数举例：

```
Short   Name( x )                  // 函数声明
    {
        x++;                       //把主函数中调用的 x 自加 1
```

```
        }
Main()
{
    Short   x=1;                    //定义一个变量 x 且赋值为 1
    x=Name(x );                     //调用 Name 这个函数且把调用后的值赋值给 x
                                      调用后，函数 x 值自加 1，则返回值为 2 赋值给 x
}
```

21.4 宏指令举例

21.4.1 读写一个位值

1）从 LB0 读取一个位值，并将该位值写入 LB100

```
main()
{
    bool   b;
    //读取
    b=Get("Local HMI", LB, 0, bool);
    //写入
    Set("Local HMI", LB, 100, b);
}
```

2）从 LW_BIT[0:a]读取一个位值，并将该位值写入 LW_BIT [100:b]

```
main()
{
    bool   b;
    //读取
    b=Get("Local HMI", LW_BIT, 0:a, bool);
    //写入
    Set("Local HMI", LW_BIT, 100:b, b);
}
```

21.4.2 读写一个 16 位整数

从 LW0 读取一个 16 位整数，并将该 16 位整数写入 LW100

```
main()
{
    short   s;
    //读取
    s=Get("Local HMI", LW, 0, short);
    //写入
```

```
    Set("Local HMI", LW, 100, s);
}
```

21.4.3 读写一个 32 位整数

从 LW0~LW1 读取一个 32 位整数，并将该 32 位整数写入 LW100~LW101

```
main()
{
    int   i;
    //读取
    i=Get("Local HMI", LW, 0, int);
    //写入
    Set("Local HMI", LW, 100, i);
}
```

21.4.4 读写一个 32 位浮点数

从 LW0~LW1 读取一个 32 位浮点数，并将该 32 位浮点数写入 LW100~LW101

```
main()
{
    float   f;
    //读取
    f=Get("Local HMI", LW, 0, float);
    //写入
    Set("Local HMI", LW, 100, f);
}
```

21.4.5 读写 3 个位值

1）从 LB0~LB2 读取 3 个位值，并写入到 LB100~LB102

```
main()
{
    bool   ba[3];
    //读取
    GetBlock("Local HMI", LB, 0, ba, 0, 3);
    //写入
    SetBlock("Local HMI", LB, 100, ba, 0, 3);
}
```

2）从 LW_BIT[0:a~c]读取 3 个位值，并写入到 LW_BIT [100:b~d]

```
main()
{
    bool    ba[3];
    //读取
    GetBlock("Local HMI", LW_BIT, 0:a, ba, 0, 3);
    //写入
    SetBlock("Local HMI", LW_BIT, 100:b, ba, 0, 3);
}
```

21.4.6 读写 3 个 16 位整数

从 LW0~LW2 读取 3 个 16 位整数，并写入到 LW100~LW102

```
main()
{
    short    sa[3];
    //读取
    GetBlock("Local HMI", LW, 0, sa, 0, 3);
    //写入
    SetBlock("Local HMI", LW, 100, sa, 0, 3);
}
```

21.4.7 读写 3 个 32 位整数

从 LW0~LW5 读取 3 个 32 位整数，并写入到 LW100~LW105

```
main()
{
    int    ia[3];
    //读取
    GetBlock("Local HMI", LW, 0, ia, 0, 3);
    //写入
    SetBlock("Local HMI", LW, 100, ia, 0, 3);
}
```

21.4.8 读写 3 个 32 位浮点数

从 LW0~LW5 读取 3 个 32 位浮点数，并写入到 LW100~LW105

```
main()
{
    float    fa[3];
    //读取
```

```
    GetBlock("Local HMI", LW, 0, fa, 0, 3);
    //写入
    SetBlock("Local HMI", LW, 100, fa, 0, 3);
}
```

21.4.9 读写 6 个字符

从 LW0~LW2 读取 6 个字符，并写入到 LW100~LW102

```
main()
{
    char    ca[6];
    //读取
    GetBlock("Local HMI", LW, 0, ca, 0, 6);
    //写入
    SetBlock("Local HMI", LW, 100, ca, 0, 6);
}
```

21.4.10 累加

下面程序 1 秒对 i 累加一次，结果以 32 位整数形式输出到 LW0~LW1，程序将持续运行一个小时。

```
main()
{
    int    i;
    for(i=0; i<3600; i++)     //循环执行 3600 次
    {
        Set("Local HMI", LW, 0, i);     //将 i 的值存入 LW0
        Delay(1000);      //每次循环延迟 1000ms
    }
}
```

21.4.11 函数

下面程序演示自定义函数 reverse( )将 32 位整数的四个字节在机器的存储顺序逆转过来，运行结果是 LW0~LW1 存储了 0x78563412。

```
int    reverse(int    i)                    //定义函数
{
    i = SwapWord(i);
    return SwapByte( HiWord(i) )<<16 | SwapByte( LoWord(i) ) & 0xffff ;   //逆转 32 位数
}
main()
```

```
{
    int i=0x12345678;              //赋值变量
    i=reverse(i);                  //调用函数 reverse 并把值传到 i 变量中
    Set("Local HMI",LW,0,i);     //把调用后的 i 值写到 LW0 寄存器内
}
```

21.4.12 循环

下面程序 1 秒检查一次 LB0 的位值，如果为 ON，则将 16 位随机整数写入到 LW0，跳出循环并结束程序；否则继续下一次检查。

```
main()
{
    bool   b;
    short   s;
    //监视循环
    while(true)
    {
        //读取 LB0
        b=Get("Local HMI",LB,0,bool);
        //如果为 ON
        if(b)
        {
            //生成 16 位随机整数
            s=Rand();
            //写入到 LW0
            Set("Local HMI",LW,0,s);
            //跳出循环
            break;
        }
        //延时 1 秒
        Delay(1000);
    }
}
```

21.4.13 阶乘

下面程序计算正整数 9 的阶乘，结果以 32 位整数形式输出到 LW0~LW1

```
int   fac(int   n)
{
```

```
    if(n<2)
        return 1;
    return n*fac(n-1);
}
main()
{
    int   i=fac(9);
    Set("Local HMI", LW, 0, i);
}
```

//特别注意：整数常量的类型规定为 int(32bit)；浮点数常量的类型规定为 double(64bit)。

```
main()
{
    short s=99;
    float f=3.14;
    //16 位整数 s 写入 LW0
    Set("Local HMI", LW, 0, s);
    //32 位整数常量 99 写入 LW0~LW1
    Set("Local HMI", LW, 0, 99);
    //32 位浮点数 f 写入 LW0~LW1
    Set("Local HMI", LW, 0, f);
    //64 位浮点数常量 3.14 写入 LW0~LW3
    Set("Local HMI", LW, 0, 3.14);
}
```

21.5 应用中的实例

21.5.1 读取数值都为负值

功能：读取的设备数值都为负值。

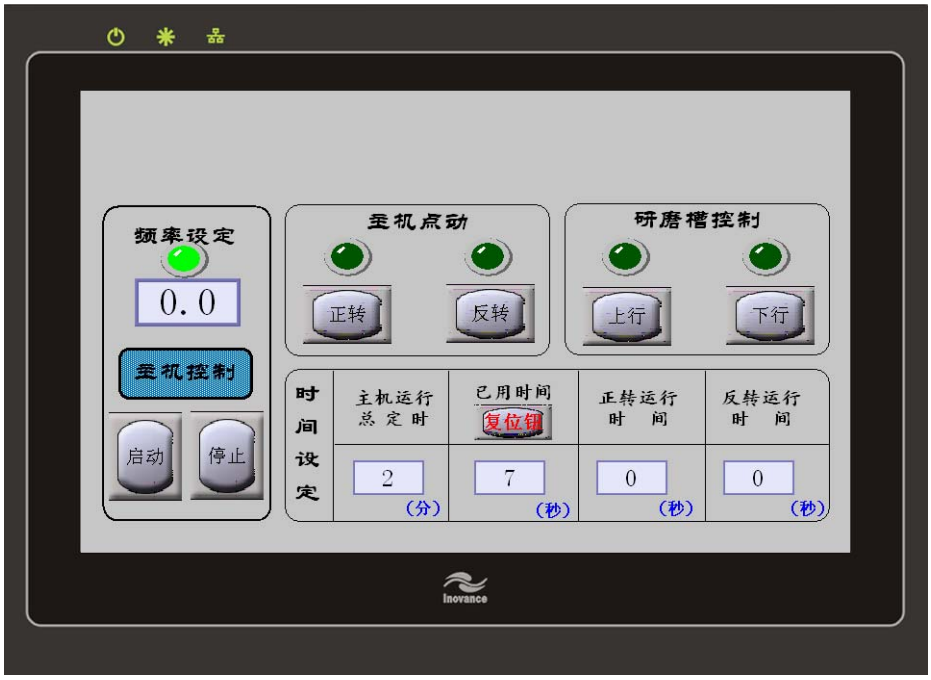
宏指令如下：

```
main()
{
    short a,b;
    a=Get("Modbus_RTU", dev_6x, 17222, short);    //读取设置一个寄存器值
    if(a>0)                                       //判断值的正负
        b=a*(-1);                                 //为正则写为负数
    else                                          //如果为负数
        b=a;                                      //直接赋值
```

```
    Set("Modbus_RTU", dev_6x, 17222, b);              //写入寄存器值
}
```

21.5.2 使用宏指令控制变频器启动停止功能

如图，可以通过宏指令手动对主机进行正反转操作以及自动启动。



手动操作：

正转：让变频器正传运行

```
 main()
{
    Set("变频器", dev_6x, 8192,3); //设置变频器启动地址，进行正传启动。
}
```

停止：使变频器停止运行

```
main()
{
    Set("变频器", dev_6x, 8192,5); //设置变频器启动地址，进行停止。
}
```

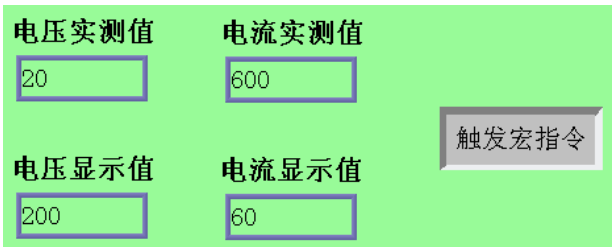
自动操作：设置主机运行总时间后，点击启动按钮，主机自动运行，到达设定时间后，自动停止运行。

```
main()
{
    bool   lb0,lb11,lb2,l=1,b=0;
    short   lw0,rw0,rw1,rw2,y1=16,y2=0;
    int   lw1,lw2,lw3;
    //初始化读取数值
```

```
    lb0=Get("Local HMI", LB, 0, bool);              //赋值
    rw0=Get("Local HMI", RW, 0, short);              //变频器设定时间
    Set("变频器", dev_6x, 2050,lw1);
    Set("Local HMI", LW, 3,0);                      //计数清零
    for(;lb0==1;)                                   //循环启动
    {
        lw0=Get("Local HMI", LW, 0, short); //读取时间
        lw3=Get("Local HMI", LW, 3, int);   //读取时间
        if(lw3>=20)                         //判断前 20 秒
        Set("变频器", dev_6x, 772,y1);      //前 20s 变频
        if(lw0>=rw0*60)                      //变频判断，是否到达时间
        {
            Set("变频器", dev_6x, 8192,5); //时间到变频停机
            Set("Local HMI", LB, 0,b);      //  自动切换手动
            Set("Local HMI", LW, 0,0);          //计数清零
        }
            lb0=Get("Local HMI", LB, 0, bool);   //判断是否手动停止操作。
    }
            Set("变频器", dev_6x, 8192,5); //变频停机
            Set("变频器", dev_6x, 772,y2); //变频停机
}
```

21.5.3 使用宏指令改变电压电流实测值的大小

功能，把读上来的电压实测值放到十倍，电流实测值缩小十倍。



宏指令如下：

```
main()
{
    short   a,b;
    //初始化读取数值
    while(1)
    {
        //读取电压值
```

```
        a=Get("Local HMI", LW, 50, short);
        //读取电流值
        b=Get("Local HMI", LW, 51, short);
        //电压值放大 10 倍
        Set("Local HMI", LW, 52, a*10);
        //电流值缩小 10 倍
        Set("Local HMI", LW, 53, b/10);
    }
}
```

22

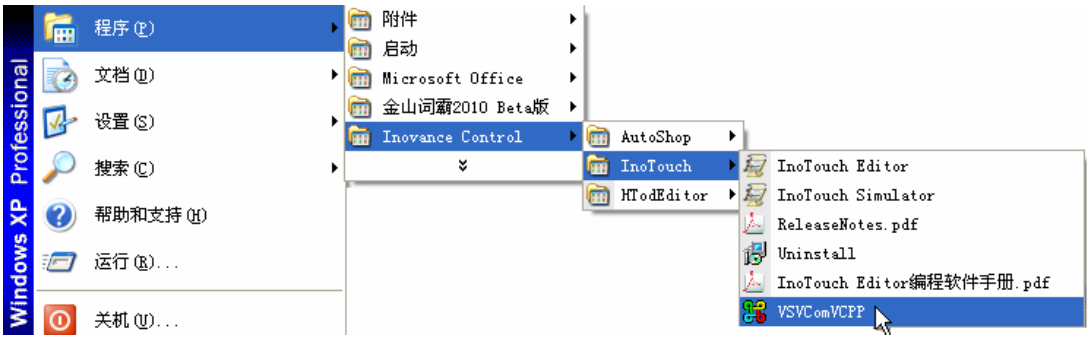
穿透通讯功能

第二十二章　穿透通讯功能

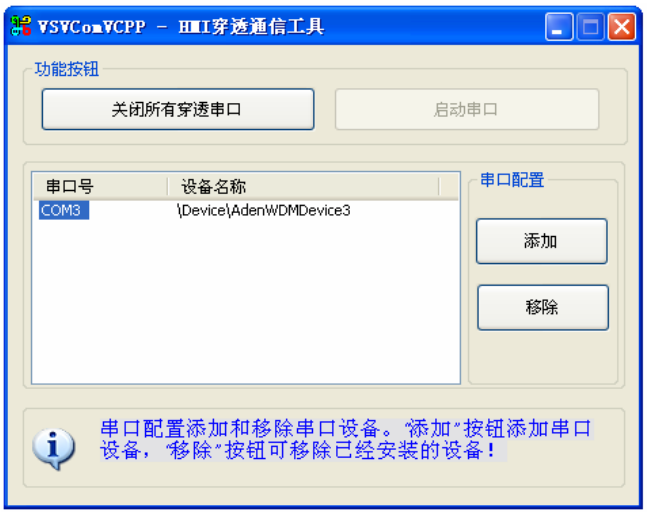
InoTouch Editor 软件提供的穿透通讯功能，是在计算机上使用PLC的编程软件，通过计算机所连接的人机界面，连接到与该人机界面连接的PLC上。这样，就可以监控，或者上传、下载PLC的程序。此时，InoTouch 系列HMI扮演一个转换器的角色。穿透通讯功能，分为串口连接的穿透通讯功能和以太网(虚拟串口)穿透通讯功能两种方式。

22.1　穿透工具软件介绍

安装 InoTouch Editor V1.10 版本或更新版本软件，使用鼠标点击“开始/程序/Inovance Control/Inotouch/VSVComVCPP”：

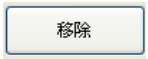


单击 VSVComVCPP 可以打开操作界面，如下图所示：



1）添加串口设备按钮 

，添加完成后，会出现如上图的串口设备列表。

2）移除串口驱动按钮 

，会卸载已经安装的串口驱动。

3）关闭串口按钮 关闭所有穿透串口，点击此按钮会关闭所有穿透串口；会关闭所有的已经启动的穿透串口。

4）串口设备列表

显示已安装的串口设备，双击列表中的串口号可以打开穿透通信的设置页面。

22.2 功能使用说明

1）HMI 通信方式选择：可分为以太网和 USB 两种方式。如下图所示：

2）本机 PLC 调试软件使用的串口

安装完驱动后，虚拟串口号会自动配置为有效的串口。默认值不需要更改。虚拟 COM 口号必须与 PLC 接口一致。可查看 PLC 串口号，点击 AutoShop 的工具栏“工具/通讯配置”，弹出如下对话框：

3） HMI 穿透功能启动

点击下图的"启动穿透通信"按钮，正常启动可以看到 HMI 状态栏内显示"HMI 已连接"，代表连接成功。

4）设置连接参数

HMI IP、端口：为 HMI 的网络通信 IP 和端口。端口后面可以选择通信协议类型，TCP 或 UDP。

HMI 中 PLC 端口: 此端口为 PLC 接在 HMI 上的串口号。

设置好的参数，可以通过“保存配置”按钮，将配置好的参数保存到配置文件中，下次软件启动将自动应用保存的配置参数。

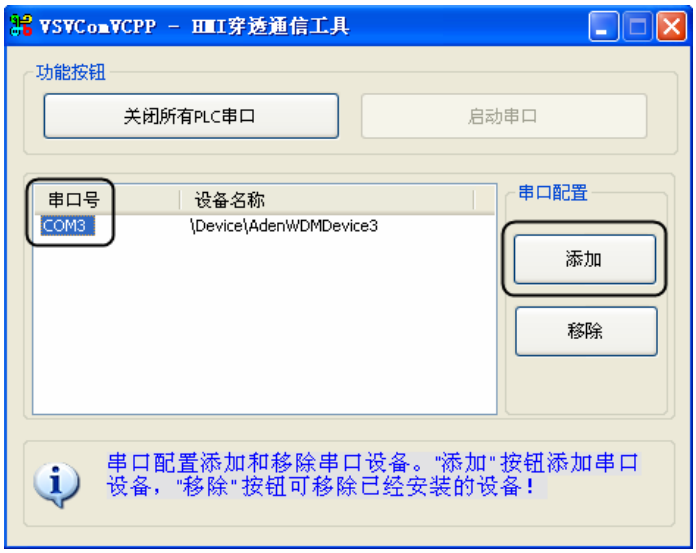
正常连接 HMI 后，PLC 后台可以使用启动的串口设备，直接与 PLC 通信。

注意：HMI 与 PLC 穿透的过程中，只能有一个功能进行通讯。例如穿透监控 PLC 数据，HMI 就不能读取其 PLC 的值。

22.3 穿透举例说明

例子 1：使用 USB 口与 HMI 穿透。

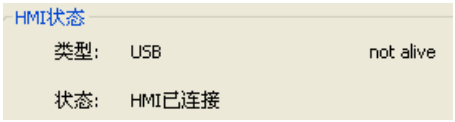
步骤 1：打开 VSVComVCPP 软件；点击“添加”，就自动添加一个串口“COM3”，如下图所示：



步骤 2：双击串口号“COM3”。弹出如下对话框， 选择 USB 连接方式。虚拟串口号为默认串口号，HMI 中 PLC 端口：为 HMI 与 PLC 通讯连接的 COM 口，设为：COM1。

步骤 3：点击“启动穿透通信”，开始进行通讯。穿透工具里的 COM 口号与 PLC 通讯的 COM 口号要一致。

步骤 4：完成上述操作后，HMI 状态显示“HMI 已连接”如下图所示，可对 HMI 及 PLC 进行上传下载功能。



例子 2：使用以太网与 PLC 进行穿透

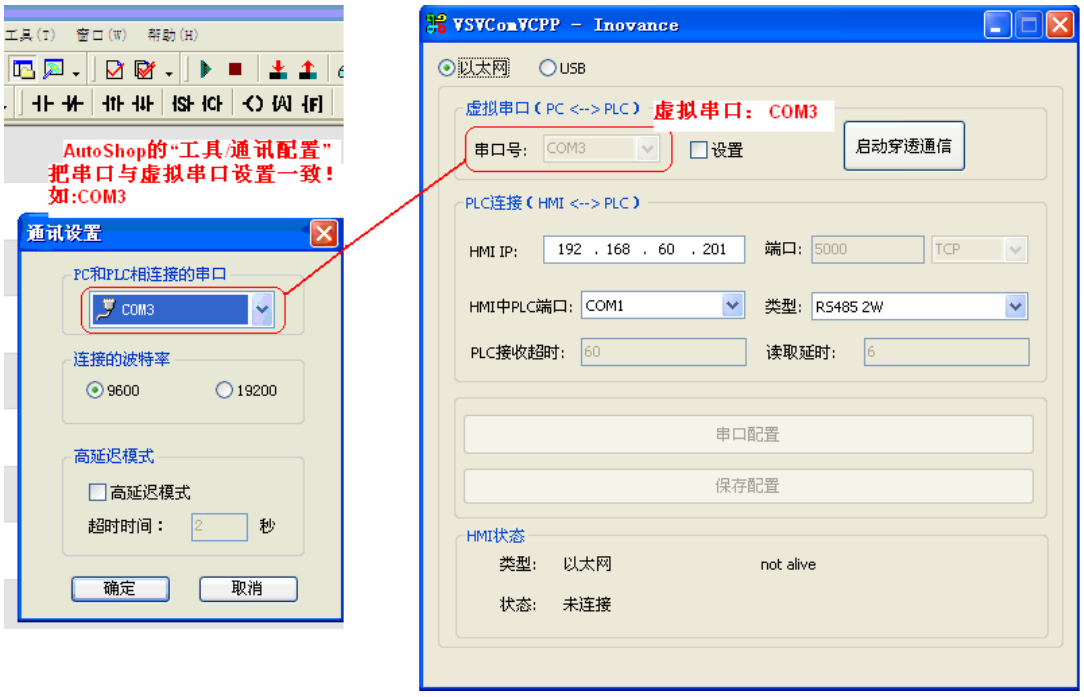
步骤 1：与例子 1 的步骤 1 相同。

步骤 2：双击串口号“COM3”。弹出如下对话框。选择以太网连接方式，虚拟串口号为默认串口号。

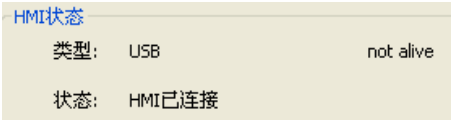
HMI IP地址：为远程HMI的地址，参考（**1.4 InoTouch** 系列人机界面的系统设定/a、设定人机界面的IP 地址）；最后请确认计算机的IP与HMI的IP互相访问；

HMI 中 PLC 端口：为 HMI 与 PLC 通讯连接的 COM 口。

步骤 3：点击“启动穿透通信”，开始进行通讯。穿透工具里的 COM 口号与 PLC 通讯的 COM 口号要一致。



步骤 4：完成上述操作后，HMI 状态显示“HMI 已连接”如下图所示，可对 HMI 及 PLC 进行上传下载功能。



注意：如果使用以太网连接中，对方 IP 下有路由器分流后连接触摸屏的，VSVComVCPP 软件中以太网地址仍为远端 IP 地址，路由器下面的 HMI 要在路由器中开放端口 5000 作为 HMI 通讯的端口进行穿透。